WORKSHEET

A8.1 Logic gates and notation

Boolean algebra and logic circuits · A level · OCR H446 1.4.3, AQA 7517 4.6.4.1, Eduqas A500QS 1.2 · about 20 min

BugBotLab

Name _____ Class _____ Date _____

What this lesson is about

NOT, AND, OR, XOR, NAND and NOR, the notation each board uses, and why NAND alone can build anything.

- The six gates
- Three ways to write the same thing
- Gates in Python
- NAND can build anything
- A robot example

Questions

6 marks in all

1. When does a NAND gate output 0? [1 mark]

- ☐ **A** Only when both inputs are 1
- ☐ **B** Only when both inputs are 0
- ☐ **C** When the inputs are different
- ☐ **D** Never

2. When does a NOR gate output 1? [1 mark]

- ☐ **A** Only when both inputs are 0
- ☐ **B** When at least one input is 1
- ☐ **C** Only when both inputs are 1
- ☐ **D** When the inputs are different

3. In OCR notation, which expression means A XOR B? [1 mark]

- ☐ **A** $A \veebar B$
- ☐ **B** $A \vee B$
- ☐ **C** $A \wedge B$
- ☐ **D** $\neg(A \wedge B)$

4. What does this program print? [1 mark]

```
a = 1
b = 0
print(1 - (a & b), 1 - (a | b), a ^ b)
```

5. What does this program print? [1 mark]

```
a = 1
print(~a, 1 - a, a ^ 1)
```

6. Why are NAND and NOR called universal gates? [1 mark]

- ☐ **A** Any logic circuit can be built from NAND gates alone, or from NOR gates alone
- ☐ **B** They are used in every processor made
- ☐ **C** They have the most rows in their truth tables
- ☐ **D** They work with any number of inputs

The task: everything from NAND

You are given `NAND(a, b)`, which takes two bits (each 0 or 1) and returns 0 or 1. Write four functions, each taking bits and returning 0 or 1, built **only** by calling `NAND` or the gates you have already built from it: `NOT(a)`, `AND(a, b)`, `OR(a, b)` and `XOR(a, b)`. Your program may not use `and`, `or`, `not`, comparisons such as `==`, a minus sign, or the operators `&`, `|`, `^` and `~`. Then use loops to print one line for each of the four input combinations, in binary order, in exactly this form: `A=0 B=1 NOT_A=1 AND=0 OR=1 XOR=1`

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

def NAND(a, b):
    return [1, 1, 1, 0][2 * a + b]

def NOT(a):
    return 0

def AND(a, b):
    return 0

def OR(a, b):
    return 0

def XOR(a, b):
    return 0
```

Plan your program here, then type it in and press Run.

Do it on the robot

www.bugbotlab.com/learn/a8-1-logic-gates-and-notation/

The simulator checks it and tells you when it passes. Nothing to install, no account.

Challenges

1. Build NOT, AND and OR from `NOR` alone. Which is easier to build from NOR than from NAND?
2. How many NAND gates does your XOR use in total, counting the ones inside the gates it calls? Can you do it in four?
3. Write `A ⊻ B` using only AND, OR and NOT, in both OCR and AQA notation.