Description

The one bit adder was implemented using four OR gates, four AND gates, and two NOT gates. There are three input bits A, B, C, and two output bits sum, and carry. A and B are the two numbers to be added, C is the carry in bit, sum is the result, and carry is the carry out bit. A full adder can be represented by the equations:

$sum = (A \oplus B) \oplus C$
$carry = AB + C(A \oplus B)$

I used eight signals (D-K) to perform all the intermediate steps in these calculations.

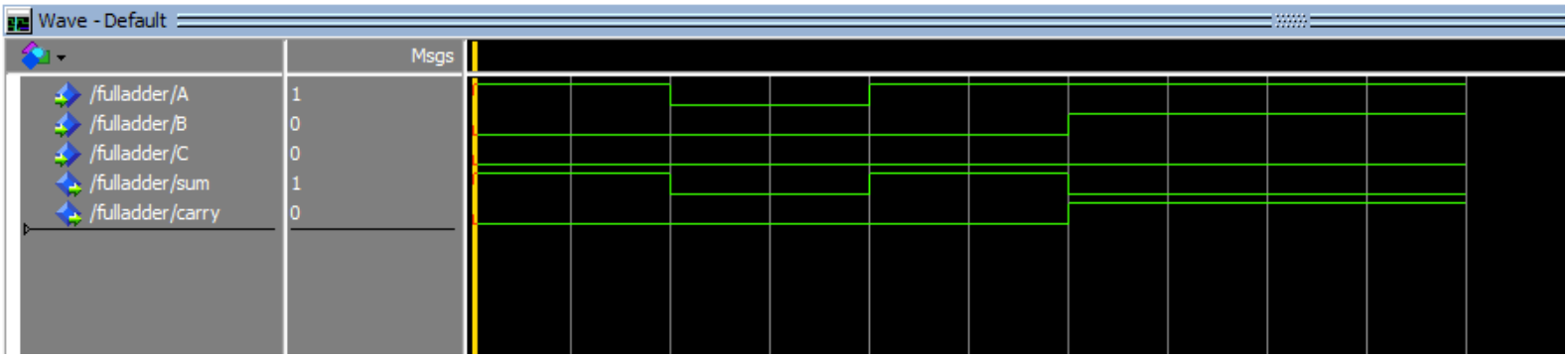
VDHL Code

```
library ieee;
use ieee.std_logic_1164.all;

entity FULLADDER is
	port(A, B, C : in std_logic;
		sum, carry : out std_logic);
end FULLADDER;

architecture fulladder_behav of FULLADDER is
	signal D, E, F, G, H, I, J, K : std_logic;
begin
	D <= A OR B;
	F <= A AND B;
	E <= D AND C;
	G <= E NOR F;
	H <= A OR B OR C;
	I <= A AND B AND C;
	J <= H AND G;
	K <= J NOR I;
	sum <= NOT K;
	carry <= NOT G;
end fulladder_behav;
```

Simulation



Analysis

The full adder got initial values of A=1 and B=0. The result was sum=1 and carry=0. Then 0 and 0 were added and the result was sum=0 and carry=0. Again 1 and 0 were added to produce sum=1 and carry0. Finally, 1 and 1 were added to produce sum=0 and carry=1.