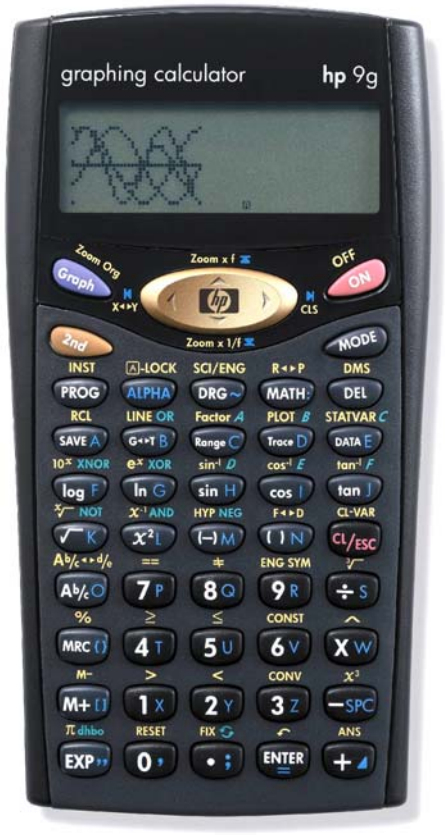
hp calculators



HP 9g Logical Operations with Base-N Numbers

Numbers in Different Bases

The Base-N Mode

Logical Operations with Base-N Numbers

Practice Performing Logical Operations in Base-N Mode

Numbers in different bases

Our number system (called the Hindu-Arabic number system) is a decimal system (it's also sometimes referred to as denary system) because it counts in 10s and powers of 10. Its base, i.e. the number on which a number system is built, is therefore 10. While base 10 numbers are extensively used, this not the only possible base. There have been number systems with base 20 (Mayas), mixed bases of 10 and 60 (used by the Babylonians), of 5 and 10 (ancient Romans), etc. Even nowadays the sexagesimal system (base 60) is used in some measurements of time and angle. The HP 9g enables you to work with numbers that are expressed in base 2, base 8, base 10 and base 16 numbers. All these bases are important in computing. Base 2 numbers are called binary numbers and their digits are limited to 1 and 0. A common abbreviation of binary digit is bit, which is either 1 or 0. Base 8 are called octal numbers, whose digits are 0, 1, 2, 3, 4, 5, 6 and 7. Finally, base 16 numbers are called hexadecimal numbers: 0, 1, 2, ,3 ,4 ,5 ,6 ,7 ,8 ,9, A, B, C, D, E and F.

The main difference between all these numbers is the value a digit have because of its place in a numeral. For example, in the decimal number 378 the digit 3 has value 300, 7 has value 70 and 8 has value 8. In other words:

$$378 = 3 \cdot 10^2 + 7 \cdot 10^1 + 8 \cdot 10^0$$

But if 378 were a hexadecimal number then its decimal value would be:

$$378\text{h} = 3 \cdot 16^2 + 7 \cdot 16^1 + 8 \cdot 16^0 = 888\text{d}$$

A small h, b, d and o after or before a number mean that this number is expressed in hexadecimal, binary, decimal or octal base respectively.

The Base-N mode

The HP 9g has a special operating mode in which binary, hexadecimal, octal and decimal operations and conversions are performed: the Base-N mode. Press MODE 2Y to set this mode. It works as the Main mode, there's an entry line, in which calculations are entered and a result line where results are displayed. In the left side of the display does always appear a small h, b, d , and o: it is the current base, on which numbers are entered and results are displayed. To set the current base press 2nd EXP‹› or simply EXP‹› to display the "dhbo" menu and select DEC, HEX, BIN or OCT with the arrow keys and press ENTER . Bear in mind that only those digits allowable in the current digits can be entered: for example 102 is an invalid number in binary mode, but you can enter numbers in a different base than the current one, by prefixing this number by the letter corresponding to its base. These letters (p, d, h and b) are in the "dhbo" menu, displayed by EXP‹› .

Not all the operations and functions that are available in the Main mode are available in Base-N. The valid operations are the basic arithmetic (+ × − ÷) and the functions MAX (MATH: MATH: 0›), MIN (MATH: MATH: 1X), SUM (MATH: MATH: 2Y) and AVG (MATH: MATH: 3Z) in the MATH menu, as well as all the memory related operations (running memory, variables, ANS). In addition to these operations, the HP 9g provides six logical operations which are described below. Even though basic arithmetic functions will work with numbers in different bases, these numbers must be *integers* – both arguments and results. In fact, when Base-N mode is active, the . ; key is disabled.

For an introduction to basic arithmetic with Base.N numbers and base conversions, refer to the HP 9g learning module *Base Conversions and Arithmetic.*

Logical Operations with Base-N Numbers

In addition to those operations, the HP 9g provides six logical operations which are listed below: (in parenthesis is the corresponding key – notice that pressing 2nd first is *not* necessary)

- AND (x^{-1} AND on the x^2 L key): Logical bit-by-bit AND of two arguments. E.g. b1010 AND b1100 returns b1000.
- NEG (HYP NEG on the (−) M key): Two's complement of the argument. That is to say, complements each bit and ads 1. Only in non-decimal bases. In base 10, NEG behaves as (−) M in Main mode, changing the sign of the argument. E.g. NEG b1100 returns b11111111111111111111111111110100.
- NOT ($\sqrt[y]{}$ NOT on the √ K key): One's complement of the argument. Each bit in the result is the complement of the corresponding bit in the argument. E.g. NOT b1100 returns b11111111111111111111111111110011.
- OR (LINE OR on the G↔T B key): Logical bit-by-bit OR of two arguments. E.g. b1010 OR b1100 returns b1110.
- XNOR (10^x XNOR on the log F key): Logical bit-by-bit XNOR of two arguments. *a* XNOR *b* is the same as NOT (*a* XOR *b*). This operation is sometimes called equivalence. E.g. b1010 XNOR b1100 returns b11111111111111111111111111111001.
- XOR (e^x XOR on the ln G key): Logical bit-by-bit exclusive OR of two arguments. E.g. b1010 XOR b1100 returns b0110.

Base-N mode is assumed in all the following examples. To set this mode now press MODE 2 Y .

Practice performing logical operations in Base-N mode

Example 1: Add the binary number 11011 to its one's complement and subtract its two's complement to that result.

Solution: Make sure the current base is 2 by pressing EXP , then selecting BIN if needed, and finally pressing ENTER . The calculation in question is 11011 + NEG(11011) – NOT (11011) . We'll store 1101 in the variable ANS first, to save a few keystrokes.

1 X 1 X 0 › 1 X 1 X ENTER

We can now perform the calculation by pressing

+ HYP NEG 2nd ANS − SPC $\sqrt[y]{}$ NOT 2nd ANS

Note that parentheses are not needed because NEG and NOT have priority over the addition or subtraction.

Answer: 11100 base 2.

Example 2: De Morgan's theorem states that NOT(*a* OR *b*) = (NOT *a*) AND (NOT *b*). Verify this equation for a = hFF23 and b = d4502.

Solution: We'll set the current base to 16. To do so press

EXP‹‹ ◀ ENTER

Let's evaluate the left side of the equation by pressing

NOT ( ) N tan⁻¹ F tan⁻¹ F 2 Y 3 Z LINE OR EXP‹‹ ▲ ENTER 4 T 5 U 0 , 2 Y ENTER

This results in the hex number FFFF0048. Parentheses were mandatory in this case because NOT takes priority over OR and AND. But, for the same reason, they are not necessary when evaluating the right side:

NOT tan⁻¹ F tan⁻¹ F 2 Y 3 Z x⁻¹ AND NOT EXP‹‹ ▲ ENTER 4 T 5 U 0 , 2 Y

Answer: The result of the right side is also hFFFF0048. So this equation is true for these particular values.

Example 3: A Logic student needs to write the truth table for the sentence $(p \rightarrow q) \vee \neg (p \leftrightarrow \neg q)$. Will his sister's HP 9g be of any use to him?

Solution: Logic connectives can be written in terms of the NOT, OR, AND and XNOR functions. This is the equivalence:

$p \rightarrow q$	NOT p OR q
$\neg p$	NOT p
$p \vee q$	p OR q
$p \wedge q$	p AND q
$p \leftrightarrow q$	p XNOR q

Therefore, that sentence can be written as: (NOT p OR q) OR NOT(p XNOR NOTq), which the HP 9g can evaluate provided p and q contain the binary digits 0 or 1 (which correspond to False and True respectively). In order to automate the process, the best option is to write a small program that inputs p and q and evaluates the expression. To enter the following program, press MODE 3 Z 0 , 1 X select an unused program number, and then enter the following four lines:

Program Line	Command	Keys
Line 1	INPUT P	2nd INST ▲ ▲ ▲ 0 , ALPHA 7 P ENTER
Line 2	INPUT Q	2nd INST ▲ ▲ ▲ 0 , ALPHA 8 Q ENTER
Line 3	A=(NOT P OR Q) OR NOT (P XNOR NOT Q)	ALPHA SAVE A SAVE A ENTER ( ) N NOT ALPHA 7 P LINE OR ALPHA 8 Q ▶ LINE OR NOT ( ) N ALPHA 7 P 10ˣ XNOR NOT ALPHA 8 Q ▶ ENTER
Line 4	PRINT A	2nd INST ▲ ▲ ▲ 3 Z ALPHA SAVE A

The program must be run in Prog Run mode: MODE 3 Z 1 X . Then, select the program number where the program has been stored and press ENTER Values for P and Q are then requested, make sure that BIN mode is set (otherwise, press EXP‹‹ select BIN and press ENTER). All possible combinations of 1's and 0's must be entered to complete the truth value. Keep in mind that the results are 32-bit numbers and that the significant digit is the one displayed at the end of the entry line, so press ALPHA ▶| to view it. If this digit is 0 then the sentence is false for the given values of p and q, whereas if it's 1 then the sentence is true. To repeat the evaluation just press ENTER and the calculator will prompt you for a new value of P.

Answer: Running the above program four times with the following values of P and Q, returns the values shown in the third column. Remember that T means 1 and F means 0.

p	q	$(p \rightarrow q) \vee \neg (p \leftrightarrow \neg q)$
T	T	T
T	F	F
F	T	T
F	F	T