Proof for the Highest Integer that can be Represented in Binary given a Specific Number of Bits

Protean Onion

1 Motivation

From experiments in interactive Python, I saw that

$$2^n - 1 = \sum_{i=0}^{n-1} 2^i, \tag{1}$$

where n represents the number of bits allocated to an array.

However, it wasn't intuitively clear to me. Why is it that the equation holds?

2 Approach: Proof

I decided to attempt a proof, of which workings have been given below. The method of proof that I understood the logic of was proof-by-induction: check if the statement holds for a given value of a variable, assume it for k, and show that if the statement holds for k, it must hold for $k + 1$. So I decided to follow that method.

2.1 Proof by Induction

Setting $n = 1$, we have

$$LHS : 2^1 - 1 = 1, \text{and} \tag{2}$$

$$RHS : \sum_{i=0}^{0} 2^i = 2^0 = 1. \tag{3}$$

We see that equation (1) holds for $n = 1$.

Let us now assume that the equation holds for $n = k$. Thus, we have

$$2^k - 1 = \sum_{i=0}^{k-1} 2^i. \tag{4}$$

Setting $n = k + 1$, we have

$$\begin{aligned} 2^{k+1} - 1 &= \sum_{i=0}^{k} 2^i \\ \implies 2^k.2 - 1 &= 2^0 + 2^1 + 2^2 + ... + 2^k \\ \implies 2^k.2 &= 1 + 2^0 + 2^1 + 2^2 + ... + 2^k \\ \implies 2^k.2 &= 2 + 2^1 + 2^2 + ... + 2^k. \end{aligned}$$

Dividing both sides by 2, we have

$$2^k = 1 + 2^0 + 2^1 + 2^2 + ... + 2^{k-1}.$$

Subtracting 1 from both sides, we have

$$2^k - 1 = 2^0 + 2^1 + 2^2 + ... + 2^{k-1}. \tag{5}$$

Equation (5) simplifies to equation (4), which we assume to be true. Since the equation holds for $n = 1$, and since we have shown that, given that the equation holds for $n = k$, it must necessarily hold for $n = k + 1$, it follows that the equation holds. Thus, we have proven, through the method of induction, that equation (1) is true.

3 Conceptual Intuition

Though we have proven that the equation holds, it is still not clear why it expresses the numerical truth of the limits of binary representation of number. To state the question clearly, mathematically, it is clear that the two expressions in equation (1) are equivalent. However, given an 8-bit array, for example, representing a memory location of an integer, and given that each bit in the array represents a value in the form of

$$2^{\text{ordinal index of bit in array}} \times \text{Value of Bit},$$

and that the integer the array represents is calculated by summing over the values represented by each bit in the array, why is it that $2^n - 1$ is a valid alternative expression for the upper limit of the integer? In other words, how is it that the expression fittingly quantifies the (mental) phenomenon of binary representation of number? Is it just the case that $2^{n+1} - 1$ is a substitute for $\sum_{i=0}^{n} 2^i$ only insofar as they are operationally equivalent, or is there more to the story?

3.1 Ideas

I have no f-ing idea. Help!