การออกแบบวงจรเข้ารหัสถอดรหัสแบบ DES ด้วยภาษาวีเอชดีแอล

DES-SYSTEM DESIGN USING VHDL

บรรจง ปิยธำรง และ วัชรากร หนูทอง

Bunjong Piyatamrong and Watcharakron Noothong

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Faculty of Engineering, King Mongkut's Institute of Technology Chaokuntaharn Ladkrabang

บทคัดย่อ

จุดประสงค์ของงานวิจัยนี้ เพื่อแสดงขั้นตอนการออกแบบวงจรเข้ารหัสและถอดรหัสแบบ DES (Data Encryption Standard) ด้วยภาษาวีเอชดีแอล (VHDL) ซึ่งเป็นภาษาคอมพิวเตอร์ระดับสูงใช้ในการเขียนบรรยายฟังก์ชันการทำงานวงจรดิจิตอลได้ในลักษณะต่าง ๆ ได้แก่ การอธิบายพฤติกรรมของระบบ (Behavioral description) การไหลของข้อมูลในระบบ (Data flow description) จนถึงอธิบายถึงการเชื่อมต่อกันระหว่าง ส่วนต่าง ๆ จนเป็นโครงสร้างของระบบ (Structure description) ด้วยประโยชน์ของภาษานี้ รวมกับความสามารถของระบบที่ช่วยออกแบบทางอิเล็กทรอนิกส์แบบอัตโนมัติ (Electronic Design Automation, EDA) ทำให้การออกแบบวงจรดิจิตอลขนาดใหญ่เป็นไปได้อย่างรวดเร็ว ในงานวิจัยนี้จะแสดงขั้นตอนการออกแบบวงจร DES (Data Encryption Standard) ซึ่งเป็นอัลกอริทึมที่ใช้ในการถอดรหัสข้อมูลขนาด 64 บิท และเข้ารหัสข้อมูลขนาด 64 บิท ซึ่งมีความปลอดภัยสูง ในขอบข่ายงานวิจัยนี้จะเริ่มทำจากการเขียน Source Code VHDL, Compile, Debug และทำการ Simulation และจากนั้น synthesize จนได้เป็นผังวงจร (Schematic) แล้วนำไป Map ลงบน FPGA (Filed Programmable Gate Array) ซึ่งสามารถนำมาทดสอบการใช้งานได้จริง

ABSTRACT

The purpose of the research is to show the step of DES crpto-system circuit design using VHDL to explain the model of digital system in many characteristic such as behavior, Data flow and structure of system. The high performance of Electronic Design Automation, EDA make the development cycle of digital system faster and easier. This research will show the step of DES (Data Encryption Standard), which is high security algorithum to decode and encode 64-bit data. The scope will simulate and verify the functional of the source code VHDL, Compile, Debug and Synthesis to schematic diagram. With the schematic, it can develop on the FPGA (Field Programmable Gate Array) to implement the real working digital design.

1. บทนำ

ภาษาวีเอชดีแอล (VHDL) ย่อมาจาก VHSIC Hardware Description Language เป็นภาษาที่ใช้เขียนอธิบายฟังก์ชันการทำงานวงจรดิจิตอลได้ในลักษณะต่างๆ ได้แก่ การอธิบายพฤติกรรมของระบบ (Behavioral) การไหลของข้อมูลในระบบ (Data flow) จนถึงอธิบายถึงการเชื่อมต่อกันระหว่างส่วนต่างๆ จนเป็นโครงสร้างของระบบ (Structure) ตัวภาษาถูกริเริ่มสร้างจากโครงสร้าง VHSIC (Very High Speed Intergated Circuit) ของ Department OF Defence (DOD) ของสหรัฐอเมริกา เนื่องจากมีบริษัทที่สร้าง VHSIC Chip หลายบริษัทได้ร่วมโครงการที่จะพัฒนา ในขณะนั้นหลายบริษัทใช้ภาษาซึ่งแตกต่างกัน ในการที่อธิบายการทำงาน Chip ของตน ด้วยเหตุนี้ทำให้เกิดความแตกต่าง แต่ละบริษัทไม่สามารถแลกเปลี่ยนเทคโนโลยีให้กันและกันได้ทำให้ DOD เกิดปัญหาในการที่จะพัฒนาและซ่อมบำรุงในภายหลัง จึงเกิดความต้องการภาษา VHDL ซึ่งเป็นมาตรฐานในการที่จะอธิบายถึงตัว Design นั้นๆ ดังนั้น DOD จึงมอบให้บริษัท IBM, TEXAS INSTRUMENT และ INTERMETICS ร่วมกันพัฒนาและกำหนดมาตรฐานของภาษา VHDL ขึ้นมา ในปี 1983 หลังจากนั้น VHDL VERSION 7.2 ได้ทำการพัฒนาและออกเผยแพร่ต่อสาธารณชนในปี 1985 ได้รับความสนใจเป็นอย่างมากในอุตสาหกรรม โดยเฉพาะอย่างยิ่งบริษัทที่ทำ VHSIC CHIP จากผลสำเร็จนี้ทำให้เกิดมาตรฐาน IEEE ของ VHDL ในปี 1986 ภายหลังจากนั้นก็มีการพัฒนาขยายขีดความสามารถของตัวภาษา VHDL เพิ่มขึ้นจากในวงการอุตสาหกรรม มหาวิทยาลัยและ DOD ก็มีการปรับปรุงและจดมาตรฐานใหม่ IEEE ในปี 1987 อีกครั้ง ซึ่งเป็นที่รู้จักกันในชื่อว่า IEEE STD 1076-1986

2. ทฤษฎีของ DES

2.1 ที่มาและประวัติของอัลกอริทึม DES

อัลกอริทึม DES (Data Encryption Standard) เป็นกลวิธีการเข้ารหัสข้อมูลและถอดรหัสข้อมูลซึ่งได้รับการเผยแพร่ในปี ค.ศ. 1977 โดย National Bureau of Standard (NBS) ประเทศสหรัฐอเมริกา (ปัจจุบันเปลี่ยนเป็น National Institute of Standard and Technology) โดยมีจุดมุ่งหมายเพื่อใช้ทำการเข้ารหัสข้อมูลและถอดรหัสข้อมูลทางธุรกิจและข้อมูลของรัฐบาลที่ไม่เกี่ยวข้องกับความลับระดับประเทศ อัลกอริทึมของ DES ถูกดัดแปลงมาจากอัลกอริทึมของ IBM ที่ชื่อว่า Lucifer Cipher ซึ่งได้ทำการพัฒนาต่อร่วมกับ National Security Agency (NSA) ทำให้อัลกอริทึมนี้ เป็นที่ยอมรับโดยทั่วไปเป็นมาตรฐานของ ANSI (American National Standards Institute) ในปี ค.ศ. 1980

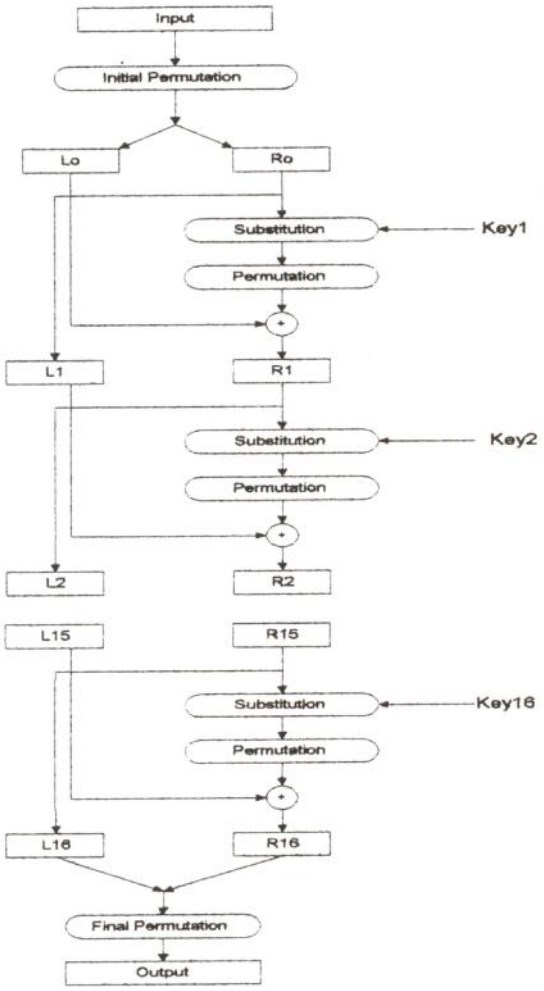
2.2 ระบบความปลอดภัยของอัลกอริทึม DES

อัลกอริทึม DES เป็นวิธีการเข้ารหัสที่ใช้ Private Key โดยการรับข้อมูลจำนวน 64 บิทและใช้ Key ขนาด 64 บิท แต่จะตัดทุกบิทที่ 8 ออกโดยใช้เป็น Parity bit อัลกอริทึม DES เป็นวิธีที่มีประสิทธิภาพมาก ในการออกแบบให้อยู่ในรูปแบบของฮาร์ดแวร์ แต่จะช้ามากถ้าถูกนำไปออกแบบในรูปของซอฟต์แวร์

ในการเข้ารหัสและถอดรหัสของ DES จะประกอบไปด้วยข้อมูลที่จะเข้ารหัสเรียกว่า Plaintext และข้อมูลที่เข้ารหัสแล้ว เรียกว่า Ciphertext ถ้าผู้ที่ต้องการเจาะระบบของ DES ในกรณีนี้คือการหา คีย์ ซึ่งจะใช้ค้นหา Plaintext ไปเป็น Ciphertext ถ้าผู้เจาะระบบอาศัยซอฟต์แวร์เป็นตัวค้นหาคีย์ที่กล่าว จะต้องใช้เวลาประมาณครึ่งปีในการค้นหาซึ่งใช้เวลามาก แต่ถ้าผู้เจาะระบบอาศัยฮาร์ดแวร์ในการค้นหาคีย์ ซึ่งทำได้เร็วกว่าก็จริงแต่จะต้องเสียค่าใช้จ่ายสูงซึ่งนาย Diffie และ Hellman ได้ทำการวิเคราะห์ค่าใช้จ่ายในการสร้าง ฮาร์ดแวร์ดังกล่าวไว้ประมาณ 20 ล้านดอลลาร์ ในการสร้าง Chip 1 ล้านตัวเพื่อใช้ในการค้นหาคีย์ในเวลา 12 ชั่วโมง ซึ่งจะเห็นได้ว่าอัลกอริทึม DES มีความปลอดภัยค่อนข้างสูงซึ่งเหมาะสำหรับการใช้ในการเก็บความลับของข้อมูลในการสื่อสาร

2.3 โครงสร้างระบบ DES

ระบบ DES เป็นระบบที่มีโครงสร้างพื้นฐานของข้อมูลอินพุทขนาด 64 บิท จะถูกส่งผ่านการทำ Initial Permutation เป็นการสลับตำแหน่งของอินพุทบิท ซึ่งจะได้ข้อมูลขนาด 64 บิท จากนั้นนำ Key ขนาด 64 บิท มาทำการตัดบิท Parity ทำให้ Key เหลือขนาด 56 บิท จากนั้นนำ Key ขนาด 56 บิท มาสร้าง Key ซึ่งใช้สำหรับแต่ละรอบขนาด 48 บิทจำนวน 16 ชุด ในการเข้ารหัสจะกระทำ 16 รอบ ซึ่งแต่ละรอบจะใช้อินพุทขนาด 64 บิทของรอบก่อนและใช้ Key 48 บิทของแต่ละรอบ ซึ่งครบ 16 รอบจะได้เอาท์พุทขนาด 64 บิท หลังจากรอบที่ 16 เอาท์พุท 64 บิท จะถูกส่งไปทำการ Final Permutation อีกครั้ง จนได้เอาท์พุทขนาด 64 บิทที่ใช้ในการส่งข้อมูลดังรูป



รูปที่ 1 แสดงโครงสร้างของระบบ DES

วิธีการสลับตำแหน่งในการทำ Initial Permutation และ Final Permutation ใน DES ถูกกำหนดรายละเอียดดังนี้

Initial Permutation (IP)

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Final Permutaion $(IP)^{-1}$

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

ตัวเลขในตารางข้างบนเป็นการกำหนดบิทของอิทพุทที่ผ่านการสลับตำแหน่ง ลำดับของตัวเลขในตารางจะไปเป็นตำแหน่งของเอาท์พุท ตัวอย่างเช่น Inital Permutation จะนำอิทพุทบิทที่ 58 ไปเป็นเอาท์พุทบิทที่ 1 และอินพุทบิทที่ 50 ไปเป็นเอาท์พุทบิทที่ 2 เป็นต้น

2.4 การสร้างคีย์สำหรับแต่ละรอบ

Key ที่ใช้มีขนาด 64 บิท แต่จะถูกตัด 8 บิท เพื่อใช้เป็น Parity bit โดยได้แก่ บิทที่ 8, 16 24, 32, 40, 48, 56, 64 จากนั้นทำการ Initial Permutation ที่ตัดเหลือ 56 บิทและจากนั้นแบ่งคีย์ออกเป็นสองส่วนขนาด 28 บิท 2 ชุด เรียกว่า Co และ Do และทำการสลับตำแหน่ง ในแต่ละชุดอีกครั้งดังนี้

ชุด Co : 47, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18, 10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36

ชุด Do : 63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22, 14, 6, 61, 53, 45, 37, 29, 21, 3, 5, 28, 20, 12, 4

จากนั้นนำชุด Co, Do ที่ได้นำมาเลื่อนบิทซ้าย จำนวนของบิทที่ถูกเลื่อนจะแตกต่างกันในแต่ระรอบ ดังนี้ ในรอบที่ 1, 2, 9 และ 16 จะถูกเลื่อนซ้าย 1 ครั้ง ส่วนในรอบอื่นๆ จะถูกเลื่อนซ้าย 2 ครั้ง การเลื่อนรอบนี้จะได้คีย์ชุดใหม่เรียกว่า C_i, D_i จากนั้นนำแต่ละชุดมาทำ Permuted Choice โดยที่บิทที่ 9, 18, 22 และ 28 จะถูกตัดทิ้ง ทั้งสองชุดดังนี้

ชุด C_i : 14, 17, 11, 24, 1, 5, 3, 28, 15, 6, 21, 10, 23, 19, 12, 4, 26, 8, 16, 7, 27, 20, 13, 2

ชุด D_i : 41, 52, 31, 37, 47, 55, 30, 40, 51, 45, 33, 48, 44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32

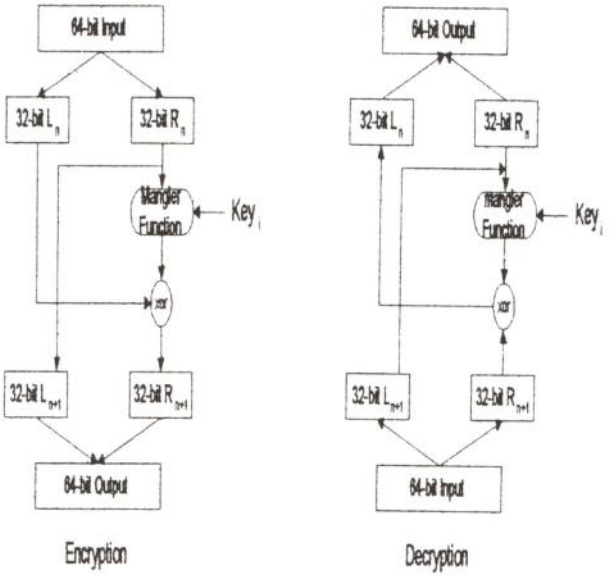
จากนั้นนำชุด C_i, D_i มารวมกันเพื่อสร้างคีย์เพื่อนำไปใช้ในแต่ละรอบในการเข้ารหัสและถอดรหัสต่อไป

2.5 วงรอบของอัลกอริทึม DES

ในการเข้ารหัสอินพุท 64 บิท จะถูกแบ่งเป็น 2 ครึ่ง ครึ่งละ 32 บิท เรียกว่า Ln และ Rn แต่ละรอบจะสร้างเอาท์พุทขนาด 32 บิท คือ Ln+1 และ Rn+1 โดยนำ Ln+1 และ Rn+1 มารวมกันจะได้เอาท์พุทขนาด 64 บิทของแต่ละรอบ

Ln+1 ได้จาก Rn ส่วน Rn+1 ได้มาดังนี้ เริ่มจาก Rn และ Kn เป็นอินพุทเข้าสู่ Mangler Function ซึ่งจะได้เอาท์พุทขนาด 32 บิท นำเอาท์พุทที่ได้ไปทำการ Exclusive OR กับ Ln จะได้ Rn+1 Mangler Function จะนำอินพุทขนาด 32 บิท กับคีย์ ขนาด 48 บิทมาสร้างเอาท์พุทขนาด 32 บิท

จากที่กล่าวมาถ้า RUN DES ย้อนกลับเพื่อถอดรหัสสมมติว่ามี Ln+1 และ Rn+1 ทำอย่างไรจึงจะได้ Ln และ Rn เราทราบว่า Rn ก็คือ Ln+1 เพราะฉะนั้นเราจะได้ Rn, Ln+1, Rn+1 และ Kn และจากที่เราทราบว่า Rn+1 และ Ln Exclusive OR กับ Mangler (Rn, Kn) จากนั้น Exclusive OR ผลที่ได้กับ Rn+1 จะได้ผลลัพธ์เป็น Ln



รูปที่ 2 แสดงวงรอบของอัลกอริทึม DES

2.5.1 กระบวนการทำ Mangler Function ใน DES

กระบวนการทำ Mangler Function คือการนำอินพุท Rn ขนาด 32 บิท และนำ Key_i ขนาด 48 บิท นำมาสร้างเอาท์พุทขนาด 32 บิท และนำไป Exclusive OR กับ Ln เพื่อสร้าง Rn+1 ตัวต่อไป

วิธีการเริ่มต้นจากการขยาย Rn จากเดิมขนาด 32 บิทให้มีขนาด 48 บิท ซึ่งทำที่เรียกว่า Expansion Permutation คือ แยก Rn เป็นชุด ชุดละ 4 บิท จำนวน 8 ชุด จากนั้นขยายแต่ละชุดให้มีขนาด 6 บิท โดยนำบิทที่ติดกันของแต่ละชุดมาต่อกันโดยถือว่าบิทซ้ายสุดและขวาสุดของ Rn ในแต่ละชุดเป็นบิทเริ่มต้นและบิทสุดท้ายของ Rn ชุดที่ถูกขยายตัวอย่างเช่น Rn ชุดที่ 1 คือ บิทที่ 1-4 ถูกขยายเป็น 6 บิท โดยนำเอา

บิทที่ 32 และบิทที่ 5 มาต่อเป็นบิทหน้าและบิทหลังตามลำดับ

จากนั้นแยก Key_i ออกเป็นชุดละ 6 บิท เพื่อนำมา Exclusive OR กับชุด Rn ที่ถูกทำการขยายแล้วจะได้เอาท์พุทขนาด 6 บิท ซึ่งแต่ละชุดที่ได้เรียกว่า B1, B2, B3,.....,B8 และนำแต่ละชุดของเอาท์พุทที่ได้ผ่านกระบวนการทำ S-BOX ซึ่งเป็นการสร้างเอาท์พุทที่มีขนาด 4 บิท (จากการเปิดตาราง) จากนั้นนำเอาท์พุท 4 บิทของ S-BOX ทั้ง 8 ชุดมารวมเป็น 32 บิท จากนั้นนำบิทเหล่านั้นไปผ่าน P-BOX (เป็นการสลับตำแหน่งของเอาท์พุทที่เกิดขึ้น)

3. ขั้นตอนการทำวิจัย

1. ออกแบบ Model จากตัวอัลกอริทึมที่กล่าวมาข้างต้นให้อยู่ในรูปของภาษา VHDL โดยแยกหน้าที่ในระบบออกเป็น Block หรือ Molude ตามหน้าที่ หรือเขียนในรูปของ State Machine, Truth Table

2. สร้าง Model ที่ใช้ภาษา VHDL เขียนอธิบายโดยอาศัย Software Tools ที่เรียกว่า Workview PLUS for Windows ของ Viewlogic Inc. เป็นตัวช่วยในการสร้าง ออกแบบการทดสอบการทำงาน

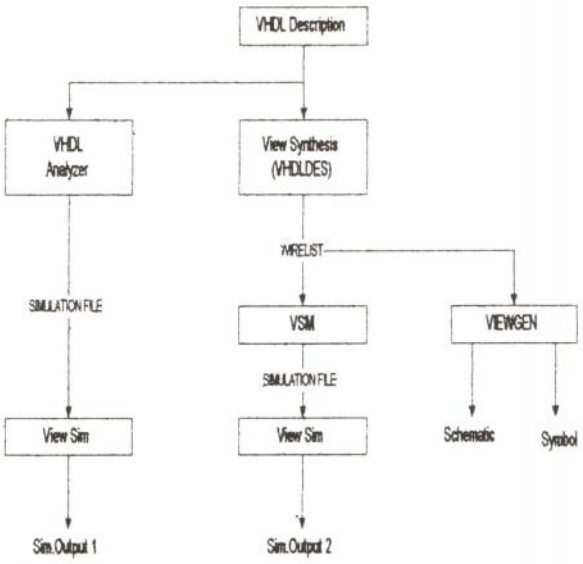
3. ทำการสังเคราะห์ (Synthesis) เพื่อได้ผังวงจร (Schematic) โดยอาศัย VIEWSynthesis (VHDLDES)

4. วิธีการทำโครงงานวิจัย

สามารถแบ่งออกเป็น 3 ขั้นตอนหลัก ดังนี้

4.1 DESIGN ENTRY

รูปที่ 3 แสดงถึงขั้นตอนการทำ Design Entry จะเห็นว่าเริ่มแรกจะทำการเขียน VHDL Code ขึ้นมาก่อนจากที่เรากำหนด จากนั้นจึงทำการตรวจสอบไวยากรณ์โดยใช้ VHDL Analyzer เป็นตัวคอมพายล์ (Compile) โดยลักษณะของการเขียน



รูปที่ 3 แสดงการทำ Design Entry โดยใช้ VHDL

ด้วยภาษา VHDL นี้จะต้องตามหลักของ VHDL Synthesis Guide

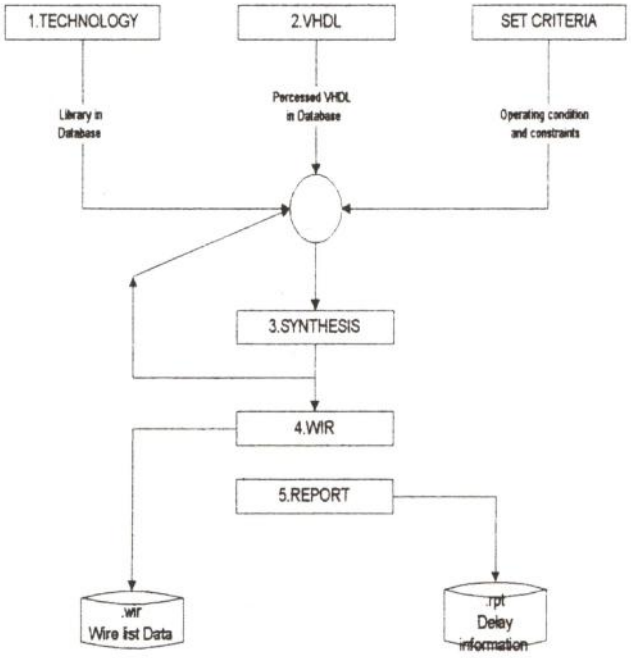
4.2 DESIGN SIMULATION

เมื่อได้ VHDL Model ที่ผ่านการคอมพายล์แล้วจะต้องนำมาทำการจำลองการทำงาน (Simulate) เพื่อตรวจสอบว่าทำงานได้ตามต้องการหรือไม่ซึ่งมีวิธีทำได้ 2 ทางคือ

ทางเลือกที่ 1 หลังจากทำการ compile VHDL Model แล้วให้ใช้ View Sim ทำการจำลองการทำงาน (Simulate) สามารถเก็บไว้เป็นไฟล์ได้

ทางเลือกที่ 2 ถ้าต้องการทดสอบในลักษณะของผังวงจร (Schematic) ให้ใช้ VIEW SYNTHESIS ทำการสังเคราะห์ VHDL Model ให้กลายเป็น WIRELIST จากนั้นให้ใช้ VIEWGEN ทำการสร้างผังวงจร (Schematic) โดยต้องกำหนดว่าใช้ Library ของอุปกรณ์จากที่ใด (ในที่นี้ใช้ X3000 ของบริษัท XILINX) จากนั้นใช้ VSM ทำการแปลงไฟล์ที่เป็น WIRELIST ให้เป็น File.VSM ซึ่งสามารถทำการจำลองการทำงานได้โดย View Sim เพื่อเปรียบเทียบว่าผลลัพธ์ที่ได้เป็นอย่างไร เมื่อได้ผลลัพธ์ถูกต้องเรียบร้อยแล้วก็เข้าสู่ขั้นตอนต่อไป

4.3 DESIGN SYNTHESIS



รูปที่ 4 แสดงการทำ Design Synthesis

จากรูปจะเห็นว่าที่จะทำการ Synthesis ได้นั้นจะต้องมี Input 3 อย่างคือ

1. Technology ว่าต้องการจะสร้างวงจรด้วย Component จากที่ใดซึ่งในงานวิจัยนี้เลือกใช้ X3000 XILINX

2. VHDL Code ที่ VIEWSYNTHESIS อ่านซึ่งต้องผ่านการ Compile แล้วเข้ามาใน Database ภายใน

3. การกำหนดข้อบังคับในการทำ Synthesis ต่างๆ

เมื่อได้กำหนดครบแล้ว Software จะเริ่มทำการ Synthesis จนได้ผลลัพธ์อยู่ในรูปของ Netlist จากนั้นให้ใช้คำสั่ง wir เพื่อสร้าง wirelist ของวงจรและใช้คำสั่ง report เพื่อสร้างรายงานเหตุการณ์ต่างๆ ที่เกิดขึ้นจากนั้นให้ใช้คำสั่ง viewgen เพื่อสร้างผังวงจร (Schematic) เพื่อนำไปเปิดใน VIEWDRAW ต่อไปเพื่อทำการ Map ขาเพื่อแปลงลงสู่ FPGA โดยใช้ xdm ของ XACT Development Software ต่อไป

5. ผลการวิจัย

ผลจากสังเคราะห์วงจรด้วย VHDLDES ของ Viewlogic Inc. โดยใช้ Technology X3000 ของ XILINX จะได้รายละเอียดของอุปกรณ์ที่ใช้สร้างผังวงจรเข้ารหัสและถอดรหัสแบบ DES ดังนี้

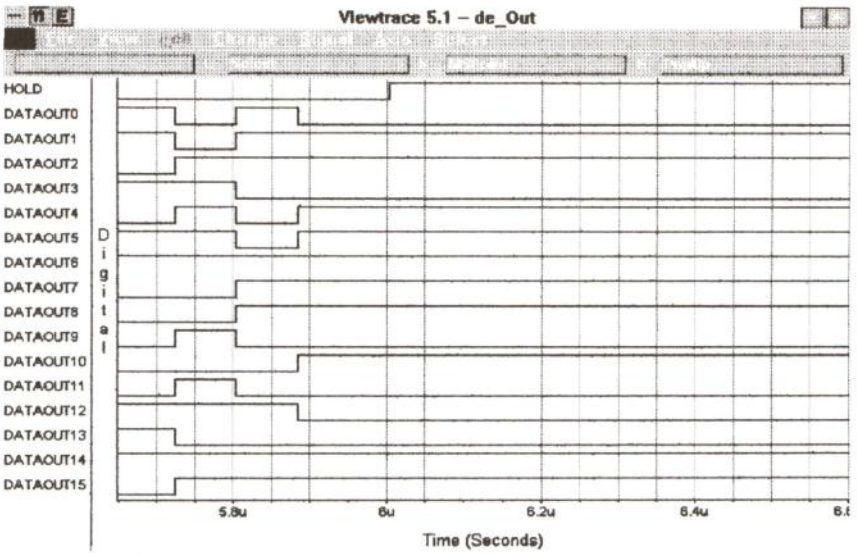
Cell	จำนวน	Area/Cell
MX3000:FDRD	513	0.00
X3000:AND2	4374	0.25
X3000:INV	527	0.00
X3000:NAND3	6	0.50
N3000:NAND5	112	0.75
X3000:NOR4	67	0.75
N3000:OR3	18	0.50
X3000:OR5	2148	0.75
MX3000:LD	32	2.00
X3000:AND5	3008	0.75
X3000:NAND2	1565	0.25
X3000:NAND4	100	0.75
X3000:NOR3	8	0.50
X3000:OR2	1908	0.25
X3000:OR4	219	0.75
X3000:XOR2	80	0.25
Total	14685	6302.25

ผลการทดสอบการทำงานของวงจรที่ถูกสร้างขึ้นดังนี้

1. ผลการสอบการถอดรหัส (DECRYPTION) (ขาสัญญาณ DEEN =1) โดยกำหนดให้

DATAIN มีค่า 0011 1011 0101 0101, 1001 0011 0110 1100, 0101 1100 1111 1100, 0000 1001 0111 0000

KEYIN มีค่า 1001 0010 0111 1000, 0011 0000 1111 0010, 1001 0100 1000 1101, 1111 1001 0101 0010 จะได้ผลดังรูป



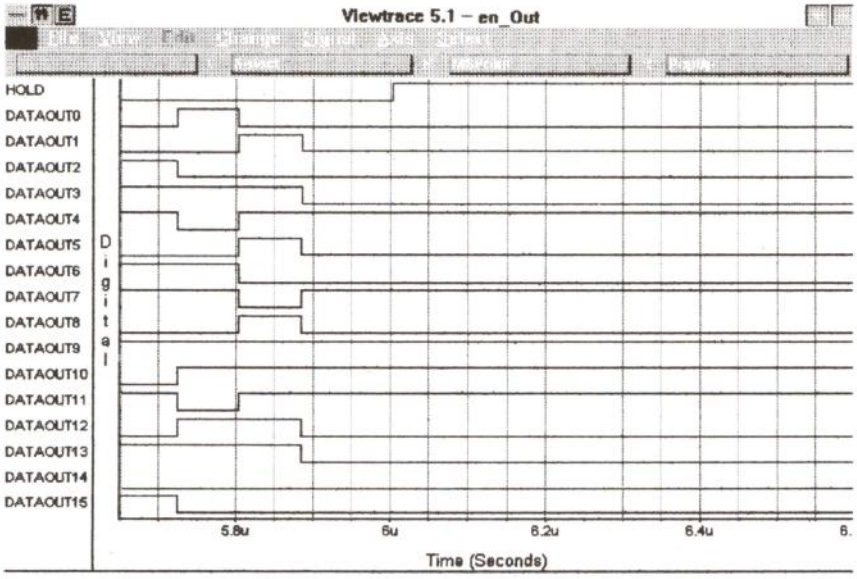
รูปที่ 5 แสดง waveform ของ DATAOUT 0-15 ในการทำ DECRYPTION

จากรูป waveform ที่ได้พบว่าการทำ DECRYPTION ดังกล่าวจะให้ DATAOUT ดังนี้

1101 0110 0000 1110, 0011 1110 0101 1011, 1110 0011 1000 1011, 0110 1111 1010 0011

2. ผลการทดสอบวงจรในการเข้ารหัส (ENCRYPTION) (ขาสัญญาณ DEEN = 0) โดยกำหนดให้ DATA ดังนี้ 1101 0110 0000 1110, 0011 1110 0101 1011, 1110 0011 1000 1011, 0110 1111 1010 0011

KEYIN ดังนี้ 1001 0010 0111 1000, 0011 0000 1111 0010, 1001 0100 1000 1101, 1111 1001 0101 0010 จะได้ผลดังรูป



รูปที่ 6 แสดง waveform ของ DATAOUT 0-15 ในการทำ ENCRYPTION

จากรูป waveform ที่ได้ พบว่าการทำ ENCRYPTION ดังกล่าวจะให้ DATAOUT ดังนี้

0011 1011 0101 0101, 1001 0011 0110 1100, 0101 1100 1111 1100, 0000 1001 0111 0000

6. บทสรุป

จากผลการจำลองการทำงานของวงจรที่ถูกออกแบบสามารถทำการเข้ารหัสและถอดรหัส (ENCRYPTION and DECRYPTION) ได้อย่างถูกต้อง เพราะเนื่องจากผลลัพธ์หรือ DATAOUT ที่ได้จากการ ENCRYPTION จะต้องมีค่าเท่ากับ DATAIN ที่ให้ในการทำ DECRYPTION โดยที่ KEYIN จะต้องเป็นคีย์ชุดเดียวกัน และผลลัพธ์หรือ DATAOUT ที่ได้จากการ DECRYPTION จะต้องมีค่าเท่ากับ DATAIN ที่ให้ในการทำ ENCRYPTION เช่นเดียวกัน ดังนั้นจะเห็นได้ว่าในการออกแบบวงจรที่สลับซับซ้อนเป็นไปได้อย่างสะดวกรวดเร็วเพราะเนื่องจากเราใช้ภาษา VHDL เป็นตัวอธิบาย model ที่เรากำหนดก่อนนำไปสร้างเป็นฮาร์ดแวร์ขึ้นจริง ดังนั้นในการแก้ไขหรือเปลี่ยนแปลงวงจรที่เราออกแบบจึงไม่จำเป็นจะต้องไปยุ่งในระดับฮาร์ดแวร์โดยตรง เราสามารถแก้ไขในส่วนของโค้ดโปรแกรมของเราและทำการ compile, simulate, synthesize ใหม่เท่านั้น

ดังนั้นในปัจจุบันจะเห็นว่ามีวงจรรวม ASIC ใหม่ๆ ถูกผลิตภัณฑ์ออกมามากมาย เนื่องมาจากเทคโนโลยีในการออกแบบวงจรรวมได้พัฒนาขึ้นไปนั่นเอง การศึกษาโครงงานนี้ทำให้สามารถนำความรู้ที่ได้ไปใช้ประโยชน์ได้ในวงการอุตสาหกรรมจริงๆ และคิดว่าในอนาคตจะมีการใช้เครื่องมือที่มีอยู่ออกแบบและพัฒนาสิ่งประดิษฐ์ใหม่ๆ สู่วงการอิเล็กทรอนิกส์ต่อไป

7. บทวิจารณ์

ระบบ Electronic Design Automation, EDA เป็นการออกแบบที่ใช้คอมพิวเตอร์ทั้ง Hardware และ Software ช่วยในการออกแบบวงจรอิเล็กทรอนิกส์ต่าง ๆ เป็นเทคโนโลยีแบบใหม่ทันสมัยมาก มีมากมายหลาย Platform ทั้ง PC และ Workstation ซอฟต์แวร์ที่ใช้บนแต่ละ Platform ก็แตกต่างกันไป การเรียนรู้แต่ละรูปแบบใช้เวลานานมากเนื่องจากไม่คุ้นเคย ทำให้งานวิจัยนี้ใช้เวลานานในการศึกษา Tools ต่างๆ เหล่านี้ และบางครั้งเกิดปัญหาความเข้ากันไม่ได้ระหว่าง Software ทำให้เกิดปัญหาในการพัฒนาต่อ ฉะนั้นในการออกแบบควรจะศึกษา Tools ที่จะใช้ให้ดีก่อนที่เริ่มลงมือออกแบบ

เอกสารอ้างอิง

[1] Jayaram Bhasker, *"VHDL Primer"*, Printice Hall, First Edition, 1992.

[2] Warwick Ford, *"Computer Communications Security"*, Prentice Hall, First Edition, 1994, pp. 69-71.

[3] Charlie Kaufman, Radia perlman and Mike Speciner, *"Network Security"*, Prentice Hall, First Edition, 1995, pp. 60-73.

[4] Viewlogic *"Using Workview plus for Windows"*, Viewlogic, 1993.

[5] Viewlogic, *"ViewDraw Reference Manual"*, Viewlogic, 1993.

[6] Viewlogic, *"ViewSim Reference Manual"*, Viewlogic, 1993.

[7] Viewlogic, "VHDL User's Guide", Viewlogic, 1993.

[8] Viewlogic, *"VHDL Reference Manual"*, Viewlogic, 1993.

[9] XILINX, *"XACT Development System Reference Guide"*, XILINX, 1994.

[10] XILINX, *"XACT Viewlogic Interfrace Use Guide"*, XILINX, 1994.

[11] XILINX, *"The Programmable Logic Data Book"*, XILINX, 1994.