TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

목차

Verilog to VHDL Code Converting

1. PWM

Verilog

```
reg [31:0] counter = 0;

//simple counter
always @(posedge S_AXI_ACLK) begin
    if(counter < PWM_COUNTER_MAX-1)
        counter <= counter + 1;
    else
        counter <= 0;
end
```

VHDL

```
process (S_AXI_ACLK) is
begin
    if(rising_edge(S_AXI_ACLK)) then
        if( to_integer(unsigned(counter)) < PWM_COUNTER_MAX - 1) then  -- x"001e847F"  = 2000000 - 1
            counter <= std_logic_vector(unsigned(counter) + 1);
        else
            counter <= (others => '0');
        end if;
    end if;
end process;

process (slv_reg0, counter) is
begin
    if( slv_reg0 < counter) then -- slv_reg00에 num 값이 들어감.
        PWM0 <= '0';
    else
        PWM0 <= '1';
    end if;
```

1. PWM

Verilog

```
always @(posedge S_AXI_ACLK) begin

if(pwm_sig == 1 && past_pwm_sig ==0) begin
   rising_edge_flag = rising_edge_flag + 4'd1;

   if(rising_edge_flag == 4'd1) begin
         cap1<=counter;
   end

   else if(rising_edge_flag == 4'd2) begin
         cap3<=counter;


         duty_e <= cap2 - cap1;
         duty <= duty_e;

         if(cap3 > cap1) begin

         period_e <= cap3 - cap1;
         end
         else if(cap1 > cap3) begin

         period_e <= cap1 - cap3;

         end

         period <= period_e;


         counter <= 32'd0;
       rising_edge_flag <= 4'd0;
   end

end

else if(pwm_sig == 0 && past_pwm_sig ==1) begin
         cap2<=counter;
end
   past_pwm_sig <= pwm_sig;
   counter <= counter + 32'd1;

end
```

VHDL

```
process(S_AXI_ACLK) is
     begin
       if (rising_edge (S_AXI_ACLK)) then

       past_pwm_sig <= pwm_sig;
       counter <= counter + 1;

         if(pwm_sig = '1' and past_pwm_sig = '0' ) then   --rising moment
             rising_edge_flag <= rising_edge_flag + 1;
             if( rising_edge_flag = 1) then
                 cap1 <= counter;
             elsif( rising_edge_flag = 2) then
                 cap3 <= counter;

                 if( cap3 > cap1) then
                    period <= std_logic_vector( cap3 - cap1);

                 elsif( cap1 > cap3) then
                    period <= std_logic_vector( cap1 - cap3);

                 end if;

                 duty <= std_logic_vector( cap2 - cap1);
                 counter <= (others => '0');
                 rising_edge_flag <= (others => '0');
             end if;

         elsif (pwm_sig = '0' and past_pwm_sig = '1' ) then   -- falling moment
                 cap2 <= counter;
         end if;

       end if;
     end process;
```

```
---- Number of Slave Registers 4
signal slv_reg0 :std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0);
signal slv_reg1 :std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0);
signal slv_reg2 :std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0);
signal slv_reg3 :std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0);
signal slv_reg_rden : std_logic;
signal slv_reg_wren : std_logic;
signal reg_data_out :std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0);
signal byte_index   : integer;
signal aw_en    : std_logic;

signal duty :std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0) := (others => '0');
signal period :std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0) := (others => '0')

signal cap1 :unsigned(C_S_AXI_DATA_WIDTH-1 downto 0) := (others => '0');
signal cap2 :unsigned(C_S_AXI_DATA_WIDTH-1 downto 0) := (others => '0');
signal cap3 :unsigned(C_S_AXI_DATA_WIDTH-1 downto 0) := (others => '0');
signal counter :unsigned(C_S_AXI_DATA_WIDTH-1 downto 0) :=(others => '0');
signal rising_edge_flag : unsigned(3 downto 0) := (others => '0');
signal past_pwm_sig : std_logic := '0';
```

1. PWM + ECAP SDK CODE

```c
#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"
#include "xparameters.h"
#include "xil_io.h"

int main()
{

    int i;
    int num =0;

    while(1){
        if(num > 1000000)
            num =0;
        else
            num += 5000;

        Xil_Out32(XPAR_MY_PWM_CORE_0_S00_AXI_BASEADDR, num);
        xil_printf("period : %d\r\n",Xil_In32(XPAR_MY_ECAP_CORE_0_S00_AXI_BASEADDR));
        xil_printf("duty : %d\r\n",Xil_In32(XPAR_MY_ECAP_CORE_0_S00_AXI_BASEADDR +4));
        xil_printf("ecap1 : %d\r\n",Xil_In32(XPAR_MY_ECAP_CORE_0_S00_AXI_BASEADDR + 8));
        xil_printf("ecap3 : %d\r\n",Xil_In32(XPAR_MY_ECAP_CORE_0_S00_AXI_BASEADDR + 12));

        for(i=0;i<10000000;i++);
    }

    return 0;
}
```

slv_reg 레지스터에서 값만 읽어오면 됨.