```vhdl
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date:    13:56:46 11/29/2018 
-- Design Name: 
-- Module Name:    BikeLockDebug - Behavioral 
-- Project Name: 
-- Target Devices: 
-- Tool versions: 
-- Description: 
--
-- Dependencies: 
--
-- Revision: 
-- Revision 0.01 - File Created
-- Additional Comments: 
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity BikeLockDebug is
   Port( clk,H,reset : in STD_LOGIC; -- Both H and reset are active high
            X : in STD_LOGIC_VECTOR (4 DOWNTO 0);
            L,A,R_state,L_state,A_state : out STD_LOGIC
            );
end BikeLockDebug;

architecture Behavioral of BikeLockDebug is
   Type state_type is (s0,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19,
                        s20,s21,s22,s23,s24,s25,s26,s27,s28,s29,s30,s31,s32,s33,s34,s35,s36,s37,s38,s39,
                        s40,s41,s42,s43,s44,s45,s46,s47,s48,s49,s50,s51,s52,s53,s54,s55,s56,s57,s58,s59);
   
   Signal p_state,n_state : state_type;

begin
   Process(clk,reset)
      begin
      if(reset='1') then
         p_state <= s0;
      elsif(clk' event and clk='1') then
         p_state <= n_state;
      end if;
   end process;
```

```

Process(X,H,p_state)
begin
   case h is
      when '1' =>
         case p_state is

   --This is where the Unarmed or Unlocked state begins
            when s0 =>
               case X is
                  when "00000" => n_state <= s0; L<='0'; A<='0';
                  when "00001" => n_state <= s0; L<='0'; A<='0';
                  when "00010" => n_state <= s0; L<='0'; A<='0';
                  when "00100" => n_state <= s1; L<='0'; A<='0';
                  when "01000" => n_state <= s0; L<='0'; A<='0';
                  when "10000" => n_state <= s0; L<='0'; A<='0';
                  when others =>
               end case;
            when s1 =>
               case X is
                  when "00000" => n_state <= s2; L<='0'; A<='0';
                  when "00100" => n_state <= s1; L<='0'; A<='0';
                  when others =>
               end case;
            when s2 =>
               case X is
                  when "00000" => n_state <= s2; L<='0'; A<='0';
                  when "00001" => n_state <= s3; L<='0'; A<='0';
                  when "00010" => n_state <= s59; L<='0'; A<='0';
                  when "00100" => n_state <= s59; L<='0'; A<='0';
                  when "01000" => n_state <= s59; L<='0'; A<='0';
                  when "10000" => n_state <= s59; L<='0'; A<='0';
                  when others =>
               end case;
            when s3 =>
               case X is
                  when "00000" => n_state <= s4; L<='0'; A<='0';
                  when "00001" => n_state <= s3; L<='0'; A<='0';
                  when others =>
               end case;
            when s4 =>
               case X is
                  when "00000" => n_state <= s4; L<='0'; A<='0';
                  when "00001" => n_state <= s59; L<='0'; A<='0';
                  when "00010" => n_state <= s59; L<='0'; A<='0';
                  when "00100" => n_state <= s59; L<='0'; A<='0';
                  when "01000" => n_state <= s5; L<='0'; A<='0';
                  when "10000" => n_state <= s59; L<='0'; A<='0';
                  when others =>
               end case;
            when s5 =>
               case X is
                  when "00000" => n_state <= s6; L<='0'; A<='0';
                  when "01000" => n_state <= s5; L<='0'; A<='0';
                  when others =>
               end case;
            when s6 =>
```

```vhdl
                     case X is
                        when "00000" => n_state <= s6; L<='0'; A<='0';
                        when "00001" => n_state <= s59; L<='0'; A<='0';
                        when "00010" => n_state <= s59; L<='0'; A<='0';
                        when "00100" => n_state <= s59; L<='0'; A<='0';
                        when "01000" => n_state <= s7; L<='0'; A<='0';
                        when "10000" => n_state <= s59; L<='0'; A<='0';
                        when others =>
                     end case;
                  when s7 =>
                     case X is
                        when "00000" => n_state <= s8; L<='0'; A<='0';
                        when "01000" => n_state <= s7; L<='0'; A<='0';
                        when others =>
                     end case;
                  when s8 =>
                     case X is
                        when "00000" => n_state <= s8; L<='0'; A<='0';
                        when "00001" => n_state <= s59; L<='0'; A<='0';
                        when "00010" => n_state <= s9; L<='0'; A<='0';
                        when "00100" => n_state <= s59; L<='0'; A<='0';
                        when "01000" => n_state <= s59; L<='0'; A<='0';
                        when "10000" => n_state <= s59; L<='0'; A<='0';
                        when others =>
                     end case;
                  when s9 =>
                     case X is
                        when "00000" => n_state <= s10; L<='0'; A<='0';
                        when "00010" => n_state <= s9; L<='0'; A<='0';
                        when others =>
                     end case;
                  when s10 =>
                     case X is
                        when "00000" => n_state <= s10; L<='0'; A<='0';
                        when "00001" => n_state <= s59; L<='0'; A<='0';
                        when "00010" => n_state <= s59; L<='0'; A<='0';
                        when "00100" => n_state <= s59; L<='0'; A<='0';
                        when "01000" => n_state <= s59; L<='0'; A<='0';
                        when "10000" => n_state <= s11; L<='0'; A<='0';
                        when others =>
                     end case;
                  when s11 =>
                     case X is
                        when "00000" => n_state <= s12; L<='0'; A<='0';
                        when "10000" => n_state <= s11; L<='0'; A<='0';
                        when others =>
                     end case;

      --This is where the Armed or Locked state begins
         --This is the 'correct input' path for the Armed or Locked State
                  when s12 =>
                     case X is
                        when "00000" => n_state <= s12; L<='1'; A<='0';
                        when "00001" => n_state <= s24; L<='1'; A<='0';
                        when "00010" => n_state <= s24; L<='1'; A<='0';
                        when "00100" => n_state <= s13; L<='1'; A<='0';
                        when "01000" => n_state <= s24; L<='1'; A<='0';
                        when "10000" => n_state <= s12; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s13 =>
                     case X is
                        when "00000" => n_state <= s14; L<='1'; A<='0';
                        when "00100" => n_state <= s13; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s14 =>
                     case X is
                        when "00000" => n_state <= s14; L<='1'; A<='0';
                        when "00001" => n_state <= s15; L<='1'; A<='0';
                        when "00010" => n_state <= s26; L<='1'; A<='0';
                        when "00100" => n_state <= s37; L<='1'; A<='0';
                        when "01000" => n_state <= s26; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s15 =>
                     case X is
                        when "00000" => n_state <= s16; L<='1'; A<='0';
                        when "00001" => n_state <= s15; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s16 =>
                     case X is
                        when "00000" => n_state <= s16; L<='1'; A<='0';
                        when "00001" => n_state <= s28; L<='1'; A<='0';
                        when "00010" => n_state <= s28; L<='1'; A<='0';
                        when "00100" => n_state <= s39; L<='1'; A<='0';
                        when "01000" => n_state <= s17; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s17 =>
                     case X is
                        when "00000" => n_state <= s18; L<='1'; A<='0';
                        when "01000" => n_state <= s17; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s18 =>
                     case X is
                        when "00000" => n_state <= s18; L<='1'; A<='0';
                        when "00001" => n_state <= s30; L<='1'; A<='0';
                        when "00010" => n_state <= s30; L<='1'; A<='0';
                        when "00100" => n_state <= s41; L<='1'; A<='0';
                        when "01000" => n_state <= s19; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s19 =>
                     case X is
                        when "00000" => n_state <= s20; L<='1'; A<='0';
                        when "01000" => n_state <= s19; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s20 =>
                     case X is
                        when "00000" => n_state <= s20; L<='1'; A<='0';
                        when "00001" => n_state <= s32; L<='1'; A<='0';
                        when "00010" => n_state <= s21; L<='1'; A<='0';
                        when "00100" => n_state <= s43; L<='1'; A<='0';
                        when "01000" => n_state <= s32; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s21 =>
                     case X is
                        when "00000" => n_state <= s22; L<='1'; A<='0';
                        when "00010" => n_state <= s21; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s22 =>
                     case X is
                        when "00000" => n_state <= s22; L<='1'; A<='0';
                        when "00001" => n_state <= s34; L<='1'; A<='0';
                        when "00010" => n_state <= s34; L<='1'; A<='0';
                        when "00100" => n_state <= s45; L<='1'; A<='0';
                        when "01000" => n_state <= s34; L<='1'; A<='0';
                        when "10000" => n_state <= s23; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s23 =>
                     case X is
                        when "00000" => n_state <= s0; L<='0'; A<='0';
                        when "10000" => n_state <= s23; L<='1'; A<='0';
                        when others =>
                     end case;

            --These is the 'failed input' path for the Armed state
                  when s24 =>
                     case X is
                        when "00000" => n_state <= s25; L<='1'; A<='0';
                        when "00001" => n_state <= s24; L<='1'; A<='0';
                        when "00010" => n_state <= s24; L<='1'; A<='0';
                        when "00100" => n_state <= s24; L<='1'; A<='0';
                        when "01000" => n_state <= s24; L<='1'; A<='0';
                        when "10000" => n_state <= s24; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s25 =>
                     case X is
                        when "00000" => n_state <= s25; L<='1'; A<='0';
                        when "00001" => n_state <= s26; L<='1'; A<='0';
                        when "00010" => n_state <= s26; L<='1'; A<='0';
                        when "00100" => n_state <= s37; L<='1'; A<='0';
                        when "01000" => n_state <= s26; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s26 =>
                     case X is
                        when "00000" => n_state <= s27; L<='1'; A<='0';
                        when "00001" => n_state <= s26; L<='1'; A<='0';
                        when "00010" => n_state <= s26; L<='1'; A<='0';
                        when "01000" => n_state <= s26; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s27 =>
                     case X is
                        when "00000" => n_state <= s27; L<='1'; A<='0';
                        when "00001" => n_state <= s28; L<='1'; A<='0';
                        when "00010" => n_state <= s28; L<='1'; A<='0';
                        when "00100" => n_state <= s39; L<='1'; A<='0';
                        when "01000" => n_state <= s28; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s28 =>
                     case X is
                        when "00000" => n_state <= s29; L<='1'; A<='0';
                        when "00001" => n_state <= s28; L<='1'; A<='0';
                        when "00010" => n_state <= s28; L<='1'; A<='0';
                        when "01000" => n_state <= s28; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s29 =>
                     case X is
                        when "00000" => n_state <= s29; L<='1'; A<='0';
                        when "00001" => n_state <= s30; L<='1'; A<='0';
                        when "00010" => n_state <= s30; L<='1'; A<='0';
                        when "00100" => n_state <= s41; L<='1'; A<='0';
                        when "01000" => n_state <= s30; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s30 =>
                     case X is
                        when "00000" => n_state <= s31; L<='1'; A<='0';
                        when "00001" => n_state <= s30; L<='1'; A<='0';
                        when "00010" => n_state <= s30; L<='1'; A<='0';
                        when "01000" => n_state <= s30; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s31 =>
                     case X is
                        when "00000" => n_state <= s31; L<='1'; A<='0';
                        when "00001" => n_state <= s32; L<='1'; A<='0';
                        when "00010" => n_state <= s32; L<='1'; A<='0';
                        when "00100" => n_state <= s43; L<='1'; A<='0';
                        when "01000" => n_state <= s32; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s32 =>
                     case X is
                        when "00000" => n_state <= s33; L<='1'; A<='0';
                        when "00001" => n_state <= s32; L<='1'; A<='0';
                        when "00010" => n_state <= s32; L<='1'; A<='0';
                        when "01000" => n_state <= s32; L<='1'; A<='0';
```

```
                        when others =>
                     end case;
                  when s33 =>
                     case X is
                        when "00000" => n_state <= s33; L<='1'; A<='0';
                        when "00001" => n_state <= s34; L<='1'; A<='0';
                        when "00010" => n_state <= s34; L<='1'; A<='0';
                        when "00100" => n_state <= s45; L<='1'; A<='0';
                        when "01000" => n_state <= s34; L<='1'; A<='0';
                        when "10000" => n_state <= s34; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s34 =>
                     case X is
                        when "00000" => n_state <= s35; L<='1'; A<='1';
                        when "00001" => n_state <= s34; L<='1'; A<='0';
                        when "00010" => n_state <= s34; L<='1'; A<='0';
                        when "01000" => n_state <= s34; L<='1'; A<='0';
                        when "10000" => n_state <= s34; L<='1'; A<='0';
                        when others =>
                     end case;

            --This state is the beginning of the Alarm states
                  when s35 =>
                     case X is
                        when "00000" => n_state <= s35; L<='1'; A<='1';
                        when "00001" => n_state <= s35; L<='1'; A<='1';
                        when "00010" => n_state <= s35; L<='1'; A<='1';
                        when "00100" => n_state <= s47; L<='1'; A<='1';
                        when "01000" => n_state <= s35; L<='1'; A<='1';
                        when "10000" => n_state <= s35; L<='1'; A<='1';
                        when others =>
                     end case;

            --This is the Armed X4 reset state
                  when s36 =>
                     case X is
                        when "00000" => n_state <= s12; L<='1'; A<='0';
                        when "00100" => n_state <= s36; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;

            --These are the states for Armed X2 resets
                  when s37 =>
                     case X is
                        when "00000" => n_state <= s38; L<='1'; A<='0';
                        when "00100" => n_state <= s37; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s38 =>
                     case X is
                        when "00000" => n_state <= s38; L<='1'; A<='0';
                        when "00001" => n_state <= s28; L<='1'; A<='0';
                        when "00010" => n_state <= s28; L<='1'; A<='0';
                        when "00100" => n_state <= s36; L<='1'; A<='0';
                        when "01000" => n_state <= s28; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s39 =>
                     case X is
                        when "00000" => n_state <= s40; L<='1'; A<='0';
                        when "00100" => n_state <= s39; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s40 =>
                     case X is
                        when "00000" => n_state <= s40; L<='1'; A<='0';
                        when "00001" => n_state <= s30; L<='1'; A<='0';
                        when "00010" => n_state <= s30; L<='1'; A<='0';
                        when "00100" => n_state <= s36; L<='1'; A<='0';
                        when "01000" => n_state <= s30; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s41 =>
                     case X is
                        when "00000" => n_state <= s42; L<='1'; A<='0';
                        when "00100" => n_state <= s41; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s42 =>
                     case X is
                        when "00000" => n_state <= s42; L<='1'; A<='0';
                        when "00001" => n_state <= s32; L<='1'; A<='0';
                        when "00010" => n_state <= s32; L<='1'; A<='0';
                        when "00100" => n_state <= s36; L<='1'; A<='0';
                        when "01000" => n_state <= s32; L<='1'; A<='0';
                        when "10000" => n_state <= s36; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s43 =>
                     case X is
                        when "00000" => n_state <= s44; L<='1'; A<='0';
                        when "00100" => n_state <= s43; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s44 =>
                     case X is
                        when "00000" => n_state <= s44; L<='1'; A<='0';
                        when "00001" => n_state <= s34; L<='1'; A<='0';
                        when "00010" => n_state <= s34; L<='1'; A<='0';
                        when "00100" => n_state <= s36; L<='1'; A<='0';
                        when "01000" => n_state <= s34; L<='1'; A<='0';
                        when "10000" => n_state <= s34; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s45 =>
                     case X is
                        when "00000" => n_state <= s46; L<='1'; A<='0';
                        when "00100" => n_state <= s45; L<='1'; A<='0';
                        when others =>
                     end case;
                  when s46 =>
                     case X is
                        when "00000" => n_state <= s46; L<='1'; A<='0';
                        when "00001" => n_state <= s34; L<='1'; A<='0';
                        when "00010" => n_state <= s34; L<='1'; A<='0';
                        when "00100" => n_state <= s36; L<='1'; A<='0';
                        when "01000" => n_state <= s34; L<='1'; A<='0';
                        when "10000" => n_state <= s34; L<='1'; A<='0';
                        when others =>
                     end case;

            --These are the Alarm unlocking states
                  when s47 =>
                     case X is
                        when "00000" => n_state <= s48; L<='1'; A<='1';
                        when "00100" => n_state <= s47; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s48 =>
                     case X is
                        when "00000" => n_state <= s48; L<='1'; A<='1';
                        when "00001" => n_state <= s49; L<='1'; A<='1';
                        when "00010" => n_state <= s58; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s58; L<='1'; A<='1';
                        when "10000" => n_state <= s58; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s49 =>
                     case X is
                        when "00000" => n_state <= s50; L<='1'; A<='1';
                        when "00001" => n_state <= s49; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s50 =>
                     case X is
                        when "00000" => n_state <= s50; L<='1'; A<='1';
                        when "00001" => n_state <= s58; L<='1'; A<='1';
                        when "00010" => n_state <= s58; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s51; L<='1'; A<='1';
                        when "10000" => n_state <= s58; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s51 =>
                     case X is
                        when "00000" => n_state <= s52; L<='1'; A<='1';
                        when "01000" => n_state <= s51; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s52 =>
                     case X is
                        when "00000" => n_state <= s52; L<='1'; A<='1';
                        when "00001" => n_state <= s58; L<='1'; A<='1';
                        when "00010" => n_state <= s58; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s53; L<='1'; A<='1';
```

```vhdl
                  when "10000" => n_state <= s58; L<='1'; A<='1';
                  when others =>
               end case;
            when s53 =>
               case X is
                  when "00000" => n_state <= s54; L<='1'; A<='1';
                  when "01000" => n_state <= s53; L<='1'; A<='1';
                  when others =>
               end case;
            when s54 =>
               case X is
                  when "00000" => n_state <= s54; L<='1'; A<='1';
                  when "00001" => n_state <= s58; L<='1'; A<='1';
                  when "00010" => n_state <= s55; L<='1'; A<='1';
                  when "00100" => n_state <= s58; L<='1'; A<='1';
                  when "01000" => n_state <= s58; L<='1'; A<='1';
                  when "10000" => n_state <= s58; L<='1'; A<='1';
                  when others =>
               end case;
            when s55 =>
               case X is
                  when "00000" => n_state <= s56; L<='1'; A<='1';
                  when "00010" => n_state <= s55; L<='1'; A<='1';
                  when others =>
               end case;
            when s56 =>
               case X is
                  when "00000" => n_state <= s56; L<='1'; A<='1';
                  when "00001" => n_state <= s58; L<='1'; A<='1';
                  when "00010" => n_state <= s58; L<='1'; A<='1';
                  when "00100" => n_state <= s58; L<='1'; A<='1';
                  when "01000" => n_state <= s58; L<='1'; A<='1';
                  when "10000" => n_state <= s57; L<='1'; A<='1';
                  when others =>
               end case;
            when s57 =>
               case X is
                  when "00000" => n_state <= s0; L<='0'; A<='0';
                  when "10000" => n_state <= s57; L<='1'; A<='1';
                  when others =>
               end case;

      --This is the Alarm reset state
            when s58 =>
               case X is
                  when "00000" => n_state <= s35; L<='1'; A<='1';
                  when "00001" => n_state <= s58; L<='1'; A<='1';
                  when "00010" => n_state <= s58; L<='1'; A<='1';
                  when "00100" => n_state <= s58; L<='1'; A<='1';
                  when "01000" => n_state <= s58; L<='1'; A<='1';
                  when "10000" => n_state <= s58; L<='1'; A<='1';
                  when others =>
               end case;

      --This is the Unarmed reset state and will lead to s0
            when s59 =>
               case X is
```

```vhdl
                      when "00000" => n_state <= s0; L<='0'; A<='0';
                      when "00001" => n_state <= s59; L<='0'; A<='0';
                      when "00010" => n_state <= s59; L<='0'; A<='0';
                      when "00100" => n_state <= s59; L<='0'; A<='0';
                      when "01000" => n_state <= s59; L<='0'; A<='0';
                      when "10000" => n_state <= s59; L<='0'; A<='0';
                      when others =>
                   end case;
                when others =>
             end case; --End of H = '1'

    --This is when the lock is not latched
          when '0' =>
             case p_state is

          -- Unlatched Unarmed states
                when s0 => n_state <= s0; L<='0'; A<='0';
                when s1 => n_state <= s0; L<='0'; A<='0';
                when s2 => n_state <= s0; L<='0'; A<='0';
                when s3 => n_state <= s0; L<='0'; A<='0';
                when s4 => n_state <= s0; L<='0'; A<='0';
                when s5 => n_state <= s0; L<='0'; A<='0';
                when s6 => n_state <= s0; L<='0'; A<='0';
                when s7 => n_state <= s0; L<='0'; A<='0';
                when s8 => n_state <= s0; L<='0'; A<='0';
                when s9 => n_state <= s0; L<='0'; A<='0';
                when s10 => n_state <= s0; L<='0'; A<='0';
                when s11 => n_state <= s0; L<='0'; A<='0';

       --These are the unlatched Armed states
                when s12 => n_state <= s35; L<='1'; A<='1';
                when s13 => n_state <= s35; L<='1'; A<='1';
                when s14 => n_state <= s35; L<='1'; A<='1';
                when s15 => n_state <= s35; L<='1'; A<='1';
                when s16 => n_state <= s35; L<='1'; A<='1';
                when s17 => n_state <= s35; L<='1'; A<='1';
                when s18 => n_state <= s35; L<='1'; A<='1';
                when s19 => n_state <= s35; L<='1'; A<='1';
                when s20 => n_state <= s35; L<='1'; A<='1';
                when s21 => n_state <= s35; L<='1'; A<='1';
                when s22 => n_state <= s35; L<='1'; A<='1';
                when s23 => n_state <= s35; L<='1'; A<='1';
                when s24 => n_state <= s35; L<='1'; A<='1';
                when s25 => n_state <= s35; L<='1'; A<='1';
                when s26 => n_state <= s35; L<='1'; A<='1';
                when s27 => n_state <= s35; L<='1'; A<='1';
                when s28 => n_state <= s35; L<='1'; A<='1';
                when s29 => n_state <= s35; L<='1'; A<='1';
                when s30 => n_state <= s35; L<='1'; A<='1';
                when s31 => n_state <= s35; L<='1'; A<='1';
                when s32 => n_state <= s35; L<='1'; A<='1';
                when s33 => n_state <= s35; L<='1'; A<='1';
                when s34 => n_state <= s35; L<='1'; A<='1';

          --S35 is the beginning of the Alarm state
                when s35 =>
                   case X is
```

```
                        when "00000" => n_state <= s35; L<='1'; A<='1';
                        when "00001" => n_state <= s35; L<='1'; A<='1';
                        when "00010" => n_state <= s35; L<='1'; A<='1';
                        when "00100" => n_state <= s47; L<='1'; A<='1';
                        when "01000" => n_state <= s35; L<='1'; A<='1';
                        when "10000" => n_state <= s35; L<='1'; A<='1';
                        when others =>
                     end case;

            --Unlatched Armed states continued
                  when s36 => n_state <= s35; L<='1'; A<='1';
                  when s37 => n_state <= s35; L<='1'; A<='1';
                  when s38 => n_state <= s35; L<='1'; A<='1';
                  when s39 => n_state <= s35; L<='1'; A<='1';
                  when s40 => n_state <= s35; L<='1'; A<='1';
                  when s41 => n_state <= s35; L<='1'; A<='1';
                  when s42 => n_state <= s35; L<='1'; A<='1';
                  when s43 => n_state <= s35; L<='1'; A<='1';
                  when s44 => n_state <= s35; L<='1'; A<='1';
                  when s45 => n_state <= s35; L<='1'; A<='1';
                  when s46 => n_state <= s35; L<='1'; A<='1';

            --Unlatched Alarm States
                  when s47 =>
                     case X is
                        when "00000" => n_state <= s48; L<='1'; A<='1';
                        when "00100" => n_state <= s47; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s48 =>
                     case X is
                        when "00000" => n_state <= s48; L<='1'; A<='1';
                        when "00001" => n_state <= s49; L<='1'; A<='1';
                        when "00010" => n_state <= s58; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s58; L<='1'; A<='1';
                        when "10000" => n_state <= s58; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s49 =>
                     case X is
                        when "00000" => n_state <= s50; L<='1'; A<='1';
                        when "00001" => n_state <= s49; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s50 =>
                     case X is
                        when "00000" => n_state <= s50; L<='1'; A<='1';
                        when "00001" => n_state <= s58; L<='1'; A<='1';
                        when "00010" => n_state <= s58; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s51; L<='1'; A<='1';
                        when "10000" => n_state <= s58; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s51 =>
                     case X is
```

```
                        when "00000" => n_state <= s52; L<='1'; A<='1';
                        when "01000" => n_state <= s51; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s52 =>
                     case X is
                        when "00000" => n_state <= s52; L<='1'; A<='1';
                        when "00001" => n_state <= s58; L<='1'; A<='1';
                        when "00010" => n_state <= s58; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s53; L<='1'; A<='1';
                        when "10000" => n_state <= s58; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s53 =>
                     case X is
                        when "00000" => n_state <= s54; L<='1'; A<='1';
                        when "01000" => n_state <= s53; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s54 =>
                     case X is
                        when "00000" => n_state <= s54; L<='1'; A<='1';
                        when "00001" => n_state <= s58; L<='1'; A<='1';
                        when "00010" => n_state <= s55; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s58; L<='1'; A<='1';
                        when "10000" => n_state <= s58; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s55 =>
                     case X is
                        when "00000" => n_state <= s56; L<='1'; A<='1';
                        when "00010" => n_state <= s55; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s56 =>
                     case X is
                        when "00000" => n_state <= s56; L<='1'; A<='1';
                        when "00001" => n_state <= s58; L<='1'; A<='1';
                        when "00010" => n_state <= s58; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s58; L<='1'; A<='1';
                        when "10000" => n_state <= s57; L<='1'; A<='1';
                        when others =>
                     end case;
                  when s57 =>
                     case X is
                        when "00000" => n_state <= s0; L<='0'; A<='0';
                        when "10000" => n_state <= s57; L<='1'; A<='1';
                        when others =>
                     end case;

            --Unlatched Alarm Reset State
                  when s58 =>
                     case X is
                        when "00000" => n_state <= s35; L<='1'; A<='1';
                        when "00001" => n_state <= s58; L<='1'; A<='1';
                        when "00010" => n_state <= s58; L<='1'; A<='1';
                        when "00100" => n_state <= s58; L<='1'; A<='1';
                        when "01000" => n_state <= s58; L<='1'; A<='1';
                        when "10000" => n_state <= s58; L<='1'; A<='1';
                        when others =>
                     end case;

            --Unlatched Unarmed Reset state
                  when s59 => n_state <= s0; L<='0'; A<='0';
                  when others =>
               end case;--End of H = '0'

            when others =>
         end case; --End of H cases

      end process;

   Process(p_state)
   begin
      if (p_state = s0) then
         r_state<= '1'; l_state<= '0'; a_state <= '0';
      elsif(p_state = s12) then
         r_state<= '0'; l_state<= '1'; a_state <= '0';
      elsif(p_state = s35) then
         r_state<= '0'; l_state<= '0'; a_state <= '1';
      Else
          r_state<= '0'; l_state<= '0'; a_state <= '0';
      end if;
   end process;


   end Behavioral;
```