```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
--val is the 24 bit value to be input into the compression.
--final is the 16 bit compressed value outputted.
entity audioc is
    Port    val :       in STD_LOGIC_VECTOR(23 downto 0);
          final :      out STD_LOGIC_VECTOR (15 downto 0);
end audioc;
--audioc is entity name
powerlog2 process(val srl 8)
begin:
    if    (x and "1000000000000000"/=0) then
        result <=( ("1111000000000000") + ((x-32768) srl 3));
    elsif (x and "1100000000000000"/=0) then
        result <=( ("1110000000000000") + ((x-16384) srl 2));
    elsif (x and "1010000000000000"/=0) then
        result <=( ("1101000000000000") + ((x-8192) srl 1));
    elsif (x and "1001000000000000"/=0) then
        result <=( ("1100000000000000") + ((x-4096)));
    elsif(x and "1000100000000000"/=0) then
        result <=( ("1011000000000000") + ((x-2048) sll 1) );
    elsif (x and "1000010000000000"/=0) then
        result <=( ("1010000000000000") + ((x-1024) sll 2));
    elsif (x and "1000001000000000"/=0) then
        result <=( ("1001000000000000") + ((x-512) sll 3));
    elsif (x and "1000000100000000"/=0) then
        result <=( ("1000000000000000") + ((x-256) sll 4));
    elsif (x and "1000000010000000"/=0) then
        result <=( ("0111000000000000") + ((x-128) sll 5));
    elsif (x and "1000000001000000"/=0) then
        result <=( ("0110000000000000") + ((x-64) sll 6));
    elsif (x and "1000000000100000"/=0) then
        result <=( ("0101000000000000") + ((x-32) sll 7));
    elsif (x and "1000000000100000"/=0) then
        result <=(("0100000000000000") + ((x-16) sll 8));
    elsif (x and "1000000000010000"/=0) then
        result <=(("0011000000000000") + ((x-8) sll 9));
    elsif (x and "1000000000001000"/=0) then
        result <=(("0010000000000000") + ((x-4) sll 10));
    elsif (x and "1000000000000100"/=0) then
        result <=(("0001000000000000") + ((x-2) sll 11));
    elsif (x and "1000000000000010"/=0) then
        result =((x-1) sll 12);
    elsif x = 0 then
        result <= "0000000000000000";
    end if;
end process;



func: process( val)
    variable compressed, remainder : SIGNED( 15 downto 0)
    type finalresult is array (4 downto 0) of integer range 0 to 65535; --final result is the
```

```
    array holding the final output of the system.(65535 represents 2^16 bits)
        for i in 0 to 4 loop
begin
--start for
            compressed := pwlog2(val srl 8); --compression process begins
            remainder := val srl ((compressed >>srl 12) + 2);--remainder term to be added to
            make a better approximation
            final := (compressed or remainder);--final value is made
            finalresult[i]] := final--the for loop is for a case where we have more than one
            24bit sample to compressed
--end for
        end loop;
end process;
end behavioral;
```