บทที่ 3
สถาปัตยกรรมและการออกแบบ FPGA

3.1 บทนำ

ในปัจจุบันงานอุตสาหกรรมทางด้านอิเล็กทรอนิกส์ได้รุดหน้าไปอย่างรวดเร็วมากและอีกทั้งประสิทธิภาพที่เพิ่มขึ้นของอุปกรณ์อิเล็กทรอนิกส์ เพื่อให้รองรับกับความต้องการของการนำมาใช้กับงานที่มีความซับซ้อนมากขึ้นเรื่อย ๆ นั้น สิ่งสำคัญที่สุดย่อมขึ้นอยู่กับประสิทธิภาพของชิ้นส่วนแต่ละชิ้นที่อยู่ภายในชิ้นส่วนอิเล็กทรอนิกส์ โดยส่วนที่สำคัญอย่างมากคือ วงจรรวมหรือที่เรียกว่า ไอซี (IC: Integrate Circuit) หรือ ไมโครชิพ ซึ่งมีการพัฒนาไปอย่างเร็วมากเช่น การพัฒนาทางด้านเทคโนโลยีของไมโครโปรเซสเซอร์ (MPU) หรือไมโครคอนโทรลเลอร์ (MCU) และอุปกรณ์ประเภทหน่วยความจำ (Memory) ซึ่งในท้องตลาดปัจจุบันมีไอซีหลากหลายชนิดให้เลือกนำมาใช้งานตามความต้องการไม่ว่าจะเป็นในเรื่องของราคาและประสิทธิภาพ

3.2 เอฟ พี จี เอ (FPGA: Field Programmable Gate Array)

FPGA เป็นอุปกรณ์ที่ถูกพัฒนาต่อมาจากอุปกรณ์ประเภท LCA(Logic Cell Array File) ของบริษัทไซลิงซ์ (Xilinx) ในปี 1985 [13] โดยมีจุดเด่นตรงที่มีประสิทธิภาพการทำงานและความหนาแน่นของจำนวนเกตสูง การใช้งานที่สะดวกสามารถกำหนดฟังก์ชันการทำงานได้ตามต้องการโดยผ่านทางซอฟท์แวร์เพื่อจะทำการโปรแกรมข้อมูลลงในชิพ FPGA ให้สามารถทำงานตามที่ออกแบบไว้โดยอัตโนมัติดังนั้นทำให้เวลาในการออกแบบและพัฒนาวงจรรวมด้วย FPGA จึงมีความสะดวกและรวดเร็วกว่าวิธีอื่นและยังมีต้นทุนค่าใช้จ่ายที่ต่ำเพราะได้ลดความเสี่ยงในการที่จะต้องแก้ไขดัดแปลงวงจรต้นแบบและการเลื่อนเวลาออกแบบผลิตภัณฑ์ลงไปด้วย

หากทำการเปรียบเทียบ FPGA กับไมโครคอนโทรลเลอร์ที่เป็นชิพที่ใช้สำหรับงานทั่วไป (General Propose Device) ซึ่งหมายถึงชิพที่ได้มีการออกแบบโครงสร้างและสถาปัตยกรรมตลอดไปจนถึงชุดคำสั่งไว้เรียบร้อยแล้ว ผู้ใช้งานเพียงทำหน้าที่เรียบเรียงชุดคำสั่งเพื่อให้ชิพทำงานตาม แต่ด้วยข้อจำกัดทางด้านสถาปัตยกรรมและชุดทำสั่ง ดังนั้นในไมโครคอนโทรลเลอร์จึงไม่ใช่คำตอบกับทุกงานเสมอไปยิ่งโดยเฉพาะการนำเอาไปใช้งานในงานประมวลผลสัญญาณที่มีความเร็วสูงมาก ๆ เช่น สัญญาณภาพ เป็นต้น ซึ่งแตกต่างจาก FPGA ที่สามารถกำหนดโครงสร้างและสถาปัตยกรรม อีกทั้งยังสามารถออกแบบให้ทำงานแบบขนานกันได้ โดยในปัจจุบันยังสามารถทำงานที่สัญญาณนาฬิกาที่สูงถึง 100 MHz เลยทีเดียว ดังนั้นประสิทธิภาพการทำงานของชิพจริง ๆ จึงขึ้นอยู่กับสถาปัตยกรรมและโครงสร้างที่ได้ออกแบบเป็นหลัก นอกจากนั้นผู้ออกแบบยังสามารถ

ออกแบบชุดคำสั่งใหม่ ๆ หรือฟังก์ชันใหม่ ๆ ได้เองเพื่อความเหมาะสมกับการใช้งาน จะเห็นได้ว่า FPGA จึงเป็นอีกทางเลือกหนึ่งของการออกแบบวงจรรวมตลอดไปจนถึงการนำไปใช้ในการออกแบบวงจรต้นแบบได้เป็นอย่างดี

3.2.1 คุณลักษณะเด่นของ FPGA

จากปัญหาในการออกแบบและพัฒนาที่ได้กล่าวมาแล้วนั้นจะเห็นได้ว่า FPGA เป็นอีกหนึ่งทางเลือกหนึ่งที่มีความนิยมนำมาใช้ในงานออกแบบและพัฒนาต้นแบบผลิตภัณฑ์ที่มีข้อดีในด้านต้นทุนการออกแบบและความสะดวกในการพัฒนา ซึ่งในปัจจุบันมีบริษัทผู้ผลิตชิพหลายรายเช่น Xilinx, Altera [14] หรือ Actel [15] เป็นต้น ซึ่งหากแบ่งตามส่วนแบ่งทางการตลาดแล้ว Xilinx จะมีส่วนแบ่งทางการตลาดสูงที่สุด โดยที่แต่ละบริษัทก็จะผลิตชิพ FPGA ที่มีขนาดและราคาที่ต่างกันออกไปส่วนมากจะถูกเรียกชื่อตามโครงสร้างและขนาดความจุของชิพ FPGA ซึ่งถูกวัดด้วยค่าจำนวนเทียบเท่าเกต (Equivalent Gate) ซึ่งจะมีตั้งแต่น้อยกว่า 10,000 เกต ไปจนถึงมากกว่า 5 ล้านเกต ขึ้นอยู่กับโครงสร้างที่นำมาผลิตชิพ เช่นชิพของบริษัท Xilinx ตระกูล 95xx จะเป็นเทคโนโลยีที่มีโครงสร้างจาก EEPROM ชิพตระกูลนี้จะมีขนาดความจุไม่เกิน 10,000 เกต ในขณะที่ชิพตระกูล spatan จะมีความจุมากถึง 4 แสนเกต และตระกูล Virtex ที่มีโครงสร้างแบบ SRAM จะมีความจุมากถึง 4 ล้านเกตเลยทีเดียวสามารถนำมาออกแบบวงจรรวมที่มีขนาดใหญ่ได้อย่างสบาย ๆ

3.3 การออกแบบวงจรดิจิตอล

รูปแบบการออกแบบวงจรรวมสามารถแบ่งออกได้สองประเภทหลัก ๆ ได้ดังนี้

3.3.1 การออกแบบโดยใช้การวาดผังวงจร (Schematic Design)

ในอดีตการออกแบบวงจรรวมดิจิตอลจะเป็นการออกแบบวงจรในระดับลอจิกเกต (Logic level) โดยใช้โปรแกรมช่วยวาดผังวงจรซึ่งการออกแบบวงจรด้วยวิธีนี้จะค่อนข้างใช้เวลามากในการออกแบบ เนื่องจากการออกแบบทุก ๆ อย่างนั้นผู้ออกแบบจะต้องออกแบบเองทั้งหมดซึ่งจะไม่เหมาะกับงานออกแบบวงจรที่มีความซับซ้อนสูง จึงเป็นข้อจำกัดอย่างหนึ่งในการออกแบบในวิธีนี้

3.3.2 การออกแบบโดยใช้ภาษาระดับสูง (High Description Language)

ด้วยเทคโนโลยีการออกแบบวงจรดิจิตอลได้พัฒนาสูงขึ้นและการพัฒนาภาษาในการออกแบบวงจรดิจิตอลพร้อมกับกระบวนการออกแบบแนวใหม่ที่เป็นการเขียนด้วยภาษาอธิบายฮาร์ดแวร์ (HDL: Hardware Description Language) [16] มาใช้ในการออกแบบวงจรซึ่งจะมีอยู่สองภาษที่นิยมใช้คือภาษา Verilog และ VHDL โดยโครงสร้างของภาษา Verilog จะคล้ายกับภาษาซี ส่วนภาษา VHDL จะคล้ายกับภาษา Pascal

การออกแบบโดยใช้ภาษา VHDL จะเป็นกระบวนการออกแบบที่เราเรียกว่า Top Down Design โดยจะอาศัยซอฟต์แวร์ในการออกแบบซึ่งเรียกว่า EDA (Electronic Design Automation) ซึ่งจำลองการทำงานจากโค้ดที่เขียนขึ้น (HDL Simulations) และโค้ดที่ผ่านการจำลองการทำงานจะถูกนำไปสังเคราะห์ (Synthesis) ให้เป็นวงจรรวมเกต (Logic Level) ที่สามารถนำไปใช้งานได้ซึ่งอุปกรณ์ที่รองรับเทคโนโลยีดังกล่าวได้แก่ CPLD (Complex Programmable Logic Device) หรือ FPGA (Field Programmable Gate Array) หรือแม้กระทั่งนำไปออกแบบในระดับเอสิก (ASIC: Application Specific Integrate circuit) จะเห็นได้ว่าการออกแบบโดยใช้ภาษา VHDL จะทำให้ผู้ออกแบบสามารถออกแบบวงจรที่มีความซับซ้อนได้อย่างสะดวกรวดเร็วขึ้น และยังสามารถนำกลับมาใช้ใหม่ได้ หรือหากต้องการเปลี่ยนแปลงเทคโนโลยีของวงจรที่ออกแบบไว้ก็เพียงสังเคราะห์วงจรใหม่ก็จะได้วงจรในเทคโนโลยีใหม่ทันที

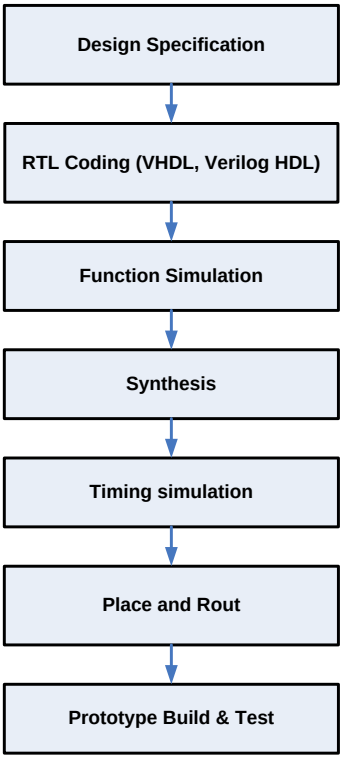
3.4 การเขียนภาษา HDL

จากที่ได้กล่าวมาแล้วว่าแนวโน้มในปัจจุบันการออกแบบวงจรดิจิตอลขนาดใหญ่จะหันไปสู่การใช้ภาษาระดับสูงมากขึ้นเนื่องจากเป็นภาษาที่มีโครงสร้างและสามารถทำความเข้าใจได้ง่าย อีกทั้งยังสามารถนำกลับมาใช้งานได้อีก และยังสามารถรองรับได้กับหลายเทคโนโลยีอีกด้วย โดยโค้ดที่เขียนขึ้นมานั้นจะไม่ขึ้นอยู่กับเทคโนโลยีนั้น ๆ หรือมีผลเพียงส่วนน้อยเท่านั้น นอกจากนั้นยังมีซอฟต์แวร์ที่มีประสิทธิภาพอีกมากมายสำหรับใช้ในการสังเคราะห์วงจร ในปัจจุบันนี้ภาษาที่ใช้อธิบายฮาร์ดแวร์จะนิยมใช้ได้แก่ภาษา VHDL และ Verilog HDL ซึ่งเป็นมาตรฐานในทางอุตสาหกรรม ซึ่งขั้นตอนในการออกแบบและพัฒนาแสดงดังรูปที่ 3.1

3.4.1 การกำหนดลักษณะเฉพาะในการออกแบบ (Design Specifications)

จากรูปที่ 3.1 แสดงขั้นตอนการออกแบบโดยใช้ภาษา HDL จะเริ่มต้นที่การเขียนโค้ดหลังจากที่ได้มีการออกแบบโครงสร้างไว้เป็นที่เรียบร้อยแล้ว เช่น การแบ่งวงจรทั้งหมดออกเป็นบล็อกย่อย ๆ เพื่อให้ง่ายต่อการออกแบบตามวิธีการ Top-down Design Methodology ซึ่งในแต่ละบล็อกย่อยจะไม่มีความซับซ้อนมากจนเกินไป และง่ายต่อการสังเคราะห์วงจรในภายหลัง จากนั้นจึงทำการเขียนภาษาอธิบายพฤติกรรมของบล็อกเหล่านั้น (Behavioral Description) ด้วยภาษา HDL แล้วจึงนำมารวมเข้าด้วยกันโดยการเรียกใช้ส่วนประกอบ และทำการเชื่อมต่อสัญญาณระหว่างบล็อกย่อย ๆ เข้าด้วยกัน จะเห็นได้ว่าการอธิบายวงจรในระดับนี้จึงเป็นการอธิบายเชิงโครงสร้าง (Structural Description) การเขียนโค้ดภาษา HDL จะมีลักษณะคล้ายกับภาษาระดับสูงอื่น ๆ สามารถใช้โปรแกรมจำพวก Text Editor ทั่ว ๆ ไปได้แต่อย่างไรก็ตามก็มีซอฟต์แวร์หลาย ๆ ตัวที่สนับสนุนการเขียนโค้ดดังกล่าวอยู่หลาย ๆ ตัวได้แก่ ISE ของบริษัท Xilinx, Max plus II ของบริษัท

Altera หรือ Moselsim ของบริษัท Model Technology, Inc เป็นต้น ซึ่งซอฟต์แวร์ดังกล่าวจะสามารถสังเคราะห์และคอมไพล์ได้ขึ้นอยู่กับผู้ใช้งานว่าเลือกใช้ FPGA ของผู้ผลิตรายใด



รูปที่ 3.1 ขั้นตอนการออกแบบวงจรด้วยภาษา HDL

3.4.2 การจำลองการทำงานของฟังก์ชัน (Functional simulation)

ขั้นตอนที่สำคัญอีกขั้นตอนหนึ่งในการออกแบบวงจรรวมก็คือ การตรวจสอบว่าแบบจำลอง (Model) ในภาษา HDL ที่ได้สร้างขึ้นนั้นสามารถทำหน้าที่ตามที่เราได้กำหนดไว้หรือไม่ ซึ่งสามารถตรวจสอบได้โดยการจำลองการทำงาน เพื่อดูการทำงานของวงจรที่สร้างขึ้น (Behavioral Simulation) จึงต้องเขียนโมดูลพิเศษขึ้นมา เพื่อใช้ในการตรวจสอบพฤติกรรมการทำงานของโมดูลที่จะใช้งาน (Design Under Test) ซึ่งโมดูลที่ใช้ทดสอบนี้เรียกว่า HDL Test bench โดยจะเป็นการกำหนดรูปแบบของสัญญาณขาเข้า (Input Signal) เพื่อที่จะดูการทำงานของโมดูลที่สร้างขึ้นและนำไปเปรียบเทียบกับสัญญาณที่บ้อนให้กับโมดูลว่าการทำงานนั้นเป็นไปตามที่คาดหวังไว้หรือไม่

3.4.3 การสังเคราะห์วงจร (Synthesis circuit)

เมื่อทำการตรวจสอบโมดูลที่สร้างขึ้นเป็นที่แน่ใจแล้วว่า วงจรที่สร้างขึ้นมานั้นสามารถทำงานได้อย่างถูกต้องตามต้องการ ขั้นตอนต่อไปก็คือการนำเอาโค้ดต้นแบบดังกล่าวไปทำการ

สังเคราะห์วงจร เพื่อให้ได้เน็ตลิสต์ในระดับเกตออกมา เครื่องมือที่ใช้ในการสังเคราะห์โค้ดภาษา HDL (HDL Synthesis Tool) เช่น ซอร์ฟต์แวร์ ISE ของบริษัท Xilinx ซึ่งสามารถสังเคราะห์วงจรสำหรับ FPGA ได้และสามารถเลือกไลบราลีให้เหมาะสมกับชิพที่เลือกใช้งานได้เช่น Vertec, spatan เป็นต้น

3.4.4 การจำลองทางเวลา (Timing simulation)

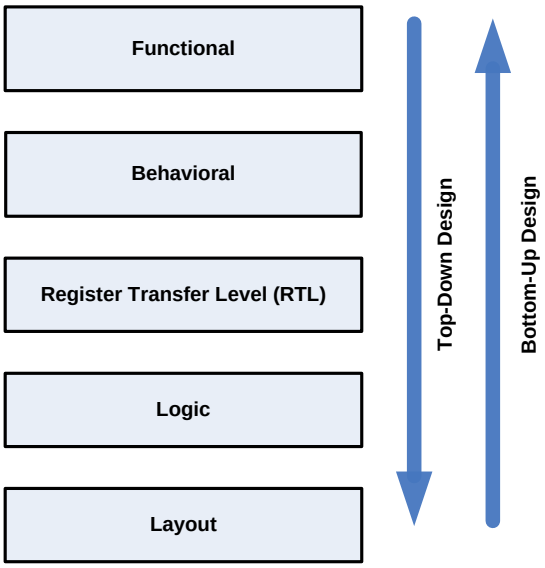
การจำลองการทำงานในระดับพฤติกรรม หรือหลังจากการสังเคราะห์วงจร แต่ยังไม่ได้ผ่านการเชื่อมวงจรเข้ากับโมดูลชุดอื่น ๆ จะเป็นการตรวจสอบความถูกต้องในระดับสัญญาณลอจิกเท่านั้น แต่ยังไม่ได้ให้รายระเอียดที่แน่นอนเกี่ยวกับเรื่องเวลา (Timing) มากนัก เช่น ความล่าช้าของสัญญาณตามเส้นทางต่าง ๆ ภายในวงจร ระยะคาบของสัญญาณเวลาที่น้อยที่สุดเท่าที่จะสามารถใช้กับวงจรได้ เป็นต้น เพราะจะต้องผ่านขั้นตอนการเชื่อมเส้นทางก่อน ดังนั้นการจำลองการทำงานทางเวลาจึงเป็นการตรวจสอบการทำงานของวงจรอีกครั้ง

3.4.5 การวาง และการเชื่อมเส้นทาง (Place and Route)

ผลที่ได้จากการสังเคราะห์นั้นจะอยู่ในรูปแบบเน็ตลิสต์ในระดับเกต โดยสามารถนำไปผ่านขั้นตอนการแปลงให้เป็นลอจิกและรวมถึงการวางและการเชื่อมเส้นทางภายในอุปกรณ์ FPGA

3.5 การออกแบบจากบนลงล่างและการออกแบบจากล่างขึ้นบน

ในการออกแบบวงจรบน FPGA จะมีวิธีการหรือกระบวนการในการออกแบบอยู่ 2 วิธีคือ การออกแบบจากบนลงล่าง (Top-Down Design) และการออกแบบจากล่างขึ้นบน (Bottom-Up Design) ดังรูปที่ 3.2 ซึ่งการออกแบบทั้งสองวิธีต่างก็มีข้อดีข้อเสียต่างกัน ในกระบวนการออกแบบจากบนลงล่าง (Top-Down Design) จะมีวิธีการมองจากระดับสูงสุดก่อนว่าวงจรที่จะออกแบบนั้นมีลักษณะอย่างไร ซึ่งเรียกระดับนี้ว่าระดับของฟังก์ชัน (Functional level) จากนั้นจึงค่อยศึกษาลึกลงไปถึงรายละเอียดของแต่ละฟังก์ชันในระดับรีจีสเตอร์ (RTL) ของแต่ละฟังก์ชันและค่อยวิเคราะห์ลึกลงไปในรายระเอียดในระดับลอจิกและในระดับสุดท้าย ซึ่งเป็นระดับล่างสุดในการออกแบบก็คือระดับผังภูมิวงจร (Layout)

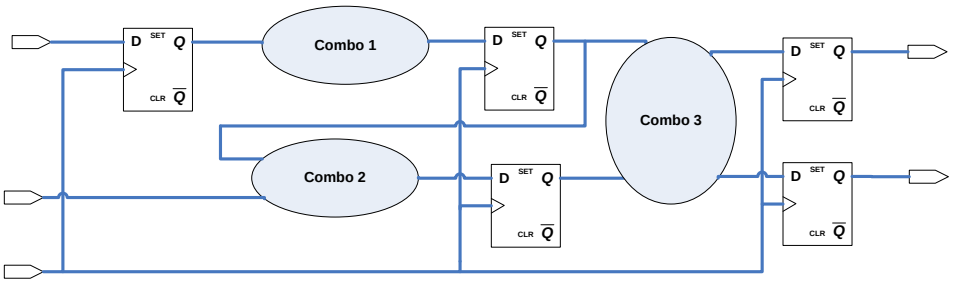


รูปที่ 3.2 กระบวนการออกแบบวงจร (Design Methodology)

ซึ่งผู้ออกแบบจะไม่ทราบรายละเอียดของวงจรที่ออกแบบ จนกว่าจะถึงระดับสุดท้ายในการออกแบบ ซึ่งจะตรงข้ามกับวิธีการออกแบบจากล่างขึ้นบน (Bottom-Up Design) ที่จะเริ่มต้นออกแบบจากระดับล่างสุดก่อนจากนั้นจึงนำแต่ละส่วนมารวมกันเป็นบล็อกในระดับบนเพื่อให้วงจรสามารถทำงานตามหน้าที่ตามฟังก์ชันนั้น ๆ ได้ กระบวนการการออกแบบลักษณะนี้ผู้ออกแบบจะทราบรายระเอียดของวงจรทั้งหมด แต่ข้อเสียของวิธีการนี้ก็คือ ผู้ที่ออกแบบจะต้องเลือกเทคโนโลยีของชิพ FPGA ก่อนที่จะเริ่มต้นออกแบบ ดังนั้นหากจำเป็นต้องเปลี่ยนชิพจำต้องทำการออกแบบใหม่ ซึ่งในกระบวนการออกแบบด้วยภาษาระดับสูงจะเป็นการออกแบบจากบนลงล่าง (Top Dow Design) โดยจะไม่ยึดติดกับเทคโนโลยีในการออกแบบวงจร เพราะการออกแบบด้วยรูปแบบภาษา HDL ซึ่งหากต้องการใช้เทคโนโลยีอะไร ก็เพียงนำไลบรารีของเทคโนโลยีมาสังเคราะห์เป็นโมเดลของวงจรในระดับล่างต่อไป

3.5.1 วงจรในระดับรีจีสเตอร์ (Register Transfer Level)

RTL เป็นการเขียนบรรยายเชิงพฤติกรรมของวงจรในระดับของรีจีสเตอร์ และวงจรคอมบิเนชั่น การออกแบบในระดับนี้ ผู้ออกแบบต้องคำนึงถึงรูปแบบการประมวลผลไทม์มิ่งการเชื่อมต่อและการเข้าจังหวะระหว่างกลุ่มวงจรคอมบิเนชันและรีจีสเตอร์ โดยจะไม่พิจารณาถึงพฤติกรรมวงจรในระดับลอจิกหรือระดับทรานซิสเตอร์โดยแสดงในรูปที่ 3.3



รูปที่ 3.3 ไดอะแกรมโครงสร้างของ Register Transfer Level (RTL)

ในการออกแบบวงจรดิจิตอลด้วย HDL ส่วนใหญ่ จะทำการเขียนบรรยายพฤติกรรมวงจรในระดับ RTL เนื่องจากมีความสะดวกในการออกแบบ และให้วงจรที่มีประสิทธิภาพมากกว่าการเขียนบรรยายในระดับลอจิก ซึ่งใช้เวลาในการออกแบบนานและหาข้อผิดพลาดลำบาก แต่วิธีการเขียนบรรยายพฤติกรรมนั้นก็ใช่ว่าจะให้ผลดีที่สุด แต่ด้วยความสะดวกในขั้นตอนการสังเคราะห์วงจร การวิเคราะห์เวลา (Timing Analysis) ซึ่งเป็นกระบวนการวิเคราะและแก้ปัญหาของค่าเวลาในวงจร รวมไปจนถึงการที่วงจรที่ถูกแยกออกเป็นบล็อกย่อย ๆ (Partitioning) เพื่อให้การสังเคราะห์วงจรได้ดีที่สุด และเข้าใจได้ง่าย ซึ่งการเขียนบรรยายพฤติกรรมจึงเป็นที่นิยมในงานที่มีความซับซ้อนสูง

3.6 ภาษาเวริล็อก (Verilog HDL)

ภาษาเวริล็อก (Verilog HDL) เป็นภาษาที่ใช้อธิบายฮาดร์แวร์ (Hardware Description Language: HDL) [17] ซึ่งใช้ในการออกแบบวงจรระบบดิจิตอล ภาษาเวริล็อกนั้นเป็นหนึ่งในสองภาษาอธิบายฮาดร์แวร์ที่นิยมใช้กันมากในการออกแบบวงจรระบบดิจิตอลในปัจจุบัน ซึ่งได้แก่ ภาษา วีเอชดีแอล (Very High Speed Integrated Circuit Hardware Description Language: VHDL) และภาษาเวริล็อก เนื่องจากภาษาเวริล็อกมีลักษณะคล้ายภาษาซี จึงทำให้วิศวกรซึ่งส่วนใหญ่นั้นมีความรู้ด้านภาษาซี อยู่แล้วสามารถเรียนรู้และเข้าใจได้ง่าย และโครงสร้างของภาษายังมีความซับซ้อนน้อยกว่าภาษาวีเอชดีแอลอีกด้วย

3.6.1 ประวัติความเป็นมา

ภาษาเวริล็อกนั้นถูกคิดค้นโดย บริษัท Gateway Design Automation ในปี 1981 ซึ่งเหตุการณ์ที่สำคัญและความเป็นมากล่าวเรียงตามลำดับได้ดังต่อไปนี้ [18]

ปี 1983 บริษัท Gateway Design Automation ได้เปิดตัวภาษาเวริล็อกนี้ซึ่งเรียกว่า “Verilog HDL” หรือเรียกง่าย ๆ ว่า “Verilog” พร้อมทั้งส่วนการจำลองการทำงานของภาษาเวริล็อก (Verilog Simulator)

ปี 1985 ภาษาเวริล็อกได้มีการปรับปรุงเพิ่มเติมและได้ออกเวอร์ชันใหม่ของส่วนการจำลองการทำงาน ซึ่งเรียกว่า “Verilog XL”

ปี 1987 ภาษาเวริล็อกเป็นที่นิยมใช้ ขนาดของวงจรที่ออกแบบนั้นเริ่มเกินความสามารถของชุดจำลองการทำงานของ Gateway Design System และบริษัทอื่น ๆ

ปี 1989 บริษัท Gateway Design System ถูกซื้อโดยบริษัท Cadence ซึ่งเป็นบริษัทผู้ผลิตซอร์ฟแวร์ในการออกแบบวงจรรวมชั้นนำของโลกในปัจจุบัน

ปี 1990 บริษัท Cadence ได้แยกภาษาเวริล็อกและส่วนการจำลองการทำงานออกจากกัน โดยได้นำภาษาเวริล็อกเปิดเผยต่อสาธารณะ และในปีนี้ผู้ใช้ภาษาเวริล็อกและบริษัทต่าง ๆ ได้ตั้งสมาคม Open Verilog International (OVI) เป็นผู้กำหนดควบคุมรายละเอียด (specification) ของภาษาเวริล็อก และได้จัดทำคู่มืออ้างอิงสำหรับภาษาเวริล็อกเวอร์ชั่น 1.0 (Verilog 1.0 Reference Manual) ขึ้นมา

ปี 1993 สมาคม OVI ได้ปรับปรุงเพิ่มเติมความสามารถของภาษาเวริล็อก เช่น ความสามารถของอาร์เรย์ และได้ออกคู่มืออ้างอิงสำหรับภาษาเวริล็อกใหม่ เป็นเวอร์ชั่น 2.0 (Verilog 2.0 Reference Manual)

ปี 1995 สมาคม OVI ได้ส่งภาษาเวริล็อกเวอร์ชั่น 2.0 ให้สมาคม IEEE (Institute of Electrical and Electronics Engineers) ซึ่งสมาคม IEEE ได้ปรับปรุงและได้กำหนดเป็นมาตรฐานใหม่ที่เรียกว่า มาตรฐาน IEEE 1394-1995

ในปี 2001 มีปรับปรุงให้เพิ่มความสามารถของภาษาเวริล็อกมาตรฐาน IEEE 1394 ทั้งด้านการสังเคราะห์วงจรและการจำลองการทำงาน อีกทั้งยังปรับปรุงให้ง่ายต่อการใช้งาน ซึ่งสมาคม IEEE กำหนดเป็นมาตรฐานใหม่ที่เรียกว่า มาตรฐาน IEEE 1394-2001

3.6.2 โครงสร้างของภาษาเวริล็อก

ภาษาเวริล็อกนั้นสามารถอธิบายพฤติกรรมของวงจรดิจิตอลได้หลายระดับ เช่น สามารถอธิบายวงจรในระดับการวาด (Layout) รีซิสเตอร์ ทรานซิสเตอร์และการเชื่อมต่อ (Wire) บนชิพไอซี (Integrated Circuit chip) ซึ่งเรียกว่า ระดับการสวิทช์ (Switch level) สามารถอธิบายการเชื่อมต่อของลอจิกเกตและฟลิบฟลอบในวงจรดิจิตอล เรียกว่า ระดับเกต (Gate level) และสามารถอธิบายการส่งข้อมูลระหว่างรีจิสเตอร์ เรียกว่า ระดับ RTL (Register Transfer Level) [16] ซึ่งการออกแบบวงจรดิจิตอลด้วยภาษาภาษาเวริล็อกส่วนใหญ่จะใช้อธิบายพฤติกรรมของวงจรดิจิตอลในระดับ RTL นี้

ภาษาเวริล็อกนั้นมีไวยกรณ์คล้ายภาษาซี เช่น จบประโยคด้วย ; หรือใช้ // เพื่อไม่ให้นำไปใช้ในการคอมไพล์ เป็นต้น โครงสร้างของภาษาเวริล็อกนั้นประกอบด้วยโมดูล (Module) ซึ่งแต่ละโมดูลนั้นสามารถรวมเข้าด้วยกันเป็นโมดูลขนาดใหญ่หรือเรียกว่า ท็อปโมดูล (Top module)

โดยแต่ละโมดูลย่อยนั้นสามารถทำงานพร้อมกัน (Concurrent) ซึ่งโครงสร้างของภาษาเวริล็อกประกอบด้วยส่วนต่าง ๆ ดังแสดงในตัวอย่างการอธิบายพฤติกรรมของ D-flip flop ได้ดังนี้

```
module dff(clk,rst,din,dout);          //  <ชื่อโมดูล> (<ชื่อพอร์ต>)
input din, clk, rst;                   //<ชื่ออินพุต>
output dout;                           //<ชื่อเอาท์พุต>
reg  dout;                                      //<ชื่อสัญญาณ>
always @(posedge clk or posedge rst)           //<รายละเอียด
โปรแกรม>
        if (rst)
                dout <= 1'b0;
        else
                dout <= din;
endmodule                              //<จบโมดูล>
```

3.7 รายละเอียดสถาปัตยกรรมของ FPGA ตระกูล Spartan III

FPGA ตระกูล Spartan III เป็น FPGA ของบริษัท xilinx ที่ใช้เทคโนโลยีในกระบวนการผลิต 90 ไมครอน [19] ที่ถูกออกแบบมาสำหรับผู้ที่ต้องการใช้งาน FPGA ที่มีความจุสูงในราคาที่ถูก โดยทำงานภายใต้สัญญาณนาฬิกาสูงสุดถึง 326 เมกกะเฮิร์ต เหมาะสำหรับงานอุตสาหกรรมโดย Spartan III จะมีความจุเกตเริ่มต้นตั้นแต่ 50,000 เกตไปจนถึง 5,000,000 เกตดังแสดงในตารางที่ 3.1 ซึ่งโครงสร้างภายในของ Spartan III ถูกพัฒนาแบบมาจาก Spartan-IIE โดยทำการเพิ่มความจุของลอจิกเกต และจำนวนอินพุต เอาต์พุตเพิ่มขึ้น ส่งผลให้ประสิทธิภาพการทำงานดีขึ้น อีกทั้งได้มีการรวมเอาโมดูลที่ใช้ในการจัดการสัญญาณนาฬิกา (DCMs) และโมดูลในการคูณที่มีอยู่ในชิพ FPGA ตระกูล Virtex™ II จำนวนขาอินพุทเอาต์พุตใช้งานของ Spartan III ขึ้นอยู่กับแพคเก็ตของชิพซึ่งจะมีจำนวนขาอินพุตเอาต์พุตตั้งแต่ 63 จนถึง 784 ขาดังแสดงในตารางที่ 3.2

ตารางที่ 3.1 จำนวนเกตและหน่วยความจำ ต่าง ๆ ภายในชิพ Spartan III แต่ละเบอร์ [19]

Device	System Gates	Logic Cells	CLB Array (one CLB = Four Slices)			Distributed RAM (bits)	Block RAM (bits)	Dedicated Multipliers	DCMs	Maximum User I/O
			Rows	Columns	Total CLBs					
XC3S50	50K	1,728	16	12	192	12K	72K	4	2	124
XC3S200	200K	4,320	24	20	480	30K	216K	12	4	173
XC3S400	400K	8,064	32	28	896	56K	288K	16	4	264
XC3S1000	1M	17,280	48	40	1,920	120K	432K	24	4	391
XC3S1500	1.5M	29,952	64	52	3,328	208K	576K	32	4	487
XC3S2000	2M	46,080	80	64	5,120	320K	720K	40	4	565
XC3S4000	4M	62,208	96	72	6,912	432K	1,738K	96	4	712
XC3S5000	5M	74,880	104	80	8,320	520K	1,872K	104	4	784

ตารางที่ 3.2 จำนวนเกตและหน่วยความจำ ต่าง ๆ ภายในชิพ Spartan III แต่ละเบอร์ [19]

	Available User I/O and Differential (Diff) I/O Pairs															
	XC3S50		XC3S200		XC3S400		XC3S1000		XC3S1500		XC3S2000		XC3S4000		XC3S5000	
	User	Diff	User	Diff	User	Diff	User	Diff	User	Diff	User	Diff	User	Diff	User	Diff
VQ100	63	29	63	29	-	-	-	-	-	-	-	-	-	-	-	-
TQ144	97	46	97	46	97	46	-	-	-	-	-	-	-	-	-	-
PQ208	124	56	141	62	141	62	-	-	-	-	-	-	-	-	-	-
FT256	-	-	173	76	173	76	173	76	-	-	-	-	-	-	-	-
FG203	-	-	-	-	221	100	221	100	221	100	-	-	-	-	-	-
FG456	-	-	-	-	264	116	333	149	333	149	-	-	-	-	-	-
FG676	-	-	-	-	-	-	391	175	487	221	489	221	-	-	-	-
FG900	-	-	-	-	-	-	-	-	-	-	565	270	633	300	633	300
FG1156	-	-	-	-	-	-	-	-	-	-	-	-	712	312	784	344

ชิพ Spartan III ใช้แรงดันทั้งหมดสามชุดได้แก่ VccInt 1.2V VccAux 2.5V และ VccIO 1.2V-3.3V

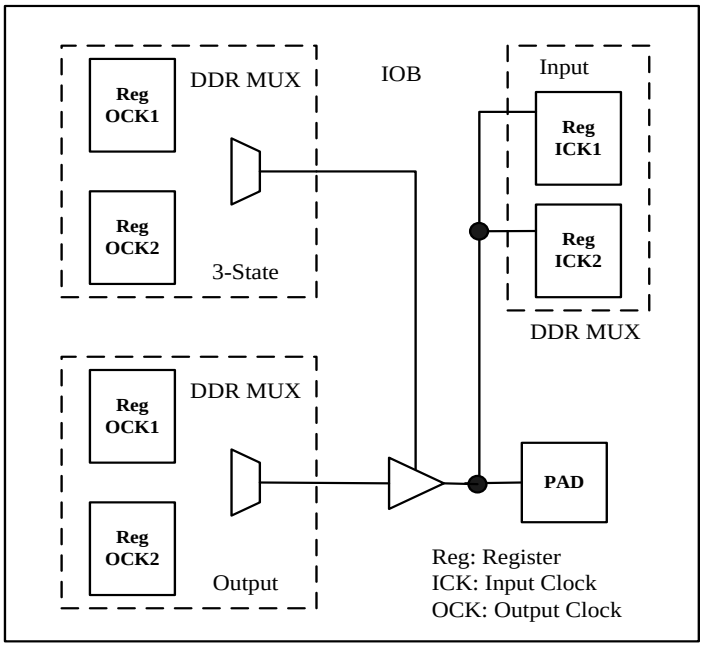
3.7.1 บล็อกอินพุตเอาต์พุต (Input/Output Block: IOB)

บล็อกอินพุตเอาต์พุต (IOB) เป็นส่วนที่ใช้ติดต่อกับวงจรภายนอก ซึ่งบล็อกอินพุตและเอาต์พุตสามารถโปรแกรมให้เป็นอินพุตหรือเอาต์พุตแบบ Single Data Rate หรือแบบ Double Data Rate (DDR) ได้ ซึ่งบล็อกอินพุตเอาต์พุตนี้ประกอบด้วย 6 รีจิสเตอร์ (Register) และ 3 DDR Multiplexers ซึ่งแสดงดังรูปที่ 3.4 โดยรีจิสเตอร์นี้สามารถโปรแกรมเป็นดีฟลิปฟลอป (D Flip-

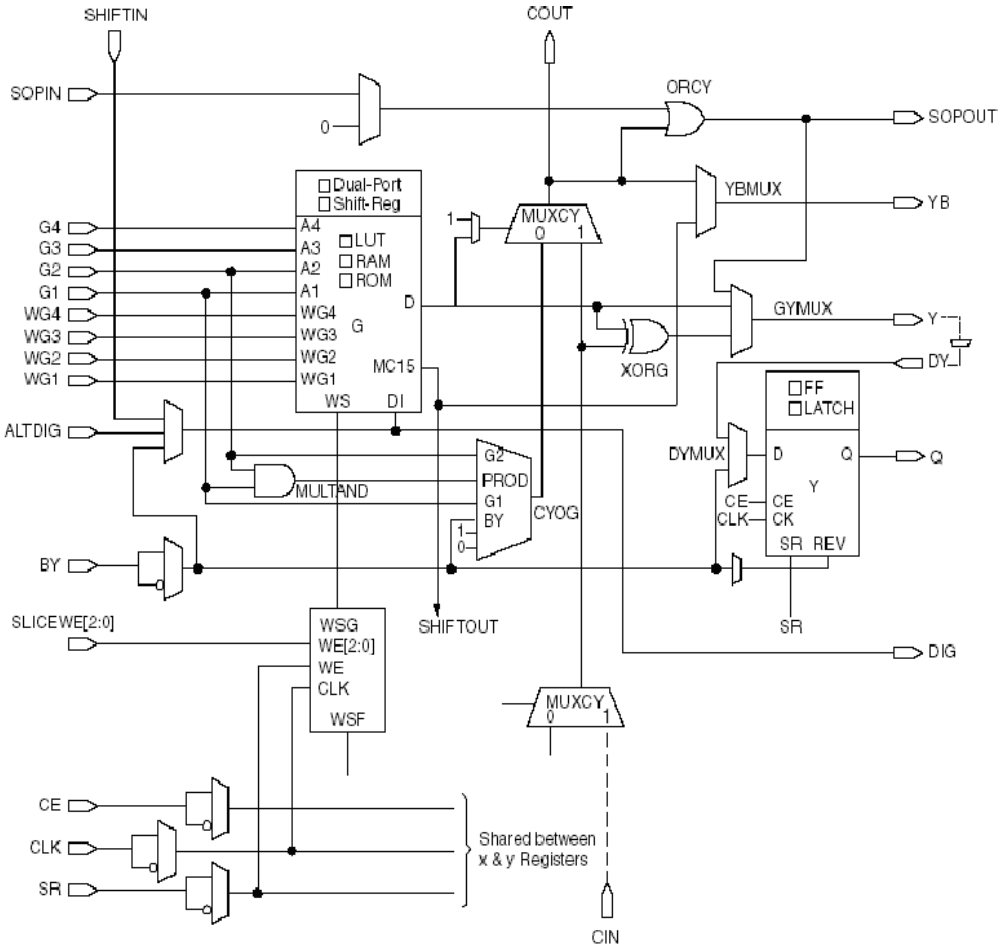
Flop) หรือแลตช์ (Latch) ได้ บล็อกอินพุตเอาต์พุตนี้ต่ออยู่กับส่วน Switch Matrix เพื่อเชื่อมต่อกับอุปกรณ์ภายในอื่น ๆ ส่วน DDR Multiplexer ใช้ในการส่งข้อมูลแบบ DDR

3.7.2 บล็อก Configurable Logic Block (CLB)

บล็อก Configurable Logic Block (CLB) ประกอบด้วย วงจรลอจิกแบบคอมไบเนชัน และแบบซิงโครนัส ซึ่ง CLB ต่ออยู่กับส่วน Switch Matrix เพื่อเชื่อมต่อกับอุปกรณ์ภายในอื่น ๆ โดยในแต่ละ CLB นั้นประกอบด้วย Slices จำนวน 4 Slices ซึ่งแต่ละ Slices มีลักษณะคล้ายกัน ซึ่ง Slice ประกอบด้วยอุปกรณ์ดังนี้ คือ ตัวสร้างฟังก์ชันสี่อินพุต (4-input Function Generators) จำนวน 2 ตัว วงจรโลจิกตัวทด (Carry Logic) วงจรบวก (Arithmetic Logic Gates) วงจรมัลติเพล็กขนาดใหญ่ (Wide Function Multiplexers) และหน่วยความจำ (Storage Elements) จำนวน 2 ตัว



รูปที่ 3.4 โครงสร้างของบล็อกอินพุตเอาต์พุต [20]



รูปที่ 3.5 โครงสร้างของ Slice [20]

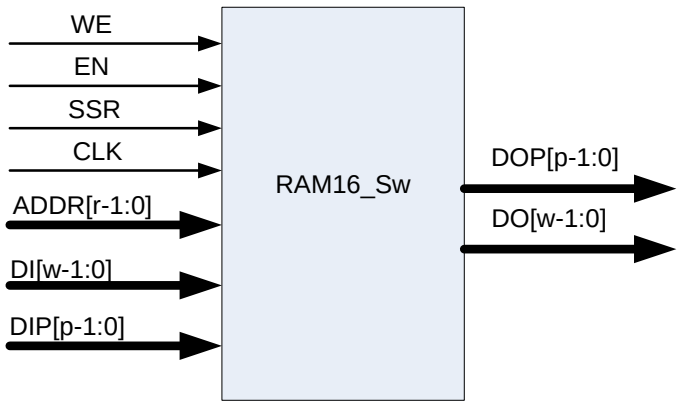
โดยตัวสร้างฟังก์ชันสี่อินพุตสามารถโปรแกรมให้เป็นตัวเปิดตารางสี่อินพุต (4-input Look-up Table) หรือโปรแกรมให้เป็นหน่วยความจำขนาด 16 บิต (16-bits of Distributed Select RAM Memory) หรือโปรแกรมให้ชิพเรจิสเตอร์ 16 บิตที่ปรับเปลี่ยนได้ (16-bit Variable Tap Shift Register) โดยเอาต์พุตของตัวสร้างฟังก์ชันนี้ต่อกับหน่วยความจำ (Storage Elements) ซึ่งเชื่อมต่อกับอุปกรณ์อื่น ๆ โดยหน่วยความจำนี้สามารถโปรแกรมเป็นดีฟลิปฟลอป (D Fip-Flop) หรือแลตช์ (Latch) ก็ได้ ซึ่งรูปที่ 3.5 นั้นแสดงโครงสร้างของ Slice

3.7.3 หน□วยความจำขนาด 18 Kb block RAM

18 Kb block RAM เป□นหน□วยความจำที่มีความเร็วสูงมาก (โดยประมาณ) 200 Mhz ที่มีอยู□ในชิพจำนวน 12 สามารถฟอร□มให□เป□น RAM หรือ ROM ที่มีขนาดต□าง ๆ กันได□รวมทั้ง FIFO ด□วย โดยรูปที่ 3 แสดงขนาด RAM Single Port ส□วนในรูปที่ 3.6 แสดงตัวอย□าง RAM แบบ Single Port ขนาดต่าง ๆ ที่สร□างจาก Block RAM แต□ละชุด

ตารางที่ 3.3 หน่วยความจำแบบพอร์ตเดี่ยวที่สร้างจากบล็อกหน่วยความจำถายใน FPGA

Organizations	Memory Depth	Data Width	Parity Width
512 x 36	512	32	4
1K x 18	1024	16	2
2K x 9	2048	8	1
4K x 4	4096	4	-
8K x 2	8192	2	-
16K x 1	16384	1	-

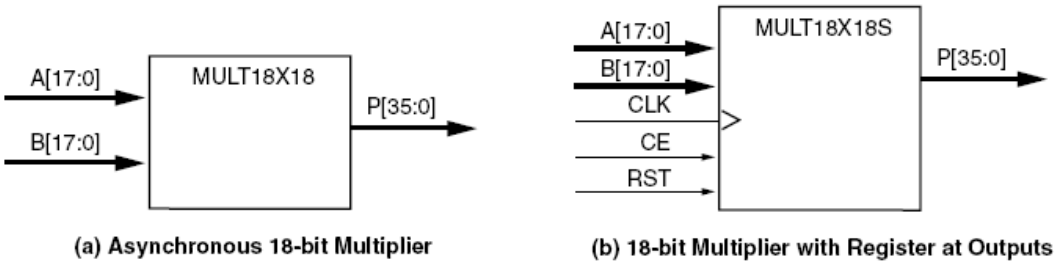


รูปที่ 3.6 โครสร้างหน่วยความจำพอร์ตเดี่ยวบน FPGA [20]

3.7.4 บล็อกวงจรคูณขนาด 18 บิต (18bits x 18bits Hardware multiplier)

บล็อกวงจรคูณขนาด 18 x 18 บิตนี้เป็นวงจรคูณที่สามารถปรับเปลี่ยนขนาดได้ โดยมีขนาดสูงสุดคือ 18 x 18 บิต และสามารถทำการคูณแบบคิดเครื่องหมาย (2's Complement Signed Multiplier)ได้ บล็อกวงจรคูณขนาด 18 x 18 บิตนี้สามารถทำงานอย่างอิสระหรือทำงานร่วมกับบล็อกหน่วยความจำ 18 กิโลบิตในการทำงานที่เรียกว่า "Multiply-Accumulator (MAC)" ซึ่งเป็นฟังก์ชันการทำงานของการประมวลผลสัญญาณดิจิตอล (Digital Signal Processing) โดยรูปที่ 3.7

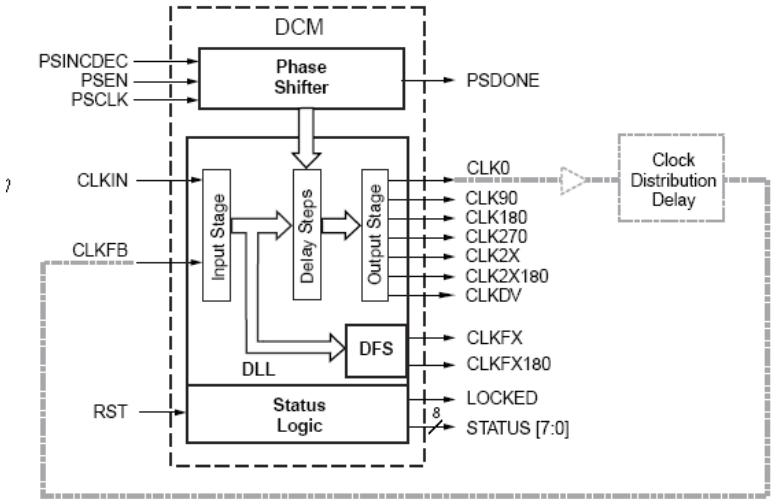
แสดงการเชื่อมต่อของบล็อกหน่วยความจำ 18 กิโลบิตและบล็อกวงจรคูณขนาด 18 x 18 บิต ซึ่งบล็อกทั้งสองจะถูกเชื่อมต่อกับ Switch Matrix



รูปที่ 3.7 โครงสร้างของวงจรคูณขนาด 18 บิต[20]

3.7.5 บล็อกจัดการสัญญาณนาฬิกา (Digital Clock Manager)

Digital Clock Manager (DCM) ในชิพตระกูล Spartan-III เป□นวงจรที่มีความสำคัญมากที่ ช่วยจัดการเกี่ยวกับสัญญาณนาฬิกา ซึ่งมีอยู□ในชิพจำนวน 4 ชุด และถือได□ว□า DCM ช□วยทำให□การออกแบบวงจรง□ายขึ้นอย□างมากเนื่องจากสามารถสร□างความถี่ต□าง ๆ ได□อย□างมากมายจากออสซิลเตอร□จากภายนอกเพียงชุดเดียว จึงไม□มีความจำเป□นใด ๆ ที่ต□องหาสัญญาณนาฬิกาจากภายนอกหลาย ๆ แหล□ง อีกต□อไป ไม□เพียงเท□านั้นสัญญาณนาฬิกาดังกล□าวยังซิงค□โครไนซ□กับสัญญาณนาฬิกาจากออสซิลเลเตอร□เดิมอีกด□วย ทำให□สามารถนำไปใช□เป□นตัวกำเนิดสัญญาณนาฬิกาความถี่ต□าง ๆ ได□โดยไม□ต□องใช□ Variable Clock จากภายนอกแต□อย□างใด DCM มีสัญลักษณ□แสดง ดังรูปที่ 3.8 ซึ่ง DCMทำงานในหน□าที่ดังต□อไปนี้



รูปที่ 3.8 โครงสร้างของ DCM [21]

- หารความถี่ (Clock Divider) เป□นวงจรหารซึ่งจะให□ความถี่เอาต□พุตเท□ากับความถี่อินพุตหารด□วยตัวเลขดังต□อไปนี้ คือ 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 9, 10, 11,12, 13, 14, 15, หรือ 16 ตามลำดับ

- สร□างความถี่สองเท□า (Clock Doublers) เป□นวงจรซึ่งจะให□ความถี่ที่เอาต□พุตจะเป□น 2 เท่าของความถี่ อินพุต

- Digital Frequency Synthesizer (DFS) เป□นวงจรซึ่งสามารถกำหนดให□ความถี่เอาต□พุตเท□ากับผลคูณของความถี่อินพุตกับอัตราส□วนของ M/D โดยที่ M = 2 ถึง 32 และ D = 1 ถึง 32 วงจรนี้นำไปใช□งาน เช□น สร□างวงจรเปลี่ยนจากการส□งข□อมูลแบบขนานเป□นอนุกรม ซึ่งต□องสร□างสัญญาณนาฬิกาสูงกว่าของเดิม เช□น 10 – 11 เท□า เป□นต□น หรืองานอื่น ๆ ที่ต□องใช□วงจรฟรีเควนซีซินธิไซเซอร□

- Delay-Locked Loop (DLL) เป□นวงจรใช□แก□ป□ญหาการเลื่อนเฟสในวงจรให□กลับมาตรงตามเฟสที่ต□องการ

- Quadrant Phase Shift เป□นวงจรเลื่อนเฟส 90, 180 และ 270 องศา ตามลำดับ

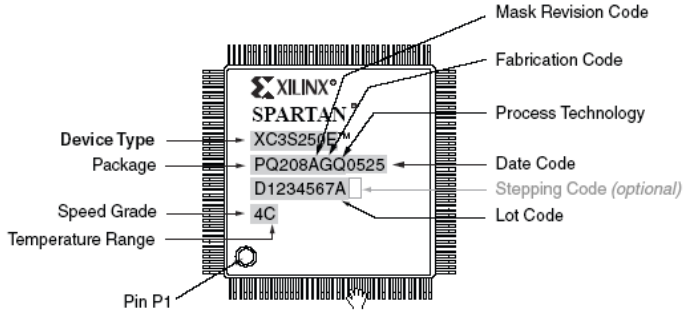
- Fine Phase Shift เป□นวงจรใช□ในการเลื่อนเฟสอย□างละเอียด มีความละเอียดอยู□ที่ 1/256 เท□าของคาบความถี่วงจรนี้มีความสำคัญมากเช□นกัน ที่ใช□ในการชดเชยการเลื่อนเฟสที่เกิดขึ้นในวงจร ทำให□การออกแบบง□ายขึ้นอย□างมาก

3.7.6 บล็อกควบคุมอิมพีแดนซ์ (Digitally Controlled Impedance: DCI)

Digitally Controlled Impedance (DCI) ใช□ป□องกันสัญญาณสะท□อนใน PCB โดยการควบคุมเอาต□พุต อิมพีแดนซ□ที่เหมาะสม

3.8 ข้อมูลเกี่ยวกับหมายเลขบนชิพ FPGA

ข้อมูลที่พิมพ์บน ตัวชิพ FPGA เป็นตัวอ้างถึงรายระเอียดต่าง ๆ เกี่ยวกับขบวนการผลิตซึ่งจะมีรายละเอียดเกี่ยวกับความจุ ตัวถัง ความเร็วเป็นต้น โดยมีรายระเอียดแสดงดังรูปที่ 3.9 ดังนี้



รูปที่ 3.9 รหัสโค้ดบนชิพ FPGA [21]

ตารางที่ 3.4 หมายเลขโค้ดบนชิพ FPGA [21]

อุปกรณ์	ความเร็ว		จำนวนขา และชนิดของแพกเกจ		ช่วงอุณหภูมิใช้งาน	
XC3S50	- 4	มาตรฐานทั่วไป	VQ100	100-Pin Plastic Very Thin Quad Flat Pack (VQFP)	C	Commercial (0 □C – 85 □C)
XC3S200	- 6	มาตรฐานสูง	TQ144	144-Pin Plastic Thin Quad Flat Pack (TQFP)		
XC3S400			PQ208	208-Pin Plastic Quad Flat Pack (PQFP)	I	Industrial (-40 □C – 100 □C)
XC3S1000			FT256	256-Ball Fine-Pitch Thin Ball Grid Array (FTBGA)		
XC3S1500			FG320	320-Ball Fine-Pitch Ball Grid Array (FBGA)		
XC3S2000			FG456	456-Ball Fine-Pitch Ball Grid Array (FBGA)		
XC3S4000			FG676	676-Ball Fine-Pitch Ball Grid Array (FBGA)		
XC3S5000			FG900	900-Ball Fine-Pitch Ball Grid Array (FBGA)		
			FG1156	1156-Ball Fine-Pitch Ball Grid Array (FBGA)		

3.9 บทสรุป

ในบทนี้ได้บรรยายถึงประวัติความเป็นมา สถาปัตยกรรม โครงสร้างองค์ประกอบต่าง ๆ ของ FPGA และรูปแบบชนิดต่าง ๆ ในการออกแบบ ขั้นตอนการออกแบบ และภาษาอธิบายพฤติกรรมทางฮาร์ดแวร์ โดยที่กล่าวมาทั้งหมดนั้นเป็นเพียงความรู้พื้นฐานในการที่จะออกแบบวงจรดิจิตอลซึ่งในการที่จะใช้งาน FPGA ให้ได้ประสิทธิภาพสูงสุดนั้นต้องมีประสบการณ์ในการออกแบบ ความเข้าใจในพื้นฐานวงจรดิจิตอล และเทคนิคในการแก้ไขปัญหาต่าง ๆ