Xilinx Zynq FPGA, TI DSP, MCU 기반의 프로그래밍 및 회로 설계 전문가 과정

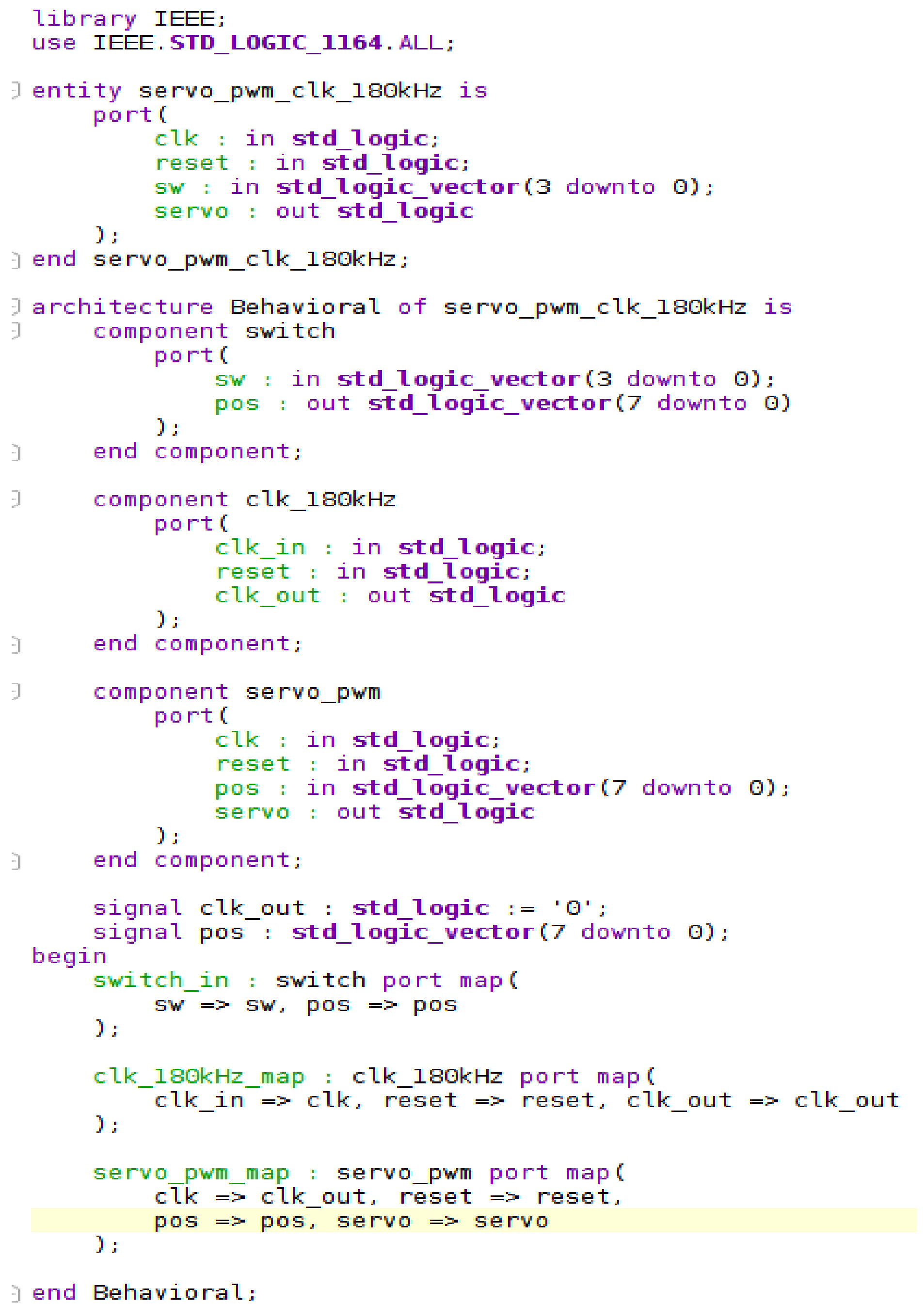
강사 – Innova Lee(이상훈)

gcccompil3r@gmail.com

학생 – hoseong Lee(이호성)

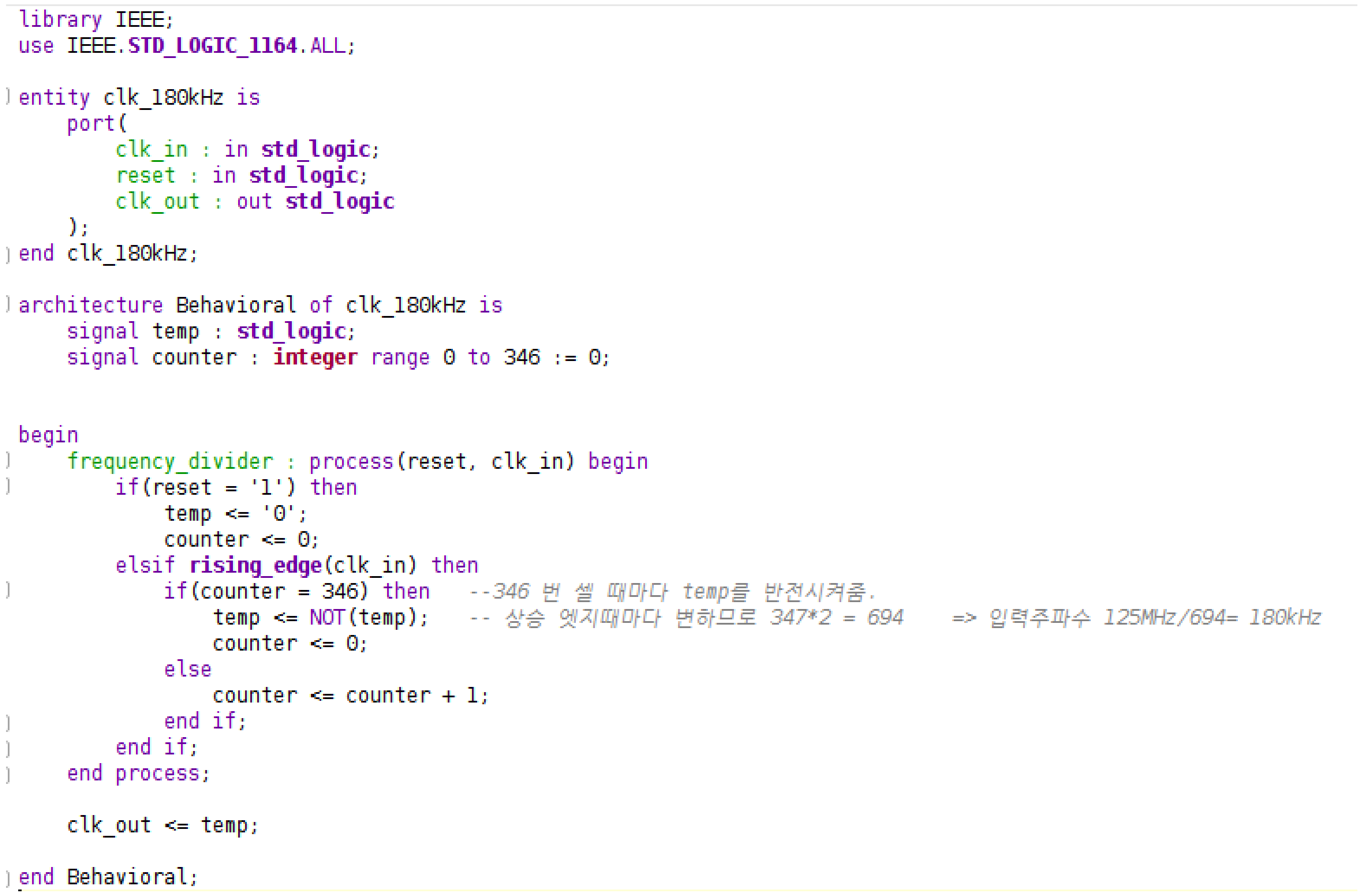
hslee00001@naver.com

-TOP module



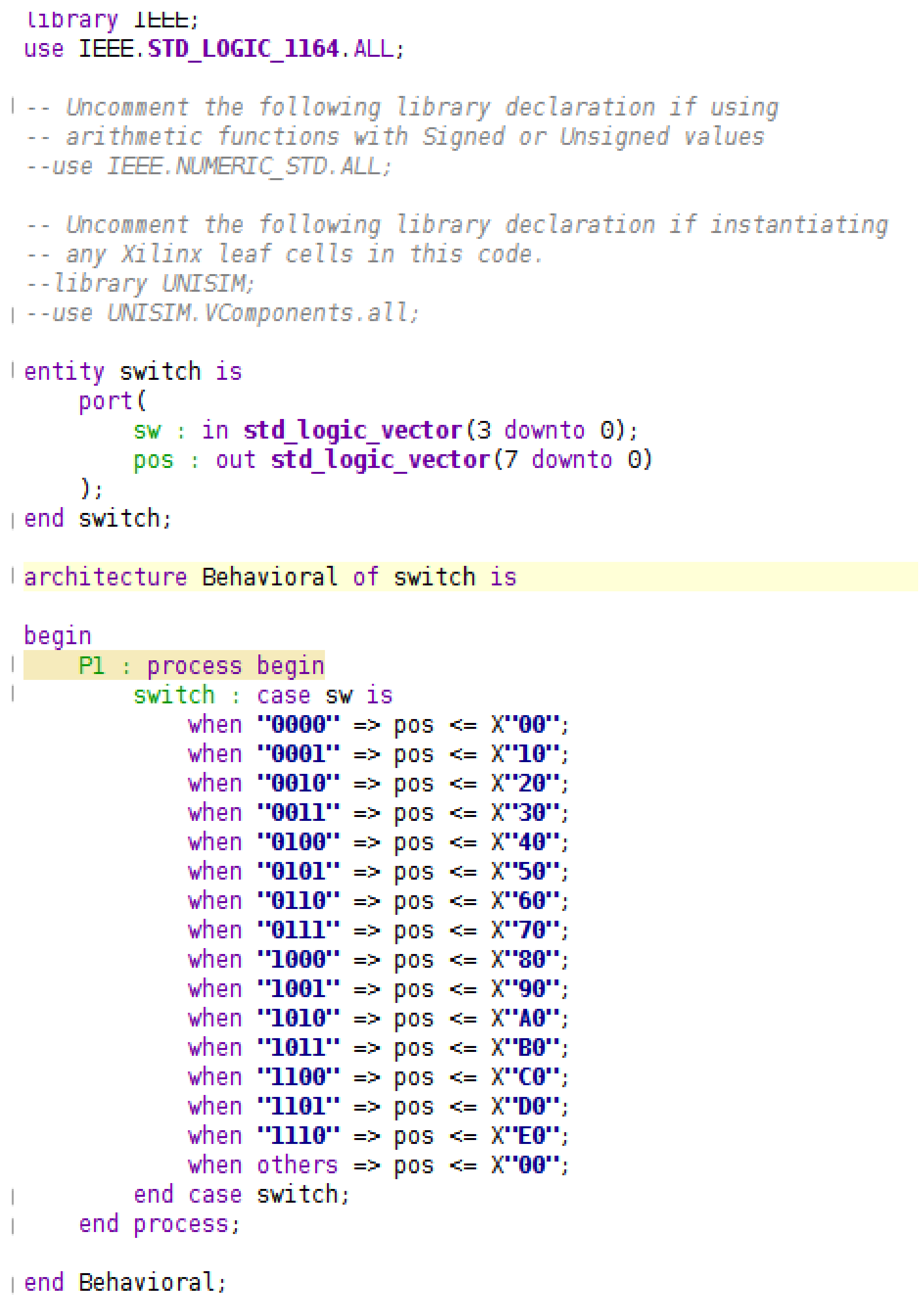
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity servo_pwm_clk_180kHz is
    port(
        clk : in std_logic;
        reset : in std_logic;
        sw : in std_logic_vector(3 downto 0);
        servo : out std_logic
    );
end servo_pwm_clk_180kHz;

architecture Behavioral of servo_pwm_clk_180kHz is
    component switch
        port(
            sw : in std_logic_vector(3 downto 0);
            pos : out std_logic_vector(7 downto 0)
        );
    end component;

    component clk_180kHz
        port(
            clk_in : in std_logic;
            reset : in std_logic;
            clk_out : out std_logic
        );
    end component;

    component servo_pwm
        port(
            clk : in std_logic;
            reset : in std_logic;
            pos : in std_logic_vector(7 downto 0);
            servo : out std_logic
        );
    end component;

    signal clk_out : std_logic := '0';
    signal pos : std_logic_vector(7 downto 0);
begin
    switch_in : switch port map(
        sw => sw, pos => pos
    );

    clk_180kHz_map : clk_180kHz port map(
        clk_in => clk, reset => reset, clk_out => clk_out
    );

    servo_pwm_map : servo_pwm port map(
        clk => clk_out, reset => reset,
        pos => pos, servo => servo
    );

end Behavioral;
```

- 하위 clk_180kHz module



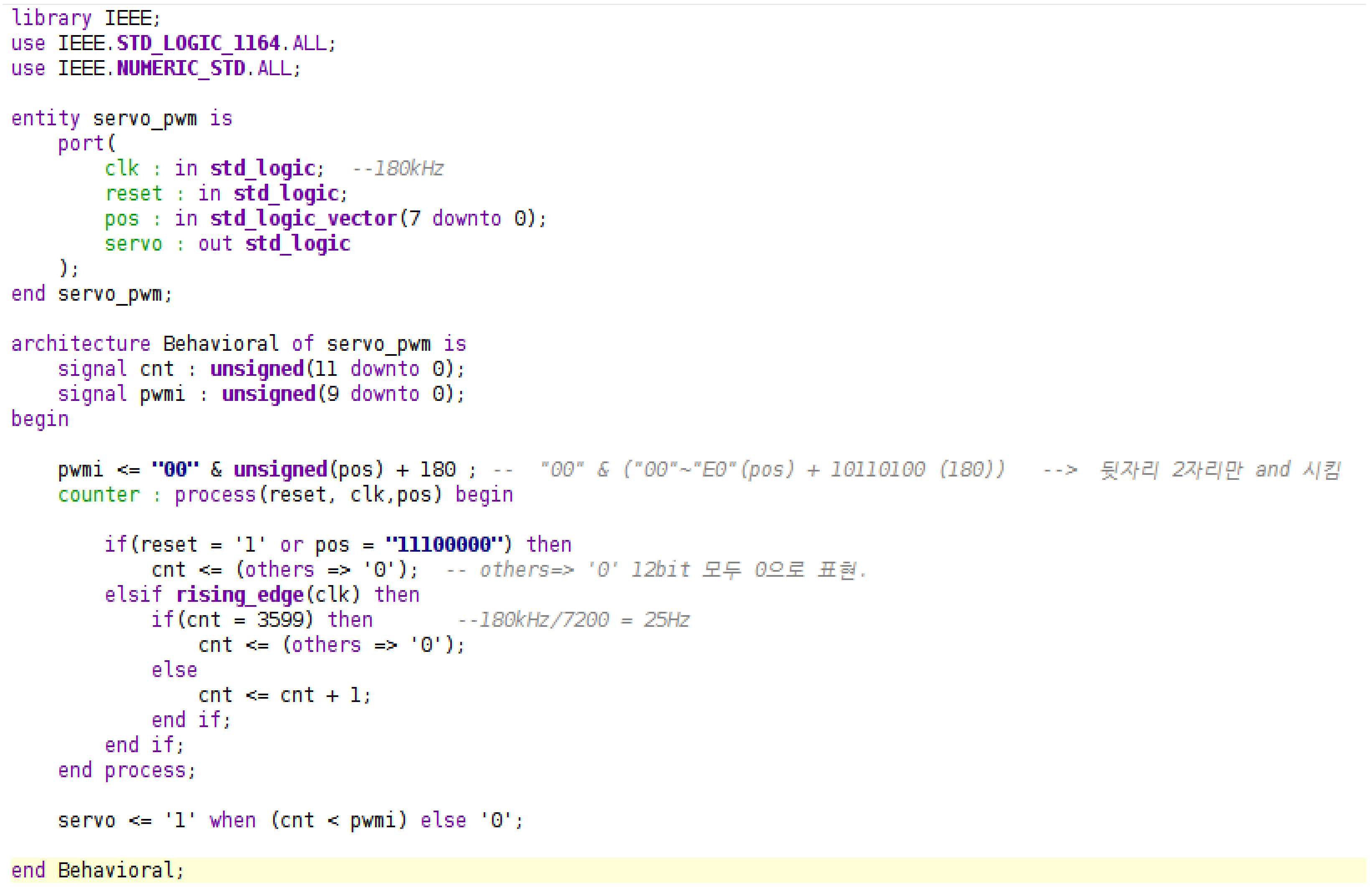
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity clk_180kHz is
    port(
        clk_in : in std_logic;
        reset : in std_logic;
        clk_out : out std_logic
    );
end clk_180kHz;

architecture Behavioral of clk_180kHz is
    signal temp : std_logic;
    signal counter : integer range 0 to 346 := 0;


begin
    frequency_divider : process(reset, clk_in) begin
        if(reset = '1') then
            temp <= '0';
            counter <= 0;
        elsif rising_edge(clk_in) then
            if(counter = 346) then   --346 번 셀 때마다 temp를 반전시켜줌.
                temp <= NOT(temp);   -- 상승 엣지때마다 변하므로 347*2 = 694    => 입력주파수 125MHz/694= 180kHz
                counter <= 0;
            else
                counter <= counter + 1;
            end if;
        end if;
    end process;

    clk_out <= temp;

end Behavioral;
```

- swich



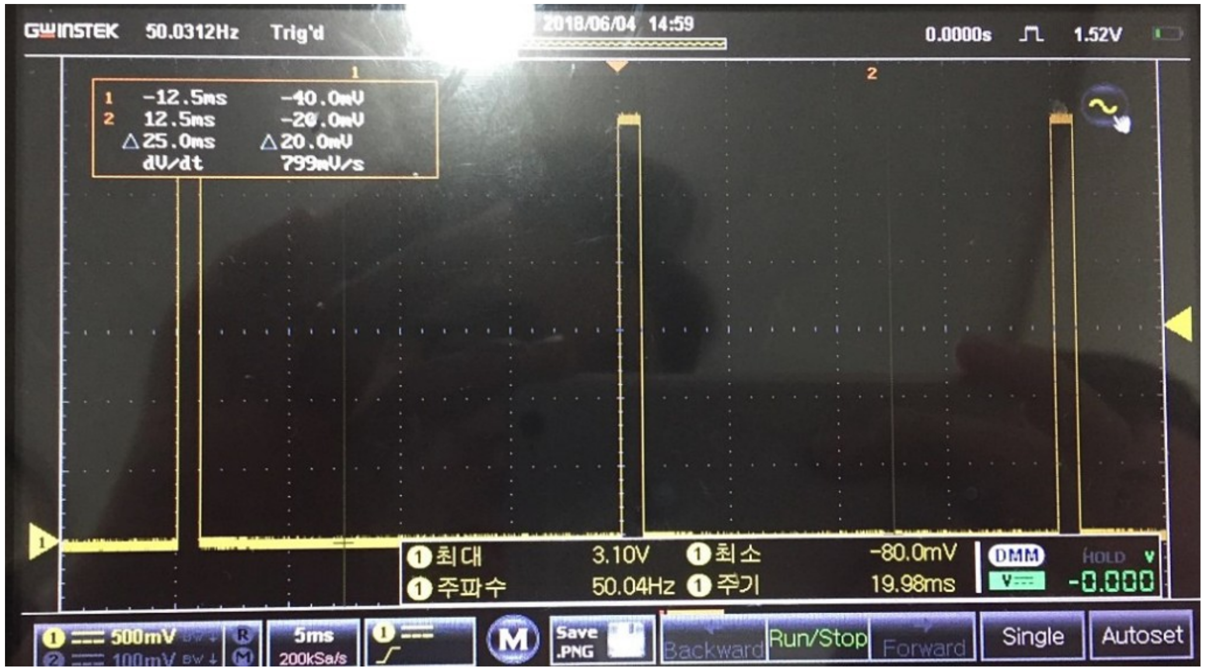
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- 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 leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity switch is
    port(
        sw : in std_logic_vector(3 downto 0);
        pos : out std_logic_vector(7 downto 0)
    );
end switch;

architecture Behavioral of switch is

begin
    P1 : process begin
        switch : case sw is
            when "0000" => pos <= X"00";
            when "0001" => pos <= X"10";
            when "0010" => pos <= X"20";
            when "0011" => pos <= X"30";
            when "0100" => pos <= X"40";
            when "0101" => pos <= X"50";
            when "0110" => pos <= X"60";
            when "0111" => pos <= X"70";
            when "1000" => pos <= X"80";
            when "1001" => pos <= X"90";
            when "1010" => pos <= X"A0";
            when "1011" => pos <= X"B0";
            when "1100" => pos <= X"C0";
            when "1101" => pos <= X"D0";
            when "1110" => pos <= X"E0";
            when others => pos <= X"00";
        end case switch;
    end process;

end Behavioral;
```

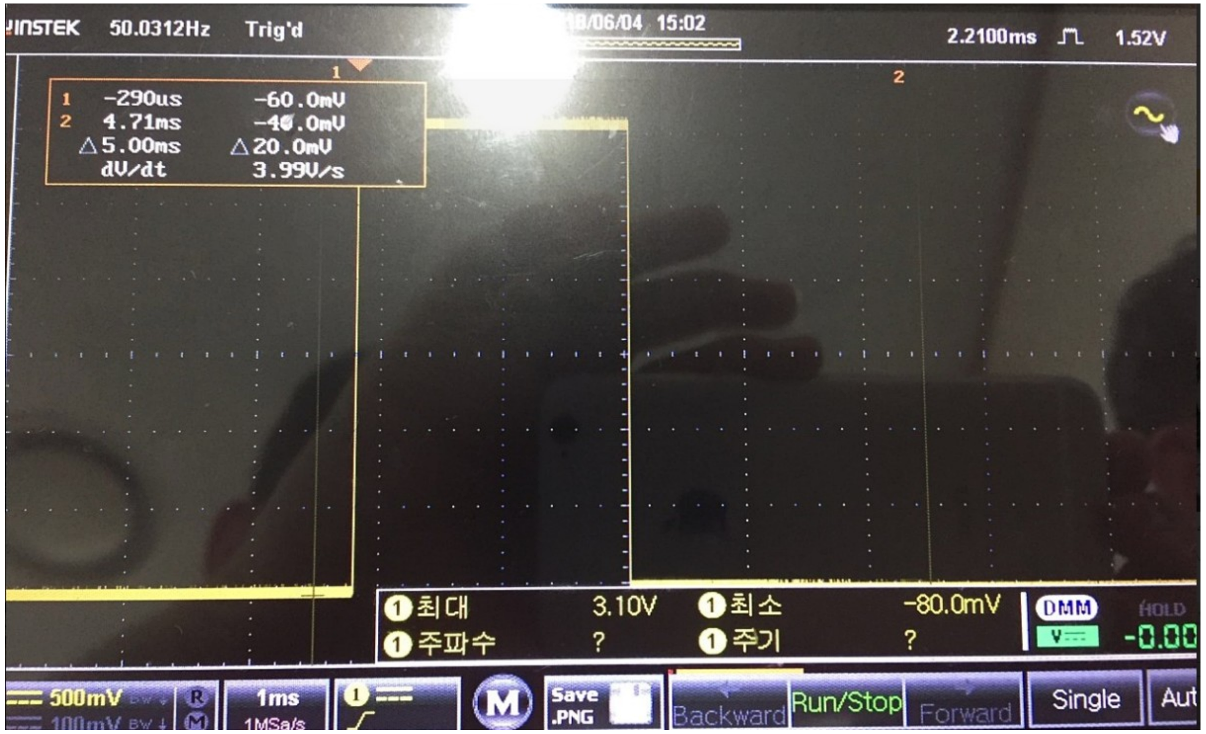
-servo_pwm



```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity servo_pwm is
    port(
        clk : in std_logic;  --180kHz
        reset : in std_logic;
        pos : in std_logic_vector(7 downto 0);
        servo : out std_logic
    );
end servo_pwm;

architecture Behavioral of servo_pwm is
    signal cnt : unsigned(11 downto 0);
    signal pwmi : unsigned(9 downto 0);
begin

    pwmi <= "00" & unsigned(pos) + 180 ; --  "00" & ("00"~"E0"(pos) + 10110100 (180))   -->  뒷자리 2자리만 and 시킴
    counter : process(reset, clk,pos) begin

        if(reset = '1' or pos = "11100000") then
            cnt <= (others => '0');  -- others=> '0' 12bit 모두 0으로 표현.
        elsif rising_edge(clk) then
            if(cnt = 3599) then        --180kHz/7200 = 25Hz
                cnt <= (others => '0');
            else
                cnt <= cnt + 1;
            end if;
        end if;
    end process;

    servo <= '1' when (cnt < pwmi) else '0';

end Behavioral;
```

-결과

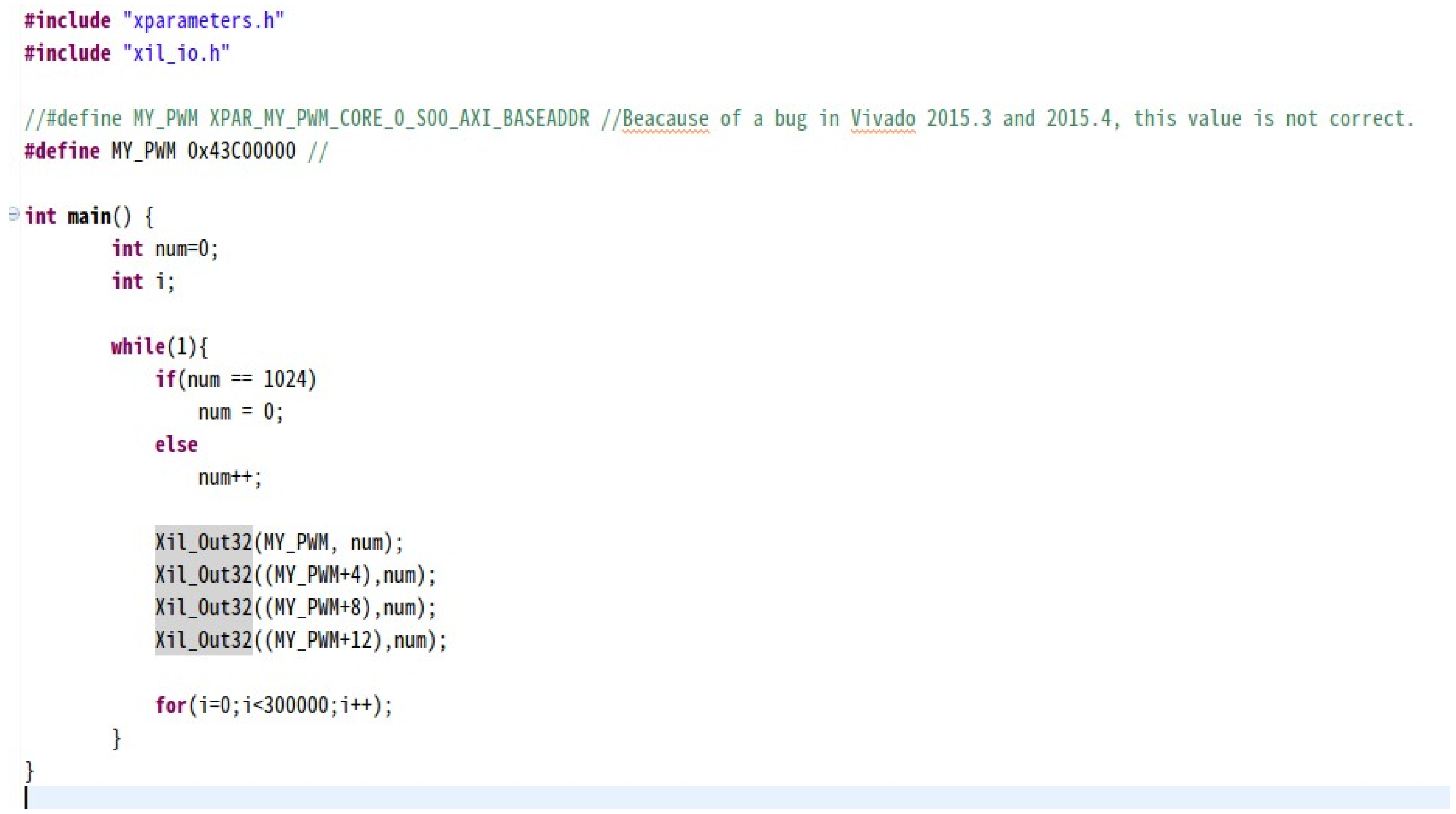

GW INSTEK 50.0312Hz Trig'd
2018/06/04 14:59
0.0000s 1.52V
1 -12.5ms -40.0mV
2 12.5ms -20.0mV
△25.0ms △20.0mV
dV/dt 799mV/s
① 최대 3.10V ① 최소 -80.0mV
① 주파수 50.04Hz ① 주기 19.98ms
DMM HOLD -0.000
① 500mV
② 100mV
5ms 200kSa/s
Save .PNG
Backward Run/Stop Forward Single Autoset


INSTEK 50.0312Hz Trig'd
18/06/04 15:02
2.2100ms 1.52V
1 -290us -60.0mV
2 4.71ms -40.0mV
△5.00ms △20.0mV
dV/dt 3.99V/s
① 최대 3.10V ① 최소 -80.0mV
① 주파수 ? ① 주기 ?
DMM HOLD -0.00
500mV
100mV
1ms 1MSa/s
Save .PNG
Backward Run/Stop Forward Single

-Coustom IP 생성 토튜리얼

- Zybo Getting Started Guide 완료: SDK 설치 https://reference.digilentinc.com/learn/programmable-logic/tutorials/zybo-getting-started-with-zynq/start?redirect=1

Tutorial 이 데모에서 Zynq Processor 의 Processing System 을 사용하여 보드상의 LED 를 조작하기 위한 기본 PWM 제어기를 구축하는 방법을 보여준다. IP Graphic Interface 에서 PWM 윈도우 크기를 변경한 다음 Processor 용으로 작성된 Duty Cycle 을 제어할 수 있다.



```
#include "xparameters.h"
#include "xil_io.h"

//#define MY_PWM XPAR_MY_PWM_CORE_0_S00_AXI_BASEADDR //Beacause of a bug in Vivado 2015.3 and 2015.4, this value is not correct.
#define MY_PWM 0x43C00000 //

int main() {
        int num=0;
        int i;

        while(1){
            if(num == 1024)
                num = 0;
            else
                num++;

            Xil_Out32(MY_PWM, num);
            Xil_Out32((MY_PWM+4),num);
            Xil_Out32((MY_PWM+8),num);
            Xil_Out32((MY_PWM+12),num);

            for(i=0;i<300000;i++);
        }
}
```