

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

```
entity PromLoader_FLEX is
port
```

```
(
  clock      : in      std_logic;
  nStatus    : in      std_logic;
  o          : in      std_logic_vector(7 downto 0);
  restart    : in      std_logic;
  Conf_Done  : in      std_logic;

  Data0      : out     std_logic;
  Dclk       : out     std_logic;
  nConfig    : buffer  std_logic;
  A          : out     std_logic_vector(15 downto 0);
  CEn        : out     std_logic
);
```

```
end;
```

```
architecture rtl of PromLoader_FLEX is
```

```
constant start          :std_logic_vector(2 downto 0) := "000";
constant wait_nCfg_2us  :std_logic_vector(2 downto 0) := "100";
constant status         :std_logic_vector(2 downto 0) := "001";
constant wait_5us       :std_logic_vector(2 downto 0) := "101";
constant config         :std_logic_vector(2 downto 0) := "011";
constant init           :std_logic_vector(2 downto 0) := "111";
```

```
signal pp:std_logic_vector(2 downto 0);
signal count:std_logic_vector(2 downto 0);
signal data0_int, dclk_int:std_logic;
signal inc:std_logic_vector(15 downto 0);
signal div:std_logic_vector(2 downto 0);
signal waitd:std_logic_vector(5 downto 0);
```

```
begin
```

```
  PROCESS (clock,restart)
```

```
  BEGIN
```

```
    if restart = '0' then
```

```
      div <= (others => '0');
```

```
    else
```

```
      IF (clock'EVENT AND clock = '1') THEN
```

```
        div <= div + 1;
```

```
        end if;  
    end if;  
END PROCESS;
```

```
process(clock,restart)  
begin
```

```
    if restart = '0' then
```

```
        pp<=start;  
        count <= (others => '0');  
        inc  <= (others => '0');  
        waitd <= (others => '0');
```

```
    else
```

```
        if clock'event and clock='1' then
```

```
            if (div = 7) then
```

```
                case pp is
```

```
                    when start =>
```

```
                        count <= (others => '0');  
                        inc  <= (others => '0');  
                        waitd <= (others => '0');  
                        pp <= wait_nCfg_2us;
```

```
                    when wait_nCfg_2us =>
```

```
                        count <= (others => '0');  
                        inc  <= (others => '0');  
                        waitd <= waitd + 1;  
                        if waitd = 20 then  
                            pp <= status;  
                        end if;
```

```
                    when status =>
```

```
                        count <= (others => '0');  
                        inc  <= (others => '0');  
                        waitd <= (others => '0');  
                        pp <= wait_5us;
```

```
                    when wait_5us =>
```

```
                        count <= (others => '0');  
                        inc  <= (others => '0');  
                        waitd <= waitd + 1;  
                        if waitd = 50 then  
                            pp <= config;  
                        end if;
```

```
                    when config =>
```

```
                        count <= count + 1;  
                        if Conf_Done='1' then
```

```

        waitd <= waitd + 1;
    end if;
    if count=7 then
        inc <= inc + 1;
    end if;
    if waitd = 60 then
        pp<= init;
    end if;

    when init =>
        count <= (others => '0');
        inc <= (others => '0');
        waitd <= (others => '0');
        if nStatus = '0' then
            pp <= start;
        else
            pp <= init;
        end if;

    when others =>
        pp <= start;
    end case;
else
    pp <= pp;
    inc <= inc;
    count <= count;
end if;
end if;
end process;

dclk_int <= div(2) when pp=config else '0';

process(count,o,pp)
begin
    if pp=config then
        case count is
            when "000" => data0_int <= o(0);
            when "001" => data0_int <= o(1);
            when "010" => data0_int <= o(2);
            when "011" => data0_int <= o(3);
            when "100" => data0_int <= o(4);
            when "101" => data0_int <= o(5);
            when "110" => data0_int <= o(6);
            when "111" => data0_int <= o(7);
            when others => null;
        end case;
    end if;
end process;

```

```
                end case;
            else
                data0_int <= '0';
            end if;
        end process;

        nConfig <= pp(0);
        CEn <= not nconfig;

        Dclk <= '0' when pp(1)='0' else dclk_int;
        Data0 <= '0' when pp(1)='0' else data0_int;
        A <= inc;

    end;
```