27/05/2024

Git (TortoiseGit)

* Clone
* Add
* Commit
* Push

.qsf

Settings File (.qsf) contains all of the project-wide and entity-level assignments and settings for the current revision of the project. A separate Quartus® Prime Settings File exists for each individual revision. The Quartus® Prime Settings File syntax is based on Tcl script syntax.

.qpf

Project File (.qpf) contains basic information about the current version of the Quartus® Prime software and the date. It also lists all of the revisions created for the project.

.c

a source code file written in C programming language.

.h

An H file is a header file referenced by a document written in C, C++, or Objective-C source code. It may contain variables, constants, and functions that are used by other files within a programming project. H files allow commonly used functions to be written only once and referenced by other source files when needed.

.sv

An ASCII text file (with the extension .sv) created with the Quartus® Prime Text Editor or any other standard text editor. A SystemVerilog Design File describes design logic in the SystemVerilog language, which is an extension to Verilog. A SystemVerilog Design File can contain any combination of the SystemVerilog constructs supported by the Quartus® Prime software.

.v

An ASCII text file (with the extension .v, .verilog, .vlg, or .vh) created with the Quartus® Prime Text Editor or any other standard text editor. A Verilog Design File contains design logic that is defined with Verilog HDL. A Verilog Design File can contain any combination of the Verilog HDL constructs supported by the Quartus® Prime software.

.vhd

An ASCII text file (with the extension .vhd or .vhdl) created with the Quartus® Prime Text Editor or another standard text editor. A VHDL Design File contains design logic that is defined with VHDL. A VHDL Design File can contain any combination of the VHDL 1987 or 1993 constructs supported by the Quartus® Prime software.

.hex

An ASCII text file (with the extension .hex). You can use a Hexadecimal (Intel-Format) File (.hex) in the Quartus® Prime software to store the initial memory values for a memory block, such as RAM or ROM, that is implemented in an FPGA device, or to build software project executables. You can use .hex files as input files in the Quartus® Prime software in the following ways:

* The Memory Editor can create a .hex file for memory initialization in the Compiler and Simulator. You can also use a Memory Initialization File (.mif) to provide memory initialization data.
* The In-System Memory Content Editor can use and create a .hex file to import and export data.

.sof

A binary file (with the extension .sof), generated by the Compiler's Assembler module, or by the makeprogfile command-line utility. A SOF contains the data for configuring all SRAM-based Intel devices supported by the Quartus® Prime software, using the Programmer.

.pof

The Programmer Output File (POF) carries the programming data for the actual Flash EPROM (FEPROM) that the FPGA will load configuration files from. Some FPGAs can select between multiple files.

###

.qsys

.qip

.tcl

.sopcinfo

.sdc

.jic

.cof

.vho

.stp

.do

DECA (MAX10)

<https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=&No=944&PartNo=1>

&&&

1. ทำความเข้าใจ FPGA ใหม่ตั้งแต่ต้น
   1. สร้างโปรเจคเอง ตั้งแต่ต้น ตั้งแต่ การใช้ PLL, pinmap

A diagram of a flowchart

Description automatically generated

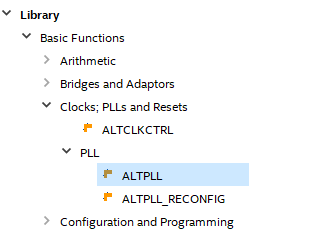
PLL

การใช้งาน :

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated



PLL IP core Quartus II

A screenshot of a computer

Description automatically generated

Pin Planner

LED[7..0] : BANK-8 , VCCIO = 1.2V

LED[0] 🡪 LED[7] : C7, C8, A6, B7, C4, A5, B4, C5

SW[1..0] : BANK-6 , VCCIO = 1.5V Schmitt Trigger

SW[0], SW[1] : J21, J22

KEY[1..0] : BANK-6 , VCCIO = 1.5V Schmitt Trigger

KEY[0], KEY[1] : H21, H22

MAX10\_CLK1\_50 : BANK-2 , VCCIO = 2.5V , M8

MAX10\_CLK2\_50 : BANK-3 , VCCIO = 3.3V , P11

- FPGA dev flow

A diagram of a software development process

Description automatically generated

1. Design เป็นการกำหนดรายละเอียดของวงจรที่ต้องการ ซึ่งจะบ่งบอกถึงส่วนประกอบต่างๆ ของวงจร
2. Synthesis ขั้นตอนการสังเคราะห์วงจรฮาร์ดแวร์ที่เกิดจากโค้ด VHDL ประกอบด้วยการแปลงโค้ดให้เป็นวงจรลอจิก แล้วทำการเทียบวงจรเข้ากับ FPGA (mapping) สำหรับใช้ในการทำ Place & Route ในการสังเคราะห์จะมีการเลือกFPGA เป้าหมายที่ใช้สำหรับการสังเคราะห์วงจร
3. Simulation เป็นขั้นตอนการจำลองการทำงานของโค้ด VHDL หรือ Verilog ในระดับ RTL (Register Transfer Level) ในการจำลองการทำงาน อาจจะใช้การเขียน testbench เพื่อสร้างสัญญาณป้อนเข้าไปยังวงจรที่ต้องการทดสอบ
4. Place & Route ซึ่งเป็นการนำเอาวงจรลอจิกมาจัดวางบน FPGA จากนั้นซอฟต์แวร์จะทำการคำนวณค่าความหน่วงของส่วนลอจิก (Logic Delay) พร้อมทั้งค่าความหน่วงของสายสัญญาณ (Interconnect Delay) จริงออกมา ในรูปแบบของไฟล์ SDF (Standard Delay Format) ซึ่งจะนำไปใช้สำหรับการจำลองการทำงานทางเวลา (Timing Simulation)
5. Timing Analysis เป็นการจำลองการทำงานของวงจร โดยมีการคิดค่าความหน่วงของเกต (Gate Delay) และความหน่วงการเชื่อมต่อที่เกิดขึ้นจริง (Interconnect Delay) ในการกำหนดการวางลอจิกและการเชื่อมต่อสัญญาณใน FPGA
6. Programming ขั้นตอนการโปรแกรม Configuration File ลงบน FPGA ใช้วิธีโปรแกรมไฟล์บิทลงบนหน่วยเก็บความจำแบบแฟลช (Flash Memory) หรือ EPROM หรือ EEPROM

28/05/2024

- ศึกษา FPGA circuit

<https://drive.google.com/file/d/1G71Zre32YsKcez-d5QMOgyZTda1KCDVm/view?usp=drive_link>

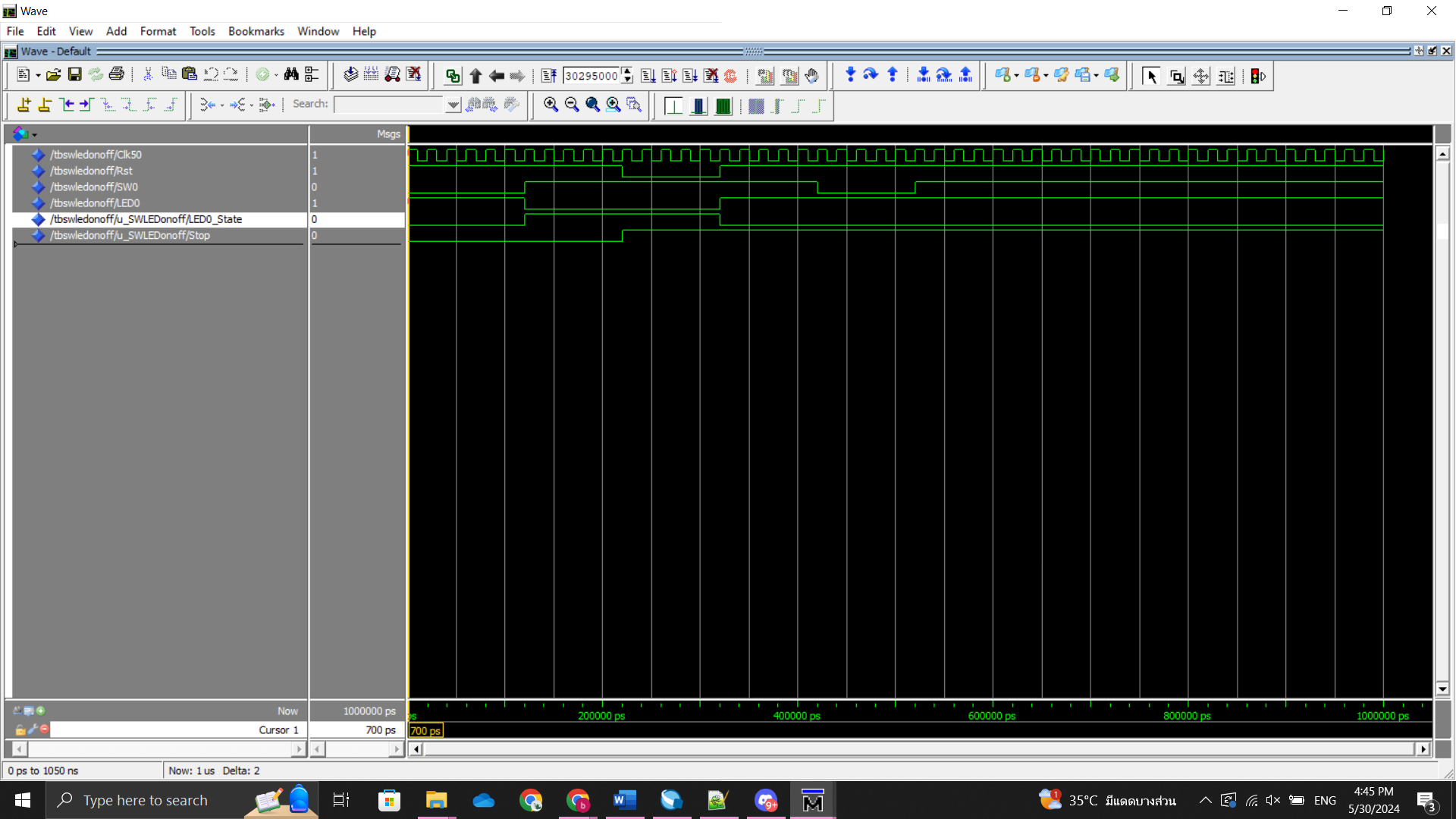
* 1. เขียน VHDL ง่ายๆ + simulation : slide SW -> LED ON/OFF เมื่อกด sw reset แล้ว ทุกอย่างจะไม่ทำงาน

Difference between RTL and behavioral code :

However, digital designers usually use the term behavioural model for non-synthesizable models (faster to simulate and easy to debug) and the term RTL for the synthesizable models.

The entity defines the interface, the architecture defines the function.

- ศึกษาวิธี simulation

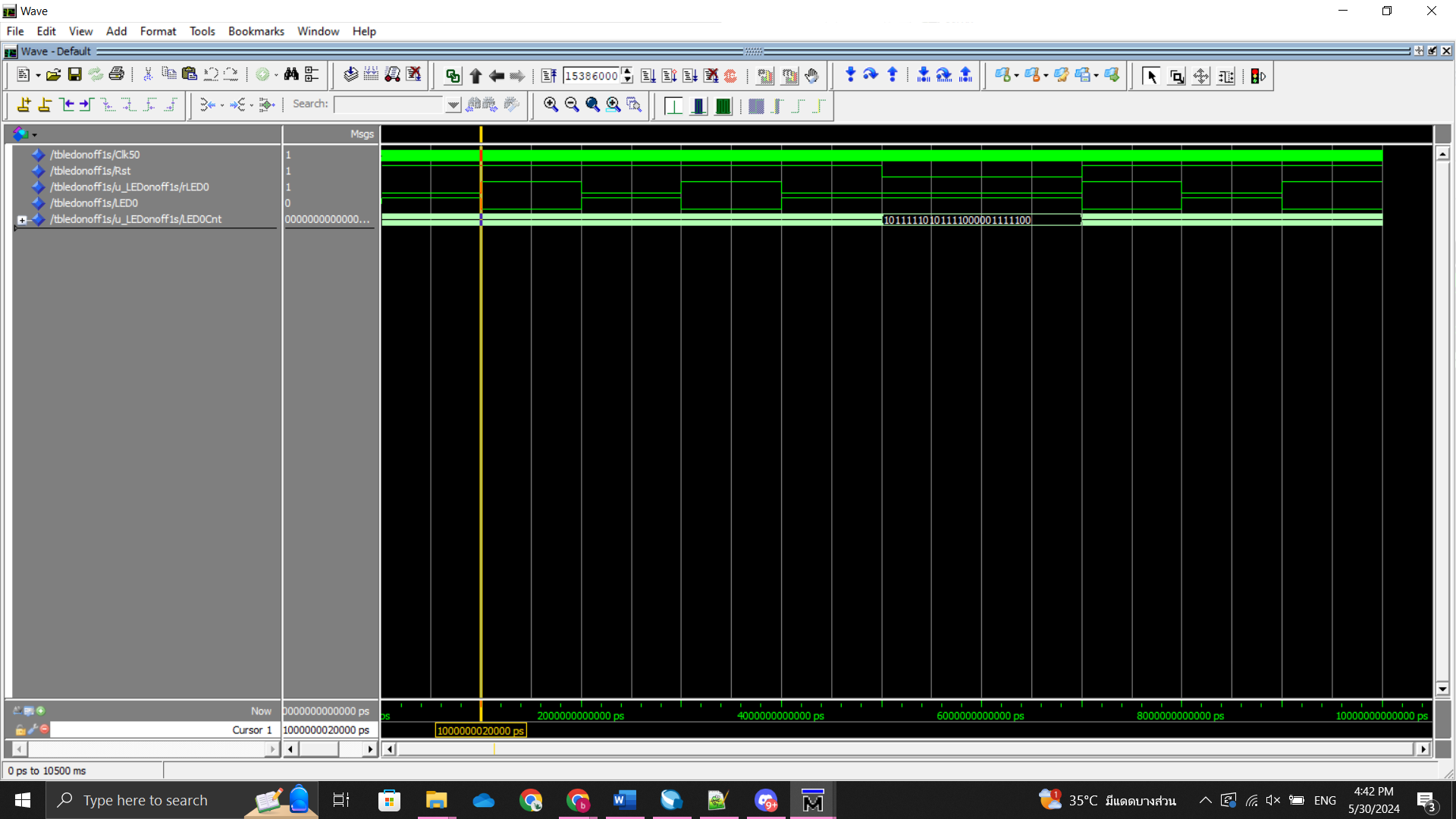


29/05/2024

* 1. เขียน LED on/off ทุกๆ 1 second + simulation

- delay in VHDL

CLK+Counter?



30/05/2024

* 1. เขียน LED on/off ทุกๆ xxx second (ปรับได้จาก Tact SW) + simulation

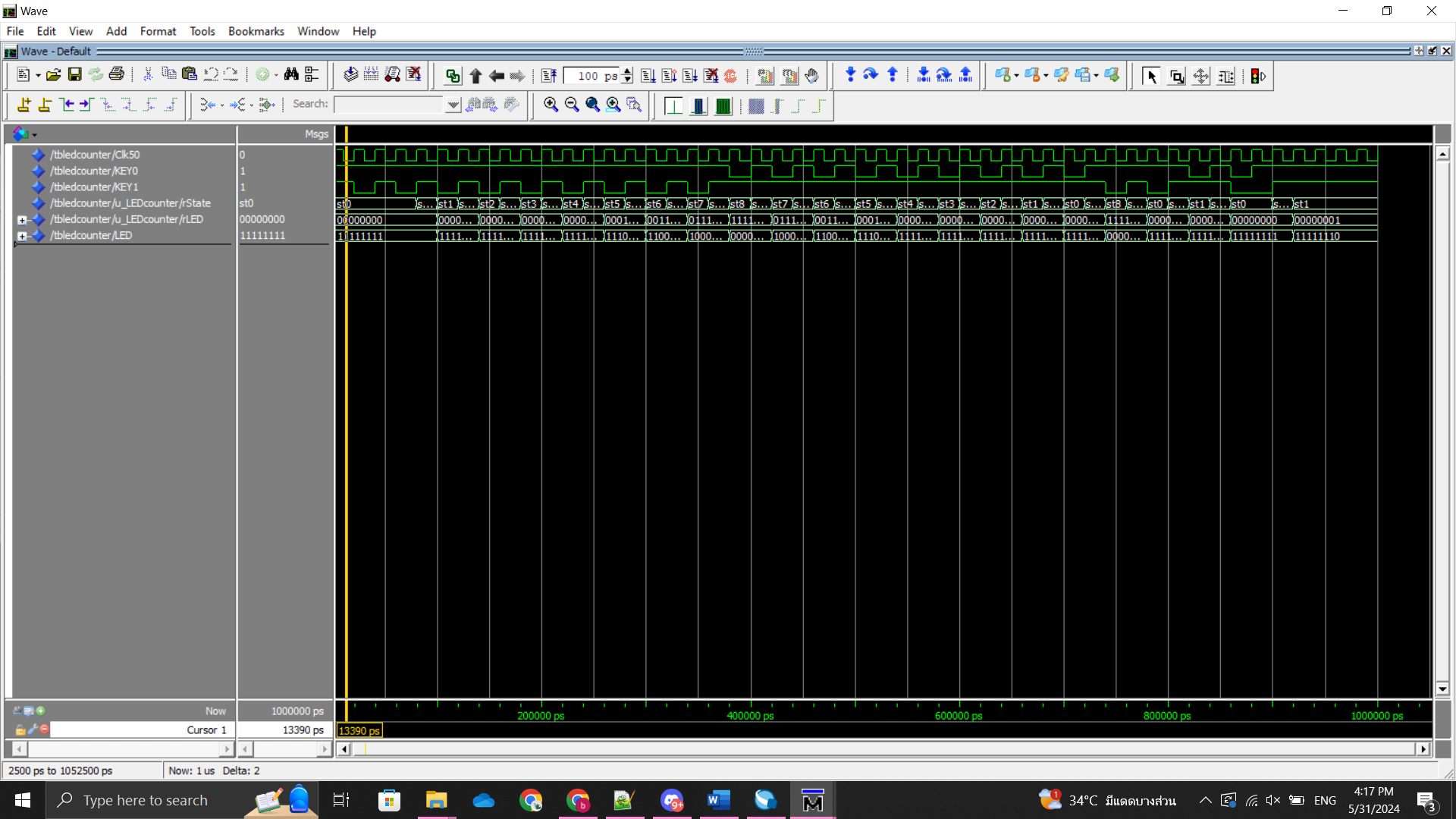
Counter use general purpose LEs 🡪 PLL more accuracy & efficiency ?

A computer screen shot of a black and green screen

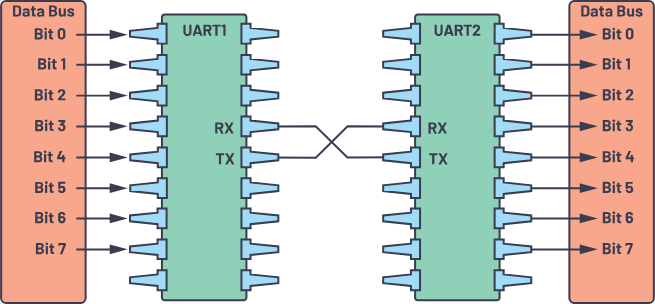
Description automatically generated

31/05/2024

* 1. กด tact SW แล้ว LED นับเพิ่ม , กด tact SW อีกตัว นับลง LED น่าจะมี 8 ตัว



UART





* Asynchronous Communication : การสื่อสาร UART ไม่ใช้สัญญาณนาฬิกาในการส่งข้อมูลระหว่างอุปกรณ์ ทั้งตัวส่งและตัวรับต้องใช้ Buad Rate ที่เหมือนกัน
* Data Framing : UART ประกอบด้วยบิตเริ่มต้น (start bit) บิตข้อมูล (data bits) (5-9 บิต) บิตพาริตี้ (parity bit) และบิตหยุด (stop bits) หนึ่งบิตหรือมากกว่า
* Baud Rate : ความเร็วในการส่งข้อมูล โดยปกติวัดเป็นบิตต่อวินาที (bps) อัตราบอดทั่วไป ได้แก่ 9600, 19200, 38400, 57600, และ 115200

04-06/06/2024

* 1. กด tact SW แล้ว LED นับเพิ่ม , กด tact SW อีกตัว นับลง นับแบบ binary 0x00 -> 0x01 -> 0x02 -> 0xFF (นับเมื่อปล่อยปุ่ม)

- Block diagram

A diagram of a computer program

Description automatically generated

- oneshot

A one-shot is a circuit that produces a stable output (logic 1 or 0) until a trigger (+ or - edge) occurs. The trigger will cause the one-shot to produce a quasi-stable output for a time period determined by the circuit configuration. After the specified period of time, the output returns to the stable state.

A quasi-stable state, on the other hand, refers to a state that appears stable for a certain period of time but is not a true equilibrium state. In a quasi-stable state, the system may eventually move to a different state if the conditions change significantly.

- Signal name start with

i :    Module to Module Signal  
w :    Module Internal Signal & Resister  
k :    Constant  
s :    State  
v :    variable

- CLK\_DIVIDER

ใช้สร้างสัญญาณนาฬิกาที่ความถี่ 1 kHz โดยการใช้ Counter นับสัญญาณนาฬิกาที่ความถี่ 50 MHz

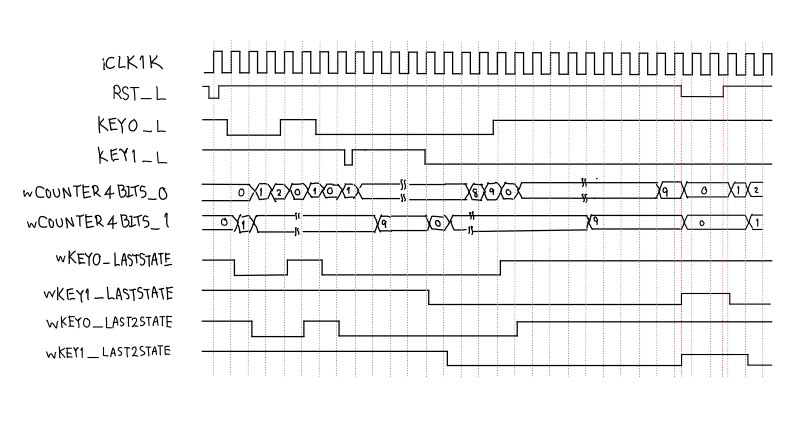
- DEBOUNCE

ทำการ Debounce ปุ่ม KEY0 และ KEY1 โดยการใช้ Counter นับสัญญาณปุ่มดังกล่าวตามขอบขาขึ้นของสัญญาณนาฬิกาความถี่ 1 kHz (1 ms) โดยเปรียบเทียบกับสัญญาณก่อนหน้าที่เก็บไว้ใน Register โดย Counter นับขึ้นเมื่อสัญญาณเหมือนเดิม และเมื่อนับได้ถึง 10 รอบ จึงถือว่าสัญญาณนั้นเกิดขึ้นจริง

- LED\_CONTROL

ควบคุมการแสดงผลของ LED โดยแสดงตาม Counter ที่นับขึ้นเมื่อ KEY1 ถูกกด และ นับลงเมื่อKEY0 ถูกกด

- Timimg Diagram



A graph paper with lines and numbers

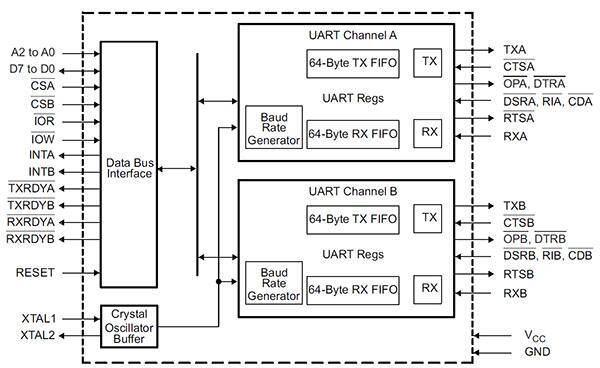
Description automatically generated with medium confidence

- Process (xxx,yyy)

Process can be sensitive to changes in certain signals, which means it will execute whenever any of these signals change. Processes are only executed when necessary, making the simulation more efficient.

07/06/2024

เขียน UART ???



UART (Universal Asynchronous Receiver-Transmitter) เป็นโปรโตคอลสำหรับการสื่อสารข้อมูลแบบอนุกรม โดยทั่วไป UART ใช้การเชื่อมต่อแบบ full-duplex หมายถึงสามารถส่งและรับข้อมูลได้พร้อมกัน UART มีการส่งและรับข้อมูลแบบ asynchronous - การส่งข้อมูล (Tx):

ข้อมูลที่จะส่งจะถูกส่งในรูปแบบอนุกรมทีละบิต

การส่งข้อมูลเริ่มต้นด้วย "Start Bit" ที่มีค่าต่ำ (0)

ตามด้วย "Data Bits" จำนวน 8 บิต (หรือมากกว่านั้นขึ้นอยู่กับการตั้งค่า)

หลังจากนั้นจะมี "Parity Bit" (ถ้ามีการใช้) สำหรับตรวจสอบความถูกต้องของข้อมูล

ปิดท้ายด้วย "Stop Bit" ที่มีค่าสูง (1) เพื่อบอกว่าแพ็กเกจข้อมูลสิ้นสุดลงแล้ว

- การรับข้อมูล (Rx):

เมื่อตรวจพบ "Start Bit" (ค่า 0) มันจะเริ่มการรับข้อมูล

อ่านค่า "Data Bits" และบันทึกใน buffer

ตรวจสอบ "Parity Bit" เพื่อความถูกต้องของข้อมูล (ถ้ามีการใช้)

"Stop Bit" จะบอกว่าแพ็กเกจข้อมูลสิ้นสุดแล้ว

USB PHY UART ?

1. NIOS soft CPU

Nios II is soft processor core designed by Intel (formerly Altera) for use in their FPGA devices.

The Nios II is a microprocessor that is implemented using the programmable logic of an FPGA.

* Open Qsys (Platform Designer) from the Tools menu.
* Add a Nios II processor to the design.
* Add necessary peripherals like on-chip memory, JTAG UART, and PIO (Parallel I/O)
* Connect all components together
* Generate the system in Qsys and integrate it into the main Quartus project.
* I/O is a broad term referring to any input/output operations.
* GPIO refers to versatile, general-purpose digital pins on a microcontroller or FPGA.
* PIO is a specific component in FPGA designs for creating parallel I/O interfaces, often used in Nios II processor systems to control external digital devices like LEDs and switches.

PIO component

(Parallel I/O)

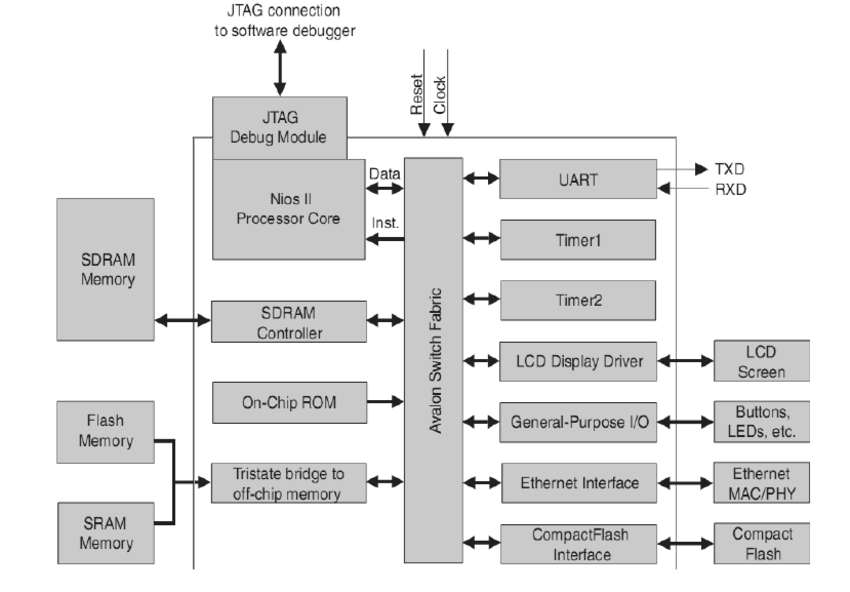
Nios II processor

(Microcontroller)

External Devices

FPGA I/O

Avalon MM



10-11/06/2024

PB0 H/L

1. โครงสร้างภายในของพอร์ต I/O บน ATmega328p

แต่ละพอร์ต I/O บน ATmega328p มีโครงสร้างภายในที่ประกอบด้วยวงจรควบคุมหลายส่วน เช่น:

Data Direction Register (DDR): ควบคุมพอร์ตว่าเป็นอินพุตหรือเอาต์พุต

PORT Register: ใช้ในการตั้งค่าเอาต์พุต ถ้าพอร์ตถูกตั้งเป็นเอาต์พุต

PIN Register: ใช้ในการอ่านค่าของพอร์ต ถ้าพอร์ตถูกตั้งเป็นอินพุต

1. การตั้งค่าพอร์ตเป็นเอาต์พุต (Output)

พอร์ต PB0 ตั้งค่าให้เป็นเอาต์พุต โดยการตั้งค่า DDRB (Data Direction Register for Port B) ค่า Register DDRB ในตำแหน่งบิต 0 (PB0) จะถูกตั้งค่าเป็น 1

1. การส่งสัญญาณจากพอร์ตเอาต์พุต

หลังจากที่พอร์ต PB0 ถูกตั้งค่าเป็นเอาต์พุตแล้ว การส่งสัญญาณไป PB0 จะเกี่ยวข้องกับการตั้งค่า PORTB Register ในตำแหน่งบิต 0:

ถ้าเราตั้งค่า PORTB Register บิต 0 เป็น 1 วงจรภายในจะเชื่อมต่อขา PB0 กับ Vcc

ถ้าเราตั้งค่า PORTB Register บิต 0 เป็น 0 วงจรภายในจะเชื่อมต่อขา PB0 กับ GND

1. การกำหนดให้ output ที่ PB0 เป็น HIGH/LOW

DDRB : Address 0x04 (0x24)

* 0x04: ที่อยู่ของ DDRB ใน Data Space
* 0x24: ที่อยู่ของ DDRB ใน I/O Memory Space

PORTB : Address 0x05 (0x25)

* 0x05: ที่อยู่ของ PORTB ใน Data Space
* 0x25: ที่อยู่ของ PORTB ใน I/O Memory Space

Data Space: ใช้สำหรับเก็บข้อมูลที่โปรแกรมต้องการในการประมวลผล เช่น ตัวแปร, ข้อมูลชั่วคราว เป็นต้น

I/O Memory Space: ใช้สำหรับเก็บข้อมูลที่ใช้ในการสื่อสารกับอุปกรณ์ภายนอก เช่น การควบคุมพอร์ต I/O, การสื่อสารผ่าน UART, SPI, I2C เป็นต้น

เมื่อเขียนโปรแกรมสำหรับ ATmega328P จะใช้ทั้งสองที่อยู่เพื่อเข้าถึง โดยไม่มีผลต่อการทำงานของโปรแกรม

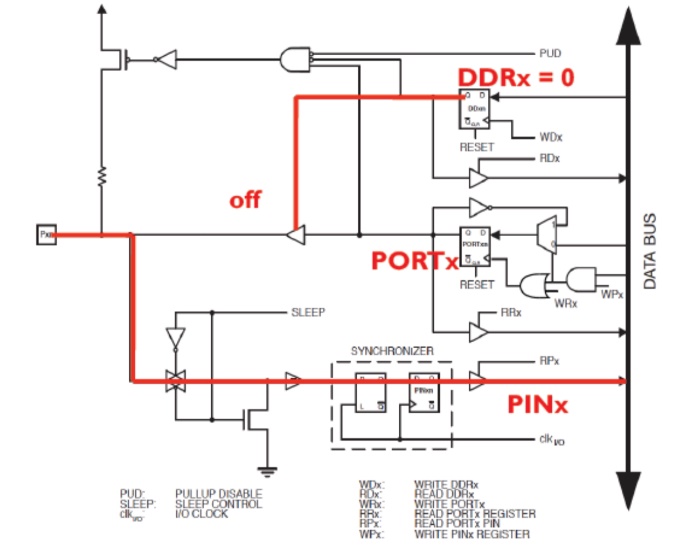
DDRB Register Bit0 🡪 ‘1’ (0x04 / 0x24 Bit0 🡪 ‘1’)

PORTB Register Bit0 🡪 ‘1’ (0x05 / 0x25 Bit0 🡪 ‘1’)

A diagram of a device

Description automatically generated

PORTx = Output



PORTx = Input

2.1 ลองสร้างโปรเจคที่รัน nios ดู console ออกทาง usb blaster jtag

A white table with blue text

Description automatically generated

* Flash Memory: ใช้เก็บโปรแกรมและข้อมูลถาวร

RAM (Random Access Memory): ใช้เก็บข้อมูลชั่วคราวระหว่างการประมวลผล

EEPROM: ใช้เก็บข้อมูลที่ต้องการให้คงอยู่แม้ไม่มีไฟเลี้ยง เช่น การตั้งค่าต่าง ๆ

* MAX 10 FPGA devices contain on-chip flash that is segmented into two part:
* Configuration Flash Memory (CFM)—store the hardware configuration data for MAX

10 FPGA

* User Flash Memory (UFM)—stores the user data or software applications
* Configuration and Booting Flow for Option 3 (Boot BSP)

CFM จะทำการ Initialized ในขณะที่ FPGA กำลัง Config อยู่ ทำให้หลังจาก reset แล้ว Nios II อยู่ในสภาพพร้อมทำงาน ซึ่ง Boot Option นี้ จะไม่รองรับการทำงานแบบ Dual Boot ( มี 2 Image ใน FPGA 1 ตัว ) Software ทั้งหมดจะถูกรันบน OCRAM เท่านั้น ในขั้นตอนการออกแบบ จะต้อง check Initialize flash content ใน RAM หรือ หากมีการแก้ไข Software ใหม่ จะต้องทำการ Compile FPGA ทุกครั้ง ไม่เหมาะสำหรับการ Debug เพราะใช้เวลา Compile ค่อนข้างนาน

"image" หมายถึงไฟล์หรือโค้ดที่ประกอบไปด้วยโปรแกรมหรือซอฟท์แวร์ที่ต้องการรันบนโปรเซสเซอร์ Nios II

A diagram of a software process

Description automatically generated

nios2-terminal

12/06/2024

* 1. ใช้ IP ของ altera รับค่าจาก SW -> LED ON/OFF
* IP
* onchip\_memory2\_0

Initialize memory content

คือการกำหนดค่าและจัดเตรียมหน่วยความจำ (เช่น RAM หรือ Flash) เพื่อให้สามารถเก็บและเรียกใช้รหัสโปรแกรมได้เมื่อบูตระบบ ซึ่งจะต้องกำหนดไฟล์ที่ใช้ในการเริ่มต้นหน่วยความจำ เช่น .hex

Enable non-default initialization file

กำหนดไฟล์ที่ต้องการใช้ในการเริ่มต้นหน่วยความจำ

* nios2\_gen2\_0

Reset vector memory: เป็นพื้นที่ที่เก็บที่อยู่ของโค้ดที่ต้องการให้ไมโครคอนโทรเลอร์เริ่มต้นการทำงานหลังจากถูกรีเซ็ต ซึ่งใช้สำหรับกำหนดการเริ่มต้นของโปรแกรมหลักของระบบ

Exception vector memory: เป็นพื้นที่ที่เก็บที่อยู่ของโค้ดที่ต้องการให้ไมโครคอนโทรเลอร์ทำงานเมื่อเกิดข้อผิดพลาดของระบบ ซึ่งใช้สำหรับการจัดการข้อผิดพลาดและกู้คืนสถานะของระบบให้กลับสู่สถานะปกติได้

* jtag\_uart\_0
* pio\_0

Bidirectional (tristate) ports : In this mode, each PIO bit shares one device pin for driving

and capturing data. The direction of each pin is individually selectable. To tristate an FPGA I/O pin, set the direction to input.

Input ports only In this mode the PIO ports can capture input only.

Output ports only In this mode the PIO ports can drive output only.

Both input and output ports In this mode, the input and output ports buses are

separate, unidirectional buses of n bits wide.

* Avalon Memory-Mapped Interface

Avalon-MM Interface ใช้ในการสื่อสารระหว่าง master และ slave ผ่านการ mapping memory

memory-mapped: การสื่อสารระหว่าง master และ slave ผ่านที่อยู่หน่วยความจำ (memory address)

สามารถใช้ในการเชื่อมต่อกับหน่วยความจำ, register-mapped peripherals และอื่นๆ.

มีสัญญาณควบคุมเช่น read, write, address, และ data

Data Master เป็นช่องทางการเข้าถึงข้อมูล (Data Path) ของ Nios II Processor ซึ่งใช้ในการอ่านและเขียนข้อมูลไปยังอุปกรณ์หรือหน่วยความจำอื่น ๆ บน FPGA

ตัวอย่าง: การอ่านข้อมูลจากหน่วยความจำหรือการส่งข้อมูลไปยังพอร์ต GPIO

Instruction Master เป็นช่องทางการเข้าถึงคำสั่ง (Instruction Path) ของ Nios II Processor ซึ่งใช้ในการดึงคำสั่งจากหน่วยความจำเพื่อประมวลผล

ตัวอย่าง: การดึงคำสั่งโปรแกรมจาก Flash Memory เพื่อให้ CPU ประมวลผล.

