

## การพัฒนาอุปกรณ์เครือข่ายบน NetFPGA เพื่อการจัดการ Loop

นางสาวสุกัศ สุขประเสริฐ

นางสาวณัฏฐ์รัตน์ เจริญสุขภาพ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

ปีการศึกษา 2557

# Network Device Implementation using NetFPGA to Handle Loops Problem


Ms. Supak      Sukprasert

Ms. Rununrath Jarornsukkhaphap


A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS  
FOR THE DEGREE OF BACHELOR OF COMPUTER ENGINEERING  
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING  
FACULTY OF ENGINEERING  
KING MONGKUT'S UNIVERSITY OF TECHNOLOGY NORTH BANGKOK  
ACADEMIC YEAR 2014

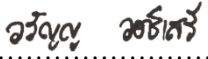
ปรินุญนัพนธ์เรื่อง : การพัฒนาอุปกรณ์เครือข่ายบน NetFPGA เพื่อการจัดการ Loop  
 ชื่อ : นางสาวสุกัศ สุขประเสริฐ  
 นางสาวณัณนัทรรัตน์ เจริญสุขภาพ  
 สาขาวิชา : วิศวกรรมคอมพิวเตอร์  
 ภาควิชา : วิศวกรรมไฟฟ้าและคอมพิวเตอร์  
 คณะ : วิศวกรรมศาสตร์  
 อาจารย์ที่ปรึกษา : รองศาสตราจารย์ ดร.วรา วราวิทย์  
 ปีการศึกษา : 2557

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ อนุมัติให้  
 ปรินุญนัพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต สาขา  
 วิชาวิศวกรรมคอมพิวเตอร์

  
 ..... หัวหน้าภาควิชาวิศวกรรมไฟฟ้า  
 (ผู้ช่วยศาสตราจารย์ ดร.นภดล วิวัชรโกเศศ) และคอมพิวเตอร์

  
 ..... ประธานกรรมการ  
 (รองศาสตราจารย์ ดร.วรา วราวิทย์)

  
 ..... กรรมการ  
 (รองศาสตราจารย์ ดร.ณชล ไชยรัตน์)

  
 ..... กรรมการ  
 (ผู้ช่วยศาสตราจารย์ ดร.วรัญญู วงษ์เสรี)

ลิขสิทธิ์ของภาควิชาวิศวกรรมไฟฟ้า และคอมพิวเตอร์คณะวิศวกรรมศาสตร์  
 มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

Projected Report Title : Network Device Implementation using NetFPGA to Handle  
Loops Problem

Name : Ms. Supak Sukprasert  
Ms. Rununrath Jarornsukkhaphap

Major Field : Computer Engineering


Department : Electrical and Computer Engineering


Faculty : Engineering


Project Advisor : Assoc. Prof. Dr. Vara Varavithya


Academic Year : 2014

Accepted by the Faculty of Engineering, King Mongkut's University of Technology North  
Bangkok in Partial Fulfillment of the Requirement for the Degree of Bachelor of Computer  
Engineering

  
..... Chairperson of Department of Electrical  
(Asst. Prof. Dr. Noppadol Wiwatcharagoses) and Computer Engineering

  
..... Chairperson  
(Assoc. Prof. Dr. Vara Varavithya)

  
..... Member  
(Assoc. Prof. Dr. Nachol Chaiyaratana)

  
..... Member  
(Asst. Prof. Dr. Waranyu Wongseree)

Copyright of the Department of Electrical and Computer Engineering, Faculty of Engineering  
King Mongkut's University of Technology North Bangkok

## บทคัดย่อ

การพัฒนาอุปกรณ์ในระบบเครือข่ายในอนาคต มีอุปกรณ์ลักษณะหนึ่งที่เรียกว่า NetFPGA ที่สามารถนำมาใช้เพื่อการพัฒนาต้นแบบอุปกรณ์ต่าง ๆ โดยไม่ได้ใช้โปรโตคอลมาตรฐานในการให้บริการ กระบวนการที่จะนำ NetFPGA มาใช้งานและขั้นตอนในการออกแบบ เพื่อสร้างให้เป็นอุปกรณ์ในระบบเครือข่ายมีความซับซ้อนและเข้าใจได้ยาก ปัจจุบัน NetFPGA ถูกนำมาใช้งานอย่างหลากหลาย และมีคนนำไปพัฒนาในหลายลักษณะการทำงาน เช่น พัฒนาเป็น Hub, Router, Switch และ OpenFlow โดยในการทดลองนี้ได้พัฒนาเป็นอุปกรณ์ที่เรียกว่า ARP-Path Switch ปรินซ์เล่มนี้ได้ทำการศึกษาการใช้งาน NetFPGA มีการเก็บรวบรวมกระบวนการในการนำ NetFPGA มาใช้งานอย่างเป็นระบบ ได้แก่ การติดตั้ง NetFPGA ศึกษาแฟ้มและไฟล์ของข้อมูลต่างๆ ที่เกี่ยวข้อง การตั้งค่าการทำงานให้เป็น ARP-Path Switch และการทดลองพัฒนาเป็น ARP-Path Switch ที่สามารถจัดการปัญหาได้โดยอาศัยการทำงานของ ARP โปรโตคอล สามารถเลือกเส้นทางในการส่งข้อมูลได้อย่างมีประสิทธิภาพ ซึ่งได้มีการทดสอบการทำงานของ ARP-Path Switch คือ ระบบเครือข่ายที่เกิดรูป ขนาด 6 node และ ขนาด 7 node ซึ่งผลการทำงานแสดงให้เห็นว่าสามารถจัดการปัญหาภายในระบบเครือข่ายได้ และได้เส้นทางที่ใช้ในการส่งข้อมูลเป็นเส้นทางที่ดีที่สุด เพื่อแก้ไขปัญหาการจัดการรูปในระบบเครือข่ายของ Reference Switch ที่ไม่สามารถจัดการปัญหาได้

## **Abstract**

In development of next generation networks, the NetFPGA has been widely adopted as a prototype platform. A researcher can study new protocols as well as services. The development process of programming and configuration of NetFPGA is rather complex and difficult to comprehend. Recently, the NetFPGA has been used in implementing diverse devices such as a hub, a router, a Ethernet switch, and an openflow switch. In this work, we studied the hand-on process in implementing NetFPGA. The related files were collected and systematically organized. The previously implementation of the ARP-Path algorithm were selected and re-implemented in the testbed. We set the NetFPGA to work as ARP-PART switches which eliminate loops using ARP Protocol. The testbed consists of 6 nodes and 7 nodes. The process of loop eliminations was demonstrated.

## กิตติกรรมประกาศ

ปริญญานิพนธ์เล่มนี้ได้สำเร็จลุล่วงไปได้ด้วยดีโดยความกรุณาและความช่วยเหลืออย่างดียิ่งของ รองศาสตราจารย์ ดร.วรา วราวิทย์ อาจารย์ที่ปรึกษาปริญญานิพนธ์ ผู้ช่วยศาสตราจารย์ ดร.มารอง ผดุงสิทธิ์ อาจารย์ที่ปรึกษาปริญญานิพนธ์พิเศษ ที่ได้ให้คำปรึกษา คำแนะนำ ข้อคิดเห็น ข้อมูล ซึ่งเป็นประโยชน์อย่างยิ่งสำหรับปริญญานิพนธ์เล่มนี้ และช่วยสนับสนุนอุปกรณ์สำหรับการพัฒนาอุปกรณ์เครือข่ายบน NetFPGA เพื่อการจัดการ Loop Problems ขอขอบพระคุณทุกท่านที่คอยช่วยเหลือทุกสิ่งจนกระทั่งปริญญานิพนธ์เล่มนี้สำเร็จลุล่วงด้วยดี

ท้ายนี้ข้าพเจ้าขอขอบพระคุณบิดา มารดา และ ครอบครัว ที่ได้ให้ความสนับสนุนและให้กำลังใจข้าพเจ้าเสมอมาจนสำเร็จการศึกษา

สุภักดิ์                      สุขประเสริฐ  
ถนันทน์รัตน์    เจริญสุขภาพ

## สารบัญ

	หน้า
บทคัดย่อภาษาไทย	จ
บทคัดย่อภาษาอังกฤษ	ฉ
กิตติกรรมประกาศ	ช
สารบัญตาราง	ณ
สารบัญภาพ	ญ
บทที่ 1. บทนำ	1
บทที่ 2. NetFPGA และ ARP-Path Switch	4
2.1 บอร์ด NetFPGA	4
2.2 การทำงานของบอร์ด NetFPGA	6
2.3 โปรโตคอล ARP	9
2.4 Reference Switch และ ARP-Path Switch	13
2.5 การทำงานของ ARP-Path Switch	16
บทที่ 3. ขั้นตอนการดำเนินงาน	23
3.1 โทโปโลยีที่ใช้ในการทดลอง	23
3.2 ภาพรวมการดำเนินการทดลอง	24
3.3 การติดตั้งบอร์ด NetFPGA และแพ็คเกจ	25
3.4 การติดตั้งแพ็คเกจ ARP-Path	25
3.5 การดาวน์โหลดโปรแกรมลงบอร์ด NetFPGA	26
3.6 การทดสอบการทำงานของ ARP-Path Switch	28
บทที่ 4. ผลการทดสอบ ARP-Path Switch	29
4.1 ผลการติดตั้งบอร์ด NetFPGA	29
4.2 การทดสอบการโปรแกรมบอร์ดด้วย Selftest	31
4.3 ผลการติดตั้งแพ็คเกจ NetFPGA และ ARP-Path Switch	32
4.4 ผลการทดสอบ ARP-Path Switch	33
บทที่ 5. สรุปผลการดำเนินการ	43
เอกสารอ้างอิง	44
ประวัติผู้แต่ง	45



## สารบัญตาราง

ตารางที่	หน้า
2-1 แสดงแพ็คเกจใน DMA ที่ได้รับการส่งมาจากซอฟต์แวร์ออกจากอินเทอร์เฟซ nf2cx	8
2-2 ตัวอย่าง ARP Hardware Type Values	10
2-3 ARP Operation Values	11
2-4 แสดง module ของ Reference Switch	14
2-5 แสดง module ของ ARP-Path Switch	14
2-6 แสดง state ของ Reference Switch	14
2-7 แสดง state ของ ARP-Path Switch	15
2-8 แสดง table ของ ARP-Path Switch	15
4-1 ตารางเก็บข้อมูลจาก Learning Table ของ ARP-Path Switch ทั้ง 5 เครื่อง	39
4-2 ตารางเก็บข้อมูลจาก Learning Table ของ ARP-Path Switch ทั้ง 5 เครื่อง	41

## สารบัญภาพ

ภาพที่	หน้า
2-1 บอร์ด NetFPGA	4
2-2 ภาพ Block Diagram แสดงส่วนประกอบของบอร์ด NetFPGA	5
2-3 แสดงสถานะการทำงานของแต่ละ Module	6
2-4 แสดง “Reference Pipeline”	7
2-5 แสดงรูปแบบเฟรมของ ARP ที่ใช้กับโปรโตคอล IP	10
2-6 แสดงการส่ง ARP Request ไปยังทุกเครื่องในเน็ตเวิร์ก	12
2-7 แสดงการตอบกลับโดย ARP Reply	13
2-8 แสดงกลไกการเมื่อ เครื่องคอมพิวเตอร์ค้นหา ต้องการติดต่อกับเครื่องปลายทาง	16
2-9 รหัสเทียมการทำ Forwarding	18
2-10 รหัสเทียมการทำ Forwarding และ Path Recovery	18
2-11 code การทำงานของ state WAIT_TILL_DONE_DECODER	19
2-12 code การทำงานของ state WRITE_HDR	19
2-13 code การทำงานของ state SKIP_HDRS	19
2-14 code การทำงานของ state WAIT_EOP	20
2-15 code การทำงานของ state ARP_CHECK_TABLE	22
3-1 โทโปโลยีที่ใช้ในการทดลอง	23
3-2 ฟังก์ชันดำเนินงานส่วนบอร์ด NetFPGA	24
3-3 ฟังก์ชันดำเนินงานการทดสอบระบบ ARP-Path Switch	25
3-4 แสดง Project Directory หลังทำการย้าย Arppath File	26
4-1 แสดงรายละเอียดของ Port เมื่อทำการเชื่อมต่อคอมพิวเตอร์และบอร์ด NetFPGA	30
4-2 การเชื่อมต่อสาย LAN ระหว่าง Port เพื่อทำการ Selftest	31
4-3 ผลการรันคำสั่ง Selftest เพื่อตรวจสอบการใช้งานของ Port	31
4-4 แสดง Packet พื้นฐานของ NetFPGA ก่อนทำการติดตั้ง ARP-Path Switch	32
4-5 แสดง Packet พื้นฐานของ NetFPGA หลังทำการติดตั้ง ARP-Path Switch	33

## สารบัญภาพ (ต่อ)

ภาพที่	หน้า
4-6 การใช้คำสั่งเพื่อดาวน์โหลด cpci_reprogram.bit	33
4-7 การตรวจสอบสถานะของ Port	34
4-8 ใช้คำสั่งเพื่อดาวน์โหลด arppath.bit เพื่อให้บอร์ด NetFPGA	35
4-9 โทโปโลยีการทดสอบ	35
4-10 แสดงคำสั่งในการกำหนดค่าของตาราง	36
4-11 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-1	36
4-12 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-2	37
4-13 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-3	37
4-14 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-4	38
4-15 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-5	38
4-16 แสดงเส้นทางการส่งข้อมูลระหว่าง Host1 และ Host2	39
4-17 โทโปโลยีการทดสอบเมื่อ Link5 ไม่ได้เชื่อมต่อ	39
4-18 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-3	40
4-19 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-5	40
4-20 แสดงเส้นทางการส่งข้อมูลระหว่าง Host1 และ Host2	41
4-21 การ Ping จาก Host1 ไป Host2	42
4-22 การ Ping จาก Host2 ไป Host1	42

# บทที่ 1

## บทนำ

เทคโนโลยีในการติดต่อสื่อสารถือว่าเป็นปัจจัยที่สำคัญในการดำเนินงานในด้านต่างๆ เพื่อช่วยให้การดำเนินงานสามารถเข้าถึงได้ง่าย และมีความรวดเร็วในการดำเนินงาน ในปัจจุบัน เทคโนโลยีในการติดต่อสื่อสารมีการเติบโตและพัฒนาอย่างรวดเร็ว เพื่อให้สามารถรองรับกับการใช้งานของมนุษย์ในด้านต่างๆ

ระบบเครือข่าย (Network) เป็นอีกเทคโนโลยีในการติดต่อสื่อสารที่สำคัญมากในปัจจุบัน โดยระบบเครือข่ายเป็นการเชื่อมต่อกันของอุปกรณ์ระบบเครือข่าย หรือ คอมพิวเตอร์ตั้งแต่ 2 เครื่อง ขึ้นไป ทำให้ผู้ใช้งานคอมพิวเตอร์ภายในระบบเครือข่ายสามารถ แลกเปลี่ยนข้อมูล ใช้งานอุปกรณ์ต่างๆในระบบเครือข่ายร่วมกัน และ สื่อสารกันได้อย่างมีประสิทธิภาพ นอกจากนี้ยังสามารถใช้ทรัพยากรร่วมกันทำให้ประหยัดทรัพยากรในการใช้งานเช่น Printer, Hard disk, CD ROM, Scanner โดยระบบเครือข่ายมีหลายขนาดแล้วแต่ลักษณะการใช้งาน คือ LAN ( Local Area Network ) ระบบเครือข่ายท้องถิ่น ระยะทางไม่เกิน 10 กิโลเมตร, MAN ( Metropolitan Area Network ) ระบบเครือข่ายเมือง, WAN ( Wide Area Network ) ระบบเครือข่ายกว้างไกล เรียกว่า World Wide ของระบบเครือข่ายระดับประเทศ

ระบบเครือข่ายคอมพิวเตอร์คอมพิวเตอร์ มีเครื่องคอมพิวเตอร์เป็นองค์ประกอบพื้นฐาน ไม่จำเป็นต้องเป็นรุ่นเดียวกัน หรือประเภทเดียวกัน รวมถึงสามารถเชื่อมต่อเข้ากับอุปกรณ์เครือข่ายอื่นๆ เช่น โมเด็ม (Modem), การ์ดเครือข่าย (Network Adapter), เกตเวย์ (gateway), เราเตอร์ (Router), ฮับ (Hub) และ สวิตช์ (Switch) ปรินูญานิพนธ์นี้จะศึกษาในเรื่องของ สวิตช์ (Switch) ที่เป็นอุปกรณ์เครือข่ายที่มีความสำคัญ ทำหน้าที่สำหรับเชื่อมต่อเครือข่ายท้องถิ่นประเภทเดียวกัน ใช้โปรโตคอลเดียวกัน ส่องวงเข้าด้วยกัน มีความสามารถในการเชื่อมต่อฮาร์ดแวร์ และตรวจสอบข้อผิดพลาดของการส่งข้อมูลได้ ซึ่งหน้าที่ของสวิตช์เหมือนฮับ คือเป็นตัวการกระจายสัญญาณ โดยมีการางในการจำค่าของ Mac Address ของการ์ดแลนค์ และหน้าที่สำคัญที่อย่างคือ การเพิ่มพอร์ทในการใช้งานให้มากขึ้น

บอร์ด NetFPGA เป็นอุปกรณ์ที่นำมาใช้ในการศึกษาการทำงานของอุปกรณ์ในระบบเครือข่าย ซึ่งสามารถโปรแกรมลงไปยังบอร์ด NetFPGA เพื่อสั่งการควบคุมพฤติกรรมการทำงานของบอร์ด โดยการใช้ Binary File ซึ่งพื้นฐานของบอร์ด NetFPGA สามารถทำงานมีพฤติกรรมเป็นสวิตช์ เรียกว่า Reference Switch ที่ทำงานบนเลเยอร์ที่ 2 ในระบบเครือข่าย แต่ในปัจจุบันการเชื่อมต่ออุปกรณ์บนเครือข่ายมีความหลากหลายและความซับซ้อนที่เพิ่มขึ้น ส่งผลทำให้เกิดปัญหาต่างๆในการติดต่อสื่อสารกันภายในเครือข่าย เช่น ปัญหาการเกิดลูปภายในเครือข่าย ซึ่งเป็นปัญหาที่มักหลีกเลี่ยงไม่ได้ เนื่องจากการต่อระบบเครือข่ายแบบลูปเป็นการช่วยรักษาเสถียรภาพการทำงานของเครือข่าย ถ้าหากเกิดปัญหาที่เส้นหนึ่งไม่สามารถส่งข้อมูลได้ ก็ยังสามารถที่จะใช้เส้นทางอื่นเป็นเส้นทางทดแทนในการส่งข้อมูลต่อไป ทำให้การดำเนินงานต่างๆไม่เกิดการหยุดชะงัก ทำให้มีประสิทธิภาพมากขึ้น แต่การทำงานของ Reference Switch บน NetFPGA จะทำงานอยู่บนเลเยอร์ 2 ไม่สามารถจัดการกับปัญหาการเกิดลูปในระบบเครือข่ายได้ ทำให้เมื่อเส้นทางใดเกิดความผิดพลาด การส่งข้อมูลไม่สามารถทำได้ จะทำให้การดำเนินงานต่างๆที่ทำผ่านระบบเครือข่ายที่ไม่สามารถแลกเปลี่ยนข้อมูลกันได้ เกิดความเสียหายในการดำเนินงาน จึงต้องมีการคิดค้นหาวิธีที่จะจัดการกับปัญหาของการเกิดลูปในเครือข่าย

โปรโตคอลที่เกี่ยวข้องกับการคำนวณเส้นทางในการส่งข้อมูลในระบบเครือข่ายขนาดใหญ่ในปัจจุบัน มีความซับซ้อนในการคำนวณและการควบคุมการแลกเปลี่ยนข้อมูล ยิ่งถ้าหากเส้นทางระหว่างสวิตช์ในระบบเครือข่ายมีความซับซ้อน หลากหลายของเส้นทางมากเท่าไร ก็ย่อมเสียทรัพยากรในการคำนวณหาเส้นทางมากเท่านั้น

โปรโตคอลหนึ่งที่ถูกนำมาใช้ในการจัดการกับปัญหาการเกิดลูปในเครือข่ายคือ Spanning Tree Protocol เพราะเป็นโปรโตคอลที่สามารถจัดการกับลูปในเครือข่ายได้ แต่เนื่องจากข้อจำกัดในการใช้งาน เช่น ขนาดของระบบเครือข่าย และเส้นทางที่เลือกในการส่งข้อมูลนั้นไม่ใช่เส้นทางที่สั้นที่สุด ทำให้ประสิทธิภาพในการทำงานยังไม่ได้ดีเท่าที่ควร จึงมีการใช้งาน ARP Protocol ที่สามารถนำมาจัดการกับลูปในเครือข่าย แก้อัปเดตของ Reference Switch ที่ไม่สามารถทำงานได้เมื่อเจอลูปภายในเครือข่าย ให้สามารถทำงานได้ ที่สามารถจัดการกับปัญหาลูปในเครือข่ายได้ เหมือนกับ Spanning Tree Protocol แต่มีประสิทธิภาพในการทำงานที่มากกว่า คือ ไม่มีข้อจำกัดในเรื่องของขนาดเครือข่าย เส้นทางที่ถูกเลือกนำมาใช้ในการส่งข้อมูลเป็นเส้นทางที่สั้นที่สุด รวมถึง ARP Protocol เป็นโปรโตคอลพื้นฐานในการทำงานอยู่แล้ว สามารถเข้าใจการทำงานได้ง่าย ไม่ยุ่งยากและซับซ้อน

โดยการทำงานของโปรโตคอล ARP เป็นโปรโตคอลในการสื่อสาร ทำหน้าที่จับคู่ระหว่าง IP Address ทาง Logical กับ Physical เนื่องจากระบบของการส่งข้อมูลในระบบ IP ไม่ขึ้นกับ Hardware ใดๆ คือ เมื่อระบบ IP ต้องการส่งข้อมูล ต้องร้องขอบริการจากระดับชั้น Data Link Layer ในระบบเครือข่าย แต่ในชั้นนี้ไม่รู้จัก IP Address จึงต้องทำการหา Hardware Address เพื่อที่จะสร้างข้อมูลในการส่งได้

ในปริณิญาณิพนธ์นี้ได้ศึกษาการทำงานของ ARP-Path Switch คือ อุปกรณ์เครือข่ายพื้นฐาน Switch บนบอร์ด NetFPGA มีพฤติกรรมการทำงานเป็น Switch ที่สามารถจัดการกับดูลูปในเครือข่ายได้โดยใช้ ARP Protocol ในการทำงาน การหาเส้นทางในการส่งข้อมูลระหว่างเครื่องอุปกรณ์ต้นทางไปยังเครื่องอุปกรณ์ปลายทาง อาศัย packet ใน ARP Protocol 2 ชนิด คือ ARP Request และ ARP Reply มาสนับสนุนการทำงานนี้

## บทที่ 2

### NetFPGA และ ARP-Path Switch

ในขั้นตอนของการศึกษาทฤษฎีเพื่อทำการเรียนรู้การทำงานของอุปกรณ์ต้นแบบ ARP-PATH switch เพื่อให้เกิดความเข้าใจ และง่ายต่อการนำมาใช้ ทำให้จำเป็นต้องมีศึกษาทฤษฎีที่เกี่ยวข้องต่าง ๆ เพื่อให้พร้อมสำหรับดำเนินการทดลองพัฒนาอุปกรณ์ต้นแบบ ARP-PATH switch โดยวิธีการดำเนินงาน ดังต่อไปนี้

1. บอร์ด NetFPGA
2. การทำงานของบอร์ด NetFPGA
3. โปรโตคอล ARP
4. Reference Switch และ ARP-Path Switch
5. การทำงานของ ARP-Path Switch

ซึ่งในแต่ละขั้นตอนสามารถอธิบายการดำเนินงานได้ดังนี้

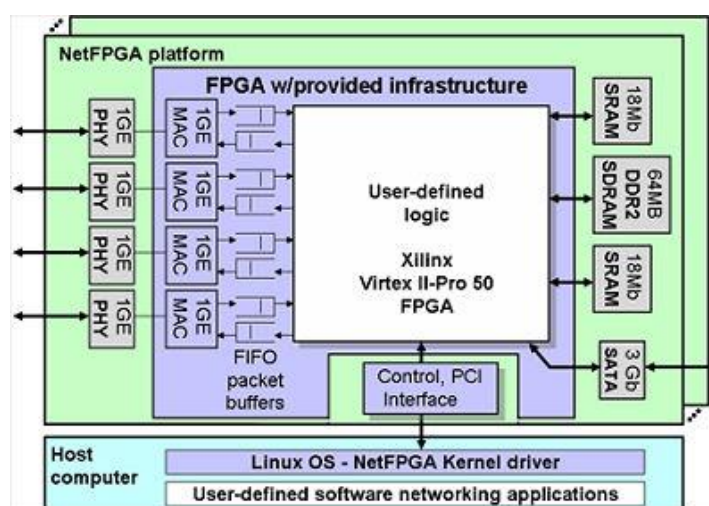
#### 2.1 บอร์ด NetFPGA

บอร์ด NetFPGA ถูกคิดค้นโดย Stamford International University ซึ่งเป็น platform ต้นทุนต่ำที่ออกแบบมาให้สามารถนำมาใช้ในการศึกษาเรื่องของฮาร์ดแวร์ของระบบเครือข่าย โดยถือว่าเป็นเครื่องมือที่มีประโยชน์สำหรับนักวิจัยระบบเครือข่าย ซึ่งตอนแรกที่บอร์ดถูกพัฒนาถูกเรียกว่า NetFPGA-1G และต่อมาได้มีการพัฒนาเพื่อให้สามารถรองรับความเร็วของเครือข่ายที่เพิ่มมากขึ้นจึงได้มีการพัฒนาต่อมาเป็น NetFPGA-10G



ภาพที่ 2-1 บอร์ด NetFPGA บอร์ด NetFPGA

บอร์ด NetFPGA เป็นอุปกรณ์ที่สามารถโปรแกรมลงไปยังบอร์ดเพื่อสั่งการให้บอร์ดกลายเป็นอุปกรณ์ในระบบเครือข่ายต่างๆได้ เช่น Ethernet Switch , Hub และ Internet Protocol Routers โดยวิธีการดาวน์โหลดไฟล์ประเภทไบนารี (Binary File) โดยใช้ภาษาเวอริลล็อก (Verilog Language) ในการพัฒนา ซึ่งในการดาวน์โหลดไฟล์ลงบอร์ด NetFPGA ต้องอาศัย NetFPGA Packet บนระบบปฏิบัติการลินุกซ์ (Linux OS) โดยบอร์ด NetFPGA 1 บอร์ด สามารถสร้างเส้นทาง (route) เพื่อทำการส่ง packet ที่ครอบคลุมถึงเครือข่ายย่อย (subnet) และสามารถนำบอร์ด NetFPGA หลายบอร์ดทำการติดตั้งบนเครื่องคอมพิวเตอร์เครื่องเดียวกันได้



ภาพที่ 2-2 ภาพ Block diagram แสดงถึงส่วนประกอบของบอร์ด NetFPGA

### 2.1.1 ส่วนประกอบของบอร์ด NetFPGA

#### 1. SRAM และ DRAM

- SRAM คือ Static Random Access Memory ใช้ในการเก็บตารางของการส่งข้อมูลมีขนาดทั้งหมด 4.5 เมกกะไบต์ มีหน่วยความจำ ZBT (Zero-Bus Turnaround) ทำงานแบบขนานกัน (Parallel) โดยใช้ขนาดลอจิกในการทำงานเท่ากัน มีขนาดเท่ากับ 18 เมกะบิต

- DRAM คือ Dynamic Random Access Memory เป็นแบบ Double-Data Rate Random Access Memory (DDR2 DRAM) ใช้ในการเก็บ packet แบบชั่วคราว (Buffering Packet) มีขนาดทั้งหมด 64 เมกกะไบต์

2. GigE (1 Gigabit/second Ethernet) จำนวน 4 port ซึ่งสามารถเชื่อมต่อกับหัวสายแลน RJ45 รองรับ Cat5E และ Cat6



3. Xilinx Virtex II Pro 50 FPGA เป็นชิปประมวลผล มีขนาดลอจิก (Logic) 53,136 โลจิกเซลล์ (Logic Cells) โดยสามารถใช้ JTAG cable ในการเชื่อมต่อเพื่อสั่งการให้ Xilinx Chip ทำงาน

4. Interface PCI สำหรับเชื่อมต่อกับเครื่องคอมพิวเตอร์ หรือ เครื่องเซิร์ฟเวอร์ โดยทำงานเชื่อมกันระหว่าง Xilinx Virtex II Pro 50 FPGA กับ GigE ทั้ง 4 port

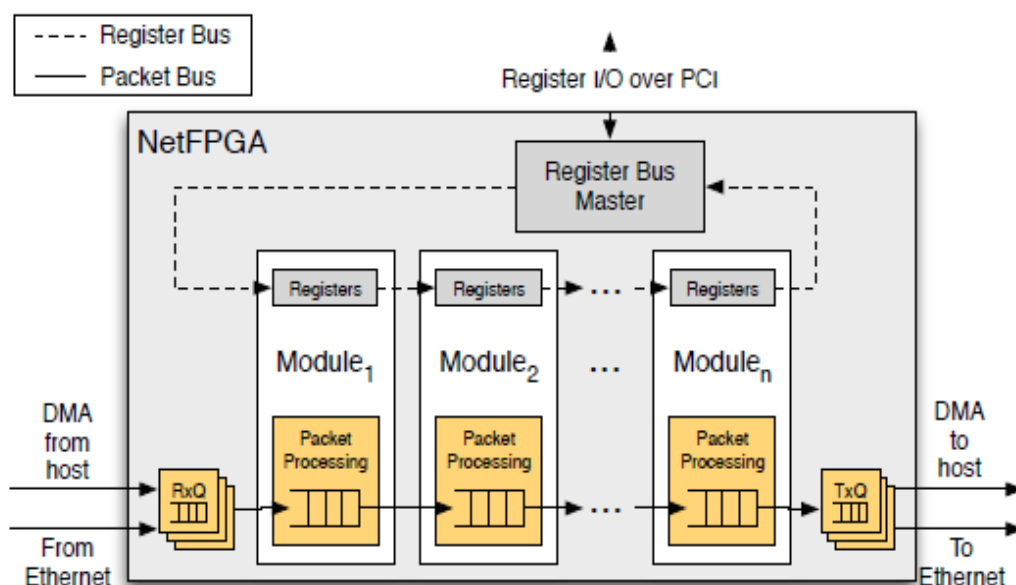
### 2.1.2 ภาษาที่ใช้ในการพัฒนาบอร์ด NetFPGA

บอร์ด NetFPGA เป็นบอร์ดที่สามารถโปรแกรมลงบอร์ดได้ ซึ่งทำให้บอร์ด NetFPGA มีความสามารถในการเปลี่ยนเป็นอุปกรณ์ในระบบเครือข่ายต่าง ๆ ได้ โดยวิธีการดาวน์โหลดไฟล์ประเภทไบนารี (Binary File) ลงไปในบอร์ด ซึ่งการพัฒนาบอร์ด NetFPGA เพื่อให้เปลี่ยนเป็นอุปกรณ์ต่าง ๆ ได้นั้น ต้องใช้ภาษาเวอริล็อก (Verilog Language) ในการพัฒนาสำหรับการดาวน์โหลดไฟล์ลงบอร์ด NetFPGA จำเป็นต้องลงแพ็คเกจ NetFPGA ก่อนในการที่จะพัฒนาบอร์ด NetFPGA บนระบบปฏิบัติการลินุกซ์ (Linux OS)

## 2.2 การทำงานพื้นฐานของบอร์ด NetFPGA

### 2.2.1 การทำงานของโมดูล

จากการศึกษาทำให้ทราบถึงการส่งข้อมูลเพื่อทำการสื่อสารสถานะการทำงานของแต่ละโมดูลย่อย โดยการทำงานสามารถทำได้ดังนี้



ภาพที่ 2-3 แสดงสถานะการทำงานของแต่ละโมดูล

จากภาพที่ 2-3 แสดงการทำงานของสถาปัตยกรรมภายใน NetFPGA ทั้งหมดที่มี โดยขั้นตอนแต่ละส่วนจะถูกเชื่อมกันด้วย bus ซึ่งในที่นี้คือ packet bus และ register bus ที่นำมาใช้ในการถ่ายโอนข้อมูลจากสถานะทำงานหนึ่งไปสู่อีกสถานะการทำงานหนึ่ง โดยใช้ FIFO packet-based push interface ขนาด 64 บิต ทำงานที่ 125 MHz (อัตรารวมของ 8 Gbps) โดยที่เลือกใช้การทำงานแบบ FIFO ในการทำงานนั้น เนื่องจาก FIFO มีความได้เปรียบในเรื่องของการทำงานแบบแยกป็นสถานะ และยังเป็นรูปแบบการทำงานที่ง่ายที่สุดที่สามารถใช้ในการส่งผ่านข้อมูลและจัดการควบคุมการไหลได้อย่างมีประสิทธิภาพ และมีประสิทธิภาพสูงสุด

## 2.2.2 ส่วนประกอบของบอร์ด NetFPGA เชิงพัฒนา มี 3 ส่วนประกอบด้วยกัน คือ

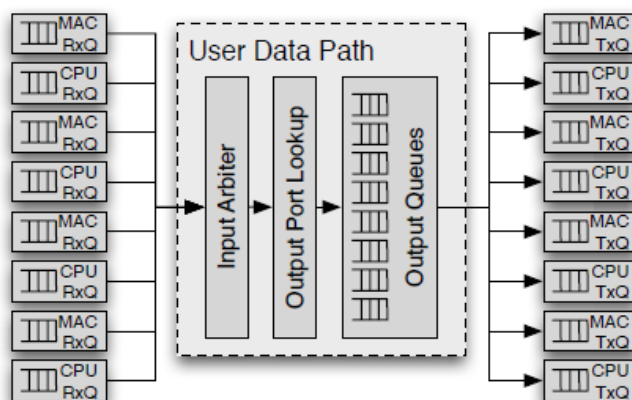
### 1. Hardware

จากที่เราได้กล่าวไปแล้วในข้างต้น NetFPGA เป็นแพลตฟอร์มฮาร์ดแวร์ ดันทุนต่ำ Reconfigurable เหมาะสำหรับอุปกรณ์เครือข่ายความเร็วสูง NetFPGA รวมถึงทรัพยากรตรรกะหน่วยความจำและการเชื่อมต่อแบบ Gigabit Ethernet จำเป็นในการอุปกรณ์เครือข่ายต่างสร้างสวิตช์ที่เรเตอร์และหรืออุปกรณ์รักษาความปลอดภัย

### 2. Gateway

เป็นส่วนของการทำงานที่เชื่อมต่อระหว่างการทำงานของ Hardware และ Software เราสามารถอธิบายการทำงานในส่วนนี้ได้โดยใช้แนวความคิดของ Reference pipeline เป็นแนวทางในการศึกษา

Reference pipeline of NetFPGA เป็นขั้นตอนกระบวนการที่เป็นพื้นฐานของการทำงานของทุกอุปกรณ์ ประกอบด้วย 5 ขั้นตอน คือ Input, Input Arbiter, Output Port Lookup, Output Queues, Output



ภาพที่ 2-4 แสดง “Reference Pipeline” สถานการณ์ทำงานพื้นฐานของบอร์ด NetFPGA

### ส่วนที่ 1 Input

การป้อนข้อมูล เป็นขั้นตอนแรกในการทำงานประกอบด้วยคิวหลายอย่างที่เรารู้จักว่า RxQueue โดยคิวเหล่านี้ได้รับแพ็คเก็ตจากพอร์ต IO เช่นพอร์ต Ethernet PCI และ DMA และอินเตอร์เฟซที่เหลือของระบบ การออกแบบในภาพด้านบนมี 4 Ethernet Rx queues และ 4 CPU DMA queues ทั้งสองจะแสดงข้อมูลดังตารางต่อไปนี้

ตารางที่ 2-1 แสดงแพ็คเก็ตเกิดใน DMA ที่ได้รับการส่งมาจากซอฟต์แวร์ออกจากอินเตอร์เฟซ nf2cx

RxQueue 0	Ethernet port 0
RxQueue 1	DMA port 0
RxQueue 2	Ethernet port 1
RxQueue 3	DMA port 1
RxQueue 4	Ethernet port 2
RxQueue 5	DMA port 2
RxQueue 6	Ethernet port 3
RxQueue 7	DMA port 3

### ส่วนที่ 2 Input Arbiter

เป็นขั้นตอนในการตัดสินใจของอินพุตที่ Rx queues ที่จะให้บริการต่อไปและดึงแพ็คเก็ตจากที่ Rx queues และส่งต่อไปยังโมดูลต่อไปในการทำงาน

### ส่วนที่ 3 Output Port Lookup

เป็นขั้นตอนของโมดูลในการค้นหาพอร์ตการส่งออก ซึ่งในส่วนนี้จะเป็นผู้รับผิดชอบต่อการตัดสินใจเลือกพอร์ตที่จะส่งแพ็คเก็ตออกไป

### ส่วนที่ 4 Output Queues

เมื่อแพ็คเก็ตผ่าน Output Port Lookup จะขึ้นอยู่กับ การตัดสินใจของแพ็คเก็ต อยู่ในขณะนี้ส่งให้โมดูลคิวเอาต์พุตที่เก็บแพ็คเก็ตในการรอคิวการส่งออกสอดคล้องกับพอร์ตออกจนกว่า Rx queues คือพร้อมที่จะยอมรับ แพ็คเก็ตสำหรับการส่ง

### ส่วนที่ 5 Output

Tx queues มีความคล้ายคลึงกับ Rx queues โดยทั้งสองส่วนนี้ต่างเป็นส่วนการทำงานที่ส่งแพ็คเก็ตออกจากพอร์ต IO แทนที่การรับข้อมูลเข้าในช่วงแรกของการทำงาน

### 3. Software

เป็นส่วนบนสุดของการทำงานที่ใช้คำสั่งในการควบคุมการทำงานและพฤติกรรมของอุปกรณ์ให้เป็นไปตามที่เราต้องการ โดยจะใช้ภาษา Verilog ในการทำงานสังการ และเข้ากระบวนการพัฒนาเพื่อให้ออกมาเป็น binary file ที่สามารถทำให้ NetFPGA เข้าใจการทำงาน และประมวลผลตามที่เราสังการได้

#### 2.3 โพรโทคอล ARP

ARP (Address Resolution Protocol) เป็นโพรโทคอลเป็นโพรโทคอลชนิดหนึ่งที่เป็นตัวกลางในการสื่อสารที่ทำหน้าที่ในการจับคู่ระหว่างไอพีแอดเดรส ซึ่งเป็นแอดเดรสทาง logical กับฮาร์ดแวร์แอดเดรสซึ่งเป็นแอดเดรสทาง physical (Map IP Address  $\leftrightarrow$  MAC Address) ที่เชื่อมโยงเครือข่ายของระบบ เพื่อให้สามารถสื่อสารกันระหว่างระบบเครือข่ายต่าง ๆ ได้ สามารถส่งข้อมูลระหว่างคอมพิวเตอร์ที่ติดต่อกัน โดยมีฮาร์ดแวร์สร้างเฟรมข้อมูลแล้วโพรโทคอล ARP จะนำข้อมูลเหล่านั้นเข้าที่เครื่อง host ในระบบเครือข่ายต่อไป

โพรโทคอล ARP จะทำหน้าที่นี้การทำงานของ ARP เมื่อแพ็คเก็ตเข้ามาที่ระบุเครื่อง host ในระบบเครือข่ายมาถึง Gateway เครื่องที่ Gateway จะเรียกโปรแกรม ARP ให้หาเครื่อง host หรือ MAC address ที่ตรงกับ IP address โปรแกรม ARP จะหาใน ARP cache เมื่อพบแล้วจะแปลงแพ็คเก็ต เป็นแพ็คเก็ตที่มีความยาวและรูปแบบที่ถูกต้อง เพื่อส่งไปยังเครื่องที่ระบุไว้ แต่ถ้าไม่พบ ARP จะกระจายแพ็คเก็ตในรูปแบบพิเศษ (ARP Request) ไปยังเครื่องทุกเครื่องในระบบ และถ้าเครื่องใดเครื่องหนึ่งทราบว่า IP address ตรงกันก็จะตอบกลับมาที่ ARP โปรแกรม ARP จะปรับปรุง ARP cache และส่งแพ็คเก็ตไปยัง MAC address หรือเครื่องที่ตอบมาโพรโทคอล ARP ได้กำหนดไว้เป็นมาตรฐานภายใต้ RFC 826 โดยการทำงานของ ARP จะมีรูปแบบการทำงานในแบบ Broadcast ดังนั้นเครือข่ายที่ใช้กับโพรโทคอล ARP ได้จึงต้องเป็นเครือข่ายที่มีการทำงานในแบบ Broadcast ซึ่งระบบแลนส่วนใหญ่จะมีการทำงานเป็นแบบ Broadcast อยู่แล้ว จึงสามารถทำงานร่วมกันกับโพรโทคอล ARP ได้เป็นอย่างดี

### 2.3.1 โครงสร้างของเฟรม ARP

Hardware		Protocol
HLEN	PLEN	Operation
Source Hardware Address (0-3)		
Source Hardware Address (4-5)		Source IP Address (0-1)
Source IP Address (2-3)		Destination Hardware Address (0-1)
Destination Hardware Address (0-1)		
Destination IP Address		

ภาพที่ 2-5 แสดงรูปแบบเฟรม ของ ARP ที่ใช้กับโปรโตคอล IP สำหรับรายละเอียดของแต่ละฟิลด์ภายในเฟรม ARP นั้น มีดังต่อไปนี้

#### – Hardware Type

ใช้ 16 บิต ที่ระบุประเภทของฮาร์ดแวร์ที่ใช้ใน Network Interface Layer หลังจากได้รับเฟรมข้อมูลของ ARP โหนด IP ตรวจสอบว่าค่า Hardware Type ของ ARP ว่าตรงกับค่า Hardware Type ของอินเตอร์เฟซที่เฟรมข้อมูลที่ได้รับมาของ ARP หากไม่ตรง เฟรมข้อมูลก็จะถูกยกเลิกหรือโยนทิ้งไป สำหรับรายชื่อทั้งหมดของ ARP Hardware Type ดูได้ที่

<http://www.iana.org/assignments/arp-parameters/arp-parameters.xhtml>

#### ตารางที่ 2-2 ตัวอย่าง ARP Hardware Type Values

Hardware Type Value	Data Link Layer Technology
1 (0x00-01)	Ethernet
6 (0x00-06)	Networks IEEE 802.5 (Token Ring)
15 (0x00-0F)	Frame Relay
16 (0x00-10)	ไม่ตรงกัน Transfer Mode (ATM)

#### – Protocol Type

ใช้ 16 บิต สำหรับระบุชนิดของโปรโตคอลที่ ARP ใช้ในการค้นหาคำแหน่ง Address ฟิลด์นี้ใช้ค่าเดียวกันกับฟิลด์ Ethernet II Ether Type สำหรับการแก้ปัญหาหมายเลข IP

Address ที่ช่องฟิลด์ข้อมูล Protocol Type ต้องกำหนด Ether Type เป็น 0x0800 หลังจากได้รับเฟรมข้อมูลของ ARP, โหนด IP ยืนยันว่าช่องฟิลด์ ARP Protocol Type ถูกกำหนดเป็น 0x0800 หากไม่ใช่ 0x0800 เฟรมข้อมูลก็จะถูกโยนทิ้งหรือไม่สนใจเฟรมข้อมูลนั้นนั่นเอง

– Hardware Address Length (HLEN)

ใช้ 8 บิต สำหรับที่ระบุความยาวของจำนวนไบต์ของ Hardware Address ของผู้ส่งและ Hardware Address ของผู้รับด้วย สำหรับ Ethernet และ Token Ring ช่อง Hardware Address Length กำหนดเป็น 6 สำหรับ Frame Relay ฟิลด์ Hardware Address Length โดยทั่วไปจะกำหนดเป็น 2 (ทั่วไปใช้ 2 ไบต์สำหรับฟิลด์ Frame Relay Address)

– Protocol Address Length (PLEN)

ใช้ 8 บิต ที่ระบุขนาดของจำนวนไบต์ของหมายเลขโปรโตคอลในส่วนทั้งผู้ส่งและผู้รับ สำหรับโปรโตคอล IP ความยาวของหมายเลข IP Addresses ถูกกำหนดไว้จำนวน 4 ไบต์

– Operation (Opcode)

ใช้ 16 บิต เพื่อใช้ระบุชนิดของเฟรมข้อมูล ARP ตาราง 2-2 รายการค่า ARP Operation ที่มักใช้งานส่วนใหญ่ สำหรับรายการ ARP Operation ทั้งหมดดูได้ที่

<http://www.iana.org/assignments/arp-parameters>

ตารางที่ 2-3 ARP Operation Values

ค่าของ Operation	ชนิดของ ARP Frame
1 (0x00-01)	ARP Request
2 (0x00-02)	ARP Reply
8 (0x00-08)	Inverse ARP Request
9 (0x00-09)	Inverse ARP Reply

– Sender Hardware Address (SHA)

ฟิลด์ที่เป็นความยาวของฟิลด์ Hardware Address Length และยังบรรจุค่า Hardware หรือ Data Layer Address ของเฟรมข้อมูล ARP ของผู้ส่ง สำหรับ Ethernet และ Token Ring ฟิลด์ SHA บรรจุค่า Mac Address ของโหนดที่ส่งเฟรมข้อมูลของ ARP

– Sender Protocol Address (SPA)

เป็นฟิลด์ที่เป็นความยาวของค่าของฟิลด์ Protocol Address Length และยังบรรจุ Protocol Address ของเฟรมข้อมูล ARP ของผู้ส่งสำหรับ IP ฟิลด์ SPA บรรจุ IP Address ของโหนดที่ส่งไปของเฟรมข้อมูล ARP

– Target Hardware Address (THA)

ฟิลด์ที่เป็นความยาวของค่าของฟิลด์ Hardware Address Length และมีฮาร์ดแวร์หรือที่อยู่ Data Link Layer ของเฟรมข้อมูล ARP ปลายทาง สำหรับ Ethernet และ Token Ring ฟิลด์ THA ถูกกำหนดเป็น 0x00-00-00-00-00-00 สำหรับเฟรม ARP Request และยังมีการกำหนด MAC Address ของ ARP Requester สำหรับเฟรมข้อมูลของ ARP Reply

– Target Protocol Address (TPA)

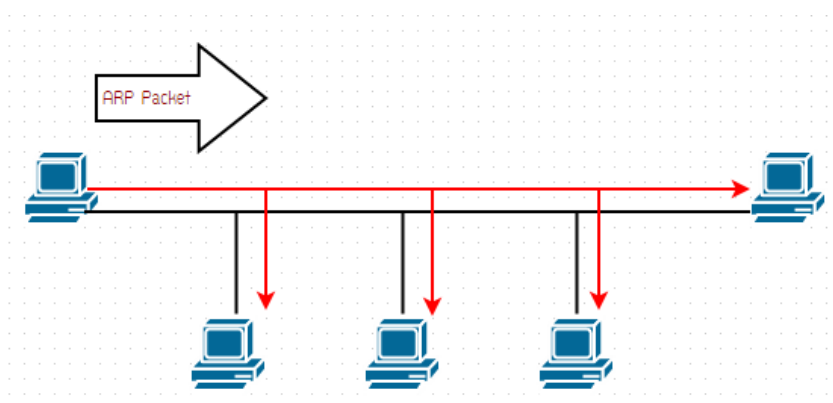
ฟิลด์ที่เป็นความยาวของค่าของฟิลด์ Protocol Address Length และยังบรรจุ Protocol Address ของเฟรมข้อมูลปลายทางของ ARP สำหรับ IP ช่องฟิลด์ TPA ถูกกำหนดให้กับ IP Address ที่ถูกจองในเฟรม ARP Request และถูกกำหนดให้ IP Address ของ ARP Requester ในเฟรมข้อมูล ARP Reply

### 2.3.2 การทำงานของโปรโตคอล ARP

การทำงานของ ARP เป็นเรื่องไม่ซับซ้อน มีเพียง 2 ขั้นตอนเท่านั้นคือ

#### 1. ARP Request

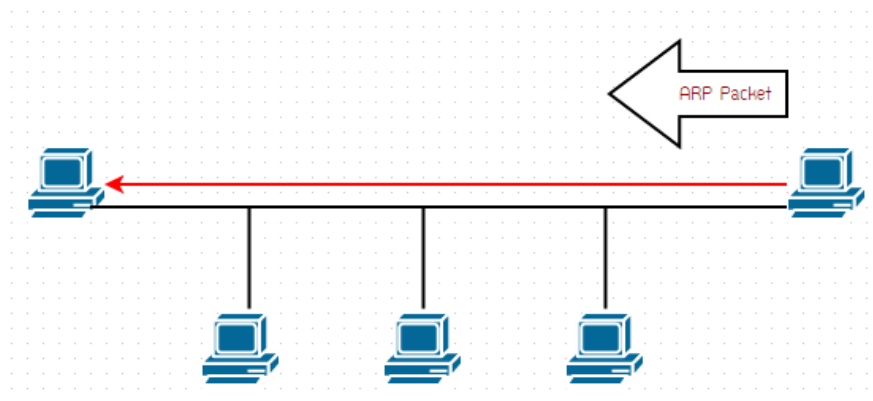
เมื่อเครื่องคอมพิวเตอร์ต้องการสอบถาม MAC Address จะทำการส่ง ARP packet ซึ่งภายในบรรจุ IP , MAC Address ของตนเอง และ IP Address ของเครื่องที่ต้องการทราบ MAC Address ส่วน MAC Address ปลายทางนั้น จะถูกกำหนดเป็น FF:FF:FF:FF:FF:FF ซึ่งเป็น Broadcast Address เพื่อให้ ARP packet ถูกส่งไปยังเครื่องทุกเครื่องที่อยู่ในเน็ตเวิร์กเดียวกัน



ภาพที่ 2-6 ARP Request จะถูกส่งไปยังเครื่องทุกๆเครื่องในเน็ตเวิร์ก

## 2. ARP Reply

เฉพาะเครื่องคอมพิวเตอร์ที่มี IP Address ตรงกับที่ระบุใน ARP Packet ซึ่งได้รับมาจากการทำ ARP Request จะทำการตอบกลับด้วย ARP Packet เช่นกัน โดยใส่ MAC Address และ IP Address ของตนเองเป็นผู้ส่ง และใส่ MAC Address และ IP Address ของเครื่องที่ส่งมาเป็นผู้รับ packet ที่ตอบกลับ



ภาพที่ 2-7 การตอบกลับด้วย ARP Reply

### 2.4 Reference Switch และ ARP-Path Switch

Reference Switch เป็น Project หนึ่งในแพ็คเกจพื้นฐานของบอร์ด NetFPGA ซึ่งอยู่ใน Directory ดังนี้ netfpga/project/reference\_switch การทำงานของ Reference Switch จะมีพฤติกรรมเหมือนกับ Switch Layer2 ซึ่งไม่สามารถจัดการกับปัญหาที่เกิดขึ้นในเน็ตเวิร์กได้ นอกจากจะมีการเรียกใช้ Spanning Tree เข้ามาช่วยในการจัดการ

ARP-Path Switch มีการทำงานที่แก้ปัญหของ Reference Switch ในเรื่องของการจัดการกับปัญหาในเน็ตเวิร์ก โดยใช้การทำงานของโปรโตคอล ARP ซึ่งการจัดการด้วยโปรโตคอล ARP นั้นมีข้อดีกว่าการใช้ Spanning Tree Protocol

#### 2.4.1 ข้อแตกต่างการทำงานของ ARP-Path Switch และ Reference Switch

- Reference Switch เมื่อมีแพ็คเกจเข้ามาใหม่ จะทำการเรียนรู้ (learning) ใหม่ตลอดเพื่อตรวจสอบเส้นทางอยู่เสมอ แต่ ARP-Path Switch จะทำการเรียนรู้เส้นทาง (learning) เฉพาะเวลาที่ได้รับแพ็คเกจที่มีการแจ้งความผิดปกติ ทำให้ในระบบเน็ตเวิร์กมีจำนวนแพ็คเกจที่ใช้ในการตรวจสอบเส้นทางน้อยกว่า

- การจัดการรูปที่เกิดขึ้นใน Reference Switch นิยมใช้ Spanning Tree Protocol ส่วน ARP-Path switch ใช้ ARP Protocol ที่เป็นโปรโตคอลพื้นฐาน ง่ายต่อการใช้งาน



– การแก้ปัญหาด้วย Spanning Tree Protocol ใน Reference Switch มีข้อจำกัดหลายประการ เช่น ขนาดของเน็ตเวิร์ก, เส้นทางที่เลือกใช้ในการสื่อสารไม่ได้เป็นเส้นทางที่สั้นที่สุด (low latency path) แต่ ARP-Path Switch ไม่มีข้อจำกัดด้านขนาดของเน็ตเวิร์ก และเส้นทางที่เลือกนั้นจะเป็นเส้นทางที่สั้นที่สุด (low latency path) รวมถึงสามารถทำ Load Balance ในการส่งข้อมูลได้

#### 2.4.2 เปรียบเทียบ Reference Switch กับ ARP-Path Switch

##### 1. Module การทำงาน

– Reference Switch มี 5 module

ตารางที่ 2-4 แสดง module ของ Reference Switch

<b>5 module</b>	1. ethernet_parser : eth_parser	4. small_fifo : dst_port_fifo
	2. mac_cam_lut : mac_cam_lut	5. op_lut_regs : op_lut_regs
	3. small_fifo : input_fifo	

– ARP-Path Switch มี 6 module

ตารางที่ 2-5 แสดง module ของ ARP-Path Switch

<b>6 module</b>	1. fallthrough_small_fifo : input_fifo	4. parse_pkt : parse_pkt
	2. reverse_bytes	5. fallthrough_small_fifo : parse_fifo
	3. generic_regs : arppath_regs	6. lookup_module : lt, bt, rt, hot

##### 2. State การทำงาน

– Reference Switch มี 4 state

ตารางที่ 2-6 แสดง state ของ Reference Switch

<b>4 state</b>	1. wait_till_done_decoder	3. skip_hdrs
	2. write_hdr	4. wait_eop

- ARP-Path Switch มี 15 state

ตารางที่ 2-7 แสดง state ของ ARP-Path Switch

15 state	1. wait_for_parsing	6. drop_pkt	11. send_eth_src_mac_llc_dsap_ssap
	2. pathfall_check_tables	7. sent_pkt	12. send_llc_ctrl_apppath_type_dst_mac
	3. pathreq_check_table	8. send_arppath	13. send_apppath_src_mac
	4. do_regular_lookups	9. send_eth_dst_src_mac	14. send_fill
	5. regular_check_table	10. send_arppath_and_drop	15. arp_check_table

### 3. ตารางที่ใช้ในการทำงานของ ARP-Path Switch

ตารางที่ 2-8 แสดง table ของ ARP-Path Switch

4 table	1. learning table (LT)	4. host table (HoT)
	2. broadcast table (BT)	5. hello table (HeT)
	3. repair table (RT)	

- Learning Table เป็นตารางที่ใช้ในการส่งแพ็คเก็ตต่อออกไปยัง Output Port ซึ่งภายในประกอบด้วย Mac Address และ Port ใช้ในการจับคู่ระหว่างเพื่อใช้ในการเลือก Port เพื่อส่งแพ็คเก็ตไปยังเส้นทางที่ถูกต้อง

- Broadcast Table เป็นตารางที่ใช้ในการส่งแพ็คเก็ตแบบ Broadcast

- Repair Table เป็นตารางที่ใช้เก็บ Mac Address ชั่วโมงของเครื่องที่มีความผิดปกติเพื่อทำการแก้ไข หาเส้นทางใหม่

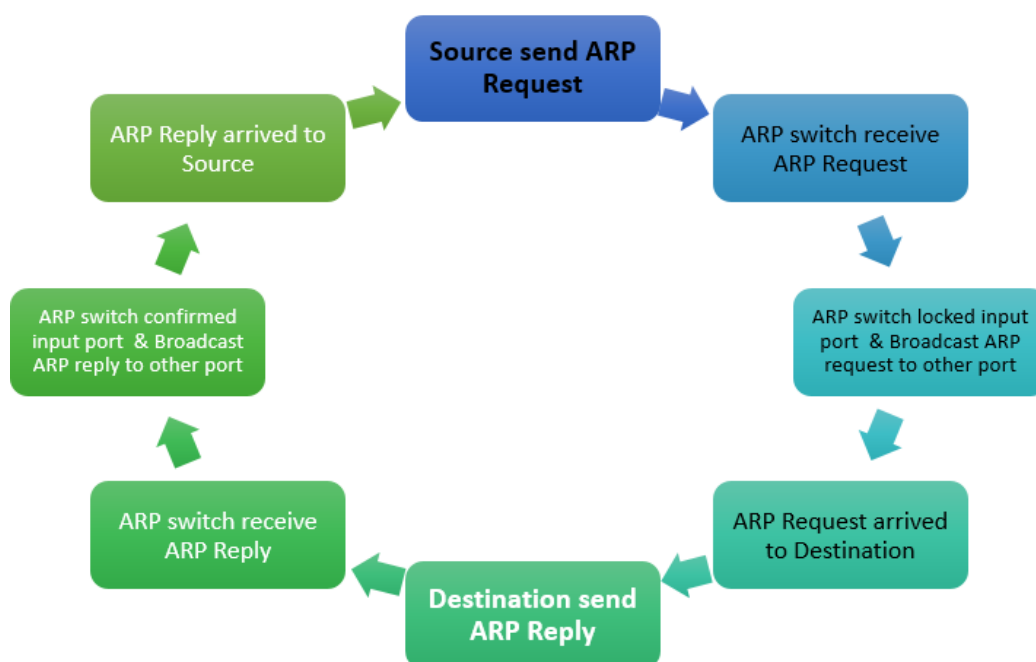
- Host Table เป็นตารางที่ใช้เก็บ Mac Address ของเครื่องคอมพิวเตอร์ที่ต่อ ซึ่งข้อมูลภายในตารางนี้จะอยู่ในตาราง Learning Table อยู่แล้ว สามารถใช้ได้จาก Learning Table เลย

- Hello Table เป็นตารางที่ใช้เก็บรายชื่อของ Port พร้อมกับค่า True/False ถ้าหากเป็น True คือ port นั้นเชื่อมต่อกับ ARP-Path Switch เครื่องอื่นอยู่ แต่ถ้าเป็น False คือ port นั้นไม่ได้ทำการเชื่อมต่อกับ ARP-Path Switch ทำงานเวลาเกิดเส้นทางที่มีความผิดปกติ Hello packet เป็นชนิดของแพ็คเก็ตที่ ARP-Path Switch ใช้ในการส่งหากัน เพื่อเรียนรู้ว่ามีเครื่องอื่นต่อกับ Port ใด ๆ หรือไม่

โดยการทำงานแบบ 5 ตารางนั้นมีความซ้ำซ้อนของตาราง ในการทำงานแบบใหม่ของ ARP-Path Switch จะใช้เพียง 3 ตาราง คือ Learning Table, Repair Table, Hello Table

## 2.5 การทำงานของ ARP-Path Switch

### 2.5.1 ภาพรวมการทำงานของ ARP-Path Switch



ภาพที่ 2-8 แสดงกลไกการเมื่อเครื่องคอมพิวเตอร์ต้นทาง ต้องการติดต่อกับเครื่องปลายทาง

จากภาพที่ 2-8 แสดงกลไกการทำงานโดยสามารถอธิบายเป็นขั้นตอนได้ ดังต่อไปนี้เมื่อ Source ต้องการติดต่อกับ Destination

1. Source send ARP Request คือ Source จะทำการส่ง ARP Request ไปยัง port ที่เชื่อมกับ Source ทุก port (Broadcast) พร้อมกับ MAC Address ของ Source แล้ว
2. ARP switch receive ARP Request คือ เมื่อได้รับ ARP Request แล้ว จะมีการเปลี่ยนสถานะของ port ที่ได้รับ ARP Request ให้มีสถานะเป็น locked ด้วย Mac address ของ Source และ จะทำการ Broadcast ARP Request ต่อไปเรื่อย ๆ ถ้าหาก ARP Request ถูกส่งไปยัง เครื่องที่มี port เป็นสถานะ locked แล้ว ARP Request จะถูกละทิ้ง (discard)

3. ARP Request arrived to Destination คือ เมื่อ ARP Request Packet ที่มีข้อมูลของ Source MAC Address เดินทางมาถึง Destination จะทำการตอบกลับด้วยการ Broadcast ARP Reply ไปยัง port ที่เชื่อมต่ออยู่

4. ARP switch receive ARP Reply คือ เมื่อได้รับ ARP Reply แล้ว และมี port หนึ่งเป็นสถานะ locked แล้ว จะทำการเปลี่ยน port ที่รับ ARP Reply มาเป็น confirmed ด้วย Mac address ของ Destination ถ้าไปถึงเครื่องที่ไม่มี port ในสถานะ locked แล้ว ARP Reply จะถูกละทิ้ง

5. ARP Reply arrived to Source คือ เมื่อ ARP Reply มาถึง Source แสดงว่ามีเส้นทางในการเชื่อมต่อสื่อสารกันระหว่าง Source และ Destination สมบูรณ์แล้ว

2.5.2 การจัดการปัญหาการเกิด loop ของ ARP-Path Switch ใช้ ARP Request และ ARP Reply ของ ARP Protocol ในการกำหนดเส้นทาง โดยเส้นทางที่ได้จากการใช้ ARP-Request และ ARP-Reply เพื่อหาเส้นทางสื่อสารที่เป็นเส้นทางที่สั้นที่สุด (low latency path)

– ARP Request เมื่อเครื่องคอมพิวเตอร์ต้นทางต้องการติดต่อไปยังเครื่องปลายทาง เครื่องต้นทางจะทำการส่ง ARP Request ออกไปทุก Port เรียกว่าการ Broadcast ไปหาเครื่องปลายทาง เมื่อ

– ARP Reply เมื่อเครื่องปลายทางได้รับ ARP Request แล้วจะตอบกลับด้วย Mac address ของตนเองด้วย ARP Reply เมื่อเครื่องต้นทางได้รับ แสดงว่าเส้นทางสื่อสารระหว่างเครื่องต้นทางและปลายทางเสร็จเรียบร้อยแล้ว สามารถใช้ในการส่งข้อมูลหากันได้ แม้จะเกิด loop ในระบบเน็ตเวิร์กที่ใช้งาน

2.5.3 การทำ Forwarding จะเกี่ยวข้องกับตารางที่ใช้ในการทำงาน 2 ตาราง คือ Learning Table และ Broadcast Table

– Learning Table ใช้เก็บ Mac address และ port ที่จะใช้ในการจับคู่ของ Mac address กับ Output Port โดยค่าอายุของข้อมูลภายในตารางต้องมีค่ามากกว่า 10 วินาที ตารางจะถูกปรับเปลี่ยนได้ต่อเมื่อได้รับ ARP Request หรือ ARP Reply เท่านั้น

– Broadcast Table หรือ Blocking Table

```

if dst_mac is BROADCAST or MULTICAST then
  if (src_mac is not in BT) || (src_mac is in BT && input_port == BT_port) then
    if (ARP Request && src_mac is not in LT) then
      Update LT (new src_mac and input_port entry)
    Update BT (new src_mac and input_port entry, or refresh timer)
    Broadcast frame (through all ports but the incoming one)
  else
    Discard frame
else if dst_mac is UNICAST then
  if (ARP Reply && src_mac is not in LT) then
    Update LT (new src_mac and input_port entry)
  Update LT (refresh timer of dst_mac)
  Forward frame (through port associated to LT entry of dst_mac)

```

ภาพที่ 2-9 รหัสเทียมการทำ Forwarding แสดงเงื่อนไขการทำงานของการทำงานการทำ Forwarding  
ซึ่งแสดงในรูปแบบของ รหัสเทียม (Pseudo code)

2.5.4 การทำ Path Recovery เมื่อเกิดความผิดปกติในเน็ตเวิร์ก ทำให้ตารางที่ใช้ในการจับคู่ Mac address กับ Output Port คือ Learning Table ต้องถูกปรับเปลี่ยนค่าเพื่อให้สามารถทำงานได้ปกติ เมื่อมีการตรวจพบที่เกิดความผิดพลาดที่มีผลต่อการส่งข้อมูล ARP-Path Switch จะทำการสร้าง PathFail แพ็คเก็ต ซึ่งประกอบด้วย Mac Address ของเครื่องที่เกิดความผิดปกติ และทำ Broadcast ออกไปยังทุก Port ที่ทำการเชื่อมต่ออยู่ทุกเครื่องที่ได้รับ PathFail จะทำการตรวจสอบกับตารางของตนเองว่ามีข้อมูลตรงกับ Mac address ที่มีความผิดปกติหรือไม่ ถ้าหากมีจะลบข้อมูลนั้นทิ้ง แล้วตอบกลับด้วย LinkReply เพื่อแจ้งให้มีการซ่อมแซมเส้นทางในการส่งข้อมูลในเฉพาะส่วนที่เกิดความผิดพลาด ซึ่งไม่ได้ทำการซ่อมแซมเส้นทางใหม่ทั้งหมด

```

if dst_mac is BROADCAST or MULTICAST then
  if (src_mac is not in BT) || (src_mac is in BT && input_port == BT_port) then
    if ((ARP Request && src_mac is not in LT) || LinkFail) then
      Update LT (new src_mac and input_port entry)
    Update BT (new src_mac and input_port entry, or refresh timer)
    if LinkFail && encapsulated mac directly connected then
      Create LinkReply packet with:
        src_mac= encapsulated mac directly connected (mac being repaired)
        dst_mac=src_mac of LinkFail
      Forward frame (through input_port)
    else
      Broadcast frame (through all ports but the incoming one)
  else
    Discard frame
else if dst_mac is UNICAST then
  if ((ARP Reply && src_mac is not in LT) || LinkReply) then
    Update LT (new src_mac and input_port entry)
  Update LT (refresh timer of dst_mac)
  Forward frame (through port associated to LT entry of dst_mac)

```

ภาพที่ 2-10 รหัสเทียมของการทำ Forwarding และ Path Recovery

### 2.5.5 อธิบาย code การทำงานของ Reference switch และ ARP-Path Switch

- Reference switch มี state การทำงานทั้งหมด 4 state

#### 1. WAIT\_TILL\_DONE\_DECODE

```
WAIT_TILL_DONE_DECODE: begin
    if(!dst_port_fifo_empty) begin
        dst_port_rd      = 1;
        state_next       = WRITE_HDR;
        in_fifo_rd_en    = 1;
    end
end
end
```

ภาพที่ 2-11 code การทำงานของ state WAIT\_TILL\_DONE\_DECODE เมื่อมีแพ็คเก็ตเข้ามาใน FIFO แล้วจะนำแพ็คเก็ตนั้นมาทำงานถอดรหัส (decode)

#### 2. WRITE\_HDR

```
WRITE_HDR: begin
    if(out_rdy) begin
        if(in_fifo_ctrl_dout==`IO_QUEUE_STAGE_NUM) begin
            out_data[`IOQ_DST_PORT_POS + NUM_OUTPUT_QUEUES - 1: `IOQ_DST_PORT_POS] = dst_ports_latched;
        end
        out_wr      = 1;
        in_fifo_rd_en = 1;
        state_next  = SKIP_HDRS;
    end
end
end
```

ภาพที่ 2-12 code การทำงานของ state WRITE\_HDR เมื่อทราบรายละเอียดของแพ็คเก็ตแล้วจะนำมาวิเคราะห์พิจารณาว่าแพ็คเก็ตนั้นต้องการส่งไปยังผู้รับเครื่องไหนในเน็ตเวิร์ค (destination) และหา Output Port ที่จะใช้ในการส่งแพ็คเก็ต

#### 3. SKIP\_HDRS

```
SKIP_HDRS: begin
    if(in_fifo_ctrl_dout==0) begin
        state_next = WAIT_EOP;
    end
    if(!in_fifo_empty & out_rdy) begin
        in_fifo_rd_en = 1;
        out_wr        = 1;
    end
end
end
```

ภาพที่ 2-13 code การทำงานของ state SKIP\_HDRS เป็นการพิจารณาว่าเมื่อใดจะเป็นการดึงข้อมูลมาถึงส่วนสุดท้ายของส่วนหัวของแพ็คเก็ต (packet header)

#### 4. WAIT\_EOP

```

WAIT_EOP: begin
    if(in_fifo_ctrl_dout!=0)begin
        if(out_rdy) begin
            state_next = WAIT_TILL_DONE_DECODE;
            out_wr      = 1;
        end
    end
    else if(!in_fifo_empty & out_rdy) begin
        in_fifo_rd_en = 1;
        out_wr        = 1;
    end
end
end // case: WAIT_EOP

```

ภาพที่ 2-14 code การทำงานของ state WAIT\_EOP เป็นส่วนที่ทำการเขียนข้อมูลให้แฟล็กเกิดเพื่อทำการส่งออกไปยัง Output Port

- ARP-Path Switch มี state การทำงานทั้งหมด 15 state

##### 1. WAIT\_FOR\_PARSING

เมื่อมีแฟล็กเกิดเข้ามาใน FIFO Queue จะถูกนำมาวิเคราะห์ทันที โดยจะถูกนำมาพิจารณาว่าแฟล็กเกิดที่เข้ามานั้นเป็นชนิดใด สามารถแบ่งได้เป็น 2 ชนิด

- ชนิดพิเศษต้องมีการทำงานที่เฉพาะ ได้แก่ Hello, From\_CPU, From\_Switch, PathFail, PathReq, ARP ซึ่งจะแยกการทำงานที่เฉพาะแต่ละชนิดของแฟล็กเกิด
- ชนิดปกติ การทำงานถัดไปคือจะไปทำการตรวจสอบตารางแบบปกติใน

สถานะ DO\_REGULAR\_LOOKUPS

##### 2. PATHFAIL\_CHECK\_TABLE

เมื่อแฟล็กเกิดที่เข้ามาเป็นชนิด PathFail แสดงว่าเกิดความผิดปกติต้องทำการ Repair ถ้าหากมี port ที่เกี่ยวข้องกับ Mac Address ที่เกิดความผิดพลาดจะแล้ว port นั้นมีการใช้เพื่อจับคู่ส่งข้อมูลใน Learning table แล้วต้องทำการร้องขอเส้นทางใหม่ ใน SEND\_ARPPATH\_AND\_DROP ถ้าหากไม่ได้มี port เกี่ยวข้องกับ Mac Address ที่เกิดความผิดพลาด ก็จะทำการ Broadcast แฟล็กเกิด PathFail ต่อไปเพื่อแจ้งให้เครื่องอื่น ๆ ทราบ

##### 3. PATHREQ\_CHECK\_TABLE

เมื่อแฟล็กเกิดที่เข้ามาเป็นชนิด Path Request จึงจะเข้ามาในสถานะนี้เพื่อจัดการ

##### 4. DO\_REGULAR\_LOOKUPS

เป็นสถานะที่ใช้ตรวจสอบว่าเกิดการชนกันของแฟล็กเกิดเมื่อใด ถ้าหากไม่เกิดการชนกันจะทำการตรวจสอบตารางต่อไปใน REGULAR\_CHECK\_TABLE

#### 5. REGULAR\_CHECK\_TABLE

เป็นการนำ Mac Address ที่เป็นเครื่องปลายทาง (destination) ไปตรวจสอบกับตารางทุกตาราง

#### 6. DROP\_PKT

ใช้ในการละทิ้ง (drop) แพ็คเก็ต

#### 7. SEND\_PKT

ใช้ในการส่งแพ็คเก็ตออก

#### 8. SEND\_ARPPATH

ใช้ในการส่งแพ็คเก็ตออกไปยัง Output Port ในกรณีที่ต้องการซ่อมแซม (repair) เส้นทางในการส่งแพ็คเก็ตในเน็ตเวิร์ก

#### 9. SEND\_ETH\_DST\_SRC\_MAC

ใช้ในการสร้างและส่งแพ็คเก็ตไปยัง Output Port ตามมาตรฐานของ IPv4

#### 10. SEND\_ARPPATH\_AND\_DROP

ใช้ในการสร้างแพ็คเก็ตสำหรับการซ่อมแซม (repair) เฉพาะกรณี

#### 11. SEND\_ETH\_SRC\_MAC\_LL\_C\_DSAP\_SSAP

ใช้ในการสร้างแพ็คเก็ตสำหรับการซ่อมแซม (repair) เฉพาะกรณี

#### 12. SEND\_LL\_C\_CTRL\_ARPPATH\_TYPE\_DST\_MAC

ใช้ในการสร้างแพ็คเก็ตสำหรับการซ่อมแซม (repair) เฉพาะกรณี

#### 13. SEND\_ARPPATH\_SRC\_MAC

ใช้ในการสร้างแพ็คเก็ตสำหรับการซ่อมแซม (repair) เฉพาะกรณี

#### 14. SEND\_FILL

การส่งแพ็คเก็ตสำหรับการซ่อมแซม (repair) เส้นทาง การส่งแพ็คเก็ตจะถูกเติมด้วยข้อมูลต่าง ๆ ซึ่งประกอบด้วย LLC Ethertype

#### 15. ARP\_CHECK\_TABLES

เป็นส่วนในการจัดการและวิเคราะห์เพื่อป้องกันการเกิดลูปในเน็ตเวิร์ก จะมีการตรวจสอบคือ If(!learning\_is\_blocked\_in\_port) ถ้าหาก Port ที่เกี่ยวข้องกับ Mac Address ที่จะต้องใช้ในการส่งข้อมูล ไม่ถูกระงับการใช้งาน (blocked) มีข้อมูลเพื่อใช้ในการจับคู่ใน Learning Table และแพ็คเก็ตเป็นชนิด broadcast ถึงจะค่อยสร้างแพ็คเก็ตเพื่อ broadcast ข้อมูล แต่ถ้าหากกรณีที่แพ็ค



เกิดเป็นแบบ unicast คาดว่าเป็นแพ็คเก็ตที่สื่อสารกันระหว่างเครื่องต้นทางและปลายทาง จึงจะนำไปตรวจสอบกับตารางทั้งหมดก่อนเพื่อหา port ที่จะใช้ในการส่งในสถานะ DO\_REGULAR\_LOOKUPS เพื่อทำงานต่อไป

```

ARP_CHECK_TABLES: begin
  if (lookup_done) begin
    if (lookup_collisions) begin
      output_port_nxt = to_cpu_port;
      state_nxt       = SEND_PKT;
    end
    else if (pkt_is_broadcast_out) begin
      if (!port_in_het) begin
        /* #ERS & #CRC / Jun 2012 */
        /* reduce the frequency of updating the HoT entry*/
        if (curr_time_short - hello_time/8 > hot_lookup_timestamp) begin
          hot_mac = eth_src_mac_out;
          hot_mac_to_learn_vld = 1'b1;
        end
      end
      if (!learning_is_blocked_in_port) begin
        lt_mac = eth_src_mac_out;
        lt_mac_to_learn_vld = 1'b1;
        inc_lt_hit = lt_lookup_success;
        output_port_nxt = BROADCAST_ALL & ~decoded_src;
        state_nxt = SEND_PKT;
      end
      else begin
        state_nxt = DROP_PKT;
      end
    end // if (pkt_is_broadcast_out)
    else begin
      // packet is unicast ARP
      lt_mac = eth_src_mac_out;
      lt_mac_to_learn_vld = 1'b1;
      inc_lt_hit = lt_lookup_success;
      hot_mac = eth_src_mac_out;
      // do lookups for all tables except
      // for the LT Table, so we don't learn and
      // lookup at the same time.
      do_lookups = 4'hf;
      do_lookups[LT_TABLE] = 1'b0;
      // can't learn and lookup at the same time, so
      // push lookups to next state
      state_nxt = DO_REGULAR_LOOKUPS;
    end // else: !if(pkt_is_broadcast_out)
  end // if (lookup_done)
end // case: ARP_CHECK_TABLES

```

ภาพที่ 2-15 code การทำงานของ state ARP\_CHECK\_TABLE

## บทที่ 3

### ขั้นตอนการดำเนินงาน

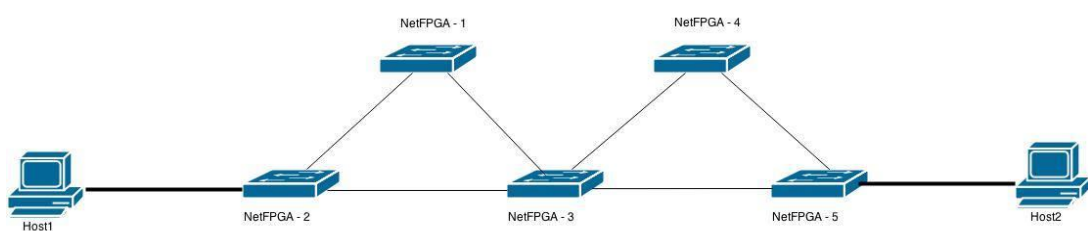
ในขั้นตอนการพัฒนาอุปกรณ์ต้นแบบ ARP-PATH switch จำเป็นต้องมีการติดตั้งซอฟต์แวร์ต่าง ๆ เพื่อใช้ในการพัฒนาและการทำให้บอร์ด NetFPGA สามารถใช้งานได้ รวมไปถึงการเชื่อมต่อเครื่องคอมพิวเตอร์เข้าด้วยกันตามโทโปโลยี ที่ได้ออกแบบ เพื่อให้พร้อมสำหรับดำเนินการทดลองพัฒนาอุปกรณ์ต้นแบบ ARP-PATH switch โดยวิธีการดำเนินงาน ดังต่อไปนี้

1. โทโปโลยีที่ใช้ในการทดลอง
2. ภาพรวมการดำเนินการทดลอง
3. การติดตั้งบอร์ด NetFPGA และ แฟลชเก็ต NetFPGA พื้นฐาน
4. การติดตั้งแฟลชเก็ต ARP-PATH Switch
5. การดาวน์โหลดโปรแกรมลงบอร์ด NetFPGA
6. ทดสอบการทำงาน ARP-PATH Switch

ซึ่งในแต่ละขั้นตอนสามารถอธิบายการดำเนินงานได้ดังนี้

#### 3.1 การเตรียมโทโปโลยีที่ใช้ในการทดลอง

ในการทดลองได้ออกแบบโครงสร้างเพื่อที่จะใช้ในการทดสอบประสิทธิภาพของ ARP\_PATH switch รวมทั้ง ซึ่งในการทดลองได้ออกแบบโครงสร้างที่แสดงให้เห็นถึงประสิทธิภาพอย่างง่ายและเหมาะสมในระดับการทดลองที่ใช้อุปกรณ์ต้นแบบบอร์ด NetFPGA มาเป็นอุปกรณ์ให้เป็น ARP\_PATH switch โดยมีโทโปโลยีที่ออกแบบมาใช้ในการทดลอง มีรายละเอียดดังต่อไปนี้



ภาพที่ 3-1 โทโปโลยีที่ใช้ในการทดลอง

จากภาพที่ 3-1 โทโปโลยีที่ใช้ในการทดลอง แสดงลักษณะการเชื่อมต่อกันระหว่างอุปกรณ์ แสดงการเชื่อมต่อกันระหว่าง ARP-PATH switch ส่วนประกอบของโทโปโลยีดังนี้

1. NetFPGA คือ เครื่องคอมพิวเตอร์ที่ทำการติดตั้งบอร์ด NetFPGA และ แฟ้มเก็บ NetFPGA พื้นฐาน
2. Host คือ เครื่องคอมพิวเตอร์ที่ใช้ในการทดสอบ

### 3.2 ภาพรวมการดำเนินการทดลอง

ขั้นตอนการดำเนินการทดลอง ซึ่งในแต่ละส่วนจะมีการดำเนินงานที่แตกต่างกันไป สามารถแยกและอธิบายได้ดังรายละเอียดต่อไปนี้

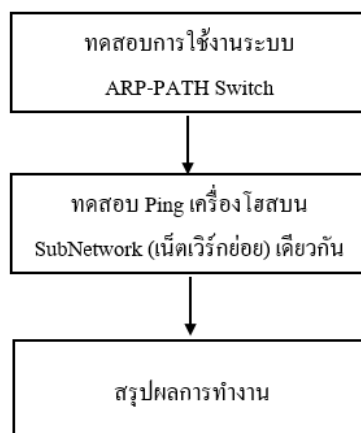
#### 3.2.1 ขั้นตอนการดำเนินการทดลอง



ภาพที่ 3-2 ผังการดำเนินงานส่วนบอร์ด NetFPGA

จากภาพที่ 3-2 แสดงขั้นตอนการทำงานเพื่อทำให้บอร์ด NetFPGA มีความสามารถเป็นอุปกรณ์เครือข่ายพื้นฐานเบื้องต้นได้ โดยดำเนินการในส่วนของการติดตั้งบอร์ด และ แฟ้มเก็บพื้นฐานที่จำเป็น

### 3.2.2 ขั้นตอนการดำเนินการทดสอบ ARP-PATH Switch



ภาพที่ 3-3 ผังการดำเนินการทดสอบระบบ ARP-PATH Switch

จากภาพที่ 3-3 แสดงถึงรายละเอียดการดำเนินการทดสอบระบบ ARP-PATH Switch ซึ่งมีการนำบอร์ด NetFPGA ที่ทำหน้าที่เป็น ARP-PATH Switch เข้ามาต่อเข้าใช้งานตามโพลีโลยีที่กำหนด แล้วทำการทดสอบงานทำงานพร้อมสรุปผลการดำเนินการ

### 3.3 การติดตั้งบอร์ด NetFPGA และแพ็คเกจ NetFGPA

การติดตั้งบอร์ด NetFPGA กับเครื่องคอมพิวเตอร์ มีขั้นตอนดังต่อไปนี้

3.3.1 ติดตั้งระบบปฏิบัติการ Fedora Core 13 with NetFPGA โดยสามารถ Download LiveDVD ในการติดตั้งได้ที่ <https://docs.google.com/file/d/0B4EuVzA5UdPRSzZyd29IYjkhTnM/edit?pli=1>

3.3.2 ขั้นตอนการติดตั้ง LiveDVD สามารถดูขั้นตอนวิธีการติดตั้งได้จาก

<https://github.com/NetFPGA/netfpga/wiki/LiveDVDInstall>

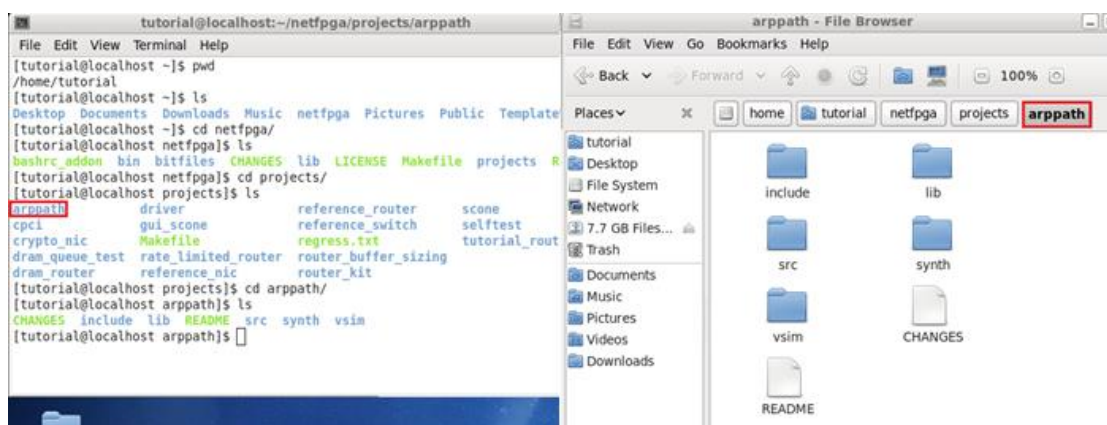
<https://sites.google.com/site/4mjaimenes/netfpga/instalacion>

### 3.4 การติดตั้งแพ็คเกจ ARP-PATH

ขั้นตอนการดำเนินการติดตั้ง แพ็คเกจ ARP-PATH Switch โดยสามารถ download ได้ที่

<https://docs.google.com/file/d/0B-pAPCvBuxmQOVZNc3h1b0hYOGc/edit?pli=1>

หลังจากนั้นทำการ unzip จะได้ไฟล์เดอร์ชื่อ arppath นำทั้ง ไฟล์เดอร์ ย้ายไปไว้ใน Project Directory ของ NetFPGA Base Packet ที่เราได้ทำการติดตั้งไว้แล้ว



ภาพที่ 3-4 แสดง project directory หลังจากทำการย้าย arppath โฟลเดอร์เข้าไปยังภายใน project directory

### 3.5 การดาวน์โหลดโปรแกรมลงบอร์ด NetFPGA

ขั้นตอนการทำให้บอร์ด NetFPGA มีพฤติกรรมการทำงานแบบ ARP-PATH Switch มีดังนี้

3.5.1 ทำการเคลียร์โปรแกรมการทำงานของบอร์ด NetFPGA ก่อนการดาวน์โหลดโปรแกรมใหม่ คำสั่ง `nf_download` ใช้ในการดาวน์โหลดโปรแกรมไปยังบอร์ด NetFPGA โดยต้องทำการเข้าไปยัง Directory ของ File ที่ต้องการใช้ดาวน์โหลด คือ Binary File การเคลียร์โปรแกรม มีชื่อไฟล์คือ `cpci_reprogram.bit` อยู่ใน Directory คือ `netfpga/bitfiles`

```
# nf_download netfpga/bitfiles/cpci_reprogram.bit
```

3.5.2 ทำการดาวน์โหลดโปรแกรมเพื่อให้บอร์ด NetFPGA มีพฤติกรรมการทำงานแบบ ARP-PATH Switch ด้วยคำสั่ง `nf_download` ไฟล์ชื่อ `arppath.bit` อยู่ใน Directory คือ `netfpga/project/arppath/synth`

```
# nf_download netfpga/project/arppath/synth /arppath.bit
```

3.5.3 กำหนดสถานะของ กิกะบิต อีเธอร์เน็ต 4 พอร์ต บนบอร์ด NetFPGA ให้เป็นสถานะพร้อมใช้งาน

```
# ifconfig nf2c0 up
```

```
# ifconfig nf2c1 up
```

```
# ifconfig nf2c2 up
```

```
# ifconfig nf2c3 up
```

ถ้าหากไม่ได้ login การใช้งานบน Fedora ด้วย User root จะติดปัญหาเรื่อง Permission (สิทธิอนุญาตการทำงานของแต่ละ User) จึงต้องใช้คำสั่ง เพื่อให้ได้สิทธิการทำงานเทียบเท่ากับการ login ด้วย root ด้วยคำสั่งด้านล่าง ก่อนการกำหนดสถานะของกิกะบิต อีเธอร์เน็ต 4 พอร์ต

```
# su root
```

3.5.4 การแสดงข้อมูลการทำงานของ ARP-PATH Switch ด้วย Command Line Interface (cli) (ส่วนต่อประสานกราฟิกกับผู้ใช้) ต้องเข้าไปใช้ Cli File ของ ARP-PATH Switch ที่อยู่ใน Directory netfpga/projects/arppath/lib/c

```
# cd netfpga/projects/arppath/lib/C
```

```
# ./cli -inf2c0 -s<mac> -F -I1000000000000 -b200000000000 -r200000000000  
-e20000000000 -p14 -m0
```

- option i เป็นการกำหนด device (อุปกรณ์) ในดึงข้อมูลออกมาแสดง
- option s เป็นการระบุ mac address ของเครื่องคอมพิวเตอร์ที่เชื่อมต่ออยู่กับบอร์ด

NetFPGA ซึ่ง ใส่เลข mac address ไปแทน <mac> เช่น -s11:22:33:44:55:66

- option l, b, r, e เป็นการกำหนด Ages (อายุของข้อมูล) ที่จะถูกเก็บในตาราง ให้กับทั้ง 4 ตาราง คือ Learning table (l) เป็น 100 วินาที , Broadcast Table (b) 2 วินาที, Repair Table (r) 2 วินาที, Hello Table (e) 2 วินาที โดยการใช้ option (คำสั่งย่อ) ในการกำหนด Ages จะกำหนดเป็นหน่วย ns (นาโนวินาที)

- option p ใช้ในการกำหนดค่าคาบของ Hello message เป็นหน่วย ns (นาโนวินาที)
- option m ใช้ในการกำหนดสถานะการส่ง Hello message ถ้าเป็น 1 คือ สามารถส่งได้ enable status ถ้าเป็น 0 คือ ไม่สามารถส่งได้ disable status ซึ่งค่าเริ่มต้นจะกำหนดให้เป็น 0

```
# ./cli nf2c0 -c
```

คำสั่งที่ใช้ให้แสดง counter (ตัวนับ) แพ็คเก็ตของกิกะบิต อีเธอร์เน็ต 4 พอร์ต และ ตารางทั้งหมด

```
# ./cli nf2c0 -dlr
```

คำสั่งที่ใช้ให้แสดงข้อมูลในตาราง Learning Table

```
# ./cli nf2c0 -dbt
```

คำสั่งที่ใช้ให้แสดงข้อมูลในตาราง Broadcast Table

```
# ./cli nf2c0 -drt
```

คำสั่งที่ใช้ให้แสดงข้อมูลในตาราง Repair Table

```
# ./cli nf2c0 -dhot
```

คำสั่งที่ใช้ให้แสดงข้อมูลในตาราง Host Table

```
# ./cli nf2c0 -H
```

คำสั่งที่ใช้ให้แสดงข้อมูลในตาราง Hello Table

```
# ./cli nf2c0 -h
```

คำสั่งที่ใช้ในการแสดง option ในการใช้ Command Line Interface พร้อมอธิบายการใช้

### 3.6 ทดสอบการทำงาน ARP-PATH Switch

การทำให้ Reference Switch ที่มีการทำงานบน Network-Layer 2 สามารถจัดการกับปัญหาการเกิดลูปในเน็ตเวิร์กด้วยการใช้ protocol ARP (โปรโตคอล ARP) เรียกว่า ARP-PATH Switch

การทดสอบเบื้องต้นสามารถทำได้โดยการใช้ Ping (คำสั่งใน Command Line เพื่อทดสอบการสื่อสารระหว่างอุปกรณ์) เครื่องคอมพิวเตอร์ ใน SubNetwork (เน็ตเวิร์กย่อย) เดียวกัน มีขั้นตอนดังนี้

3.6.1 ทำการเคลียร์โปรแกรมการทำงานของ NetFPGA ก่อนการดาวน์โหลดโปรแกรมใหม่

```
# nf_download netfpga/bitfiles/cpci_reprogram.bit
```

3.6.2 ทำการดาวน์โหลดโปรแกรมการทำงานของ ARP-PATH Switch ลงบอร์ด NetFPGA

```
# nf_download netfpga/project/arppath/synth /arppath.bit
```

3.6.3 ทำการเชื่อมต่อ ARP-PATH Switch ให้เป็นโทโปโลยีตามที่กำหนด ซึ่งมีการเกิดลูป และทำการตั้งไอพีแอดเดรสของเครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็น Source host (เครื่องต้นทาง) และ Destination Host (เครื่องปลายทาง) ให้อยู่ใน SubNetwork (เน็ตเวิร์กย่อย) เดียวกัน ในการทดสอบจะตั้ง Source Host1 169.254.211.238 และ Destination Host2 169.254.211.239

3.6.4 ทดสอบการทำงานของ ARP-PATH Switch ในการจัดการกับปัญหาการเกิดลูป ด้วยการ Ping (คำสั่งใน Command Line เพื่อทดสอบการสื่อสารระหว่างอุปกรณ์)

3.6.5 ทดสอบการ Ping จาก Source ไป Destination และ จาก Destination ไป Source

3.6.6 ทดสอบการ Ping เมื่อมีเส้นทางในโทโปโลยีเกิดการล้มเหลว ไม่สามารถส่งผ่านแพ็คเก็ตได้ ต้องมีการหาเส้นทางใหม่

## บทที่ 4

### ผลการทดลอง

จากวิธีการดำเนินการทดลองทั้งการติดตั้งบอร์ด NetFPGA พร้อมกับแพ็คเกจที่สามารถทำให้ใช้งานบอร์ด NetFPGA ได้ รวมไปถึงการติดตั้งแพ็คเกจ ARP-PATH Switch เพื่อนำมาใช้ในการส่งผ่านแพ็คเกจ เมื่อติดตั้งสำเร็จ บอร์ด NetFPGA จะมีความพร้อมในการใช้งานเสมือนดังเป็นอุปกรณ์ตัวนั้น ซึ่งจากที่กล่าวมา ทำให้ได้ผลการดำเนินการ ดังรายละเอียดต่อไปนี้

1. ผลการติดตั้งบอร์ด NetFPGA
2. การทดสอบการโปรแกรมบอร์ด Selftest
3. ผลการติดตั้งแพ็คเกจ NetFPGA และ ARP-PATH Switch
4. ผลการปฏิบัติการทดลอง ARP-PATH Switch

#### 4.1 ผลการติดตั้งบอร์ด NetFPGA

จากการดำเนินการติดตั้งบอร์ด NetFPGA ที่ได้กล่าวในการดำเนินการทดลองนั้น เมื่อทำการติดตั้งบอร์ด NetFPGA ลงบนเครื่องคอมพิวเตอร์ที่ติดตั้งระบบปฏิบัติการ Fedora Core 13 with NetFPGA แล้วนั้นซึ่งได้รวมไปถึงการติดตั้งแพ็คเกจเพื่อให้บอร์ด NetFPGA สามารถใช้งานได้โดยวิธีการตรวจสอบว่าบอร์ด NetFPGA นั้นสามารถใช้งานได้แล้วนั้นให้ใช้คำสั่ง ifconfig ตรวจสอบคู่มือเวิร์กอินเทอร์เฟซการ์ดของบอร์ด NetFPGA ที่ถูกเพิ่มขึ้นมา



```
[tutorial@localhost ~]$ ifconfig
eth2      Link encap:Ethernet  HWaddr 00:23:AE:9C:A4:8C
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:16

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:400 (400.0 b)  TX bytes:400 (400.0 b)

nf2c0     Link encap:Ethernet  HWaddr 00:4E:46:32:43:00
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:18

nf2c1     Link encap:Ethernet  HWaddr 00:4E:46:32:43:01
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:18

nf2c2     Link encap:Ethernet  HWaddr 00:4E:46:32:43:02
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:18

nf2c3     Link encap:Ethernet  HWaddr 00:4E:46:32:43:03
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:18
```

**ภาพที่ 4-1** ภาพชื่อพอร์ตของบอร์ด NetFPGA จะแสดงรายละเอียดของพอร์ตเมื่อทำเชื่อมต่อระหว่างเครื่องคอมพิวเตอร์และบอร์ด NetFPGA สำเร็จจะแสดง กิกะบิต อีเทอร์เน็ต ทั้งหมด 4 พอร์ต แสดงว่าทั้งหมดอยู่ในสถานะพร้อมใช้งาน

จากภาพที่ 4-1 แสดงถึงการติดตั้งบอร์ด NetFPGA สำเร็จจะแสดงชื่อพอร์ตสำหรับเชื่อมต่อ กิกะบิต อีเทอร์เน็ต ทั้ง 4 พอร์ต ได้แก่ nf2c0, nf2c1, nf2c2, nf2c3 โดยรายละเอียดที่แสดงข้างต้นจะมีเหมือนกับการแสดงคำสั่ง ifconfig เพื่อแสดงสถานะของเน็ตเวิร์กอินเทอร์เฟซการ์ด

#### 4.2 การทดสอบการโปรแกรมบอร์ด Selftest

หลังจากที่ติดตั้งแพ็คเกจบอร์ด NetFPGA และสามารถแสดงพอร์ตของบอร์ด NetFPGA ที่ติดตั้งเรียบร้อยแล้ว ขั้นตอนต่อไปคือ การทดสอบการใช้งานกิกะบิต อีเทอร์เน็ต ทั้ง 4 พอร์ต ด้วยวิธีการโปรแกรมบอร์ด NetFPGA ด้วย Selftest Binary File (ไบนารีไฟล์)



ภาพที่ 4-2 แสดงการเชื่อมต่อสายแลนระหว่างกิกะบิต อีเทอร์เน็ต ทั้ง 4 พอร์ต สำหรับการทำ Selftest

การทดสอบ Selftest ต้องทำการเชื่อมต่อสายแลนจากพอร์ต nf2c0 ไปยัง nf2c1 และจาก nf2c2 ไปยัง nf2c3 จากนั้นใช้คำสั่งในการเริ่มการทดสอบ เพื่อทดสอบการใช้งาน คือ

**# netfpga/projects/selftest/sw/selftest -n**

```
[tutorial@localhost ~]$ ~/netfpga/projects/selftest/sw/selftest -n
Found net device: nf2c0
CPCI Information
-----
Version: 4 (rev 1)

Device (Virtex) Information
-----
Project directory: selftest
Project name: Selftest
Project description: NetFPGA selftest -- exercises all major subsystem
Device ID: 5
Version: 1.1.0
Built against CPCI version: 4 (rev 1)

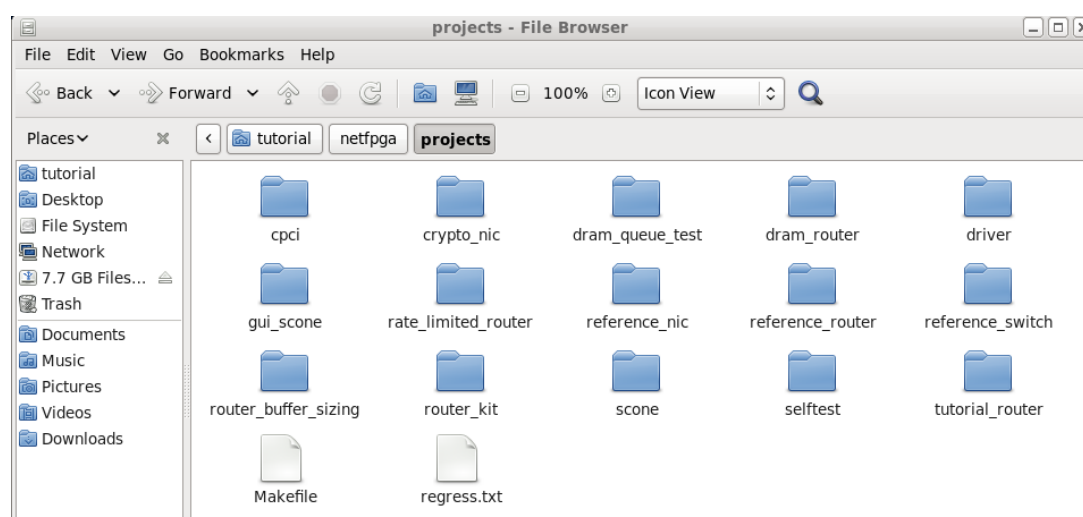
NetFPGA selftest 1.00 alpha
Running..... PASSED
```

ภาพที่ 4-3 ผลการรันคำสั่ง Selftest เพื่อตรวจสอบการใช้งานกิกะบิต อีเทอร์เน็ต ทั้ง 4 พอร์ต กรณีที่สามารถใช้งานได้ผลการรันจะแสดง เป็น PASSED

จากภาพที่ 4-3 แสดงถึงผลการเขียนโปรแกรมลงบอร์ดเพื่อให้บอร์ดทำการ Selftest กิกะบิต อีเธอร์เน็ต พอร์ตของบอร์ด NetFPGA ซึ่งใช้คำสั่ง `netfpga/projects/selftest/sw/selftest -n` เป็นการดาวน์โหลดโปรแกรมลงบอร์ด NetFPGA ซึ่งจะแสดงรายละเอียดต่าง ๆ เมื่อพอร์ตของบอร์ด NetFPGA สามารถส่งแพ็คเก็ตถึงกันได้ปกติ จะแสดงผลเป็น PASSED กรณีที่ทำการ Selftest แล้วไม่ผ่านให้ใช้คำสั่ง `netfpga/projects/selftest/sw/selftest -n -c` คำสั่งนี้จะแสดงรายละเอียดของกิกะบิต อีเธอร์เน็ต พอร์ตแต่ละ ว่าพอร์ตใดมีสถานะดับอยู่ ก็จะสามารถแสดงรายละเอียดได้อีกว่าเป็นปัญหาจากสาย LAN ที่ใช้เชื่อมต่อ หรือ กิกะบิต อีเธอร์เน็ต พอร์ตนั้นมีปัญหา

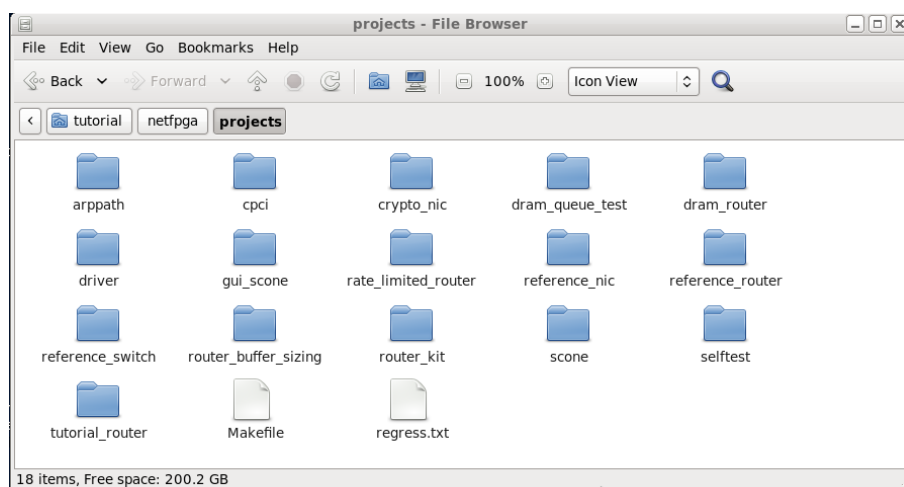
#### 4.3 ผลการติดตั้งแพ็คเกจ NetFPGA และ ARP-PATH Switch

จากขั้นตอนวิธีดำเนินการที่ต้องทำการติดตั้งแพ็คเกจบอร์ด NetFPGA และ ARP-PATH Switch เมื่อทำการติดตั้งสำเร็จจะแสดงโฟลเดอร์ที่ไว้เก็บแพ็คเกจในการส่งข้อมูลระหว่างกัน



ภาพที่ 4-4 แสดงแพ็คเกจพื้นฐานของ NetFPGA ก่อนทำการติดตั้งแพ็คเกจ ARP-Path switch

จากภาพที่ 4-4 เมื่อติดตั้งแพ็คเกจพื้นฐานของบอร์ด NetFPGA สำเร็จจะแสดงโฟลเดอร์ที่ชื่อ netfpga ภายในประกอบด้วยโฟลเดอร์ไฟล์แยกกันตามประเภทการทำงาน อย่างโฟลเดอร์ projects จะประกอบไปด้วย โฟลเดอร์ Project พื้นฐานของบอร์ด NetFPGA เช่น Selftest



ภาพที่ 4-5 แสดงแฟ้มเกิดพื้นฐานของ NetFPGA หลังทำการติดตั้งแฟ้มเกิด ARP-Path switch

จากภาพที่ 4-5 แสดงการติดตั้งแฟ้มเกิด ARP-PATH Switch เมื่อสำเร็จจะพบไฟล์เคอร์ arppath ที่ประกอบด้วยไฟล์ต่างๆที่ใช้ในการทำงานของ ARP-PATH Switch ที่สามารถจัดการปัญหาได้

#### 4.4 ผลการปฏิบัติการทดลอง ARP-PATH Switch

##### 4.4.1 ทำการเคลียร์โปรแกรมการทำงานของบอร์ด NetFPGA

**# nf\_download netfpga/bitfiles/cpci\_reprogram.bit**

```
[tutorial@localhost bitfiles]$ nf_download netfpga/bitfiles/cpci_reprogrammer.bit
Found net device: nf2c0
Bit file built from: nf2_top_par.ncd;HW_TIMEOUT=FALSE
Part: 2vp50ff1152
Date: 2010/ 7/26
Time: 18:22:55
Error Registers: 0
Good, after resetting programming interface the FIFO is empty
Download completed - 2377668 bytes, (expected -1).
DONE went high - chip has been successfully programmed.
CPCI Information
-----
Version: 4 {rev 1}

Device (Virtex) Information
-----
Project directory: cpci_reprogrammer
Project name: CPCI Reprogrammer
Project description: CPCI Reprogrammer

Device ID: 4
Version: 1.0.1
Built against CPCI version: 4 {rev 1}

Virtex design compiled against active CPCI version
[tutorial@localhost bitfiles]$
```

ภาพที่ 4-6 การใช้คำสั่งเพื่อดาวน์โหลด cpci\_reprogram.bit เพื่อล้างการทำงานของบอร์ด NetFPGA ที่มีการทำงานค้างไว้ก่อนหน้านี้

#### 4.4.2 ทำการเปลี่ยนสถานะกิกะบิต อีเทอร์เน็ต ทั้ง 4 พอร์ต ให้อยู่ในสถานะที่พร้อมใช้งาน

```
[root@localhost ~]# ifconfig
eth2      Link encap:Ethernet  HWaddr 00:23:AE:9C:A4:8C
          inet addr:192.168.106.176  Bcast:192.168.106.255  Mask:255.
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:21154 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12699 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12369890 (11.7 MiB)  TX bytes:2968871 (2.8 MiB)
          Interrupt:16

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:400 (400.0 b)  TX bytes:400 (400.0 b)

[root@localhost ~]# ifconfig nf2c0 up
[root@localhost ~]# ifconfig nf2c1 up
[root@localhost ~]# ifconfig nf2c2 up
[root@localhost ~]# ifconfig nf2c3 up
[root@localhost ~]# ifconfig
eth2      Link encap:Ethernet  HWaddr 00:23:AE:9C:A4:8C
          inet addr:192.168.106.176  Bcast:192.168.106.255  Mask:255.
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:21197 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12699 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12375503 (11.8 MiB)  TX bytes:2968871 (2.8 MiB)
          Interrupt:16

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:400 (400.0 b)  TX bytes:400 (400.0 b)

nf2c0     Link encap:Ethernet  HWaddr 00:4E:46:32:43:00
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:18

nf2c1     Link encap:Ethernet  HWaddr 00:4E:46:32:43:01
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

nf2c2     Link encap:Ethernet  HWaddr 00:4E:46:32:43:02
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:18

nf2c3     Link encap:Ethernet  HWaddr 00:4E:46:32:43:03
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:18
```

ภาพที่ 4-7 การตรวจสอบสถานะของ Gigabit Port ทั้ง 4

#### 4.4.3 ทำการดาวน์โหลดโปรแกรมการทำงาน ARP-PATH Switch ลงบอร์ด NetFPGA

# **nf\_download netfpga/project/arppath/synth /arppath.bit**

```
[tutorial@localhost ~]$ cd netfpga/projects/arppath/synth/
[tutorial@localhost synth]$ nf_download arppath.bit
Found net device: nf2c0
Bit file built from: nf2_top_par.ncd;HW_TIMEOUT=FALSE
Part: 2vp50ff1152
Date: 2014/ 4/ 9
Time: 15:27:58
Error Registers: 0
Good, after resetting programming interface the FIFO is empty
Download completed - 2377668 bytes. (expected -1).
DONE went high - chip has been successfully programmed.
CPCI Information
*****
Version: 4 (rev 1)

Device (Virtex) Information
*****
Project directory: arppath
Project name: ARPPATH Switch
Project description: Switch implementing the ARP-Path protocol

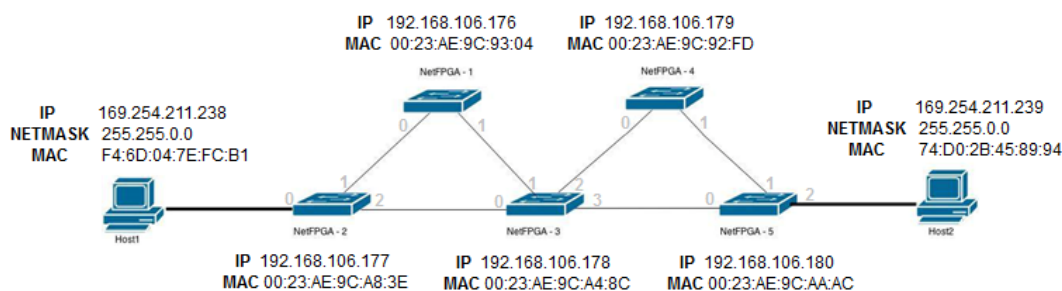
Device ID: 123
Version: 4.3.0
Built against CPCI version: 4 (rev 1)

Virtex design compiled against active CPCI version
[tutorial@localhost synth]$
```

ภาพที่ 4-8 เมื่อใช้คำสั่งเพื่อดาวน์โหลด arppath.bit เพื่อให้บอร์ด NetFPGA มีพฤติกรรมเป็น

ARP-PATH Switch

#### 4.4.4 ทำการเชื่อมต่อ ARP-PATH Switch ให้เป็นโทโปโลยีตามที่กำหนด ซึ่งมีการเกิดลูป



ภาพที่ 4-9 โทโปโลยีในการทดสอบ

ทดสอบการทำงานของ ARP-PATH Switch ในการจัดการกับปัญหาการเกิดลูปในโทโปโลยีที่กำหนด ด้วยการ Ping จาก Source ไป Destination และ จาก Destination ไป Source คำสั่งในการกำหนดค่า อายุของข้อมูลในแต่ละตาราง และตั้งค่าการทำงาน ตามขั้นตอนที่ 3.5.4 ในบทที่ 3 คือ

```
# ./cli -inf2c0 -s<mac> -F -I100000000000 -b2000000000 -r2000000000
-e2000000000 -p14 -m0
```

```
[tutorial@localhost C]$ ./cli -inf2c0 -s00:23:AE:9C:A4:8C -F -l100000000000 -b2000000000 -r2000000000 -e2000000000 -p14 -m0
Found net device: nf2c0
<netfpga1>
Initialized hardware hash functions.
Set the Learning time to 99999940608

Set the Blocking time to 1999962112

Set the Repair time to 1999962112

Set the Hello time to 1999962112

Hello period shift was set to 14, which corresponds to approximately 1073741 us.

Disabled hello sending.

</netfpga1>
```

## ภาพที่ 4-10 ภาพเมื่อใช้คำสั่งในการกำหนดค่าของตาราง

คำสั่งในการแสดงข้อมูลในแต่ละตารางคือ

1. Learning Table : `./cli nf2co -dlt`
2. Broadcast Table : `./cli nf2co -dbt`
3. Repair Table : `./cli nf2co -drt`
4. Host Table : `./cli nf2co -dhot`

### – ตารางของ NetFPGA-1

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dlt
Found net device: nf2c0
<netfpga1>
<lt>
lt table entries at time 0003497214998088 ns:
idx | mac | port | age (ns)
0009 | 74:d0:2b:45:89:94 | 0001 | 0000302052210248
0442 | f4:6d:04:7e:fc:b1 | 0000 | 0000302053389896
</lt>
</netfpga1>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dbt
Found net device: nf2c0
<netfpga1>
<bt>
bt table entries at time 0003500622733064 ns:
idx | mac | port | age (ns)
0074 | f4:6d:04:7e:fc:b1 | 0000 | 0000189820559112
0233 | 00:23:ae:9c:a8:3e | 0000 | 0000012190080776
0507 | 74:d0:2b:45:89:94 | 0001 | 0000036471168776
0820 | 00:23:ae:9c:aa:ac | 0001 | 0000071415801480
</bt>
</netfpga1>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dhot
Found net device: nf2c0
<netfpga1>
<hot>
hot table entries at time 0003514719028568 ns:
idx | mac | port | age (ns)
0198 | f4:6d:04:7e:fc:b1 | 0001 | 0000203916854616
0363 | 74:d0:2b:45:89:94 | 0000 | 0000050567464280
</hot>
</netfpga1>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -drt
Found net device: nf2c0
<netfpga1>
<rt>
rt table entries at time 0003511727217112 ns:
idx | mac | port | age (ns)
</rt>
</netfpga1>
```

## ภาพที่ 4-11 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-1

## – ตารางของ NetFPGA-2

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dlt
Found net device: nf2c0
<netfpgal>
<lt>
lt table entries at time 0003518347643224 ns:
idx | mac | port | age (ns)
0258 | 74:d0:2b:45:89:94 | 0002 | 0000422782142808
0702 | f4:6d:04:7e:fc:b1 | 0000 | 0000422781618520
</lt>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dbt
Found net device: nf2c0
<netfpgal>
<bt>
bt table entries at time 0003519835484552 ns:
idx | mac | port | age (ns)
0420 | 74:d0:2b:45:89:94 | 0002 | 0000097949987208
0880 | 00:23:ae:9c:aa:ac | 0002 | 0000021957915016
0954 | f4:6d:04:7e:fc:b1 | 0000 | 0000309636772232
</bt>
</netfpgal>
```

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dhot
Found net device: nf2c0
<netfpgal>
<hot>
hot table entries at time 0003616395112104 ns:
idx | mac | port | age (ns)
0107 | 74:d0:2b:45:89:94 | 0001 | 0000014154149544
0795 | f4:6d:04:7e:fc:b1 | 0000 | 0000008152886952
</hot>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -drt
Found net device: nf2c0
<netfpgal>
<rt>
rt table entries at time 0003522427487032 ns:
idx | mac | port | age (ns)
</rt>
</netfpgal>
```

ภาพที่ 4-12 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-2

## – ตารางของ NetFPGA-3

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dlt
Found net device: nf2c0
<netfpgal>
<lt>
lt table entries at time 0003622292430376 ns:
idx | mac | port | age (ns)
0118 | f4:6d:04:7e:fc:b1 | 0000 | 0000622656101928
0279 | 74:d0:2b:45:89:94 | 0003 | 0000622656691752
</lt>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dbt
Found net device: nf2c0
<netfpgal>
<bt>
bt table entries at time 0003624900093944 ns:
idx | mac | port | age (ns)
0310 | 00:23:ae:9c:aa:ac | 0003 | 0000147188082680
0395 | f4:6d:04:7e:fc:b1 | 0000 | 0000095586478072
0628 | 00:23:ae:9c:a8:3e | 0000 | 0000001230956536
0907 | 74:d0:2b:45:89:94 | 0003 | 0000087514933240
</bt>
</netfpgal>
```

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dhot
Found net device: nf2c0
<netfpgal>
<hot>
hot table entries at time 0003632276535080 ns:
idx | mac | port | age (ns)
0495 | f4:6d:04:7e:fc:b1 | 0001 | 0000102962919208
0975 | 74:d0:2b:45:89:94 | 0002 | 0000094891374376
</hot>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -drt
Found net device: nf2c0
<netfpgal>
<rt>
rt table entries at time 0003629044797752 ns:
idx | mac | port | age (ns)
</rt>
</netfpgal>
```

ภาพที่ 4-13 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-3



## – ตารางของ NetFPGA-4

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dlt
Found net device: nf2c0
<netfpgal>
<lt>
lt table entries at time 0003418359134712 ns:
idx | mac | port | age (ns)
0442 | f4:6d:04:7e:fc:b1 | 0000 | 0000722330974712
0980 | 74:d0:2b:45:89:94 | 0001 | 0000722329860600
</lt>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dbt
Found net device: nf2c0
<netfpgal>
<bt>
bt table entries at time 0003420871062056 ns:
idx | mac | port | age (ns)
0114 | 74:d0:2b:45:89:94 | 0001 | 0000039804989992
0249 | 00:23:ae:9c:a8:3e | 0000 | 00000004540265000
0695 | f4:6d:04:7e:fc:b1 | 0000 | 0000194158916136
0983 | 00:23:ae:9c:aa:ac | 0001 | 0000245760455208
</bt>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dhot
Found net device: nf2c0
<netfpgal>
<hot>
hot table entries at time 0003426039032472 ns:
idx | mac | port | age (ns)
0181 | f4:6d:04:7e:fc:b1 | 0001 | 0000199326886552
0989 | 74:d0:2b:45:89:94 | 0000 | 0000044972960408
</hot>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -drt
Found net device: nf2c0
<netfpgal>
<rt>
rt table entries at time 0003423223198952 ns:
idx | mac | port | age (ns)
</rt>
</netfpgal>
```

## ภาพที่ 4-14 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-4

## – ตารางของ NetFPGA-5

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dlt
Found net device: nf2c0
<netfpgal>
<lt>
lt table entries at time 0003687912231176 ns:
idx | mac | port | age (ns)
0001 | f4:6d:04:7e:fc:b1 | 0000 | 0000799567171848
0935 | 74:d0:2b:45:89:94 | 0002 | 0000799567696136
</lt>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dbt
Found net device: nf2c0
<netfpgal>
<bt>
bt table entries at time 0003692248653592 ns:
idx | mac | port | age (ns)
0000 | f4:6d:04:7e:fc:b1 | 0000 | 0003690610056984
0177 | 00:23:ae:9c:a8:3e | 0000 | 0000002425446168
0353 | 74:d0:2b:45:89:94 | 0002 | 0000026509270808
0696 | f4:6d:04:7e:fc:b1 | 0000 | 0000274226831128
</bt>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dhot
Found net device: nf2c0
<netfpgal>
<hot>
hot table entries at time 0003719096152776 ns:
idx | mac | port | age (ns)
0000 | f4:6d:04:7e:fc:b1 | 0000 | 0003717457556168
0401 | 74:d0:2b:45:89:94 | 0002 | 0000000823702216
0437 | f4:6d:04:7e:fc:b1 | 0001 | 0000301074330312
</hot>
</netfpgal>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -drt
Found net device: nf2c0
<netfpgal>
<rt>
rt table entries at time 0003695784380856 ns:
idx | mac | port | age (ns)
</rt>
</netfpgal>
```

## ภาพที่ 4-15 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-5

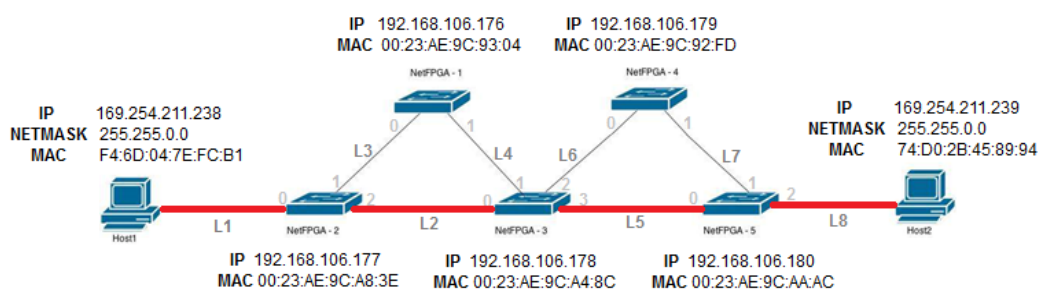
ข้อมูลจากตารางของ ARP-PATH Switch ทั้งหมด 5 เครื่องนั้น เราสามารถดูเส้นทางที่ใช้ในการส่งแพ็คเก็ตระหว่างเครื่องคอมพิวเตอร์ Host1 และ Host2 จาก Learning Table ของ ARP-PATH Switch ซึ่งการทำงานเมื่อแพ็คเก็ตเกิดเข้ามามีการแยกส่วนเพื่อวิเคราะห์ว่าเครื่องไหนเป็นคนส่ง และเครื่องไหนเป็นคนรับ ซึ่งจะนำเอา MAC Address ของเครื่องที่ระบุให้เป็นผู้รับมาตรวจสอบกับ Learning Table ของแต่ละเครื่อง ถ้าหากใน Learning Table มีข้อมูลของ MAC Address เครื่องรับ ก็จะทำการ map (จับคู่) กันระหว่าง MAC Address และ Output Port เพื่อทำการส่งแพ็คเก็ตนั้นออกไปหาเครื่องรับได้

เมื่อนำข้อมูลจาก Learning Table แต่ละเครื่องมาสรุปเป็นตารางเพื่อหาเส้นทางในการส่งข้อมูลจะได้ดังนี้

ตารางที่ 4-1 ตารางเก็บข้อมูลจาก Learning Table ของ ARP-PATH Switch ทั้ง 5 เครื่อง

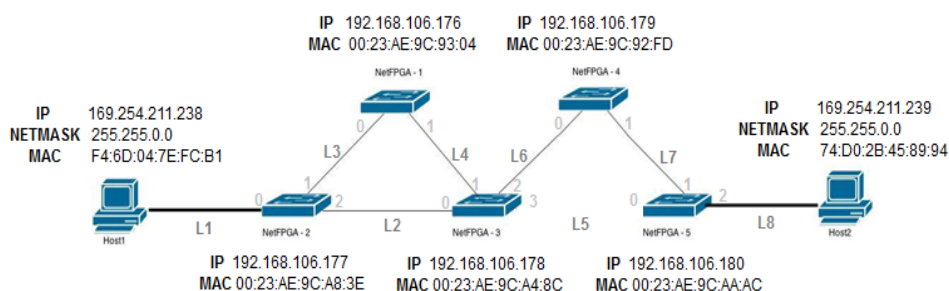
	Port to Host1 (F4:6D:04:7E:FC:B1)	Port to Host2 (74:D0:2B:45:89:94)
NetFPGA-1	0	1
NetFPGA-2	0	2
NetFPGA-3	0	3
NetFPGA-4	0	1
NetFPGA-5	0	2

จะได้เส้นทางของการส่งข้อมูลจาก Host1 ไปยัง Host2 ผ่านโทโปโลยีที่กำหนดดังนี้



ภาพที่ 4-16 แสดงเส้นทางการส่งข้อมูลระหว่าง Host1 และ Host2

4.4.5 ทดสอบการ Ping เมื่อ สาย LAN : 15 เกิดความผิดพลาด ถูกถอดออก โทโปโลยีที่ใช้ในการทดสอบเป็นดังนี้



ภาพที่ 4-17 โทโปโลยีการทดสอบเมื่อ Link 5 ไม่ได้เชื่อมต่อ

จากโทโปโลยีสังเกตได้ว่าการเชื่อมต่อของ ARP-PATH Switch จะมีการเปลี่ยนแปลงแค่ที่เครื่อง NetFPGA-3 และ NetFPGA-5 ตาราง Learning Table ที่ใช้ในการ map หา Output Port ของแพ็คเก็ตก็จะมีการเปลี่ยนแปลงแค่ NetFPGA-3 และ NetFPGA-5 โดยมีการเปลี่ยนแปลงดังนี้

#### – ตารางของ NetFPGA-3

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dlt
Found net device: nf2c0
<netfpga>
<lt>
lt table entries at time 0004463243213496 ns:
idx | mac | port | age (ns)
0118 | f4:6d:04:7e:fc:b1 | 0000 | 0000268260915896
0279 | 74:d0:2b:45:89:94 | 0002 | 0000268261505720
</lt>
</netfpga>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dbt
Found net device: nf2c0
<netfpga>
<bt>
bt table entries at time 0004469177572504 ns:
idx | mac | port | age (ns)
0310 | 00:23:ae:9c:aa:ac | 0002 | 0000181059902616
0395 | f4:6d:04:7e:fc:b1 | 0000 | 0000075435859096
0628 | 00:23:ae:9c:a8:3e | 0000 | 0000000402250904
0907 | 74:d0:2b:45:89:94 | 0002 | 0000000609475736
</bt>
</netfpga>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dhot
Found net device: nf2c0
<netfpga>
<hot>
hot table entries at time 0004480217709080 ns:
idx | mac | port | age (ns)
0495 | f4:6d:04:7e:fc:b1 | 0001 | 0000094476075672
0975 | 74:d0:2b:45:89:94 | 0002 | 0000011064148632
</hot>
</netfpga>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -drt
Found net device: nf2c0
<netfpga>
<rt>
rt table entries at time 0004473433099016 ns:
idx | mac | port | age (ns)
</rt>
</netfpga>
```

ภาพที่ 4-18 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-3

#### – ตารางของ NetFPGA-5

```
[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dlt
Found net device: nf2c0
<netfpga>
<lt>
lt table entries at time 0004091109070952 ns:
idx | mac | port | age (ns)
0001 | f4:6d:04:7e:fc:b1 | 0001 | 0000007419287656
0935 | 74:d0:2b:45:89:94 | 0002 | 0000007419877480
</lt>
</netfpga>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dbt
Found net device: nf2c0
<netfpga>
<bt>
bt table entries at time 0004227027456776 ns:
idx | mac | port | age (ns)
0000 | f4:6d:04:7e:fc:b1 | 0000 | 0004226188860168
0177 | 00:23:ae:9c:a8:3e | 0001 | 0000050343926536
0353 | 74:d0:2b:45:89:94 | 0002 | 0000070931077896
0696 | f4:6d:04:7e:fc:b1 | 0001 | 0000006392011528
</bt>
</netfpga>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -dhot
Found net device: nf2c0
<netfpga>
<hot>
hot table entries at time 0004290259336648 ns:
idx | mac | port | age (ns)
0000 | f4:6d:04:7e:fc:b1 | 0000 | 0004288620740040
0401 | 74:d0:2b:45:89:94 | 0002 | 0000016811769288
0437 | f4:6d:04:7e:fc:b1 | 0001 | 0000007810399688
</hot>
</netfpga>

[tutorial@localhost ~]$ ./netfpga/projects/arppath/lib/C/cli nf2c0 -drt
Found net device: nf2c0
<netfpga>
<rt>
rt table entries at time 0004231763009464 ms:
idx | mac | port | age (ns)
</rt>
</netfpga>
```

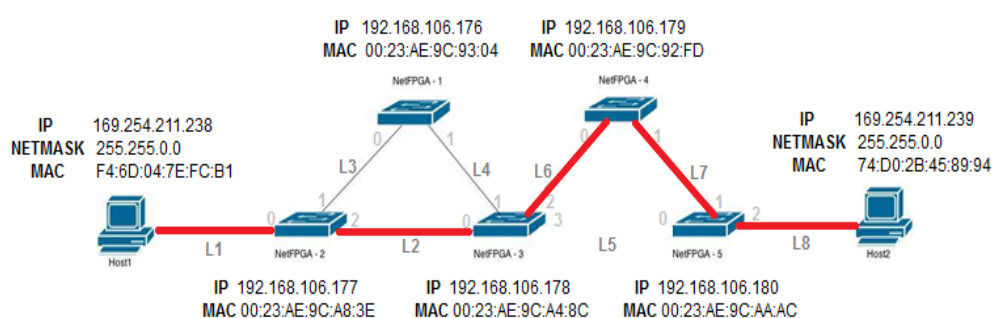
ภาพที่ 4-19 ตาราง Learning, Broadcast, Repair, Host table ของเครื่อง NetFPGA-5

เมื่อนำข้อมูลจาก Learning Table แต่ละเครื่องมาสรุปเป็นตารางเพื่อหาเส้นทางในการส่งข้อมูลจะได้ดังนี้

ตารางที่ 4-2 ตารางเก็บข้อมูลจาก Learning Table ของ ARP-PATH Switch ทั้ง 5 เครื่อง

	Port to Host1 (F4:6D:04:7E:FC:B1)	Port to Host2 (74:D0:2B:45:89:94)
NetFPGA-1	0	1
NetFPGA-2	0	2
NetFPGA-3	0	2
NetFPGA-4	0	1
NetFPGA-5	1	2

จะได้เส้นทางของการส่งข้อมูลจาก Host1 ไปยัง Host2 ผ่านโทโปโลยีที่กำหนดดังนี้

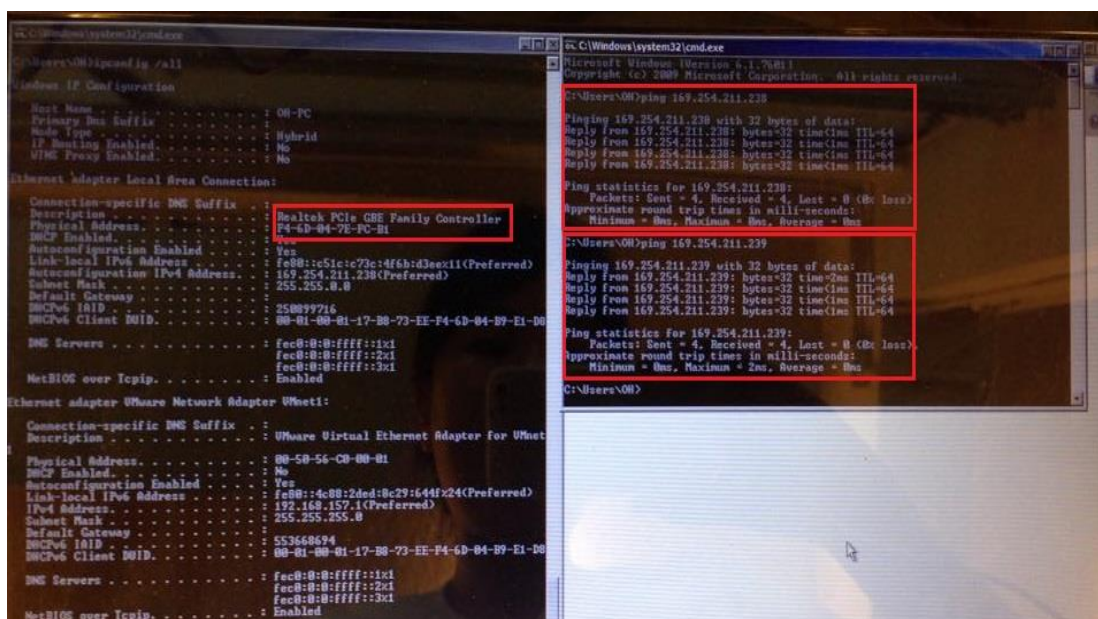


ภาพที่ 4-20 แสดงเส้นทางการส่งข้อมูลระหว่าง Host1 และ Host2

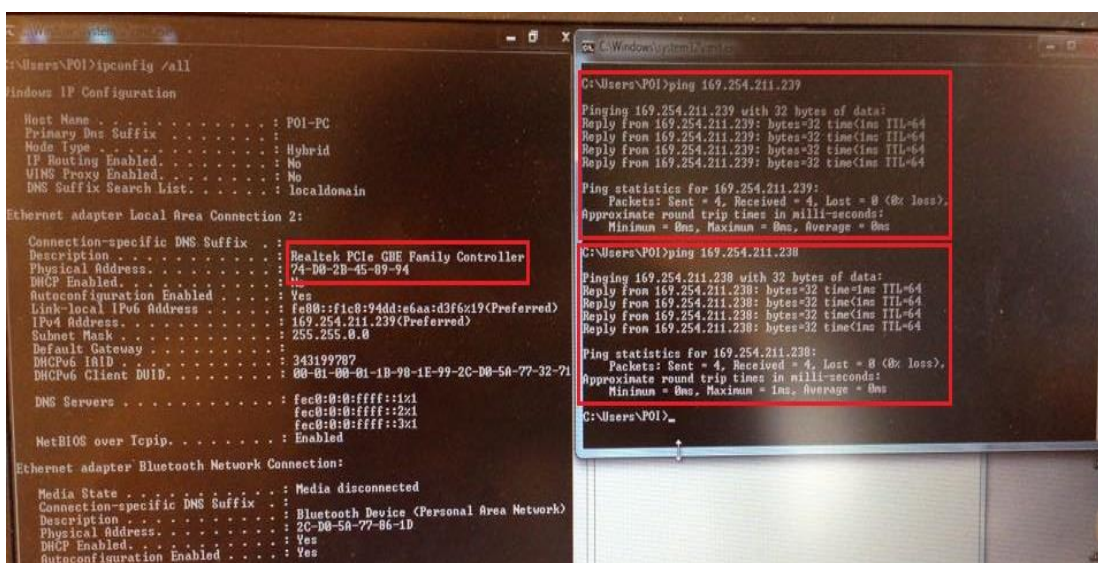
#### 4.4.6 การ Ping เครื่องคอมพิวเตอร์บน SubNetwork (เน็ตเวิร์กย่อย) เดียวกัน

Host 1 : IP 169.254.211.238                      MAC F4:6D:04:7E:FC:B1

Host 2 : IP 169.254.211.239                      MAC 74:D0:2B:45:89:94



ภาพที่ 4-21 การ Ping จาก Host 1 ไป Host 2



ภาพที่ 4-22 การ Ping จาก Host 2 ไป Host 1

จากวิธีการดำเนินการเพื่อทดสอบการทำงานของ ARP-PATH Switch เห็นได้ว่าการนำบอร์ด NetFPGA มาทำให้มีพฤติกรรมการทำงานเป็น ARP-PATH Switch เพื่อแก้ปัญหาก็เกิดอุปสรรค ซึ่ง Reference Switch ที่เป็น project พื้นฐานของบอร์ด NetFPGA มีพฤติกรรมการทำงานเป็น Switch Network-Layer 2 ไม่สามารถจัดการแก้ปัญหาก็เกิดอุปสรรคบนเน็ตเวิร์ก

## บทที่ 5

### สรุปผลการดำเนินการ

ในปฏิญานิพนธ์นี้ได้นำเสนอการพัฒนาต้นแบบในการทำ ARP-PATH Switch จากบอร์ด NetFPGA โดยใช้อุปกรณ์ที่สามารถนำมาทดลองได้ในการปฏิบัติการขนาดเล็ก ซึ่งบอร์ด NetFPGA สามารถทำงานได้เอนกประสงค์ และสามารถทำงานเป็นอุปกรณ์ต่าง ๆ ได้ดี เหมาะสำหรับผู้สนใจ และผู้ที่ต้องการศึกษา และพัฒนาอุปกรณ์เครือข่ายโดยใช้ ARP โพรโตคอลในระดับต้นเพื่อช่วยในการแก้ปัญหาการเกิดลูปในเน็ตเวิร์ก

การนำบอร์ด NetFPGA มาพัฒนาเพื่อให้เข้าใจในเรื่องของทฤษฎีการทำงานของสวิตช์ Layer 2 (Reference Switch) ที่ไม่สามารถจัดการกับปัญหาการเกิดลูปในเน็ตเวิร์กได้ โดยใช้โปรโตคอลพื้นฐาน คือ ARP โพรโตคอล จากเดิมที่ต้องใช้ Spanning Tree โพรโตคอล, Multiple STP โพรโตคอล ในการแก้ปัญหาซึ่งมีข้อจำกัดในการทำงานดังนี้

- ขนาดเน็ตเวิร์ก
- เส้นทางที่หาได้จะไม่ใช่เส้นทางที่สั้นที่สุด
- การตั้งค่า (Configuring) มีความยุ่งยาก ซับซ้อน
- ลดจำนวน message ที่ส่งภายในเน็ตเวิร์กในการแก้ปัญหา

โดยได้มีการนำบอร์ด NetFPGA มาติดตั้งลงบนเครื่องคอมพิวเตอร์พร้อมกับติดตั้งแพ็คเกจ NetFPGA พื้นฐาน และ ARP-PATH Project เพื่อทดสอบการทำงานด้านการจัดการลูปของ ARP-PATH Switch บนบอร์ด NetFPGA

จากการดำเนินการทดลอง พบว่าอุปกรณ์ต้นแบบบอร์ด NetFPGA สามารถทำงานเป็น ARP-PATH Switch ได้สามารถส่งแพ็คเกจได้เหมือนสวิตช์ปกติ แต่มีความสามารถที่เพิ่มเติมคือการจัดการกับลูปซึ่งเป็นปัญหาที่สวิตช์ทั่วไป (Reference Switch Layer2) ไม่สามารถทำได้ แสดงว่า ARP-PATH Switch มีความสามารถในการทำงานมากกว่าสวิตช์ทั่วไป และสรุปได้ว่า ARP-PATH Switch มีส่วนช่วยให้ระบบเน็ตเวิร์กมีประสิทธิภาพมากยิ่งขึ้น ทั้งทางด้านทรัพยากร และการจัดการปัญหาในด้านต่างๆ

### เอกสารอ้างอิง

1. ที่ปรึกษาพิเศษ
  - ผศ.ดร.มารอง ผดุงสิทธิ์ : ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี.
  - Elisa Rojas : Communication and Information Technologies engineering from the University of Alcala, Spain [Online]. Available: [http://www3.uah.es/elisa\\_rojas/](http://www3.uah.es/elisa_rojas/)
2. ข้อมูลเกี่ยวกับการใช้งาน NetFPGA [Online]. Available:
  - <https://github.com/NetFPGA/netfpga/wiki/Guide>
  - <https://github.com/NetFPGA/netfpga/wiki/Guide#InstallNetFPGABasePackage>
  - <https://github.com/NetFPGA/NetFPGA-public/wiki/Getting-Started-Guide>
  - <https://github.com/NetFPGA/netfpga/wiki/DevelopersGuide>
  - [https://github.com/NetFPGA/netfpga/wiki/ARP\\_Path-Switch](https://github.com/NetFPGA/netfpga/wiki/ARP_Path-Switch)
3. ข้อมูลเกี่ยวกับ ARP-Path Switch [Online]. Available:
  - [http://dspace.uah.es/dspace/bitstream/handle/10017/21144/14%20Comnet%20Rojas14\\_Revised2.pdf?sequence=1&isAllowed=y](http://dspace.uah.es/dspace/bitstream/handle/10017/21144/14%20Comnet%20Rojas14_Revised2.pdf?sequence=1&isAllowed=y)
  - <http://conferences.sigcomm.org/sigcomm/2011/papers/sigcomm/p444.pdf>
  - <http://www.ieee802.org/1/files/public/docs2011/new-ibanez-ARP-Path-and-Broad-path-0311-v5.pdf>
  - [http://www.artinalgorithms.be/\\_docs/Ibanez-ARP-Path2.pdf](http://www.artinalgorithms.be/_docs/Ibanez-ARP-Path2.pdf)
  - [http://www.ieee-lcn.org/prior/LCN37/lcn37demos/LCNDemos12\\_Ibanez.pdf](http://www.ieee-lcn.org/prior/LCN37/lcn37demos/LCNDemos12_Ibanez.pdf)



### ประวัติผู้แต่ง

ปรินญาณพนธ์เรื่อง : การพัฒนาอุปกรณ์เครือข่ายบน NetFPGA เพื่อการจัดการ Loop  
 สาขาวิชา : วิศวกรรมคอมพิวเตอร์  
 ภาควิชา : วิศวกรรมไฟฟ้า และคอมพิวเตอร์  
 คณะ : วิศวกรรมศาสตร์  
 ชื่อ : นางสาวสุกัก สุขประเสริฐ  
 ประวัติ

เกิดเมื่อวันที่ 6 พฤศจิกายน 2535 อยู่บ้านเลขที่ 99/5141 หมู่ 2 ซอยท่าอิฐ ถนนรัตนนิเบศร์ ตำบลท่าอิฐ อำเภอปากเกร็ด จังหวัดนนทบุรี สำเร็จการศึกษาในระดับชั้นมัธยมศึกษาตอนปลาย จาก โรงเรียนสวนกุหลาบวิทยาลัย นนทบุรี จังหวัดนนทบุรี สาขาวิทยาศาสตร์-คณิตศาสตร์ ปีการศึกษา 2553 และสำเร็จการศึกษาในระดับปริญญาตรี สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้า และคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ ปีการศึกษา 2557

ชื่อ : นางสาวณัฏฐนันทน์ เจริญสุขภาพ  
 ประวัติ

เกิดเมื่อวันที่ 17 พฤษภาคม พ.ศ. 2536 อยู่บ้านเลขที่ 59/140 หมู่ 5 ซอยประเสริฐมนูกิจ 27 ถนนประเสริฐมนูกิจ แขวงจระเข้บัว เขตลาดพร้าว จังหวัดกรุงเทพฯ สำเร็จการศึกษาในระดับชั้นมัธยมศึกษาตอนปลาย จากโรงเรียนสตรีวิทยา 2 จังหวัดกรุงเทพฯ สาขาวิทยาศาสตร์-คณิตศาสตร์ ปีการศึกษา 2553 และสำเร็จการศึกษาในระดับปริญญาตรี สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้า และคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ ปีการศึกษา 2557