นายวิชัย ศรีสุรักษ์
B3706985

# Week05_Mojo V3 - Mojo V3 Start, Verilog and VHDL

## 1. Install

These tutorials walk you through setting up the required software.

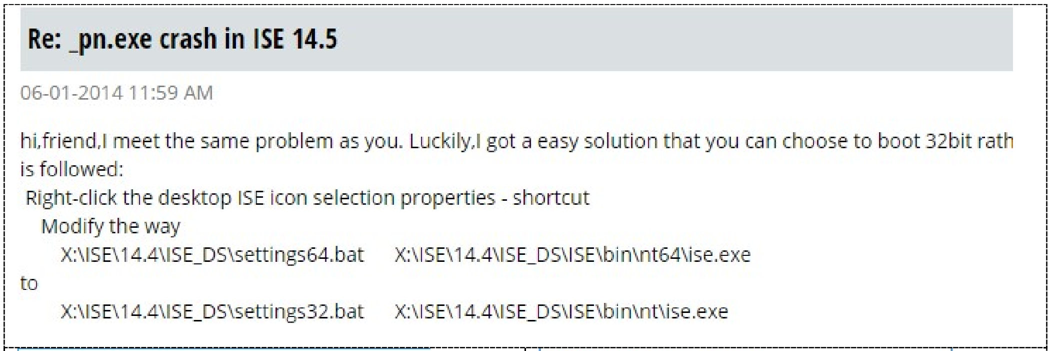
1. Xilinx's ISE
2. Mojo Loader

ISE is the tool provided by Xilinx, the guys who make the FPGA on the Mojo. It is responsible for building your projects.

The Mojo Loader is a simple tool we provide for loading the .bin files generated from ISE onto your Mojo.

### *1.1 Install Xilinx's ISE 14.7*

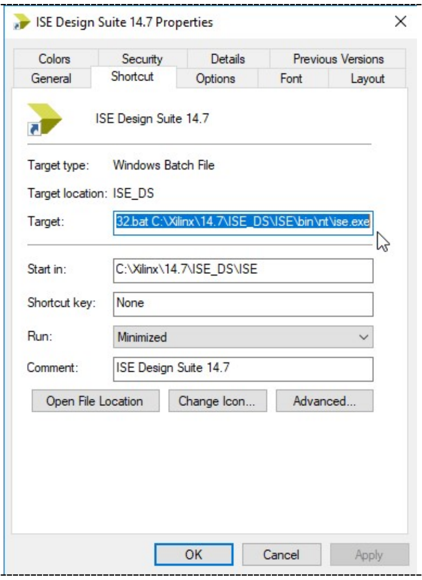
1. Installs W14.7
   "D:_Install_Digital_DigitalLab SUT\Xilinx_ISE_DS_Win_14.7_1015_1\bin\nt64 xsetup.exe"
   > Select WebPack Only
2. Run



**Re: _pn.exe crash in ISE 14.5**

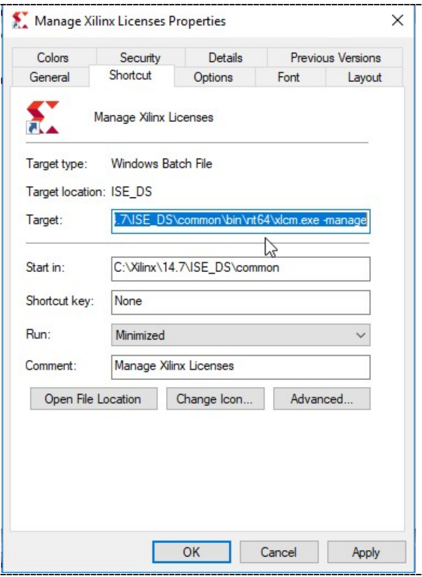
06-01-2014 11:59 AM

hi,friend,I meet the same problem as you. Luckily,I got a easy solution that you can choose to boot 32bit rath is followed:
Right-click the desktop ISE icon selection properties - shortcut
Modify the way
X:\ISE\14.4\ISE_DS\settings64.bat X:\ISE\14.4\ISE_DS\ISE\bin\nt64\ise.exe
to
X:\ISE\14.4\ISE_DS\settings32.bat X:\ISE\14.4\ISE_DS\ISE\bin\nt\ise.exe



ISE Design Suite 14.7 Properties

ISE Design Suite 14.7
Target type: Windows Batch File
Target location: ISE_DS
Target: 32.bat C:\Xilinx\14.7\ISE_DS\ISE\bin\nt\ise.exe
Start in: C:\Xilinx\14.7\ISE_DS\ISE
Shortcut key: None
Run: Minimized
Comment: ISE Design Suite 14.7



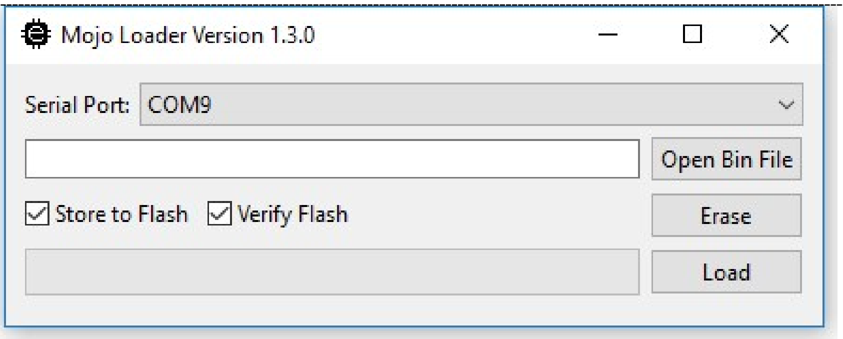
Manage Xilinx Licenses Properties

Manage Xilinx Licenses
Target type: Windows Batch File
Target location: ISE_DS
Target: .7\ISE_DS\common\bin\nt64\xlcm.exe -manage
Start in: C:\Xilinx\14.7\ISE_DS\common
Shortcut key: None
Run: Minimized
Comment: Manage Xilinx Licenses

3. Window → Xilinx Manage License → license at
   "D:_Install_Digital_DigitalLab SUT\Xilinx_ISE_DS_Win_14.7_1015_1\Xilinx.lic"

### *1.2 Install Mojo Loader*

1. Download http://cdn.embeddedmicro.com/mojo-loader/mojo-loader-1.3.0.exe
2. Test Run



Mojo Loader Version 1.3.0

Serial Port: COM9

☑ Store to Flash ☑ Verify Flash

## 2. My First Project

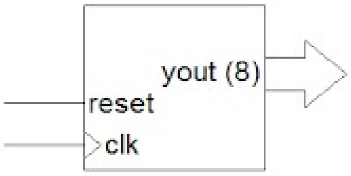
**Lab01 - สร้างไฟวิ่ง 8 ดวงออกทาง LED ของ Mojo V3 Board**

< Ref https://sites.google.com/site/eplearn/vhdl_fpga/lab/fpga_lab1 >

### วัตถุประสงค์

- เพื่อศึกษาการใช้งาน ISE webpack 14.7
- เพื่อศึกษาการใช้งาน บอร์ด MOJO V3

### ระบบ



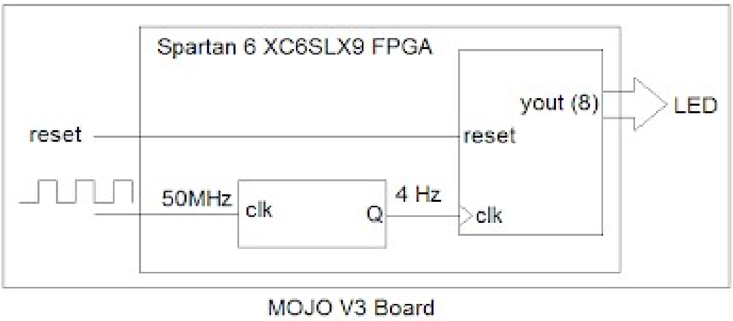
รูปที่ 1 ระบบวงจรไฟวิ่ง

### การทำงาน

| reset | การทำงาน |
|---|---|
| 0 | yout = "00000001" |
| 1 | ไฟวิ่งจากขวามือไปซ้ายมือ ( MSB <- LSB ) |

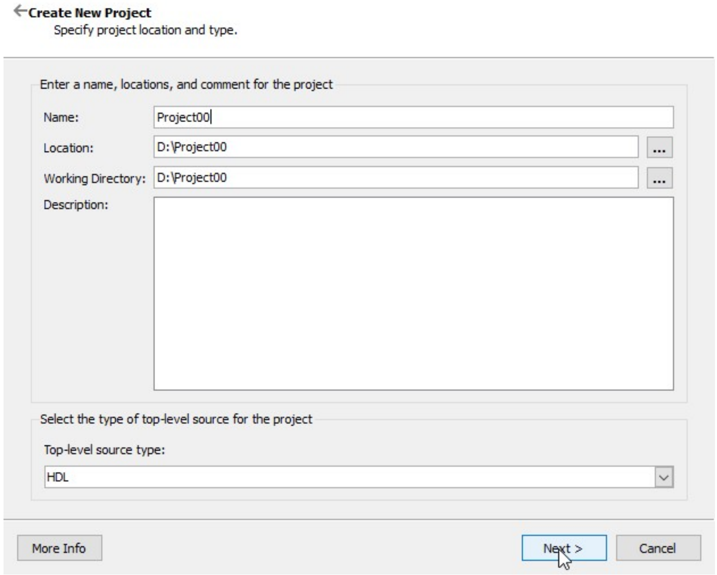
### การทดสอบบนบอร์ด MOJO V3

เนื่องจากความถีของสัญญาณนาฬิกาเป็น 50 MHz ถ้านำมาใช้กับวงจรนี้ จะไปเห็นลักษณะการวิ่งเลย ดังนั้น ต้องมีการหารความถี่ให้เหลือน้อยลง ดังนั้นระบบตามรูปที่ 1 ต้องเพิ่มวงจรหารความถี่เข้าไปด้วย



รูปที่ 2 ระบบบนบอร์ด MOJO V3 ที่มีวงจรหารความถี่

## ISE Wabpack 14.7 Start



**Create New Project**
Specify project location and type.

Enter a name, locations, and comment for the project

| | |
|---|---|
| Name: | Project00 |
| Location: | D:\Project00 |
| Working Directory: | D:\Project00 |
| Description: | |

Select the type of top-level source for the project

Top-level source type: HDL

Create New Project

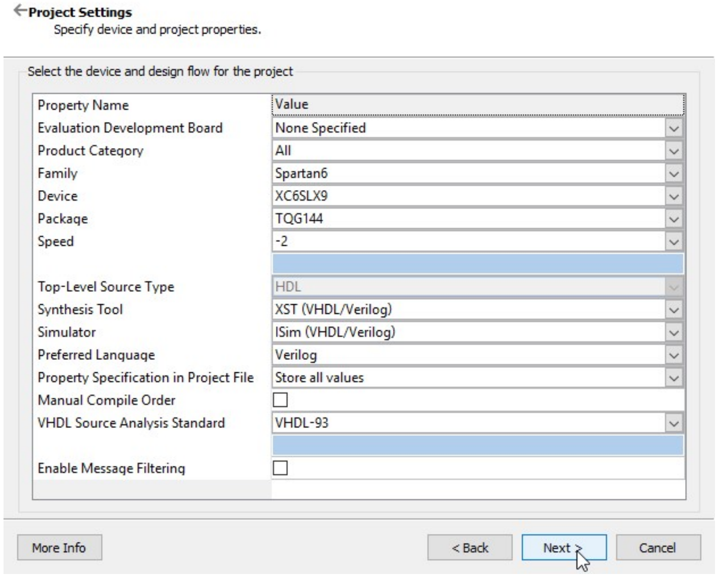
Name = Proj001

Loc. Dir = D:\xxx

Wrk. Dir = D:\xxx

HDL

Next



**Project Settings**
Specify device and project properties.

Select the device and design flow for the project

| Property Name | Value |
|---|---|
| Evaluation Development Board | None Specified |
| Product Cateqory | All |
| Family | Spartan6 |
| Device | XC6SLX9 |
| Packaqe | TQG144 |
| Speed | -2 |
| | |
| Top-Level Source Type | HDL |
| Synthesis Tool | XST (VHDL/Verilog) |
| Simulator | ISim (VHDL/Verilog) |
| Preferred Lanquage | Verilog |
| Property Specification in Project File | Store all values |
| Manual Compile Order | □ |
| VHDL Source Analysis Standard | VHDL-93 |
| | |
| Enable Messaqe Filterinq | □ |

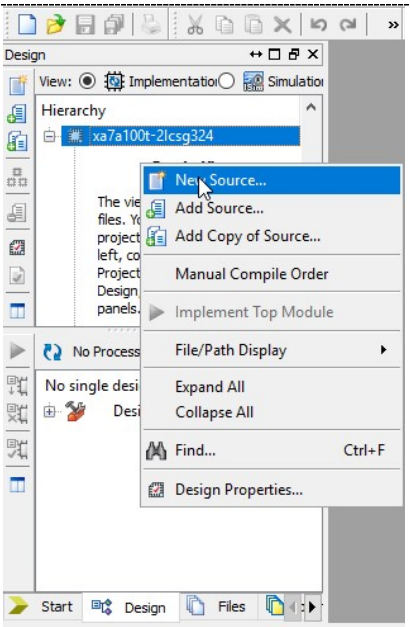
Spartan 6

XC6SLX9

TQG144

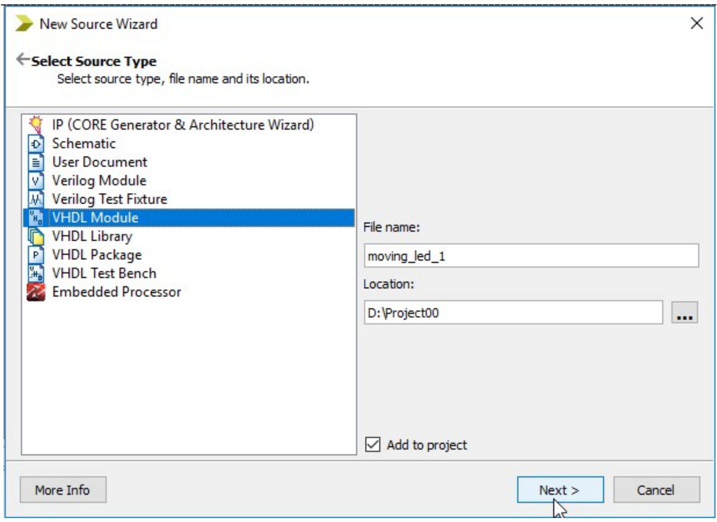
VHDL-93

Next



Click

New Source



File name: moving_led_1

Location: D:\Project00

☑ Add to project

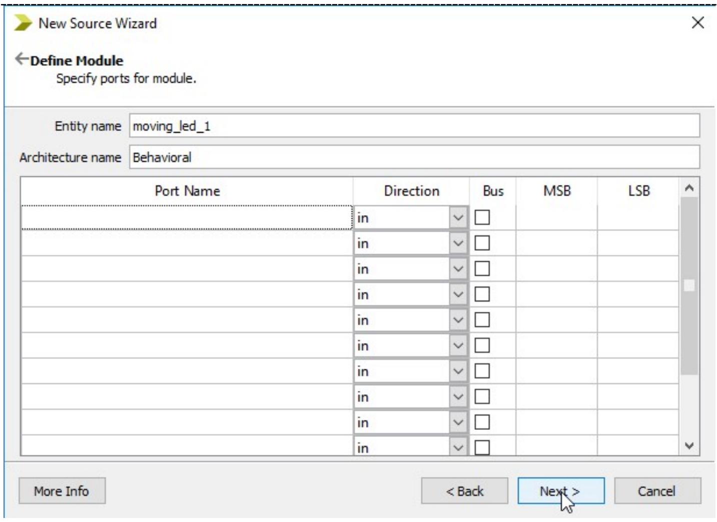
VHDL Module

Moving_led1

D:\xxxx

☑ Add to project

Next

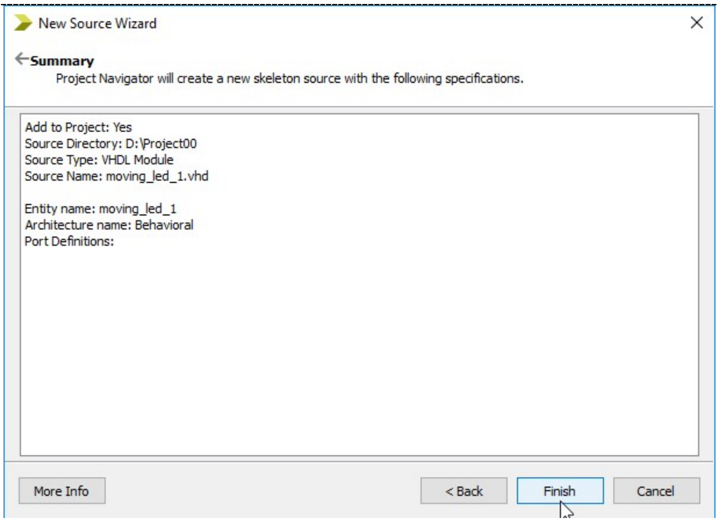


Entity name: moving_led_1

Architecture name: Behavioral

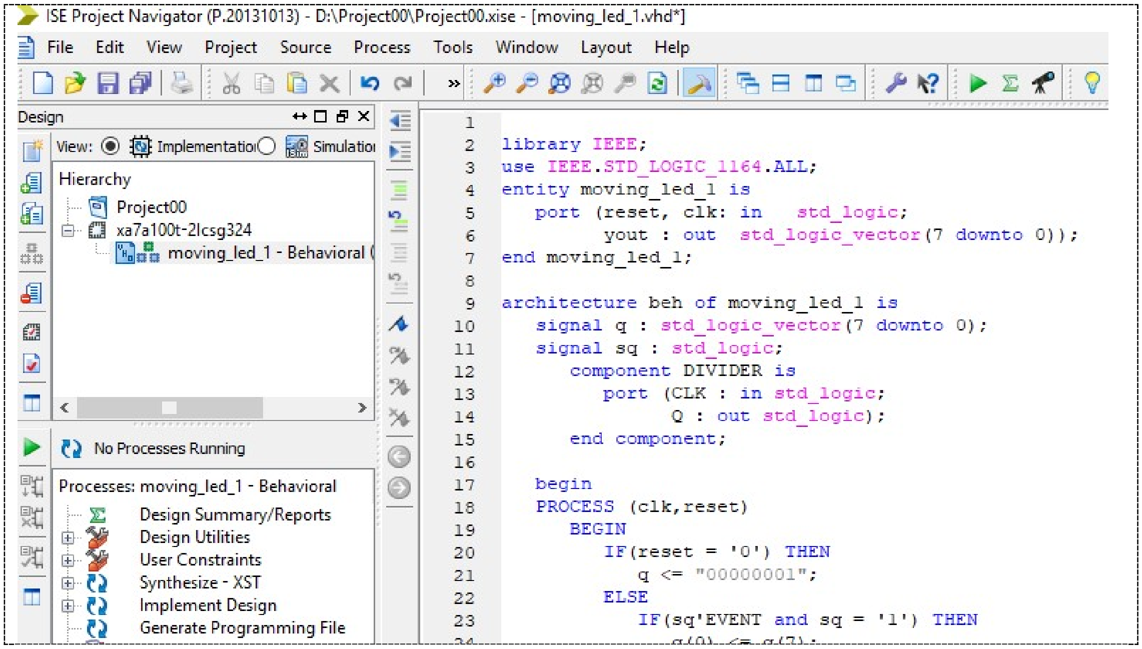
| Port Name | Direction | Bus | MSB | LSB |
|---|---|---|---|---|
| | in | □ | | |
| | in | □ | | |
| | in | □ | | |
| | in | □ | | |
| | in | □ | | |
| | in | □ | | |
| | in | □ | | |
| | in | □ | | |
| | in | □ | | |
| | in | □ | | |

Next

| | |
|---|---|
| <br> New Source Wizard<br>Summary<br>More Info<br>< Back<br>Finish<br>Cancel<br>Project Navigator will create a new skeleton source with the following specifications.<br>Add to Project: Yes<br>Source Directory: D:\Project00<br>Source Type: VHDL Module<br>Source Name: moving_led_1.vhd<br>Entity name: moving_led_1<br>Architecture name: Behavioral<br>Port Definitions: | Finish |

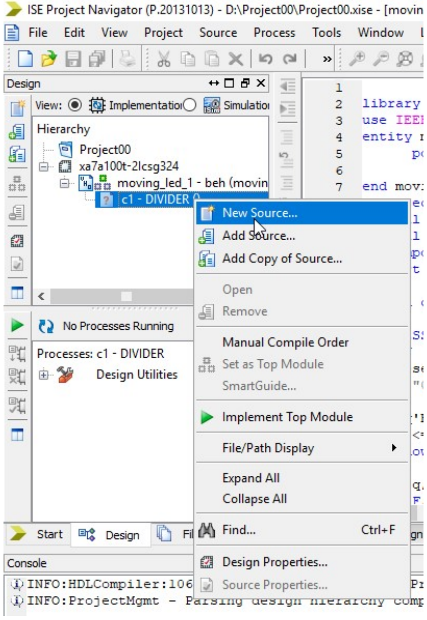
โปรแกรม moving_led_1

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity moving_led_1 is
      port (reset, clk: in   std_logic;
            yout : out  std_logic_vector(7 downto 0));
end moving_led_1;
architecture beh of moving_led_1 is
 signal q : std_logic_vector(7 downto 0);
 signal sq : std_logic;
   component DIVIDER is
   port (CLK : in std_logic;
         Q : out std_logic);
   end component;
begin

PROCESS (clk,reset)
 BEGIN
 IF(reset = '0') THEN
 q <= "00000001";
 ELSE
 IF(sq'EVENT and sq = '1') THEN
 q(0) <= q(7);
 q(7 downto 1)<=q(6 downto 0);
 ELSE
 q <= q;
 END IF;
 END IF;
 END PROCESS;
 yout <= q;
 c1: DIVIDER port map(CLK, sq);
end beh;
```


ISE Project Navigator (P.20131013) - D:\Project00\Project00.xise - [moving_led_1.vhd*]
File Edit View Project Source Process Tools Window Layout Help
Design
View: ● Implementation ○ Simulation
Hierarchy
Project00
xa7a100t-2lcsg324
moving_led_1 - Behavioral (
No Processes Running
Processes: moving_led_1 - Behavioral
Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
Implement Design
Generate Programming File

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity moving_led_1 is
    port (reset, clk: in   std_logic;
          yout : out  std_logic_vector(7 downto 0));
end moving_led_1;

architecture beh of moving_led_1 is
    signal q : std_logic_vector(7 downto 0);
    signal sq : std_logic;
        component DIVIDER is
            port (CLK : in std_logic;
                    Q : out std_logic);
        end component;

    begin
    PROCESS (clk,reset)
        BEGIN
            IF(reset = '0') THEN
                q <= "00000001";
            ELSE
                IF(sq'EVENT and sq = '1') THEN
                    q(0) <= q(7);
```

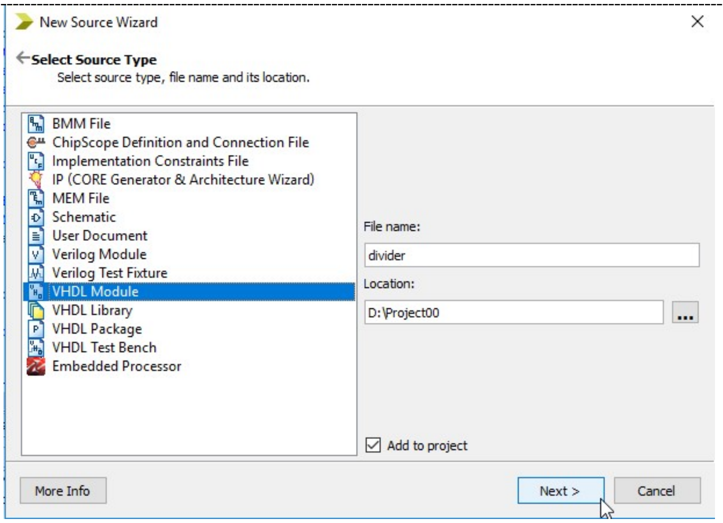
## เพิ่ม โปรแกรม Divider



```
INFO:HDLCompiler:106
INFO:ProjectMgmt - Parsing design hierarchy comp
```

Click

New Source



Select source type, file name and its location.

- BMM File
- ChipScope Definition and Connection File
- Implementation Constraints File
- IP (CORE Generator & Architecture Wizard)
- MEM File
- Schematic
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Module
- VHDL Library
- VHDL Package
- VHDL Test Bench
- Embedded Processor

File name: divider

Location: D:\Project00

☑ Add to project

VHDL Module

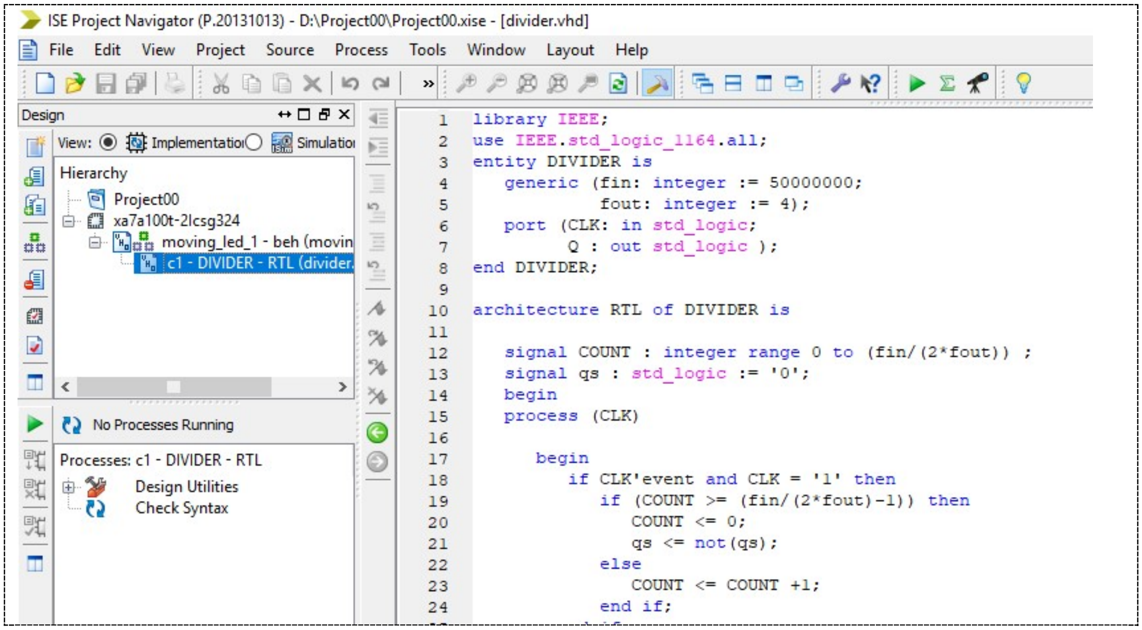
Divider

D:\xxxx

Next

## โปรแกรม Divider

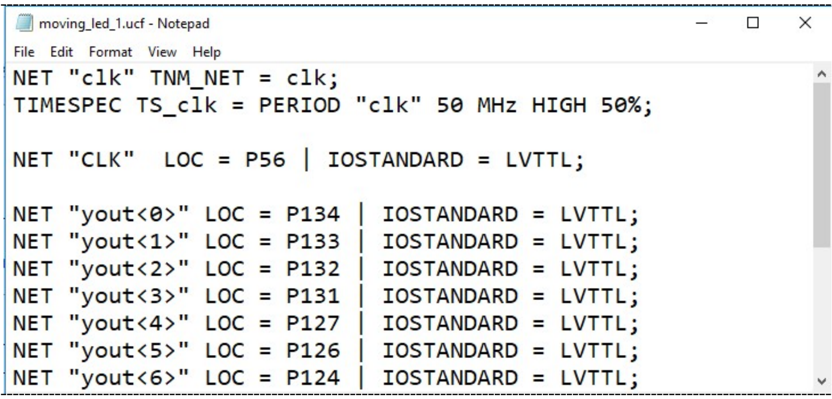
```
library IEEE;
use IEEE.std_logic_1164.all;
entity DIVIDER is
  generic (fin: integer := 50000000;
        fout: integer := 4);
  port (CLK: in std_logic;
       Q : out std_logic );
end DIVIDER;

architecture RTL of DIVIDER is

  signal COUNT : integer range 0 to (fin/(2*fout)) ;
  signal qs : std_logic := '0';
  begin
  process (CLK)

    begin
      if CLK'event and CLK = '1' then
        if (COUNT >= (fin/(2*fout)-1)) then
          COUNT <= 0;
          qs <= not(qs);
        else
          COUNT <= COUNT +1;
        end if;
     end if;
  end process;
  Q <= qs ;
end RTL;
```


ISE Project Navigator (P.20131013) - D:\Project00\Project00.xise - [divider.vhd]
File Edit View Project Source Process Tools Window Layout Help
Design
View: ● Implementation ○ Simulation
Hierarchy
Project00
xa7a100t-2lcsg324
moving_led_1 - beh (movin
c1 - DIVIDER - RTL (divider.
No Processes Running
Processes: c1 - DIVIDER - RTL
Design Utilities
Check Syntax

```
library IEEE;
use IEEE.std_logic_1164.all;
entity DIVIDER is
   generic (fin: integer := 50000000;
            fout: integer := 4);
   port (CLK: in std_logic;
         Q : out std_logic );
end DIVIDER;

architecture RTL of DIVIDER is

   signal COUNT : integer range 0 to (fin/(2*fout)) ;
   signal qs : std_logic := '0';
   begin
   process (CLK)

      begin
         if CLK'event and CLK = '1' then
            if (COUNT >= (fin/(2*fout)-1)) then
               COUNT <= 0;
               qs <= not(qs);
            else
               COUNT <= COUNT +1;
            end if;
```

สร้างไฟล์เพื่อกำหนดขาอุปกรณ

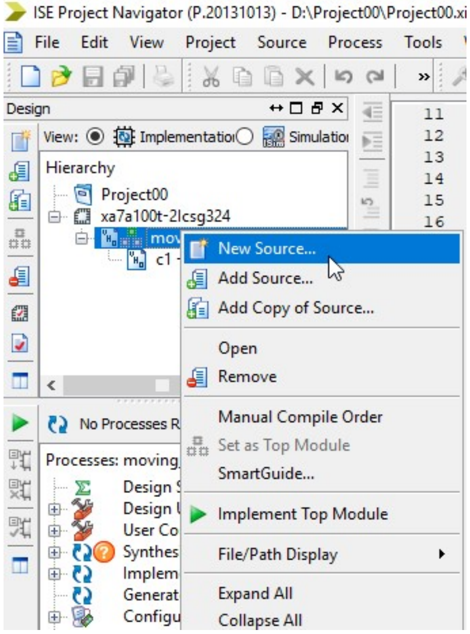
พิมพ์ ข้อความต่อไปนี้ลงใน Notepad แล้ว Save ไว้ที่โฟลดเดอร์โปรเจคที่สร้างไว้ ให้ใช้ชื่ออะไรก็ไดแต่ลงท้ายด้วย “xxxx.ucf”



```
NET "clk" TNM_NET = clk;
TIMESPEC TS_clk = PERIOD "clk" 50 MHz HIGH 50%;

NET "CLK"  LOC = P56 | IOSTANDARD = LVTTL;

NET "yout<0>" LOC = P134 | IOSTANDARD = LVTTL;
NET "yout<1>" LOC = P133 | IOSTANDARD = LVTTL;
NET "yout<2>" LOC = P132 | IOSTANDARD = LVTTL;
NET "yout<3>" LOC = P131 | IOSTANDARD = LVTTL;
NET "yout<4>" LOC = P127 | IOSTANDARD = LVTTL;
NET "yout<5>" LOC = P126 | IOSTANDARD = LVTTL;
NET "yout<6>" LOC = P124 | IOSTANDARD = LVTTL;
```

```
# “moving_led_1.ucf” File
NET "clk" TNM_NET = clk;
TIMESPEC TS_clk = PERIOD "clk" 50 MHz HIGH 50%;

NET "CLK"  LOC = P56 | IOSTANDARD = LVTTL;

NET "yout<0>" LOC = P134 | IOSTANDARD = LVTTL;
NET "yout<1>" LOC = P133 | IOSTANDARD = LVTTL;
NET "yout<2>" LOC = P132 | IOSTANDARD = LVTTL;
NET "yout<3>" LOC = P131 | IOSTANDARD = LVTTL;
NET "yout<4>" LOC = P127 | IOSTANDARD = LVTTL;
NET "yout<5>" LOC = P126 | IOSTANDARD = LVTTL;
NET "yout<6>" LOC = P124 | IOSTANDARD = LVTTL;
NET "yout<7>" LOC = P123 | IOSTANDARD = LVTTL;

NET "reset" LOC = P38 | IOSTANDARD = LVTTL;

#PACE: Start of PACE Area Constraints
#PACE: Start of PACE Prohibit Constraints
#PACE: End of Constraints generated by PACE
```

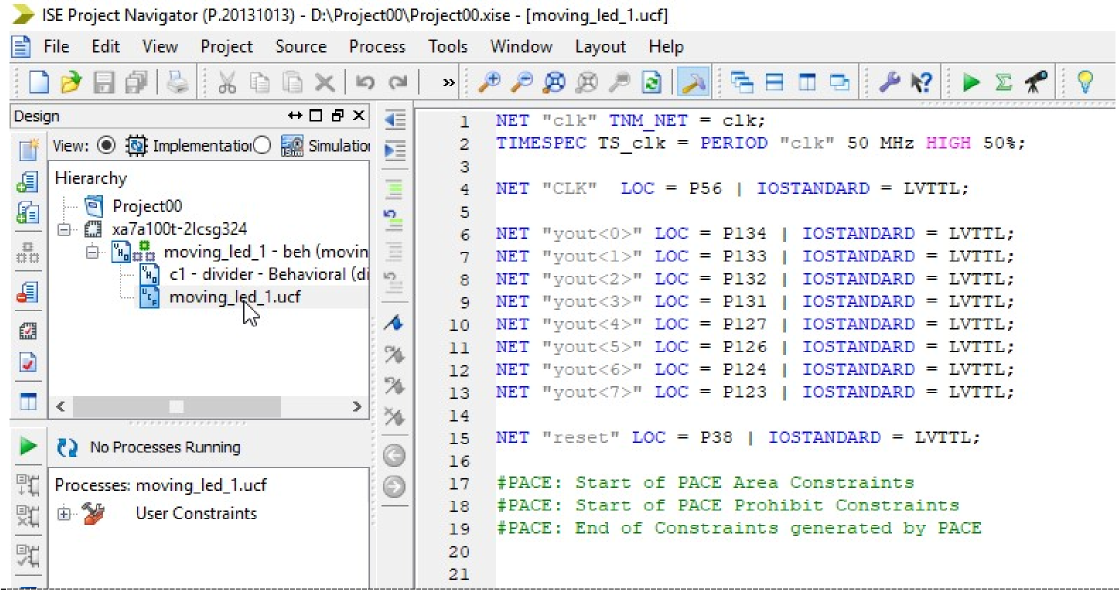
เพิ่ม “xxxx.ucf” ไฟล์เข้าไปในโปรเจ็ค



Click

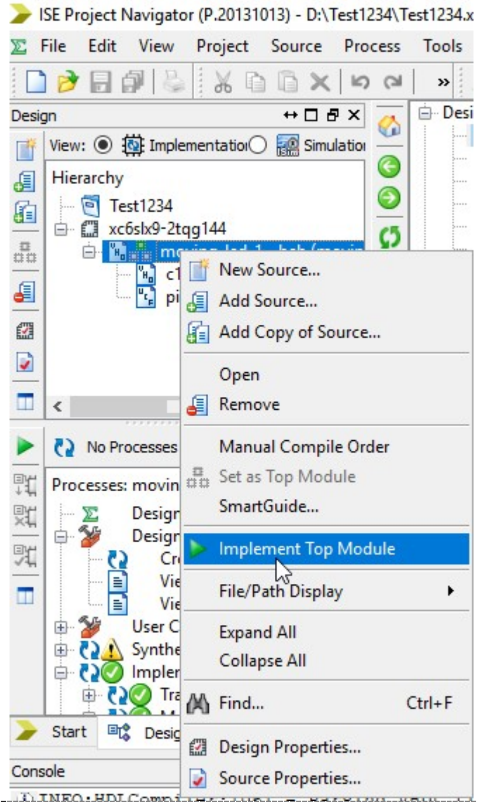
New Source

เลือกไฟล์ “xxxx.ucf.


ISE Project Navigator (P.20131013) - D:\Project00\Project00.xise - [moving_led_1.ucf]
File Edit View Project Source Process Tools Window Layout Help
Design
View: ● Implementation ○ Simulation
Hierarchy
Project00
xa7a100t-2lcsg324
moving_led_1 - beh (movin
c1 - divider - Behavioral (di
moving_led_1.ucf
No Processes Running
Processes: moving_led_1.ucf
User Constraints

```
NET "clk" TNM_NET = clk;
TIMESPEC TS_clk = PERIOD "clk" 50 MHz HIGH 50%;

NET "CLK"  LOC = P56 | IOSTANDARD = LVTTL;

NET "yout<0>" LOC = P134 | IOSTANDARD = LVTTL;
NET "yout<1>" LOC = P133 | IOSTANDARD = LVTTL;
NET "yout<2>" LOC = P132 | IOSTANDARD = LVTTL;
NET "yout<3>" LOC = P131 | IOSTANDARD = LVTTL;
NET "yout<4>" LOC = P127 | IOSTANDARD = LVTTL;
NET "yout<5>" LOC = P126 | IOSTANDARD = LVTTL;
NET "yout<6>" LOC = P124 | IOSTANDARD = LVTTL;
NET "yout<7>" LOC = P123 | IOSTANDARD = LVTTL;

NET "reset" LOC = P38 | IOSTANDARD = LVTTL;

#PACE: Start of PACE Area Constraints
#PACE: Start of PACE Prohibit Constraints
#PACE: End of Constraints generated by PACE
```

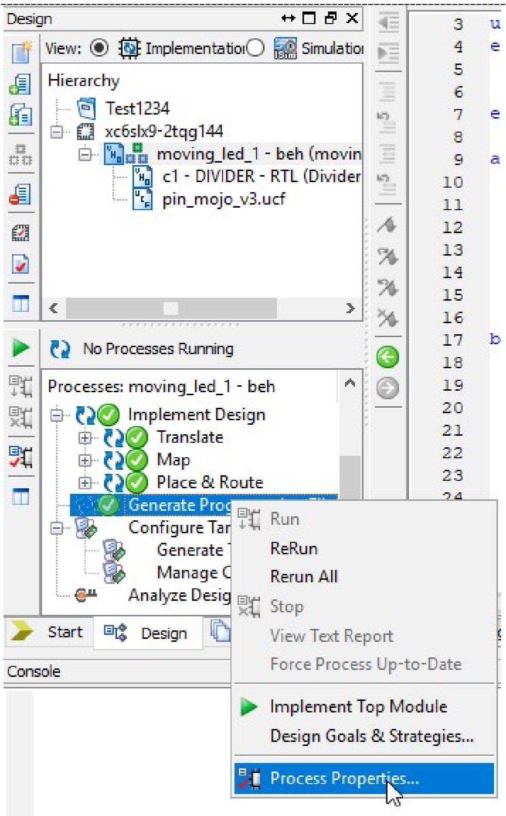
## Implement



Click

Inplement Top Mo..

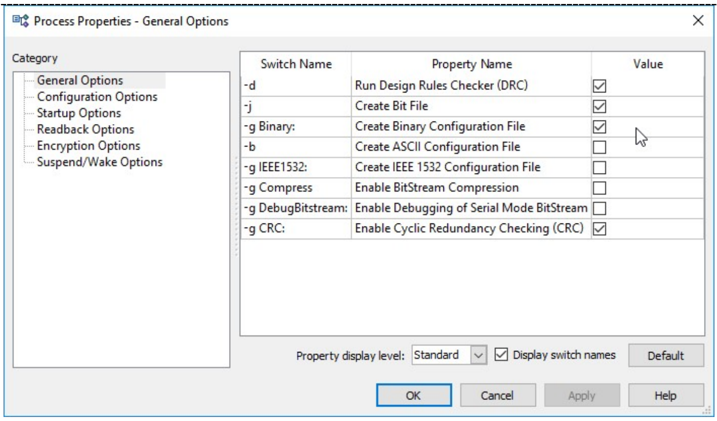
## Create Bit Stream



Click moving.v

Generate Programming..

Process Properties



Process Properties - General Options

Category

- General Options
- Configuration Options
- Startup Options
- Readback Options
- Encryption Options
- Suspend/Wake Options

| Switch Name | Property Name | Value |
|---|---|---|
| -d | Run Design Rules Checker (DRC) | ☑ |
| -j | Create Bit File | ☑ |
| -g Binary: | Create Binary Configuration File | ☑ |
| -b | Create ASCII Configuration File | □ |
| -g IEEE1532: | Create IEEE 1532 Configuration File | □ |
| -g Compress | Enable BitStream Compression | □ |
| -g DebugBitstream: | Enable Debugging of Serial Mode BitStream | □ |
| -g CRC: | Enable Cyclic Redundancy Checking (CRC) | ☑ |

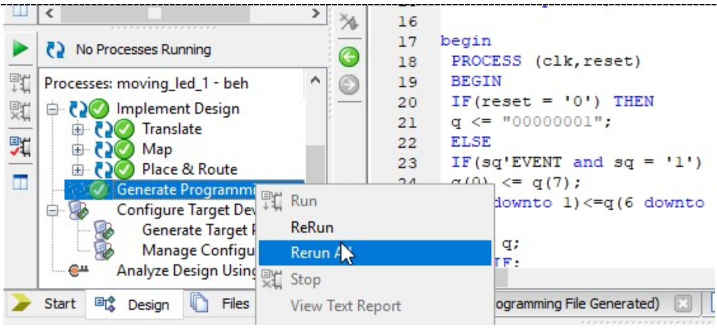
Property display level: Standard ☑ Display switch names

☑ Run Design Rules

☑ Create Bit File

☑ Create Bianry ...

Ok



No Processes Running

Processes: moving_led_1 - beh

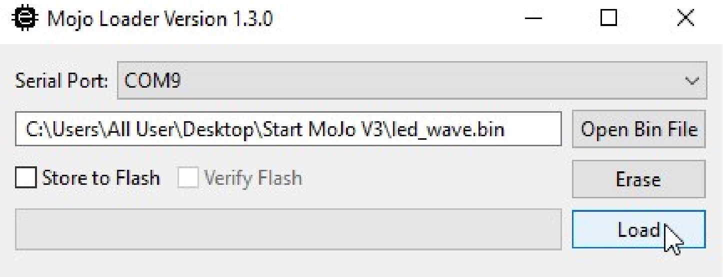
- Implement Design
  - Translate
  - Map
  - Place & Route
- Generate Programm
- Configure Target Dev
  - Generate Target F
  - Manage Configu
- Analyze Design Usin

```
begin
 PROCESS (clk,reset)
 BEGIN
 IF(reset = '0') THEN
 q <= "00000001";
 ELSE
 IF(sq'EVENT and sq = '1')
 q(0) <= q(7);
downto 1)<=q(6 downto
 q;
IF;
```

Generate Program

Rerun All

Programming with Mojo Loader Program



Mojo Loader Version 1.3.0

Serial Port: COM9

C:\Users\All User\Desktop\Start MoJo V3\led_wave.bin

□ Store to Flash □ Verify Flash

Com Port

File xx.bin

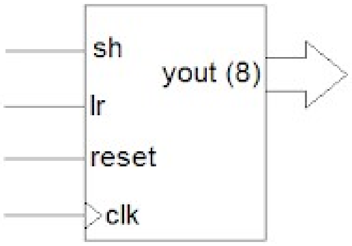
□Store... □Verify...

Load

## Lab02 - ไฟวิ่ง 8 ดวง แบบมีสัญญาณควบคุม

< Ref https://sites.google.com/site/eplearn/vhdl_fpga/lab/fpga_lab1 >

### ระบบ



รูปที่ 3 ระบบไฟวิ่งแบบมีสัญญาณควบคุมการวิ่งและทิศทางการวิ่ง

### การทำงาน

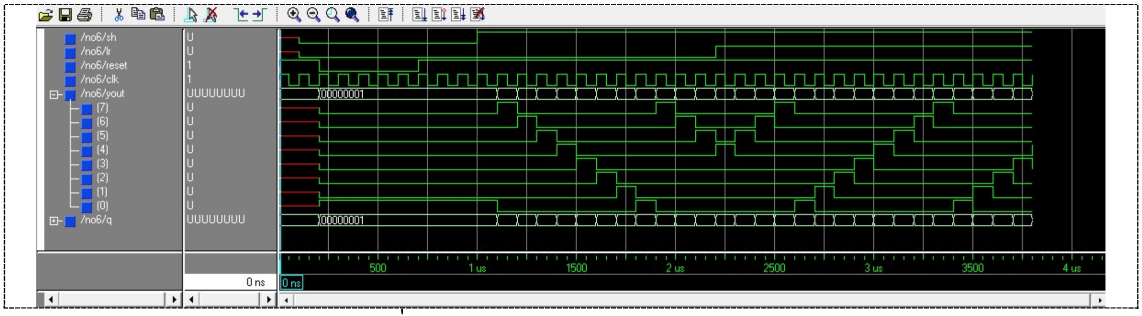
| reset | sh | lr | การทำงาน |
|---|---|---|---|
| 0 | X | X | yout = "00000001" |
| 1 | 0 | X | yout คงที่ไม่มีการเลื่อน |
| 1 | 1 | 0 | ไฟวิ่งจากซ้ายมือไปขวามือ ( MSB -> LSB ) |
| 1 | 1 | 1 | ไฟวิ่งจากขวามือไปซ้ายมือ ( MSB <- LSB ) |

หมายเหตุ x = don't care

### โปรแกรม

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity no6 is
     port (sh, lr, reset, clk: in   std_logic;
            yout : out  std_logic_vector(7 downto 0));
end no6;
architecture beh of no6 is
    signal q : std_logic_vector(7 downto 0);
begin
 PROCESS (clk,reset)
 BEGIN
 IF(reset = '0') THEN
 q <= "00000001";
 ELSE
 IF(CLK'EVENT and CLK = '1') THEN
 IF sh = '1' THEN
 IF lr = '0' THEN
 q(7) <= q(0);
 q(6 downto 0)<=q(7 downto 1);
 ELSE
 q(0) <= q(7);
 q(7 downto 1)<=q(6 downto 0);
 END IF;
 ELSE
 q <= q;
 END IF;
 ELSE
 q <= q;
 END IF;
 END IF;
 END PROCESS;
 yout <= q;
end beh;
```

ทดสอบการทำงานด้วย Model sim



รูปที่ 2 แสดงผลการจำลองการทำงาน

# Lab03 - สร้างไฟวิ่ง 8 ดวงออกทาง LED ของ Mojo V3 Board by Verilog

## 3.1 ทดสอบการทำงานโปรแกรม

### 3.1/2 Verilog Code File

```
`timescale 1ns / 1ps

module MainTest7Seg(Clk_50MHz, Reset_Onboard, LED_Output);
        input                   Reset_Onboard, Clk_50MHz;
        output          [7:0] LED_Output;

        reg             [3:0]           Counter;
        reg             [7:0]           rLED_Output;
        reg             [27:0]          Dly_Counter;

        always@(posedge Clk_50MHz or negedge Reset_Onboard )       begin
                if(Reset_Onboard == 0)
                        Counter <= 0;
                else begin
                        Dly_Counter <= Dly_Counter + 1'b1;
                        if(Dly_Counter >= 6_250_000) begin
                                Dly_Counter <= 0;
                                if(Counter >= 13) Counter <= 0;
                                else            Counter <= Counter + 1'b1;
                        end
                end
        end

        always @(Counter)
                case (Counter)
                        4'b0000:  rLED_Output = 8'b10000000;
                        4'b0001:  rLED_Output = 8'b01000000;
                        4'b0010:  rLED_Output = 8'b00100000;
                        4'b0011:  rLED_Output = 8'b00010000;
                        4'b0100:  rLED_Output = 8'b00001000;
                        4'b0101:  rLED_Output = 8'b00000100;
                        4'b0110:  rLED_Output = 8'b00000010;
                        4'b0111:  rLED_Output = 8'b00000001;
                        4'b1000:  rLED_Output = 8'b00000000;
                        4'b1001:  rLED_Output = 8'b11111111;
                        4'b1010:  rLED_Output = 8'b00000000;
                        4'b1011:  rLED_Output = 8'b11111111;
                        4'b1100:  rLED_Output = 8'b00000000;
                endcase

        assign LED_Output = rLED_Output;

Endmodule
```

### 3.2/2 Pin Definition File

```
NET "Clk_50MHz"                 LOC = P56 | IOSTANDARD = LVTTL;
NET "Reset_Onboard"             LOC = P38 | IOSTANDARD = LVTTL;

NET "LED_Output<0>" LOC = P134 | IOSTANDARD = LVTTL;
NET "LED_Output<1>" LOC = P133 | IOSTANDARD = LVTTL;
NET "LED_Output<2>" LOC = P132 | IOSTANDARD = LVTTL;
NET "LED_Output<3>" LOC = P131 | IOSTANDARD = LVTTL;
NET "LED_Output<4>" LOC = P127 | IOSTANDARD = LVTTL;
NET "LED_Output<5>" LOC = P126 | IOSTANDARD = LVTTL;
NET "LED_Output<6>" LOC = P124 | IOSTANDARD = LVTTL;
NET "LED_Output<7>" LOC = P123 | IOSTANDARD = LVTTL;
```

## 3.2 คำถาม

- ต้องการให้โปรแกรมทำงานเป็นไฟวิ่งไปแล้ววิ่งกลับ ต้องปรับแก้โปรแกรมอะไรบ้าง

# Lab04 – Mode Select with Switch

## 4.1 ทดสอบการทำงานโปรแกรม ลองกด Reset Switch

### Verilog Code File

```
`timescale 1ns / 1ps

module MainTest7Seg(Rst_OnBoard, LED_Output);
        input                   Rst_OnBoard;
        output  [7:0]           LED_Output;
        reg                     [7:0]   rLED_Output;
        reg                     rMode   = 1;

        always@(negedge Rst_OnBoard)            begin
                if(Rst_OnBoard == 0)
                        if(rMode == 1) begin
                                rLED_Output <= 8'b00000000;
                                rMode <= 0;
                                end
                        else begin
                                rLED_Output <= 8'b11111111;
                                rMode <= 1;
                                end
                end

        assign LED_Output = rLED_Output;

endmodule
```

### Pin Definition File

```
// NET "Clk_50MHz" LOC = P56 | IOSTANDARD = LVTTL;
NET "Rst_OnBoard"   LOC = P38 | IOSTANDARD = LVTTL | CLOCK_DEDICATED_ROUTE=FALSE;

NET "LED_Output<0>"     LOC = P134 | IOSTANDARD = LVTTL;
NET "LED_Output<1>"     LOC = P133 | IOSTANDARD = LVTTL;
NET "LED_Output<2>"     LOC = P132 | IOSTANDARD = LVTTL;
NET "LED_Output<3>"     LOC = P131 | IOSTANDARD = LVTTL;
NET "LED_Output<4>"     LOC = P127 | IOSTANDARD = LVTTL;
NET "LED_Output<5>"     LOC = P126 | IOSTANDARD = LVTTL;
NET "LED_Output<6>"     LOC = P124 | IOSTANDARD = LVTTL;
NET "LED_Output<7>"     LOC = P123 | IOSTANDARD = LVTTL;
```

## 4.2 คำถาม

- ต้องการให้โปรแกรมทำงาน 3 โหมด คือ (0)All On, (1)All Off, (2)Blink

# Lab05 - สร้างไฟวิ่ง 8 ดวงออกทาง LED V.2 ของ Mojo V3 Board by Verilog

## 5.1 ทดสอบการทำงานโปรแกรม ลองกด Reset Switch

Verilog Code File

```
`timescale 1ns / 1ps

module MainTest7Seg(Clk_50MHz, Rst_OnBoard, LED_Output);
        input                   Rst_OnBoard, Clk_50MHz;
        output  [7:0] LED_Output;

        parameter Start_Case_0                  = 6'b000000;
        parameter Start_Case_1                  = 6'b100000;
        parameter StepRun_Case_0        = 13;
        parameter StepRun_Case_1        = 15;
        reg     rMode                           = 1'b1;
        reg     [5:0] rCounter          = Start_Case_1;
        reg     [5:0] rStart            = Start_Case_1;
        reg     [5:0] rStop                     = Start_Case_1 + StepRun_Case_1;
        reg     [7:0] rLED_Output;
        reg     [27:0] Dly_Counter;

        always@(posedge Clk_50MHz or negedge Rst_OnBoard)      begin
                if(Rst_OnBoard == 0)            begin
                        if(rMode == 1) begin
                                rMode   <= 0;
                                rCounter <= Start_Case_0;
                                rStart  <= Start_Case_0;
                                rStop                   <= Start_Case_0 + StepRun_Case_0;
                                end
                        else begin
                                rMode   <= 1;
                                rCounter <= Start_Case_1;
                                rStart  <= Start_Case_1;
                                rStop                   <= Start_Case_1 + StepRun_Case_1;
                                end
                        end
                else begin
                        Dly_Counter <= Dly_Counter + 1'b1;
                        if(Dly_Counter >= 6_250_000) begin
                                Dly_Counter <= 0;
                                if (rCounter >= rStop) rCounter <= rStart;
                                else    rCounter <= rCounter + 1'b1;
                                end
                        end
                end

        always @(rCounter)
                case (rCounter)
                        //----------Mode 0 = 000000 to 011111 ---------
                        Start_Case_0+0:         rLED_Output = 8'b10000000;
                        Start_Case_0+1:         rLED_Output = 8'b01000000;
                        Start_Case_0+2:         rLED_Output = 8'b00100000;
                        Start_Case_0+3:         rLED_Output = 8'b00010000;
                        Start_Case_0+4:         rLED_Output = 8'b00001000;
                        Start_Case_0+5:         rLED_Output = 8'b00000100;
                        Start_Case_0+6:         rLED_Output = 8'b00000010;
                        Start_Case_0+7:         rLED_Output = 8'b00000001;
                        Start_Case_0+8:         rLED_Output = 8'b00000000;
                        Start_Case_0+9:         rLED_Output = 8'b11111111;
                        Start_Case_0+10:        rLED_Output = 8'b00000000;
                        Start_Case_0+11:        rLED_Output = 8'b11111111;
                        Start_Case_0+12:        rLED_Output = 8'b00000000;
                        //----------Mode 1 = 100000 to 111111 ---------
                        Start_Case_1+0:         rLED_Output = 8'b10000001;
                        Start_Case_1+1:         rLED_Output = 8'b01000010;
                        Start_Case_1+2:         rLED_Output = 8'b00100100;
                        Start_Case_1+3:         rLED_Output = 8'b00011000;
                        Start_Case_1+4:         rLED_Output = 8'b00011000;
                        Start_Case_1+5:         rLED_Output = 8'b00100100;
                        Start_Case_1+6:         rLED_Output = 8'b01000010;
                        Start_Case_1+7:         rLED_Output = 8'b10000001;
                        Start_Case_1+8:         rLED_Output = 8'b00000000;
                        Start_Case_1+9:         rLED_Output = 8'b11111111;
                        Start_Case_1+10:        rLED_Output = 8'b00000000;
                        Start_Case_1+11:        rLED_Output = 8'b11111111;
                        Start_Case_1+12:        rLED_Output = 8'b00000000;
                        Start_Case_1+13:        rLED_Output = 8'b11111111;
                        Start_Case_1+14:        rLED_Output = 8'b00000000;
                        //----------Mode x = default ------------------
                        default:        rLED_Output = 8'b00000000;
                endcase

        assign LED_Output = rLED_Output;

endmodule
```

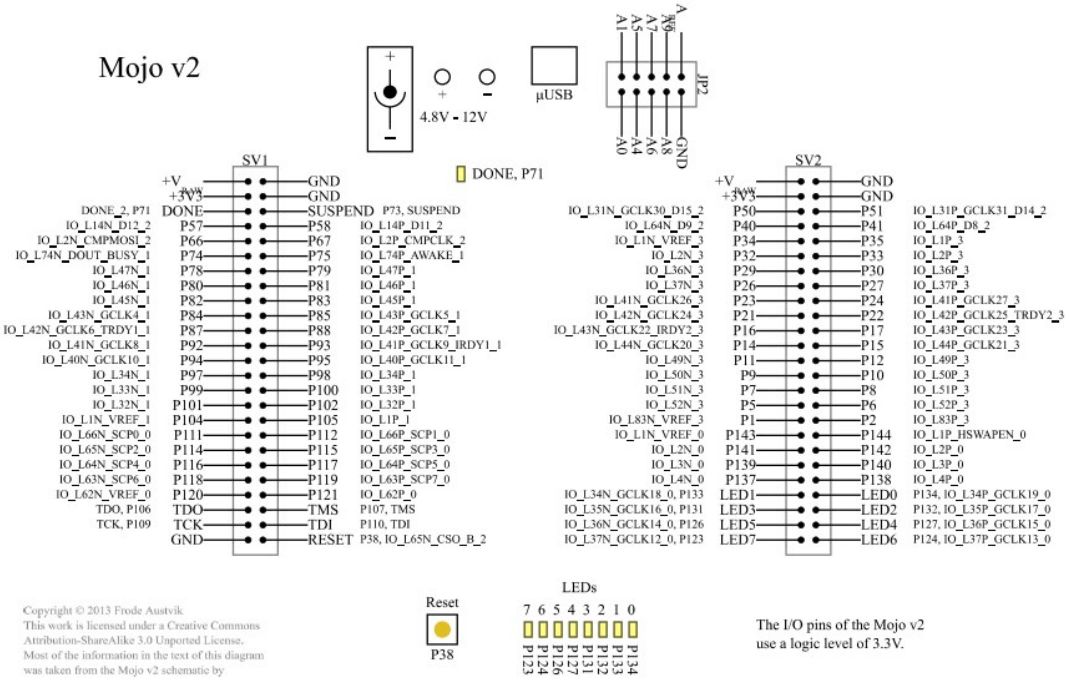
## Pin Definition File

```
NET "Rst_OnBoard"           LOC = P38 | IOSTANDARD = LVTTL | CLOCK_DEDICATED_ROUTE = FALSE;
NET "Clk_50MHz"             LOC = P56 | IOSTANDARD = LVTTL;

NET "LED_Output<0>"         LOC = P134 | IOSTANDARD = LVTTL;
NET "LED_Output<1>"         LOC = P133 | IOSTANDARD = LVTTL;
NET "LED_Output<2>"         LOC = P132 | IOSTANDARD = LVTTL;
NET "LED_Output<3>"         LOC = P131 | IOSTANDARD = LVTTL;
NET "LED_Output<4>"         LOC = P127 | IOSTANDARD = LVTTL;
NET "LED_Output<5>"         LOC = P126 | IOSTANDARD = LVTTL;
NET "LED_Output<6>"         LOC = P124 | IOSTANDARD = LVTTL;
NET "LED_Output<7>"         LOC = P123 | IOSTANDARD = LVTTL;
```

## 5.2 คำถาม

- ต้องการให้โปรแกรมทำงานเป็นไฟวิ่ง 3 รูปแบบตามแต่จิตนาการ

## Mojo V2 Pin I/O



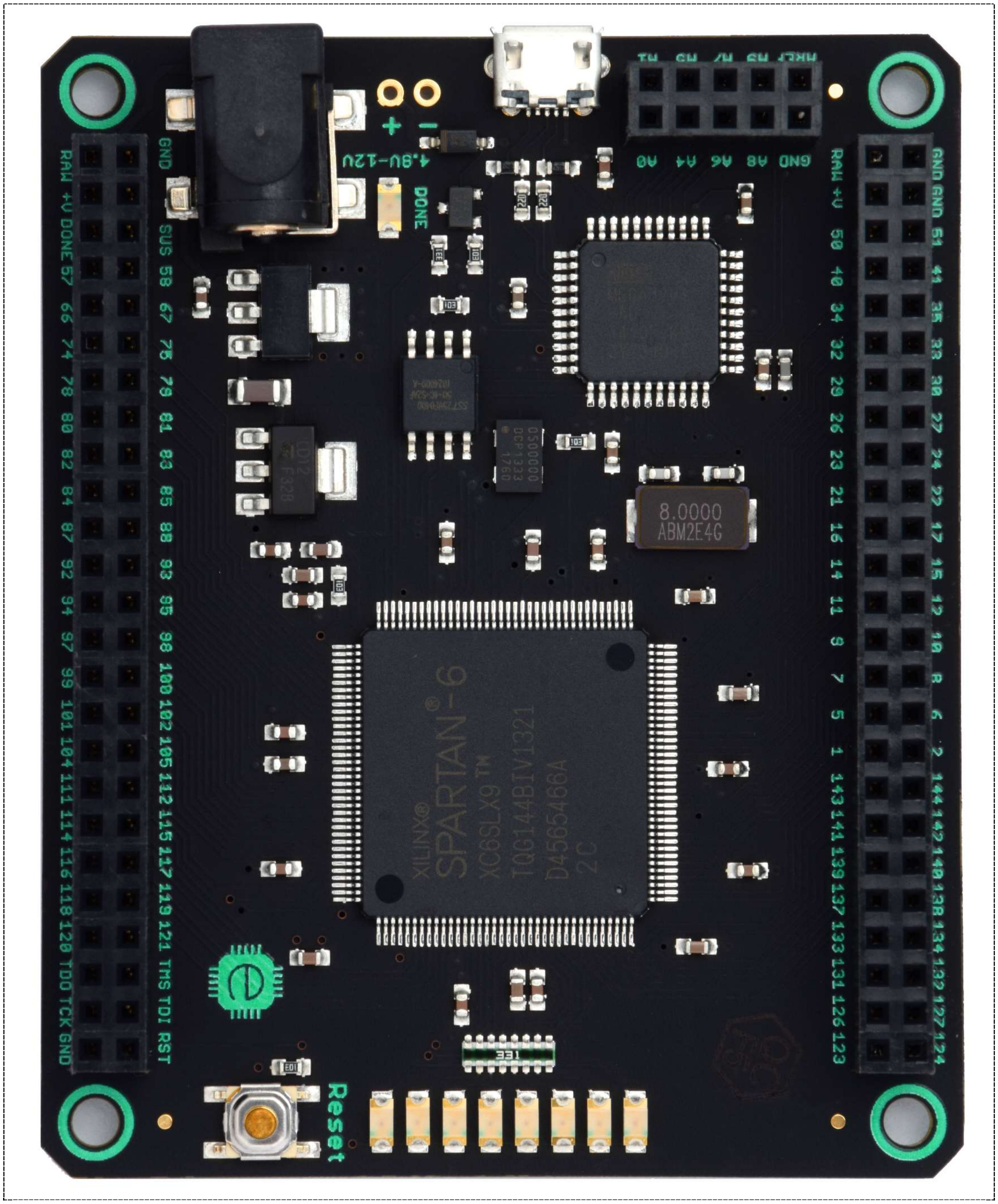
| | | | |
|---|---|---|---|
| | $+V_{RAW}$ | GND | |
| | +3V3 | GND | |
| DONE_2, P71 | DONE | SUSPEND | P73, SUSPEND |
| IO_L14N_D12_2 | P57 | P58 | IO_L14P_D11_2 |
| IO_L2N_CMPMOSI_2 | P66 | P67 | IO_L2P_CMPCLK_2 |
| IO_L74N_DOUT_BUSY_1 | P74 | P75 | IO_L74P_AWAKE_1 |
| IO_L47N_1 | P78 | P79 | IO_L47P_1 |
| IO_L46N_1 | P80 | P81 | IO_L46P_1 |
| IO_L45N_1 | P82 | P83 | IO_L45P_1 |
| IO_L43N_GCLK4_1 | P84 | P85 | IO_L43P_GCLK5_1 |
| IO_L42N_GCLK6_TRDY1_1 | P87 | P88 | IO_L42P_GCLK7_1 |
| IO_L41N_GCLK8_1 | P92 | P93 | IO_L41P_GCLK9_IRDY1_1 |
| IO_L40N_GCLK10_1 | P94 | P95 | IO_L40P_GCLK11_1 |
| IO_L34N_1 | P97 | P98 | IO_L34P_1 |
| IO_L33N_1 | P99 | P100 | IO_L33P_1 |
| IO_L32N_1 | P101 | P102 | IO_L32P_1 |
| IO_L1N_VREF_1 | P104 | P105 | IO_L1P_1 |
| IO_L66N_SCP0_0 | P111 | P112 | IO_L66P_SCP1_0 |
| IO_L65N_SCP2_0 | P114 | P115 | IO_L65P_SCP3_0 |
| IO_L64N_SCP4_0 | P116 | P117 | IO_L64P_SCP5_0 |
| IO_L63N_SCP6_0 | P118 | P119 | IO_L63P_SCP7_0 |
| IO_L62N_VREF_0 | P120 | P121 | IO_L62P_0 |
| TDO, P106 | TDO | TMS | P107, TMS |
| TCK, P109 | TCK | TDI | P110, TDI |
| | GND | RESET | P38, IO_L65N_CSO_B_2 |

| | | | |
|---|---|---|---|
| | $+V_{RAW}$ | GND | |
| | +3V3 | GND | |
| IO_L31N_GCLK30_D15_2 | P50 | P51 | IO_L31P_GCLK31_D14_2 |
| IO_L64N_D9_2 | P40 | P41 | IO_L64P_D8_2 |
| IO_L1N_VREF_3 | P34 | P35 | IO_L1P_3 |
| IO_L2N_3 | P32 | P33 | IO_L2P_3 |
| IO_L36N_3 | P29 | P30 | IO_L36P_3 |
| IO_L37N_3 | P26 | P27 | IO_L37P_3 |
| IO_L41N_GCLK26_3 | P23 | P24 | IO_L41P_GCLK27_3 |
| IO_L42N_GCLK24_3 | P21 | P22 | IO_L42P_GCLK25_TRDY2_3 |
| IO_L43N_GCLK22_IRDY2_3 | P16 | P17 | IO_L43P_GCLK23_3 |
| IO_L44N_GCLK20_3 | P14 | P15 | IO_L44P_GCLK21_3 |
| IO_L49N_3 | P11 | P12 | IO_L49P_3 |
| IO_L50N_3 | P9 | P10 | IO_L50P_3 |
| IO_L51N_3 | P7 | P8 | IO_L51P_3 |
| IO_L52N_3 | P5 | P6 | IO_L52P_3 |
| IO_L83N_VREF_3 | P1 | P2 | IO_L83P_3 |
| IO_L1N_VREF_0 | P143 | P144 | IO_L1P_HSWAPEN_0 |
| IO_L2N_0 | P141 | P142 | IO_L2P_0 |
| IO_L3N_0 | P139 | P140 | IO_L3P_0 |
| IO_L4N_0 | P137 | P138 | IO_L4P_0 |
| IO_L34N_GCLK18_0, P133 | LED1 | LED0 | P134, IO_L34P_GCLK19_0 |
| IO_L35N_GCLK16_0, P131 | LED3 | LED2 | P132, IO_L35P_GCLK17_0 |
| IO_L36N_GCLK14_0, P126 | LED5 | LED4 | P127, IO_L36P_GCLK15_0 |
| IO_L37N_GCLK12_0, P123 | LED7 | LED6 | P124, IO_L37P_GCLK13_0 |

Copyright © 2013 Frode Austvik
This work is licensed under a Creative Commons
Attribution-ShareAlike 3.0 Unported License.
Most of the information in the text of this diagram
was taken from the Mojo v2 schematic by

The I/O pins of the Mojo v2 use a logic level of 3.3V.

## Mojo V3


+
-
4.8V-12V
DONE
Reset
A1 A5 A7 A9 AREF
A0 A4 A6 A8 GND
RAW +V DONE 57 66 74 78 80 82 84 87 92 94 97 99 101 104 111 114 116 118 120 TDO TCK GND
GND SUS 58 67 75 79 81 83 85 88 93 95 98 100 102 105 112 115 117 119 121 TMS TDI RST
RAW +V 50 40 34 32 29 26 23 21 16 14 11 9 7 5 1 143 141 139 137 133 131 126 123
GND GND 51 41 35 33 30 27 24 22 17 15 12 10 8 6 2 144 142 140 138 134 132 127 124
8.0000
ABM2E4G
XILINX®
SPARTAN®-6
XC6SLX9™
TQG144BIV1321
D4565466A
2C
331