```vhdl
-------------------------------------------------------------------------
-- This component is the central processing unit of the vending machine.
-- This includes the following operations:
--      When one of the coin inputs are asserted, sum is updated
--      When one of the price_product inputs are asserted, price is update
--      If a buy is attempted:
--          sum will be deducted from price if sum >= price
--          alarm signal will be asserted if sum < price
-------------------------------------------------------------------------

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity processing_unit is
    port(clock      : in  std_logic;    -- Clock signal in 762Hz
         clk_3      : in  std_logic;    -- Clock signal in 3Hz
         buy        : in  std_logic;
         coin1      : in  std_logic;
         coin2      : in  std_logic;
         coin5      : in  std_logic;
         price_cola : in  std_logic;
         price_hash : in  std_logic;
         price_aqua : in  std_logic;
         Reset      : in  std_logic;
         sum_out    : out unsigned(5 downto 0);
         price_out  : out unsigned(5 downto 0);
         alarm_out  : out std_logic;
         cola_out   : out std_logic;
         hash_out   : out std_logic;
         aqua_out   : out std_logic);
end processing_unit;

architecture Behavioral of processing_unit is
    signal sum, price                     : unsigned(5 downto 0);
    signal alarm_count, alarm_count_next : unsigned(10 downto 0);
    signal alarm, cola, hash, aqua       : std_logic;
    signal cola_count, cola_count_next   : unsigned(10 downto 0);
    signal hash_count, hash_count_next   : unsigned(10 downto 0);
    signal aqua_count, aqua_count_next   : unsigned(10 downto 0);

begin
    alarm_count_next <= alarm_count + 1;
    cola_count_next  <= cola_count + 1;
    hash_count_next  <= hash_count + 1;
    aqua_count_next  <= aqua_count + 1;

    ---------------------------------------------------------------------
    -- This process sets the price of the 3 products. When one of the
    -- price_'product' signals are asserted, the 'product' signal is set
    -- to '1', which will cause 'product'_count to keep adding + 1 on
    -- every clock, until its MSB = '1' and the 'product' signal will be
    -- set back to '0'. For both the price_'product' and the 'product'
    -- signals, cola overrules hash, which overrules aqua.
    ----------------------------------------------------------------------

    process(price_cola, price_hash, price_aqua, clock)
    begin
        if rising_edge(clock) then
            if price_cola = '1' then
                cola  <= '1';
                price <= "010010";
            elsif price_hash = '1' then
                hash  <= '1';
                price <= "110111";
            elsif price_aqua = '1' then
                aqua  <= '1';
                price <= "001100";
            elsif cola = '1' then
                cola_count <= cola_count_next;
                if cola_count(10) = '1' then
                    cola       <= '0';
                    cola_count <= "00000000000";
                end if;
            elsif hash = '1' then
                hash_count <= hash_count_next;
                if hash_count(10) = '1' then
                    hash       <= '0';
                    hash_count <= "00000000000";
                end if;
            elsif aqua = '1' then
                aqua_count <= aqua_count_next;
                if aqua_count(10) = '1' then
                    aqua       <= '0';
                    aqua_count <= "00000000000";
                end if;
            end if;
        end if;
        cola_out <= cola;
        hash_out <= hash;
        aqua_out <= aqua;
    end process;

    ------------------------------------------------------------------
    -- This process adds the coin value to sum when a coin input is
    -- asserted. If buy is asserted sum will be deducted from price,
    -- if sum >= price, else alarm will be set to '1', which causes
    -- alarm_count to be increased by one on every clock until MSB
    -- of alarm_count equals '1' which resets alarm to '0'.
    ------------------------------------------------------------------

    process(coin1, coin2, coin5, buy, clock)
    begin
        if rising_edge(clock) then
            if Reset = '1' then
                sum <= "000000";
            elsif coin1 = '1' then
                sum <= sum + 1;
            elsif coin2 = '1' then
                sum <= sum + 2;
            elsif coin5 = '1' then
                sum <= sum + 5;
            elsif alarm = '1' then
                alarm_count <= alarm_count_next;
                if alarm_count(10) = '1' then
                    alarm       <= '0';
                    alarm_count <= "00000000000";
                end if;
            elsif buy = '1' then
                if sum >= price then
                    sum <= sum - price;
                elsif sum < price then
                    alarm <= '1';
                end if;
            end if;
            sum_out   <= sum(5 downto 0);
            price_out <= price(5 downto 0);
        end if;
        alarm_out <= alarm;
    end process;

end Behavioral;
```