Tnum <AND, OR xor AND> Proof

2.24.21

a) Let A be the set of concrete values represented by tnum a.

b) Let a_{and}, a_{or} represent functions that perform bitwise AND and bitwise OR, respectively, on all members of set A:

$$A = \{a_1, a_2, a_3, ..., a_n\}$$

$$a_{and} = a_1 \wedge a_2 \wedge a_3 \wedge ... \wedge a_n$$

$$a_{or} = a_1 \vee a_2 \vee a_3 \vee ... \vee a_n$$

c) Definition of well formed tnum: a.value $\wedge$ a.mask $= 0$

d) Definition of tnum membership: $x \in a \iff x \wedge \neg a.mask = a.value$

e) Observations:

1. If all members of set A contain 1 in the ith bit, then a_{and} will return 1 in the ith bit. This corresponds to the known 1's in the tnum.
2. If all members of set A contain 0 in the ith bit, then a_{or} will return 1 in the ith bit. This corresponds to the known 0's in the tnum.
3. Any 1 in the ith bit of the resulting bitvector $a_{or} \oplus a_{and}$ corresponds to uncertain bits in the tnum. Let $a_{uncertain} = a_{or} \oplus a_{and}$ (where $\oplus$ represents bitwise xor)
4. Any 1 in the ith bit of the resulting bitvector $\neg(a_{or} \oplus a_{and})$ corresponds to certain bits in the tnum. Let $a_{certain} = \neg(a_{or} \oplus a_{and})$

Claim: a.value $= a_{and}$, a.mask $= a_{or} \oplus a_{and}$

Proof of soundness: First, we can show that our claim produces a well formed tnum in the following way :

$$\text{a.value} \wedge \text{a.mask} = 0$$

$$= a_{and} \wedge (a_{or} \oplus a_{and}) = 0$$

$$= (a_{and} \wedge a_{or}) \oplus (a_{and} \wedge a_{and}) = 0$$

$$= (a_{and} \wedge a_{or}) \oplus (a_{and} \wedge a_{and}) = 0$$

$$= a_{and} \oplus a_{and} = 0$$

*note that, by definition, $a_{and} \subseteq a_{or}$ which implies that $a_{and} \wedge a_{or} = a_{and}$.

Let A_k be an arbitrary member of A and $A_k[i]$ denote the ith bit of member A_k. Now, using a case analysis, we show that all members of A are represented by the tnum $< a_{and}, a_{or} \oplus a_{and} >$ by satisfying the definition of tnum membership:

$$A_i \wedge \neg a.mask = a.value$$

$$= A_k \wedge \neg(a_{or} \oplus a_{and}) = a_{and}$$

$$= A_k[i] \wedge a_{certain}[i] = a_{and}[i]$$

1) $A_k[i] = 0$. This implies that $a_{and}[i] = 0$ since a_{and} will capture any 0 in the ith of a member of A if such exists. Thus the bitwise operation $A_k[i] \wedge a_{certain}[i] = a_{and}[i]$ holds regardless of the value of $a_{certain}[i]$.

2) $A_k[i] = 1$ and $a_{and}[i] = 1$. In this case the the ith bit must be a certain 1 since it is contained by a_{and}. This implies that $a_{certain}[i] = 1$, which means that $A_k[i] \wedge a_{certain}[i] = a_{and}[i]$ must also be true.

3 $A_k[i] = 1$ and $a_{and}[i] = 0$. In this case, the 1 in the ith bit is uncertain since it is not present in a_{and} which implies that $a_{certain} = 0$. Thus, $A_k[i] \wedge a_{certain}[i] = a_{and}[i]$ must hold in this case as well.

Proof of maximal precision: