$In(1,X_1)$ to denote the first input terminal for circuit X_1. A similar function $Out(n,c)$ is used for output terminals. The predicate $Arity(c,i,j)$ asserts that circuit c has i input and j output terminals. The connectivity between gates can be represented by a predicate, *Connected*, which takes two terminals as arguments, as in $Connected(Out(1,X_1), In(1,X_2))$.

Finally, we need to know whether a signal is on or off. One possibility is to use a unary predicate, $On(t)$, which is true when the signal at a terminal is on. This makes it a little difficult, however, to pose questions such as "What are all the possible values of the signals at the output terminals of circuit C_1 ?" We therefore introduce as objects two signal values, 1 and 0, representing "on" and "off" respectively, and a function $Signal(t)$ that denotes the signal value for the terminal t.

Encode general knowledge of the domain

One sign that we have a good ontology is that we require only a few general rules, which can be stated clearly and concisely. These are all the axioms we will need:

1. If two terminals are connected, then they have the same signal:
   $\forall t_1,t_2\ Terminal(t_1) \wedge Terminal(t_2) \wedge Connected(t_1,t_2) \Rightarrow$
   $Signal(t_1) = Signal(t_2)$.
2. The signal at every terminal is either 1 or 0:
   $\forall t\ Terminal(t) \Rightarrow Signal(t) = 1 \vee Signal(t) = 0$.
3. *Connected* is commutative:
   $\forall t_1,t_2\ Connected(t_1,t_2) \Leftrightarrow Connected(t_2,t_1)$.
4. There are four types of gates:
   $\forall g\ Gate(g) \wedge k = Type(g) \Rightarrow k = AND \vee k = OR \vee k = XOR \vee k = NOT$.
5. An AND gate's output is 0 if and only if any of its inputs is 0:
   $\forall g\ Gate(g) \wedge Type(g) = AND \Rightarrow$
   $Signal(Out(1,g)) = 0 \Leftrightarrow \exists n\ Signal(In(n,g)) = 0$.
6. An OR gate's output is 1 if and only if any of its inputs is 1:
   $\forall g\ Gate(g) \wedge Type(g) = OR \Rightarrow$
   $Signal(Out(1,g)) = 1 \Leftrightarrow \exists n\ Signal(In(n,g)) = 1$.
7. An XOR gate's output is 1 if and only if its inputs are different:
   $\forall g\ Gate(g) \wedge Type(g) = XOR \Rightarrow$
   $Signal(Out(1,g)) = 1 \Leftrightarrow Signal(In(1,g)) \neq Signal(In(2,g))$.
8. A NOT gate's output is different from its input:
   $\forall g\ Gate(g) \wedge Type(g) = NOT \Rightarrow$
   $Signal(Out(1,g)) \neq Signal(In(1,g))$.
9. The gates (except for NOT) have two inputs and one output.
   $\forall g\ Gate(g) \wedge Type(g) = NOT \Rightarrow Arity(g,1,1)$.
   $\forall g\ Gate(g) \wedge k = Type(g) \wedge (k = AND \vee k = OR \vee k = XOR) \Rightarrow$
   $Arity(g,2,1)$
10. A circuit has terminals, up to its input and output arity, and nothing beyond its arity:
    $\forall c,i,j\ Circuit(c) \wedge Arity(c,i,j) \Rightarrow$
    $\forall n\ (n \leq i \Rightarrow Terminal(In(n,c))) \wedge (n > i \Rightarrow In(n,c) = Nothing) \wedge$
    $\forall n\ (n \leq j \Rightarrow Terminal(Out(n,c))) \wedge (n > j \Rightarrow Out(n,c) = Nothing)$