A theory provides three names: \( set \), \( flip \), and \( ask \). It is presented by an implementation. Let \( u: bin \) be the user's variable, and let \( v: bin \) be the implementer's variable. The axioms are:

\[
\begin{align*}
set & \equiv v := T \\
flip & \equiv v := \neg v \\
ask & \equiv u := v
\end{align*}
\]

(a) Replace \( v \) with \( w: nat \) according to the data transformer \( v = even w \).

(b) Replace \( v \) with \( w: nat \) according to the data transformer \( (w=0 \Rightarrow v) \land (w=1 \Rightarrow \neg v) \). Is anything wrong?

\[ 
\text{Operation } set \text{ becomes } \\
\forall v \cdot (w=0 \Rightarrow v) \land (w=1 \Rightarrow \neg v) \Rightarrow \exists v': (w'=0 \Rightarrow v') \land (w'=1 \Rightarrow \neg v') \land (v'=T) \\
= u'=u \land w'+1
\]

Operation \( flip \) becomes

\[ 
\forall v \cdot (w=0 \Rightarrow v) \land (w=1 \Rightarrow \neg v) \Rightarrow \exists v': (w'=0 \Rightarrow v') \land (w'=1 \Rightarrow \neg v') \land (v'=v) \\
= u'=u \land (w'+0 \Rightarrow w'+1 \land (v'=w+1 \Rightarrow u'))
\]

Operation \( ask \) becomes

\[ 
\forall v \cdot (w=0 \Rightarrow v) \land (w=1 \Rightarrow \neg v) \Rightarrow \exists v': (w'=0 \Rightarrow v') \land (w'=1 \Rightarrow \neg v') \land (u'=u) \\
= (w=0 \land w'+1 \land u') \lor (w=1 \land w'+0 \land \neg u')
\]

Something is wrong. Although \( (w=0 \Rightarrow v) \land (w=1 \Rightarrow \neg v) \) is a data transformer, it is a rather weak one because when \( w \) is neither 0 nor 1 it doesn't constrain \( v \). So the result is that \( ask \) is transformed into something that's unimplementable.

(c) Replace \( v \) with \( w: nat \) according to \( (v \Rightarrow w=0) \land (\neg v \Rightarrow w=1) \). Is anything wrong?

\[ 
\text{Operation } set \text{ becomes } \\
\forall v \cdot w=0 \land (\neg v \Rightarrow w=1) \Rightarrow \exists v': (v' \Rightarrow w'=0) \land (\neg v' \Rightarrow w'=1) \land (v'=T) \\
= w=0 \land 1 \Rightarrow (w'=0)
\]

Operation \( flip \) becomes

\[ 
\forall v \cdot (v \Rightarrow w=0) \land (\neg v \Rightarrow w=1) \Rightarrow \exists v': (v' \Rightarrow w'=0) \land (\neg v' \Rightarrow w'=1) \land (v'=v) \\
= w=0 \land 1 \Rightarrow (u'=w)
\]

Operation \( ask \) becomes

\[ 
\forall v \cdot (v \Rightarrow w=0) \land (\neg v \Rightarrow w=1) \Rightarrow \exists v': (v' \Rightarrow w'=0) \land (\neg v' \Rightarrow w'=1) \land (u'=v) \\
= w=0 \land 1 \Rightarrow (u'=w)
\]

Something is wrong. We have been transforming with something that isn't a transformer; it's too strong.

\[ 
\forall w \cdot \exists v \cdot (v \Rightarrow w=0) \land (\neg v \Rightarrow w=1) \\
= \forall w \cdot w=0 \lor w=1 \\
= \bot
\]

The last line isn't a theorem, so neither is the first. Nothing constrains the implementation to start in a state where \( w=0 \land v=1 \). If it starts with \( w=2 \), then \( set \) might not set \( w \) to 0, after which \( ask \) will give the wrong answer.