hp calculators

HP 49G+ Operations on binary numbers

The BASE menu

Operations on binary numbers

Practice performing operations on binary numbers

The BASE menu

The BASE menu is the RED shifted function of the 3 key and can be accessed by pressing → BASE . The screen displays the Base menu containing eleven functions for working with numbers in different bases. The first four functions allow the base to be changed to HEX for hexadecimal base 16 (which is chosen here, showing in reverse letters as choice one), DEC for decimal base 10, OCT for octal base 8, and BIN for binary base 2. The 49G+ calls such numbers "binary" even if they are not in base 2. All such numbers are proceeded with the # sign in front of the number. To enter a number in the current base, press ← # before typing in the number.

Figure 1

Beginning with function five in the menu, there are several functions and sub-menus containing additional functions for working with binary numbers. Functions five and six convert numbers from real to binary and from binary to real. Choice seven in the menu displays a sub-menu with choices for applying logic functions to binary numbers. Choice eight displays a sub-menu for working with binary numbers at the binary bit level. Choice nine displays a sub-menu for working with binary numbers at the byte level. Choice ten and choice eleven (not shown in the display below) allow for the selection of the word size. The default word size is 64 bits.

Figure 2

Operations on binary numbers

Any binary number, regardless of the base in which it is displayed, can still be thought of as a collection of 1's and 0's. For example, the number 24 in base 10 is also the number 11000 in binary. This is because 11000 in binary is equal to $1 \times 2^4 + 1 \times 2^3$, or 24. The 49G+ contains many functions in the LOGIC, BINARY and BYTE sub-menus that operate on binary numbers. Many of these functions operate on binary numbers at the bit or byte level.

The LOGIC sub-menu contains the functions listed below. The AND function compares two binary numbers at the bit-by-bit level and creates a new binary number with a 1 for each bit position that contains a 1 in both numbers in the same position. The OR function compares two binary numbers at the bit-by-bit level and creates a new binary number with a 1 for each bit position if either original number contains a 1 at the same bit position. The XOR function (which stands for exclusive OR), does the same thing as the OR function, but only for positions where the original numbers contained a 1 and/or 0, but not where both contain a 1. The NOT function creates a new binary number where each bit position's value has been "flipped", with each 1 becoming a 0 and each 0 becoming a 1.

Figure 3

The BIT sub-menu contains the functions listed below. The RL function rotates a binary number 1 bit to the left. The SL function shifts a binary number 1 bit to the left. The ASR function performs an arithmetic shift to the right. The function SR shifts a binary number one bit to the right. The function RR rotates a binary number 1 bit to the right.

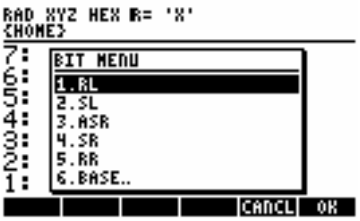

Figure 4

The BYTE sub-menu contains the functions listed below. The perform the same functions as the BIT functions above but work with bytes not bits.

Figure 5

The examples that follow illustrate some of these functions.

Practice performing operations on binary numbers

Example 1: Evaluate NOT(#4567 d). Set the wordsize to 16 bits.

Solution: First, set the wordsize to 16 bits and make sure the calculator is in DEC mode to enter the base 10 number. Assumes RPN mode.

1 6 ENTER → BASE 1 ENTER → BASE 2 ENTER ← # 4 5 6 7 ENTER
→ BASE 7 ENTER 4 ENTER

Figure 6

Answer: 60968 base 10.

Example 2: Perform an OR on these two binary numbers: #70114 o and #57610 o. Set the wordsize to 32 bits.

Solution: First, set the wordsize to 32 bits and make sure the calculator is in OCT mode to enter the base 8 number. Assumes Algebraic mode.

→ BASE 1 ENTER 3 2 ENTER → BASE 3 ENTER ← # 7 0 1 1 4 ENTER
→ BASE 7 ENTER 2 ENTER ← # 5 7 6 1 0 ENTER

Figure 7

Answer: 77714 base 8.

Example 3: Shift the binary number: #ABCD h one byte to the right. Use a 32 bit wordsize. Show the result in bases 16 and 10.

Solution: First, set the wordsize to 32 bits and make sure the calculator is in HEX mode to enter the base 16 number. Assumes RPN mode.

3 2 ENTER ↱ BASE 1 ENTER ↱ BASE ENTER ↰ # ALPHA ALPHA A B C D ENTER
↱ BASE 9 ENTER 3 ENTER

Figure 8

↱ BASE 2 ENTER

Figure 9

Answer: AB base 16 and 171 base 10.

Example 4: Rotate the binary number: 11000101 b one bit to the right. Set the wordsize to 16 bits.

Solution: First, set the wordsize to 16 bits and make sure the calculator is in BIN mode to enter the base 2 number. Assumes RPN mode.

1 6 ENTER ↱ BASE 1 ENTER ↱ BASE 4 ENTER ↰ # 1 1 0 0 0 1 0 1
ENTER ↱ BASE 8 ENTER 5 ENTER

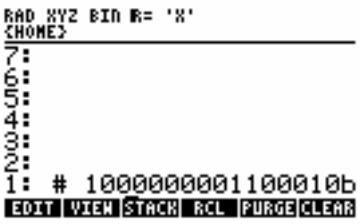

Figure 10

Answer: 1000000001100010 base 2. The extra zeroes come from the 16 bit wordsize.