บทที่ 2

ทฤษฎี

2.1 ไมโครคอนโทรลเลอร์ MCS-51

2.1.1 โครงสร้างพื้นฐาน และคุณสมบัติของ MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 ได้มีการนำไปใช้งานกันอย่างแพร่หลาย ปัจจุบันบริษัทผู้ผลิตได้พัฒนาความสามารถของไมโครคอนโทรลเลอร์ MCS-51 รุ่นใหม่ ๆ ให้สามารถทำงานได้อย่างมีประสิทธิภาพ และมีความเร็วเพิ่มขึ้นมาก แต่ยังคงโครงสร้างพื้นฐานที่สำคัญไว้ดังเดิมมีรายละเอียดดังนี้

1. เป็นไมโครคอนโทรลเลอร์ที่มีหน่วยประมวลผลกลางแบบ 8 บิต
2. มีคำสั่งคำนวณทางคณิตศาสตร์ และตรรกศาสตร์ (Boolean processor)
3. มีแอดเดรสบัสขนาด 16 บิตทำให้สามารถอ้างตำแหน่งหน่วยความจำโปรแกรม และหน่วยความจำข้อมูลได้ 64 กิโลไบต์
4. มีหน่วยความจำ (RAM) ภายในขนาด 128 ไบต์ (8051/8031) หรือ 256 ไบต์ (8052/8032)
5. มีพอร์ตอนุกรมทำงานแบบดูเพล็กซ์เต็ม (Full Duplex) 1 พอร์ต
6. มีพอร์ตอินพุต/เอาต์พุตแบบขนานจำนวน 32 บิต
7. มีไทเมอร์ 2 ตัว (8051/8031) หรือ 3 ตัว (8052/8032)
8. มีวงจรควบคุมการเกิดอินเตอร์รัปต์ 5 ประเภท (8051/8031) หรือ 6 ประเภท (8052/8032)
9. มีวงจรออสซิลเลเตอร์ภายในตัว

ไมโครคอนโทรลเลอร์ MCS-51 มีวงจรออสซิลเลเตอร์อยู่ภายใน ดังนั้นในการใช้งานจึงสามารถต่อคริสตอล และตัวเก็บประจุเข้ากับขาสัญญาณ XTAL1 และ XTAL2 ได้โดยตรงดังแสดงในรูปที่ 2.1 โดยความถี่ของคริสตอลที่ต่อเข้ากับไมโครคอนโทรลเลอร์จะเป็นตัวระบุความเร็วในการทำงานโดยตรง ในไมโครคอนโทรลเลอร์ MCS-51 ปกติ 1 แมชชีนไซเคิล (machine cycle) จะใช้สัญญาณนาฬิกาจำนวน 12 ลูก และในการทำงานแต่ละคำสั่งไมโครคอนโทรลเลอร์จะใช้เวลาในการทำงาน 1-4 แมชชีนไซเคิล ขึ้นอยู่กับความซับซ้อนของคำสั่งนั้น ทำให้ความสัมพันธ์ของเวลาที่ใช้ในการทำงาน กับความถี่คริสตอลเป็นดังสมการที่ (2.1)

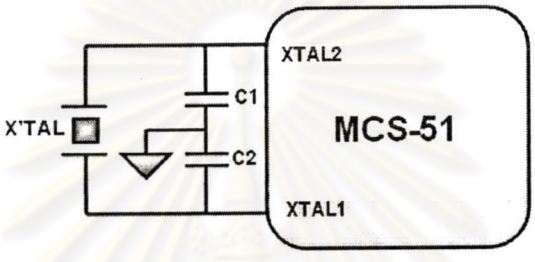
$$T = \frac{C \times 12}{f_{XTAL}} \qquad (2.1)$$

โดย T = ช่วงเวลาที่ใช้ในการทำงาน

C = จำนวนแมชชีนไซเคิล

f_{XTAL} = ความถี่ของคริสตอล

ในปัจจุบันผู้ผลิตได้พัฒนาให้ไมโครคอนโทรลเลอร์สามารถทำงานได้เร็วขึ้นโดยเพิ่มความสามารถในการรองรับคริสตอลความถี่สูงขึ้น รวมไปถึงการปรับปรุงการทำงานภายในให้ไมโครคอนโทรลเลอร์ใช้จำนวนสัญญาณนาฬิกาในการสร้างแมชชีนไซเคิลน้อยลง โดยในบางรุ่น 1 แมชชีนไซเคิลใช้สัญญาณนาฬิกาเพียงแค่ 1 ลูกเท่านั้น



รูปที่ 2.1 การต่อคริสตอลเข้ากับไมโครคอนโทรลเลอร์ MCS-51

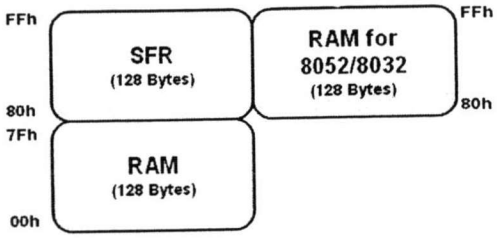
2.1.2 ระบบหน่วยความจำของ MCS-51

ระบบหน่วยความจำของไมโครคอนโทรลเลอร์ MCS-51 นั้นสามารถแบ่งออกเป็น 3 ส่วน คือ หน่วยความจำโปรแกรม (Program memory) หน่วยความจำข้อมูลภายใน (Internal RAM) และหน่วยความจำข้อมูลภายนอก (External RAM)

หน่วยความจำโปรแกรมใช้สำหรับเก็บคำสั่งที่ควบคุมการทำงานไมโครคอนโทรลเลอร์ สามารถเก็บข้อมูลได้โดยไม่ต้องอาศัยไฟเลี้ยง ไมโครคอนโทรลเลอร์ส่วนใหญ่จะมีหน่วยความจำโปรแกรมอยู่ภายในตัว โดยโครงสร้างของหน่วยความจำโปรแกรมนั้นมีหลายชนิดเช่น ROM (Read Only Memory), EEPROM (Erasable Programmable Read Only Memory), Flash เป็นต้น ในการใช้งานปกติไมโครคอนโทรลเลอร์จะไม่เขียนข้อมูล แต่อ่านคำสั่งจากหน่วยความจำโปรแกรมนี้เท่านั้น ส่วนการลงโปรแกรมให้กับไมโครคอนโทรลเลอร์นั้นมีวิธีหลากหลาย และขึ้นอยู่กับชนิดของหน่วยความจำที่ใช้

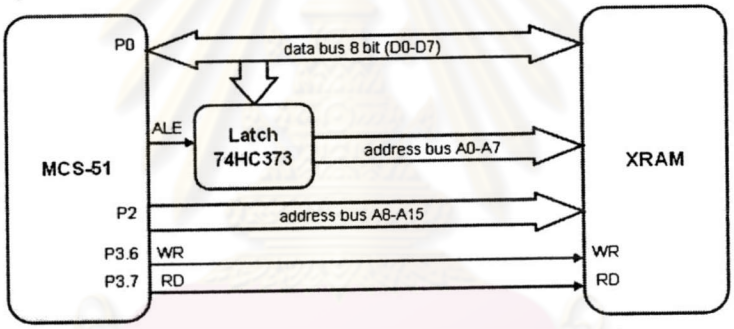
หน่วยความจำข้อมูลภายในของไมโครคอนโทรลเลอร์ MCS-51 นั้นมีขนาด 128-256 ไบต์ ขึ้นอยู่กับรุ่นของไมโครคอนโทรลเลอร์ แบ่งออกเป็น 2 ส่วนคือหน่วยความจำส่วนล่าง (00h-7Fh) และหน่วยความจำส่วนบน (80h-FFh) หน่วยความจำส่วนล่างนั้นใช้สำหรับเก็บข้อมูลทั่วไป ส่วนหน่วยความจำส่วนบนนั้นเป็นหน่วยความจำพิเศษที่ใช้สำหรับกำหนดการทำงานของไมโครคอนโทรลเลอร์ หรือเรียกว่า SFR (Special Function Register) ชิพรุ่นใหม่ ๆ ที่มีโครงสร้าง

เป็นไปตามตระกูล 8052 นั้นมีหน่วยความจำทั่วไปที่ต้องเข้าถึงด้วยวิธี indirect อยู่ในตำแหน่งหน่วยความจำส่วนบนซ้อนทับอยู่กับ SFR ดังแสดงในรูปที่ 2.2



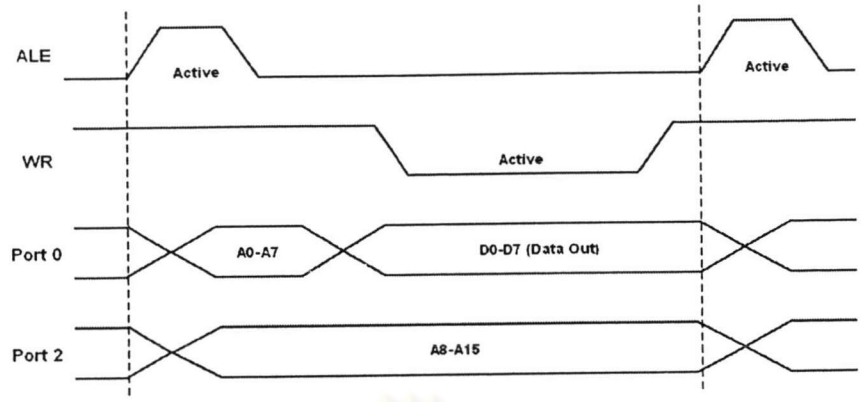
รูปที่ 2.2 โครงสร้างหน่วยความจำข้อมูลภายใน MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 สามารถต่อหน่วยความจำภายนอกแบบ memory map เพิ่มได้ถึง 64 กิโลไบต์ โดยใช้พอร์ต 0 เป็นสายสัญญาณข้อมูลขนาด 8 บิต (data bus D0-D7) และสายสัญญาณกำหนดตำแหน่งหน่วยความจำไบต์ล่าง (address bus A0-A7) ในการใช้งานนั้นต้องใช้ไอซี Latch ทำหน้าที่แยกสัญญาณทั้งสองออกจากกันด้วยสัญญาณควบคุม ALE จากไมโครคอนโทรลเลอร์ ส่วนสายสัญญาณกำหนดตำแหน่งหน่วยความจำไบต์บน (A8-A15) และสายสัญญาณควบคุม WR, RD นั้นใช้พอร์ต 2 และพอร์ต 3.6, 3.7 ได้โดยตรงตามลำดับ ดังรูปที่ 2.3



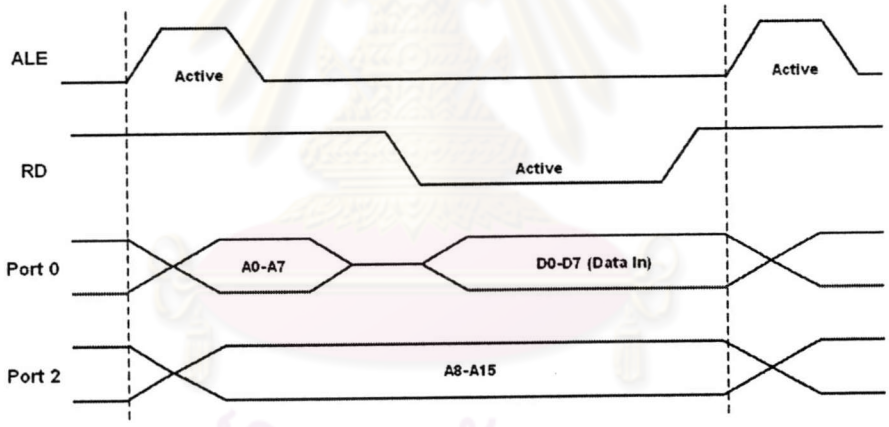
รูปที่ 2.3 การต่อหน่วยความจำข้อมูลภายนอกเพิ่ม

ในการเขียนข้อมูลลงหน่วยความจำภายนอก ในช่วงขอบขาขึ้นของสัญญาณ ALE พอร์ต 2 จะส่งตำแหน่งไบต์บน (A8-A15) ให้กับหน่วยความจำภายนอก ส่วนพอร์ต 0 จะส่งตำแหน่งไบต์ล่าง (A0-A7) เพื่อให้ไอซีแลชค้างค่าชี้ตำแหน่งแอดเดรสไบต์ล่างนี้ไว้ เมื่อสัญญาณ ALE เปลี่ยนสถานะจากลอจิก '1' เป็น '0' พอร์ต 0 จะเปลี่ยนไปส่งค่าข้อมูลขนาด 8 บิตให้กับหน่วยความจำภายนอก และสัญญาณ WR จะเปลี่ยนสถานะจาก ลอจิก '1' เป็น '0' เพื่อให้หน่วยความจำภายนอกรับข้อมูลดังแสดงในรูปที่ 2.4



รูปที่ 2.4 การเขียนข้อมูลลงหน่วยความจำข้อมูลภายนอก

ในการอ่านข้อมูลจากหน่วยความจำภายนอก ในช่วงขอบขาขึ้นของสัญญาณ ALE พอร์ต 2 จะส่งตำแหน่งไบต์ล่าง (A8-A15) ให้กับหน่วยความจำภายนอก ส่วนพอร์ต 0 จะส่งตำแหน่งไบต์ล่าง (A0-A7) เพื่อให้ไอซีแลชค้างค่าชี้ตำแหน่งแอดเดรสไบต์ล่างนี้ไว้ เมื่อสัญญาณ ALE เปลี่ยนสถานะจากลอจิก '1' เป็น '0' พอร์ต 0 จะเปลี่ยนโมดการทำงานเป็นพอร์ตอินพุตเพื่อรอรับข้อมูลจากหน่วยความจำภายนอก และเมื่อสัญญาณ RD เปลี่ยนสถานะจาก ลอจิก '1' เป็น '0' ไมโครคอนโทรลเลอร์จะอ่านข้อมูลจากหน่วยความจำภายนอกทางพอร์ต 0 ดังแสดงในรูปที่ 2.5



รูปที่ 2.5 การอ่านข้อมูลจากหน่วยความจำข้อมูลภายนอก

2.1.3 ไทเมอร์/เคาน์เตอร์ภายใน MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 มีวงจรไทเมอร์/เคาน์เตอร์ (Timer/Counter) ทำหน้าที่นับความถี่ของสัญญาณนาฬิกา หรือนับการเปลี่ยนแปลงลอจิกอยู่ในตัว โดยสามารถทำงานได้หลายโมดขึ้นอยู่กับความต้องการของผู้ใช้ ดังนั้นในการใช้งานจึงต้องกำหนดค่าให้กับรีจิสเตอร์ควบคุมที่เกี่ยวข้องต่าง ๆ เสียก่อน โดยมีรายละเอียดดังนี้

รีจิสเตอร์ TH0, TL0, TH1, TL1, TH2, TL2: เป็นรีจิสเตอร์ที่ใช้ในการนับของวงจรไทเมอร์/เคาน์เตอร์ 0, 1, และ 2 ตามลำดับ

รีจิสเตอร์ TMOD เป็นรีจิสเตอร์ขนาด 8 บิต มีแอดเดรสอยู่ที่ตำแหน่ง 0x89 ในพื้นที่ของรีจิสเตอร์ SFR ไม่สามารถเข้าถึงได้ในระดับบิต ใช้ในการกำหนดรูปแบบการทำงานของวงจรไทเมอร์/เคาน์เตอร์ 1 และ 2 มีโครงสร้างดังแสดงในรูปที่ 2.6

7	6	5	4	3	2	1	0
Gate	C/T	M1	M0	Gate	C/T	M1	M0
Timer/Counter 1				Timer/Counter 0			

รูปที่ 2.6 โครงสร้างรีจิสเตอร์ TMOD

GATE: ใช้เลือกลักษณะการควบคุมการทำงานของไทเมอร์/เคาน์เตอร์ โดยถ้าบิต GATE มีค่าเป็น '0' ไทเมอร์/เคาน์เตอร์จะทำงานเมื่อบิต TR ในรีจิสเตอร์ TCON เป็น '1' ซึ่งเป็นการควบคุมทางซอฟต์แวร์ แต่ถ้าบิต GATE มีค่าเป็น '1' ไทเมอร์/เคาน์เตอร์จะทำงานเมื่อบิต TR ในรีจิสเตอร์ TCON เป็น '1' และสถานะลอจิกที่อินพุตอินเตอร์รัปต์ INT0 และ INT1 เป็น '1' ซึ่งเป็นการควบคุมทางฮาร์ดแวร์

C/T (Timer or Counter selector): ใช้เลือกการทำงานของไทเมอร์/เคาน์เตอร์ โดยถ้าบิต C/T นี้เป็น '0' วงจรนับจะใช้สัญญาณนาฬิกาภายในไมโครคอนโทรลเลอร์เป็นอินพุต และทำงานเป็นไทเมอร์ แต่ถ้าเป็น '1' วงจรนับจะใช้สัญญาณภายนอกที่เข้ามาทางพอร์ต T0 หรือ T1 เป็นอินพุต และทำงานเป็นเคาน์เตอร์

M1, M0 (Mode selection bit): ใช้เลือกโมดการทำงานของไทเมอร์เคาน์เตอร์ โดยถ้ามีค่าเป็น "00", "01", "10", "11" วงจรนับจะทำงานในโมด 0, 1, 2, และ 3 ตามลำดับ ในการทำงานในโมด 0 เป็นการนับแบบ 13 บิต โดยใช้ 5 บิตล่างของรีจิสเตอร์ TL กับทั้ง 8 บิตของรีจิสเตอร์ TH เมื่อมีการนับไมโครคอนโทรลเลอร์จะเพิ่มค่ารีจิสเตอร์ TL ครั้งละ 1 เรื่อย ๆ จนรีจิสเตอร์ TL มีค่าเป็น "00011111_2" และเมื่อเกิดการนับเพิ่มอีกครั้ง รีจิสเตอร์ TL จะส่งบิต 1 ให้ TH นับต่อพร้อมกับรีเซตตัวเองเป็น "00000000_2" เมื่อมีการนับต่อจนกว่ารีจิสเตอร์ TL และ TH มีค่าเป็น "00011111_2" และ "11111111_2" ตามลำดับ จะเกิดการโอเวอร์โฟลว์กับบิต TF ซึ่งจะทำให้ไมโครคอนโทรลเลอร์สร้างสัญญาณอินเตอร์รัปต์สำหรับไทเมอร์/เคาน์เตอร์ และรีเซตรีจิสเตอร์ TH และ TF เป็น 0 อีกครั้ง ส่วนในโมด 1 เป็นการนับแบบ 16 บิตโดยใช้รีจิสเตอร์ TL เก็บข้อมูลไบต์ล่าง และ TH เก็บข้อมูลไบต์บน ในการทำงานจะเริ่มนับค่าจาก 0x0000 หรือจากค่าที่ตั้งเอาไว้จนถึง 0xFFFF และเกิดการโอเวอร์โฟลว์กับบิต TF ซึ่งจะทำให้ไมโครคอนโทรลเลอร์สร้างสัญญาณอินเตอร์รัปต์สำหรับไทเมอร์/เคาน์เตอร์ขึ้น และรีเซตรีจิสเตอร์ TH และ TL ให้มีค่าเป็น 0x0000 เพื่อเริ่มต้นการนับต่อไป ส่วนในโมด 2 เป็นการนับแบบตั้งค่าอัตโนมัติ 8 บิตโดยใช้รีจิสเตอร์ TL เป็นตัวนับ และใช้รีจิสเตอร์ TH เก็บค่าเริ่มต้น เมื่อรีจิสเตอร์ TL ได้นับจนมีค่าเป็น 0xFF จะเกิดการโอเวอร์โฟลว์กับบิต TF ไมโครคอนโทรลเลอร์จะสร้างสัญญาณอินเตอร์รัปต์สำหรับไทเมอร์/เคาน์เตอร์ และเปลี่ยน

ค่าของ รีจิสเตอร์ TL ให้เป็นค่าที่เก็บเอาไว้ใน TH เพื่อเริ่มต้นนับต่อไป ส่วนโมด 3 นั้นเป็นการนับ 8 บิตสำหรับวงจรไทเมอร์/เคาน์เตอร์ 0 โดยใช้รีจิสเตอร์ TH0 และ TL0 เป็นตัวนับทั้งคู่ โดยใช้ TF1 เก็บการเกิดโอเวอร์โฟลว์ของ TH0 และใช้ TF0 เก็บการเกิดโอเวอร์โฟลว์ของ TL0 การทำงานในโมดนี้จะทำให้ไทเมอร์/เคาน์เตอร์ 1 ไม่สามารถสร้างสัญญาณอินเตอร์รัปต์ได้

รีจิสเตอร์ TCON: มีขนาด 8 บิต สามารถเข้าถึงในระดับบิตได้โดยตรง ทำหน้าที่ควบคุมการทำงานของวงจรไทเมอร์/เคาน์เตอร์ และกำหนดรูปแบบการเกิดอินเตอร์รัปต์จากภายนอกมีโครงสร้างดังแสดงในรูปที่ 2.7

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

รูปที่ 2.7 โครงสร้างรีจิสเตอร์ TCON

IT0: ใช้เลือกลักษณะของสัญญาณอินเตอร์รัปต์จากภายนอกตัวที่ 0 (INT0) โดยถ้าเป็น '0' ไมโครคอนโทรลเลอร์จะใช้ขอบขาลงเป็นตัวกำเนิดสัญญาณอินเตอร์รัปต์ แต่ถ้าเป็น '1' ไมโครคอนโทรลเลอร์จะใช้ช่วงสัญญาณลอจิกต่ำเป็นตัวกำเนิดสัญญาณอินเตอร์รัปต์

IE0: ใช้ในการแสดงการเกิดอินเตอร์รัปต์จากภายนอกโดยจะมีค่าเป็น '1' เมื่อเกิดอินเตอร์รัปต์จากภายนอกขึ้น และจะกลับมามีค่าเป็น '0' เมื่อไมโครคอนโทรลเลอร์ได้ทำงานรองรับการเกิดอินเตอร์รัปต์จากภายนอกนั้นเรียบร้อยแล้ว

IT1: เหมือนกับ IT0 แต่เป็นการกำหนดสำหรับการเกิดอินเตอร์รัปต์จากภายนอกตัวที่ 1

IE1: เหมือนกับ IE0 แต่เป็นการกำหนดสำหรับการเกิดอินเตอร์รัปต์จากภายนอกตัวที่ 1

TR0: ใช้ควบคุมการเปิดปิดการทำงานของไทเมอร์/เคาน์เตอร์ 0โดยถ้าเป็น '1' วงจรไทเมอร์เคาน์เตอร์ 0 จะทำงาน แต่ถ้าเป็น '0' วงจรไทเมอร์เคาน์เตอร์ 0 จะหยุดทำงาน

TF0: เป็นบิตแสดงการเกิดโอเวอร์โฟลว์ของไทเมอร์เคาน์เตอร์ 0

TR1: เหมือนกับ TR0 แต่ใช้กับไทเมอร์/เคาน์เตอร์ 1

TF1: เหมือนกับ TF0 แต่ใช้กับไทเมอร์/เคาน์เตอร์ 1

รีจิสเตอร์ T2MOD: มีขนาด 8 บิตไม่สามารถเข้าถึงได้ในระดับบิต ใช้เลือกโมดการทำงานของวงจรไทเมอร์/เคาน์เตอร์ 2 ซึ่งมีในไมโครคอนโทรลเลอร์ MCS-51 ตระกูล 8052 เท่านั้น มีโครงสร้างดังแสดงในรูปที่ 2.8

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	DCEN

รูปที่ 2.8 โครงสร้างรีจิสเตอร์ T2MOD

DCEN: ใช้ในการกำหนดรูปแบบการนับของไทเมอร์/เคาน์เตอร์ 2 โดยถ้าเป็น '0' จะเป็นการทำงานแบบนับขึ้นเท่านั้น แต่ถ้าเป็น '1' จะทำงานแบบนับขึ้นหรือลงก็ได้โดยขึ้นอยู่กับสัญญาณอินพุตที่ขา T2EX ถ้าเป็นลอจิก '1' จะเป็นการนับขึ้น และถ้าเป็นลอจิก '0' จะเป็นการนับลง

รีจิสเตอร์ T2CON: มีขนาด 8 บิต สามารถเข้าถึงได้ในระดับบิต ใช้ควบคุมการทำงานไทเมอร์/เคาน์เตอร์ 2 มีโครงสร้างดังแสดงในรูปที่ 2.9

7	6	5	4	3	2	1	0
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2

รูปที่ 2.9 โครงสร้างรีจิสเตอร์ T2CON

CP/RL2: ใช้เลือกโมดการทำงานของไทเมอร์/เคาน์เตอร์ 2 โดยถ้าเป็น '1' จะทำงานในโมดตรวจจับสัญญาณ (Capture) และถ้าเป็น '0' จะทำงานในโมดตั้งค่านับอัตโนมัติ (Auto-Reload)

C/T2: ใช้เลือกการทำงานของไทเมอร์/เคาน์เตอร์ 2 โดยถ้าบิต C/T2 นี้เป็น '0' วงจรนับจะใช้สัญญาณนาฬิกาภายในไมโครคอนโทรลเลอร์เป็นอินพุต และทำงานเป็นไทเมอร์ แต่ถ้าเป็น '1' วงจรนับจะใช้สัญญาณภายนอกที่เข้ามาทางพอร์ต T0 หรือ T1 เป็นอินพุต และทำงานเป็นเคาน์เตอร์

TR2: ใช้ควบคุมการเปิดปิดการทำงานของไทเมอร์/เคาน์เตอร์ 2 โดยถ้าเป็น '1' วงจรไทเมอร์เคาน์เตอร์ 2 จะทำงาน แต่ถ้าเป็น '0' วงจรไทเมอร์เคาน์เตอร์ 2 จะหยุดทำงาน

EXEN2: ใช้ควบคุมการเปิดปิดวงจรตรวจจับการเปลี่ยนแปลงระดับลอจิกจาก '1' เป็น '0' ของสัญญาณที่ขา T2EX โดยถ้ามีค่าเป็น '1' จะเป็นการเปิดวงจรตรวจจับ

TCLK: ใช้เลือกไทเมอร์/เคาน์เตอร์ในการทำงานเป็นตัวกำหนดอัตราบอดสำหรับการส่งข้อมูลแบบอนุกรม โดยถ้ามีค่าเป็น '1' คือให้ไทเมอร์/เคาน์เตอร์ 2 เป็นตัวกำหนดอัตราบอดสำหรับการส่งข้อมูล แต่ถ้าเป็น '0' คือให้ไทเมอร์/เคาน์เตอร์ 1 เป็นตัวกำหนดอัตราบอดสำหรับการส่งข้อมูลแบบอนุกรม

RCLK: ใช้เลือกไทเมอร์/เคาน์เตอร์ในการทำงานเป็นตัวกำหนดอัตราบอดสำหรับการรับข้อมูลแบบอนุกรม โดยถ้ามีค่าเป็น '1' คือให้ไทเมอร์/เคาน์เตอร์ 2 เป็นตัวกำหนดอัตราบอดสำหรับการรับข้อมูล แต่ถ้าเป็น '0' คือให้ไทเมอร์/เคาน์เตอร์ 1 เป็นตัวกำหนดอัตราบอดสำหรับการรับข้อมูลแบบอนุกรม

EXF2: เป็นบิตแสดงการตั้งค่าใหม่ หรือการตรวจจับสัญญาณ ซึ่งเกิดจากสัญญาณที่ขา T2EX เปลี่ยนระดับจากลอจิก '1' เป็น '0' ในกรณีที่บิต EXEN2 ถูกตั้งให้เป็น '1'

TF2: เป็นบิตแสดงการเกิดโอเวอร์โฟลว์ของไทเมอร์เคาน์เตอร์ 2

จะเห็นได้ว่าโมดการทำงานของไทเมอร์/เคาน์เตอร์ 2 นั้นขึ้นอยู่กับรีจิสเตอร์ RCLK, TCLK, และ CP/RL2 และสามารถแบ่งการทำงานออกเป็น 3 โมดคือโมดตรวจจับสัญญาณ, โมดตั้งค่าการนับอัตโนมัติ, และโมดกำเนิดอัตราบอดในการสื่อสารข้อมูลแบบอนุกรม ในโมดตรวจจับ

สัญญาณ การใช้งานต้องกำหนดบิต EXEN2 ให้เป็น '1' ก่อนเพื่อให้วงจรตรวจจับขอบขาลงของสัญญาณที่ขาอินพุต T2EX ซึ่งตรงกับขา P1.1 ทำงาน เมื่อเปิดวงจรไทเมอร์/เคาน์เตอร์ 2 แล้ว รีจิสเตอร์ TL2 และ TH2 จะนับไปเรื่อย ๆ จนกว่าขา T2EX จะตรวจจับขอบขาลงของสัญญาณได้ เมื่อตรวจจับได้ ค่าของรีจิสเตอร์ TL2 และ TH2 จะถูกส่งต่อไปยังรีจิสเตอร์ RCAP2L และ RCAP2H ตามลำดับ พร้อมกับเซตบิต EXF2 ให้เป็น '1' เพื่อสร้างสัญญาณอินเตอร์รัปต์จากไทเมอร์/เคาน์เตอร์ขึ้น ส่วนในโมดตั้งค่าการนับอัตโนมัตินั้นสามารถทำงานได้ 2 ลักษณะขึ้นอยู่กับสถานะของบิต EXEN2 ในกรณีที่บิต EXEN2 เป็น '0' เมื่อไทเมอร์/เคาน์เตอร์ 2 นับครบรอบ จะเกิดอินเตอร์รัปต์จากการโอเวอร์โฟลว์ของบิต TF2 จากนั้นไมโครคอนโทรลเลอร์จะโหลดค่าในรีจิสเตอร์ RCAP2L และ RCAP2H เข้ามาในรีจิสเตอร์ตัวนับ TL2 และ TH2 โดยอัตโนมัติ เพื่อเริ่มนับค่าใหม่ แต่ในกรณีที่บิต EXEN2 เป็น '1' เมื่อไทเมอร์/เคาน์เตอร์ 2 นับจนครบรอบ จะเกิดอินเตอร์รัปต์ และโหลดค่าจากรีจิสเตอร์ RCAP2L และ RCAP2H เข้ามาในรีจิสเตอร์ TL2 และ TH2 เมื่อบิต EXF2 มีค่าเป็น '1' ซึ่งเกิดจากสัญญาณที่ขา T2EX เปลี่ยนแปลงจากระดับลอจิก '1' เป็น '0'

2.1.4 ระบบอินเตอร์รัปต์ของ MCS-51

การอินเตอร์รัปต์ เป็นการขัดจังหวะการทำงานโดยปกติของไมโครคอนโทรลเลอร์ ในไมโครคอนโทรลเลอร์ MCS-51 สามารถตอบสนองการอินเตอร์รัปต์ได้จาก 6 แหล่งกำเนิดคือ อินเตอร์รัปต์จากสัญญาณภายนอกที่ขา INT0 และ INT1, อินเตอร์รัปต์จากวงจรไทเมอร์/เคาน์เตอร์ 0, 1, และ 2 และอินเตอร์รัปต์จากพอร์ตสื่อสารอนุกรม ในการเลือกใช้งานอินเตอร์รัปต์นั้นสามารถทำได้โดยการตั้งค่าภายในรีจิสเตอร์ IE (Interrupt Enable) ซึ่งเป็นรีจิสเตอร์ควบคุมการเปิดปิดอินเตอร์รัปต์ต่าง ๆ มีแอดเดรสอยู่ที่ 0xA8 ใน SFR และมีโครงสร้างดังแสดงในรูปที่ 2.10

7	6	5	4	3	2	1	0
EA	-	ET2	ES	ET1	EX1	ET0	EX0

รูปที่ 2.10 โครงสร้างรีจิสเตอร์ IE

EA: ใช้ควบคุมการตอบสนองอินเตอร์รัปต์ทั้งหมดโดยถ้ามีค่าเป็น '0' ไมโครคอนโทรลเลอร์จะไม่ตอบสนองการอินเตอร์รัปต์ทั้งหมด แต่ถ้าเป็น '1' ไมโครคอนโทรเลอร์จะสามารถตอบสนองอินเตอร์รัปต์ต่าง ๆ ได้

ET2: ใช้ควบคุมการตอบสนองอินเตอร์รัปต์จากวงจรไทเมอร์/เคาน์เตอร์ 2 โดยถ้าเป็น '1' จะเป็นการเปิด และถ้าเป็น '0' จะเป็นการปิด

ES: ใช้ควบคุมการตอบสนองอินเตอร์รัปต์จากพอร์ตสื่อสารอนุกรม โดยถ้าเป็น '1' จะเป็นการเปิด และถ้าเป็น '0' จะเป็นการปิด

ET1: ใช้ควบคุมการตอบสนองอินเตอร์รัปต์จากวงจรไทเมอร์/เคาน์เตอร์ 1 โดยถ้าเป็น '1' จะเป็นการเปิด และถ้าเป็น '0' จะเป็นการปิด

EX1: ใช้ควบคุมการตอบสนองอินเตอร์รัปต์จากสัญญาณภายนอกที่ป้อนเข้ามายังขา INT1 โดยถ้าเป็น '1' จะเป็นการเปิด และถ้าเป็น '0' จะเป็นการปิด

ET0: ใช้ควบคุมการตอบสนองอินเตอร์รัปต์จากวงจรไทเมอร์/เคาน์เตอร์ 0 โดยถ้าเป็น '1' จะเป็นการเปิด และถ้าเป็น '0' จะเป็นการปิด

EX0: ใช้ควบคุมการตอบสนองอินเตอร์รัปต์จากสัญญาณภายนอกที่ป้อนเข้ามายังขา INT0 โดยถ้าเป็น '1' จะเป็นการเปิด และถ้าเป็น '0' จะเป็นการปิด

การกำหนดระดับความสำคัญให้กับสัญญาณอินเตอร์รัปต์ในไมโครคอนโทรลเลอร์ MCS-51 นั้น สามารถทำได้ 2 ระดับคือสูงและต่ำ ในการใช้งานถ้าต้องการตั้งให้อินเตอร์รัปต์หนึ่งมีความสำคัญสูงสามารถทำได้โดยการตั้งค่ารีจิสเตอร์ IP (Interrupt Priority) ซึ่งอยู่ที่ตำแหน่ง 0xB8 ใน SFR และมีโครงสร้างดังแสดงในรูปที่ 2.11 ให้ตำแหน่งอินเตอร์รัปต์นั้นมีค่าเป็น '1'

7	6	5	4	3	2	1	0
-	-	PT2	PS	PT1	PX1	PT0	PX0

รูปที่ 2.11 โครงสร้างรีจิสเตอร์ IP

PT2: อินเตอร์รัปต์จากวงจรไทเมอร์/เคาน์เตอร์ 2

PS: อินเตอร์รัปต์จากพอร์ตสื่อสารอนุกรม

PT1: อินเตอร์รัปต์จากวงจรไทเมอร์/เคาน์เตอร์ 1

PX1: อินเตอร์รัปต์จากสัญญาณภายนอกที่ป้อนเข้ามายังขา INT1

PT0: อินเตอร์รัปต์จากวงจรไทเมอร์/เคาน์เตอร์ 0

PX0: อินเตอร์รัปต์จากสัญญาณภายนอกที่ป้อนเข้ามายังขา INT0

2.2 ระบบโทรศัพท์อินเทอร์เน็ต และโพรโทคอลที่เกี่ยวข้อง

2.2.1 ระบบโทรศัพท์บนโครงข่ายอินเทอร์เน็ต

ระบบโทรศัพท์บนโครงข่ายอินเทอร์เน็ต คือระบบโทรศัพท์ที่ใช้เทคนิคการส่งข้อมูลเสียงและซิกแนลลิงผ่านโครงข่ายอินเทอร์เน็ต ซึ่งปัจจุบันโครงข่ายได้พัฒนา และขยายครอบคลุมทั่วโลก ประกอบกับค่าใช้จ่ายในการใช้บริการอินเทอร์เน็ตมีแนวโน้มต่ำลงเรื่อย ๆ ดังนั้นการใช้งานโทรศัพท์บนโครงข่ายอินเทอร์เน็ตจึงมีค่าใช้จ่ายต่ำกว่าระบบโทรศัพท์ PSTN ที่ใช้ในปัจจุบันมาก ซึ่งเป็นประโยชน์ต่อผู้ใช้งานปลายทางโดยตรง ในส่วนผู้ให้บริการก็ได้ประโยชน์เช่นกันเนื่องจากการส่งข้อมูลเสียงผ่านโครงข่ายอินเทอร์เน็ตสามารถใช้เทคนิคการบีบอัดข้อมูลเสียงให้มีขนาดเล็กลงกว่าการเข้ารหัสแบบ PCM ที่ใช้ในระบบโทรศัพท์ PSTN หลายเท่า ประกอบกับโครงข่ายอิน

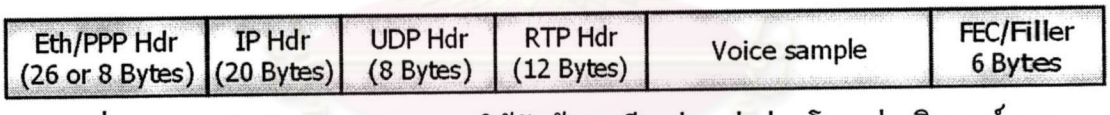
เทอร์เนตเป็นโครงข่ายแบบสวิตช์แพ็กเกตซึ่งมีการส่งข้อมูลเป็นแพ็กเกตไม่ต้องจองวงจรตลอดเวลา เหมือนกับโครงข่าย PSTN ซึ่งเป็นแบบสวิตช์วงจร จึงทำให้การใช้งานช่องสัญญาณผ่านอุปกรณ์สื่อสัญญาณมีประสิทธิภาพ และสามารถรองรับผู้ใช้งานได้มากยิ่งขึ้น

การเข้ารหัสสัญญาณเสียง เพื่อใช้ในระบบโทรศัพท์บนโครงข่ายอินเทอร์เนต สามารถทำได้หลายวิธี โดยแต่ละวิธีให้คุณภาพเสียง อัตราข้อมูล และการประวิงเวลาในการเข้ารหัสที่แตกต่างกัน ซึ่งคุณลักษณะทั้งหมดนี้เป็นปัจจัยที่สำคัญ และมีผลต่อคุณภาพของเสียง และคุณภาพของการให้บริการ ดังแสดงในตารางที่ 2.1

ตารางที่ 2.1 เปรียบเทียบคุณลักษณะของตัวอย่างการเข้ารหัสสัญญาณเสียงแบบต่าง ๆ

Voice Coding	Bit Rate (kbps)	Coding Delay (ms)	MOS (Mean Opinion Score)
G.711 PCM	64	0.125	4.3-4.4
G.726 ADPCM	32	1	4.0-4.2
G.729 CS-ACELP	8	15	4.0-4.2
G.723.1 ACELP	5.3-6.4	37.5	3.5-4.0

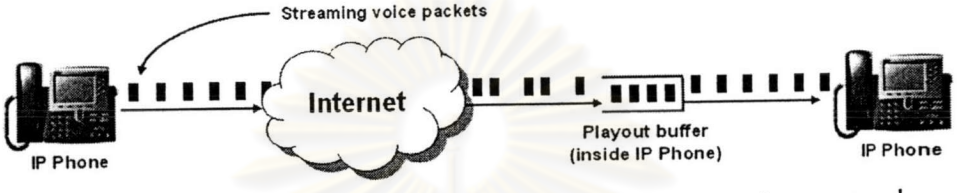
เนื่องจากการให้บริการเสียงเป็นการให้บริการที่ต้องการคุณสมบัติเวลาจริง และข้อมูลเสียงสามารถทนต่อการเกิดข้อมูลสูญหายบางส่วนได้ ดังนั้นในการส่งข้อมูลเสียงผ่านโครงข่ายอินเทอร์เนต ข้อมูลเสียงจะถูกนำมาผ่านการเติมข่าวสารการควบคุม (Encapsulation) ด้วยเฮดเดอร์ RTP (Real Time Protocol) [6] และ UDP (User Datagram Protocol) [7] ตามลำดับก่อนเพิ่มเฮดเดอร์ IP (Internet Protocol) เพื่อให้สามารถส่งผ่านโครงข่ายอินเทอร์เนตได้ดังแสดงในรูปที่ 2.12



รูปที่ 2.12 การเติมข่าวสารการควบคุมให้กับข้อมูลเสียงก่อนส่งผ่านโครงข่ายอินเทอร์เนต

การประวิงทางเวลาของข้อมูลเสียงระหว่างต้นทาง และปลายทางเป็นปัจจัยสำคัญที่มีผลต่อคุณภาพของการบริการระบบโทรศัพท์บนโครงข่ายอินเทอร์เนต ถ้าข้อมูลเสียงเกิดการประวิงเวลาระหว่างต้นทาง และปลายทางมากเกินไปจะทำให้การสนทนาไม่ต่อเนื่อง และไม่เป็นธรรมชาติได้ การประวิงทางเวลาของข้อมูลสามารถแบ่งออกได้เป็น 2 ชนิดคือ การประวิงเวลาที่เกิดจากการแพร่กระจายของสัญญาณผ่านตัวกลาง (Propagation delay) และ การประวิงเวลาที่เกิดจากการประมวลผลที่อุปกรณ์ (Handling delay) ในทางปฏิบัติการประวิงทางเวลาที่เกิดจากการเดินทางของสัญญาณผ่านตัวกลางมีค่าน้อยมากเมื่อเทียบกับการประวิงทางเวลาที่เกิดจากการประมวลผลของอุปกรณ์ปลายทาง และอุปกรณ์ภายในโครงข่าย ดังนั้นในการพัฒนาออกแบบอุปกรณ์ และการเลือกวิธีการประมวลผลแพ็กเกตข้อมูลเสียงของอุปกรณ์ภายในโครงข่าย และอุปกรณ์ปลายทางจึงเป็นสิ่งสำคัญ และมีผลต่อคุณภาพของการให้บริการโดยตรง

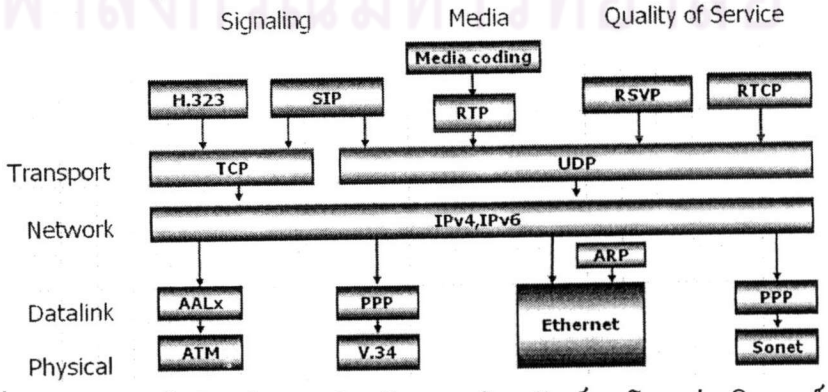
นอกจากการประวิงทางเวลาแล้ว ในการส่งข้อมูลเสียงผ่านโครงข่ายอินเทอร์เนต ข้อมูลเสียงแต่ละแพ็กเกตจะใช้เวลาในการเดินทางจากต้นทางไปยังปลายทางไม่เท่ากัน ซึ่งทำให้ข้อมูลที่ปลายทางได้รับนั้นมี Jitter เกิดขึ้น และถ้านำข้อมูลเสียงนั้นไปถอดรหัสทันทีจะทำให้เกิดความไม่ต่อเนื่องของเสียงได้ การลด Jitter สามารถทำได้โดยการนำข้อมูลเสียงที่ได้รับจากต้นทางมาเก็บไว้ในบัฟเฟอร์ก่อนช่วงเวลาหนึ่งแล้วถึงค่อยนำข้อมูลเสียงนั้นไปถอดรหัสดังแสดงในรูปที่ 2.13 โดยถ้ายิ่งบัฟเฟอร์ข้อมูลเสียงมากก็จะสามารถลด Jitter ได้มาก แต่ข้อมูลเสียงจะเกิดการประวิงทางเวลามากขึ้นเช่นกัน ดังนั้นการเลือกขนาดของบัฟเฟอร์จึงเป็นสิ่งที่สำคัญต่อคุณภาพเสียงที่ผู้รับจะได้ยิน



รูปที่ 2.13 การแก้ไข Jitter ที่เกิดกับแพ็กเกตข้อมูลเสียงโดยการบัฟเฟอร์ข้อมูลเสียงที่ภาครับ

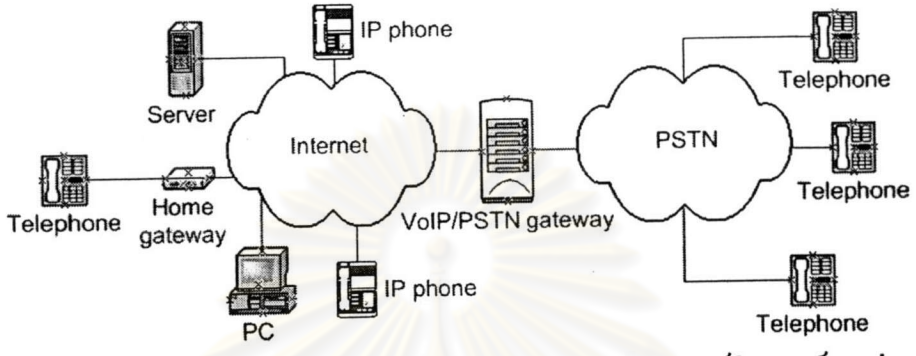
การสื่อสารข้อมูลเสียงระหว่างต้นทางกับปลายทางในระบบโทรศัพท์บนโครงข่ายอินเทอร์เนตนั้น จำเป็นต้องใช้การสัญญาณ (Signaling) เพื่อควบคุมการสร้าง สิ้นสุด และเปลี่ยนแปลงเซสชันการสนทนาระหว่างต้นทาง กับปลายทาง ซึ่งในปัจจุบันมีการสัญญาณที่นิยมใช้อยู่ 2 ชนิดด้วยกันคือ H.323 และ SIP (Session Initiation Protocol) แต่เนื่องจาก SIP นั้นถูกพัฒนาขึ้นภายหลังและมีแนวทางในการพัฒนาเพื่อรองรับการทำงานบนโครงข่ายอินเทอร์เนตโดยตรง จึงทำให้ SIP เป็นโพรโทคอลที่มีความซับซ้อนน้อย สามารถเพิ่มเติมคุณลักษณะของโพรโทคอลได้ง่าย และสามารถรองรับขนาดของโครงข่ายที่ใหญ่ได้ดีกว่า H.323

ระบบโทรศัพท์บนโครงข่ายอินเทอร์เนตมีโครงสร้างโพรโทคอลดังแสดงในรูปที่ 2.14 โพรโทคอล H.323 และ SIP อยู่ในระดับชั้นประยุกต์ใช้งาน (Application layer) ทำงานบนโพรโทคอลในชั้นเคลื่อนย้ายที่ต่ำกว่า โดย SIP นั้นสามารถทำงานได้ทั้งบน UDP และ TCP (Transport Control Protocol) ส่วน H.323 นั้นสามารถทำงานได้บน TCP เท่านั้น โพรโทคอล RTCP (Real Time Control Protocol) และ RSVP (Resource Reservation Protocol) [8] ถูกใช้ในการควบคุมคุณภาพของการบริการข้อมูลเสียง ซึ่งต้องการคุณสมบัติเวลาจริงบนโครงข่ายอินเทอร์เนต



รูปที่ 2.14 โครงสร้างโพรโทคอลสำหรับระบบโทรศัพท์บนโครงข่ายอินเทอร์เนต

เนื่องจากระบบโทรศัพท์ PSTN ที่ใช้ในปัจจุบันมีการใช้งานอย่างแพร่หลาย ดังนั้นในการเปลี่ยนมาใช้ระบบโทรศัพท์บนโครงข่ายอินเทอร์เนต จึงต้องค่อย ๆ เปลี่ยนทีละส่วน โดยเริ่มจากส่วนของผู้ให้บริการก่อน แล้วจึงค่อยเปลี่ยนในส่วนของผู้ใช้บริการซึ่งมีจำนวนมาก ในช่วงการเปลี่ยนแปลงนี้จึงต้องใช้ Gateway เพื่อให้ระบบโทรศัพท์ PSTN และระบบโทรศัพท์บนโครงข่ายอินเทอร์เนตสามารถทำงานร่วมกันได้ ดังแสดงในรูปที่ 2.15



รูปที่ 2.15 การใช้งานระบบโทรศัพท์ PSTN และระบบโทรศัพท์อินเทอร์เนตร่วมกัน

2.2.2 โพรโทคอลอีเทอร์เนต

โพรโคคอลอีเทอร์เนต เป็นโพรโทคอลชั้นเชื่อมโยงข้อมูล (datalink layer) ที่ใช้งานอย่างแพร่หลายในโครงข่ายพื้นที่ท้องถิ่น และในกรณีที่โครงข่ายพื้นที่ท้องถิ่นเป็นโครงข่ายแบบแพร่สัญญาณ โพรโทคอลอีเทอร์เนตจะใช้เทคนิค CSMA/CD (Carrier Sense Multiple Access with Collision Detection) ในการควบคุมการเข้าถึงตัวกลาง โครงสร้างเฟรมของอีเทอร์เนตนั้นมีหลายชนิด ในกรณีที่ใช้กับโพรโทคอล TCP/IP นั้นเป็นแบบ Ethernet_II ดังแสดงในรูปที่ 2.16

8 bytes	6 bytes	6 bytes	2 bytes	46-1500 bytes	4 bytes
preamble	destination address	source address	type	data	CRC

รูปที่ 2.16 โครงสร้างเฟรมข้อมูลของโพรโทคอลอีเทอร์เนตชนิด Ethernet_II

ฟิลด์ preamble มีขนาด 8 ไบต์มีค่าเป็น 1 และ 0 สลับกันไปตลอด ใช้ในการซิงโครไนซ์และบ่งบอกจุดเริ่มต้นของเฟรม

ฟิลด์ destination address มีขนาด 6 ไบต์ ใช้ระบุฮาร์ดแวร์แอดเดรสของสถานีปลายทาง

ฟิลด์ source address มีขนาด 6 ไบต์ ใช้ระบุฮาร์ดแวร์แอดเดรสของสถานีต้นทาง

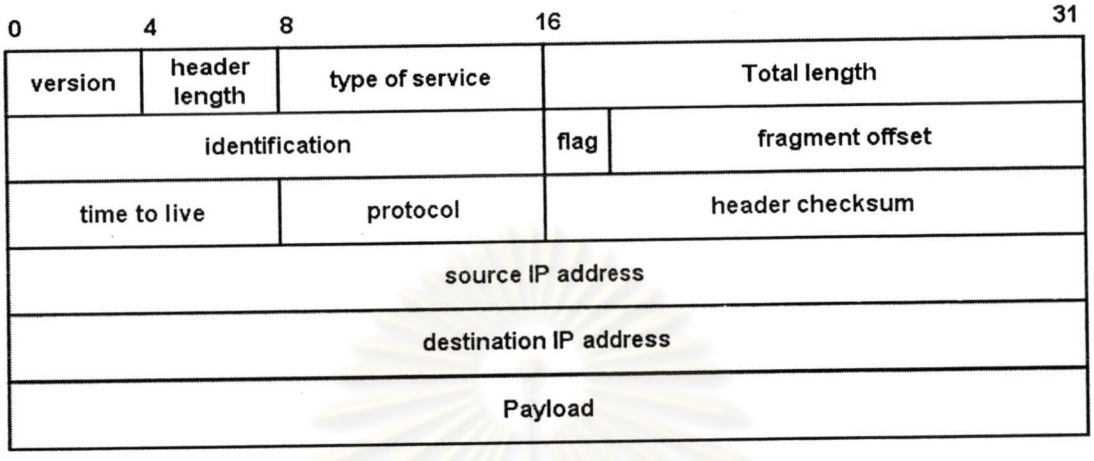
ฟิลด์ type มีขนาด 2 ไบต์ ใช้ระบุชนิดของโพรโทคอลชั้นบน โดยถ้ามีค่าเป็น 0x0800 แสดงว่าโพรโทคอลชั้นบนเป็น IP และถ้ามีค่าเป็น 0x0806 แสดงว่าโพรโทคอลชั้นบนเป็น ARP

ฟิลด์ data มีขนาดตั้งแต่ 46 ถึง 1500 ไบต์ ใช้บรรจุข้อมูลของโพรโทคอลในชั้นที่สูงกว่า

ฟิลด์ CRC มีขนาด 4 ไบต์ ใช้สำหรับตรวจวัดความผิดพลาดของข้อมูลที่อาจเกิดขึ้นระหว่างการรับส่งสัญญาณ

2.2.3 โพรโทคอลไอพี (IP Protocol)

โพรโทคอลไอพี เป็นโพรโทคอลในชั้นโครงข่าย (network layer) ใช้ในการส่งแพ็กเกตข้อมูลจากต้นทาง ไปยังปลายทางให้ถูกต้อง มีโครงสร้างดังแสดงในรูปที่ 2.17

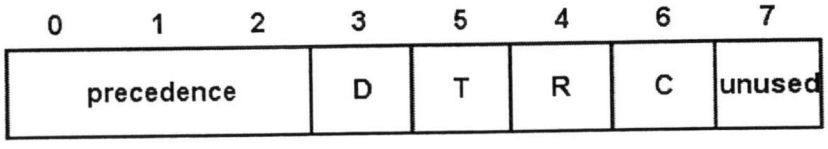


รูปที่ 2.17 โครงสร้างดาตาแกรมของโพรโทคอลไอพี

ฟิลด์ version มีขนาด 4 บิต ใช้ในการระบุเวอร์ชันของโพรโทคอล IP ที่ใช้ในแพ็กเกตนั้น ซึ่งโพรโทคอล IP ที่ใช้อยู่ในปัจจุบันเป็นเวอร์ชัน 4

ฟิลด์ header length มีขนาด 4 บิต ใช้บ่งบอกขนาดของเฮดเดอร์ของไอพีดาตาแกรมในหน่วยของเวิร์ด (32 บิต)

ฟิลด์ type of service มีขนาด 8 บิต ใช้บ่งบอกรูปแบบการให้บริการของไอพีดาตาแกรมนั้น มีโครงสร้างดังแสดงในรูปที่ 2.18 ส่วน precedence มีขนาด 3 บิตใช้บ่งบอกความสำคัญของไอพีดาตาแกรม โดยบ่งบอกได้ 8 ระดับตั้งแต่ 0 ซึ่งมีความสำคัญน้อยที่สุด จนถึงระดับ 7 ซึ่งมีความสำคัญมากที่สุด อย่างไรก็ตามปัจจุบันยังไม่มีการนำมาใช้งานจริงในทางปฏิบัติ ส่วนอีก 4 บิตที่เหลือซึ่งประกอบไปด้วยบิต D, T, R, และ C นั้นใช้ในการบ่งบอกคุณภาพของการให้บริการ โดยถ้าบิต C เป็น '1' แสดงว่าไอพีดาตาแกรมนี้ต้องการการประวิงทางเวลาต่ำ ถ้าบิต T เป็น '1' แสดงว่าไอพีดาตาแกรมนี้ต้องการช่องสัญญาณที่มีความจุสูง ถ้าบิต R เป็น '1' แสดงว่าไอพีดาตาแกรมนี้ต้องการช่องสัญญาณที่มีความเชื่อถือได้สูง และบิต C ถ้าเป็น '1' แสดงว่าไอพีดาตาแกรมต้องการช่องสัญญาณที่มีค่าใช้จ่ายน้อยที่สุด บิต D, T, R, และ C นี้จะใช้ในการประมวลผลของเราเตอร์เพื่อเลือกเส้นทาง โดยปกติแล้วต้นทางจะเซตเพียงแค่บิตใดบิตหนึ่งเท่านั้น เนื่องจากการเซตหลายบิตเพื่อให้ตอบสนองหลายอย่างนั้นในความเป็นจริงเป็นไปได้ยาก



รูปที่ 2.18 โครงสร้างของฟิลด์ type of service (TOS)

ฟิลด์ total length มีขนาด 16 บิต ใช้บ่งบอกขนาดของไอพีดาตาแกรมในหน่วยของไบต์

ฟิลด์ identification มีขนาด 16 บิต ใช้ในการระบุหมายเลขประจำตัวของไอพีดาตาแกรมแต่ละตัว มีประโยชน์ในกรณีที่มีการแบ่งข้อมูลออกเป็นส่วนย่อยๆ (fragmentation) โดยปลายทางจะใช้ค่าในฟิลด์ identification นี้เป็นส่วนประกอบในการสร้างข้อมูลเดิมกลับมา

ฟิลด์ flag มีขนาด 3 บิต บิตแรกไม่มีการใช้งาน และถูกกำหนดให้เป็น 0 เสมอ บิตที่สองเรียกว่าบิต D ใช้บ่งบอกการให้อนุญาตแบ่งไอพีดาตาแกรมออกเป็นส่วนย่อย ๆ โดยที่ถ้า D = 0 หมายความว่าสถานีต้นทางอนุญาตให้อุปกรณ์เราเตอร์แบ่งไอพีดาตาแกรมนั้นออกเป็นส่วนย่อย ๆ ได้ แต่ถ้า D = 1 แสดงว่าสถานีต้นทางไม่อนุญาตให้อุปกรณ์เราเตอร์แบ่งไอพีดาตาแกรมนั้นออกเป็นส่วนย่อย ๆ บิตที่สามเรียกว่าบิต M จะถูกใช้เพื่อระบุว่ามีการแบ่งไอพีดาตาแกรมที่เราเตอร์เกิดขึ้น โดยเมื่อเราเตอร์มีการแบ่งไอพีดาตาแกรม เราเตอร์จะเซตบิต M ของไอพีดาตาแกรมแต่ละส่วนให้เป็น 1 ยกเว้นไอพีดาตาแกรมส่วนสุดท้ายจะเป็น 0 เมื่อปลายทางได้รับส่วนของไอพีดาตาแกรมที่มี M = 1 ปลายทางจะทราบว่าไอพีดาตาแกรมนั้นผ่านการแบ่งเป็นส่วนย่อย ๆ มา และจะรอรับส่วนของไอพีดาตาแกรมต่อ ๆ ไป จนกว่าจะได้รับไอพีดาตาแกรมส่วนสุดท้ายซึ่งจะมีค่า M = 0

ฟิลด์ fragment offset มีขนาด 13 บิต ทำหน้าที่เป็นตัวชี้ตำแหน่งของดาตาแกรม ในกรณีที่ดาตาแกรมนั้นผ่านการแบ่งออกเป็นส่วนย่อย ๆ

ฟิลด์ time to live มีขนาด 8 บิต ใช้ระบุจำนวนเราเตอร์สูงสุดที่ไอพีดาตาแกรมนี้สามารถผ่านได้

ฟิลด์ protocol มีขนาด 8 บิต ใช้ระบุชนิดของโพรโทคอลในชั้นเคลื่อนย้าย ตัวอย่างโพรโทคอลที่สำคัญแสดงในตารางที่ 2.2

ตารางที่ 2.2 ตัวอย่างการกำหนดค่าโพรโทคอลที่สำคัญในฟิล์ด protocol ของไอพีดาตาแกรม

Decimal	Protocol
1	ICMP (Internet Control Message Protocol)
6	TCP (Transmission Control Protocol)
8	EGP (Exterior Gateway Protocol)
17	UDP (User Datagram Protocol)

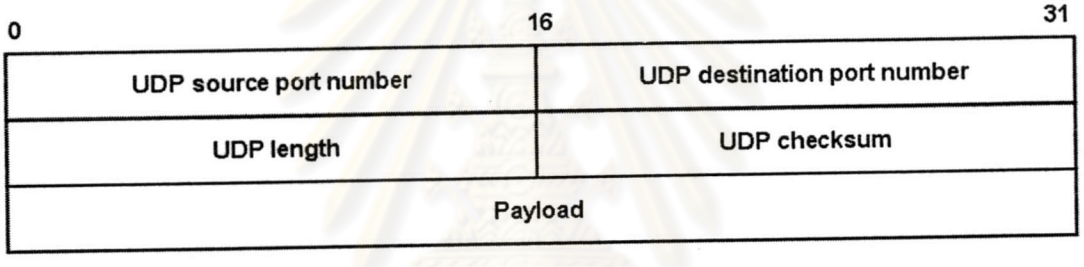
ฟิลด์ header checksum มีขนาด 16 บิต ใช้ในการตรวจวัดความถูกต้องของเฮดเดอร์ไอพี ในการสร้างฟิลด์นี้ สามารถทำได้โดยการนำเฮดเดอร์มาบวกกันแบบ one's complement ครั้งละ 16 บิตจนครบ จากนั้นนำผลรวมที่ได้มากลับบิต (bit inversion) แล้วค่อยบรรจุลงในฟิลด์ ทางภาครับเมื่อได้รับข้อมูลสามารถตรวจวัดความถูกต้องของเฮดเดอร์ได้โดย การนำค่าภายในเฮดเดอร์นั้นมาบวกกันแบบ one's complement ครั้งละ 16 บิตจนครบ ถ้าผลที่ได้มีค่าเป็น 1 ทั้ง 16 บิตแสดงว่าไม่มีความผิดพลาดเกินขึ้นในเฮดเดอร์

ฟิลด์ source IP address มีขนาด 32 บิต ใช้บ่งบอกไอพีแอดเดรสของสถานีต้นทาง

ฟิลด์ destination IP address มีขนาด 32 บิต ใช้บ่งบอกไอพีแอดเดรสของสถานีปลายทาง

2.2.4 โพรโทคอล UDP

โพรโทคอล UDP เป็นโพรโทคอลในชั้นเคลื่อนย้าย (transport layer) ทำหน้าที่มัลติเพลกซ์การใช้งานแอปพลิเคชัน และเป็นตัวกลางต่อร่วมระหว่างแอปพลิเคชัน กับระบบโครงข่าย การทำงานของโพรโทคอล UDP นั้นเป็นแบบ connectionless รับประกันความถูกต้องของ UDP ดาตาแกรมเท่านั้น แต่ไม่รับประกันความถูกต้องของข้อมูลข่าวสารทั้งหมดที่ส่ง นั่นคือข้อมูลอาจเกิดการสูญหาย หรือมาถึงปลายทางโดยไม่เรียงลำดับได้ ดังนั้นโพรโทคอล UDP จึงเป็นโพรโทคอลที่มีโครงสร้างเรียบง่าย ไม่ซับซ้อน เหมาะสมสำหรับระบบที่มีอัตราความผิดพลาดต่ำ หรือการส่งข้อมูลที่ยอมรับความผิดพลาดได้ในระดับหนึ่ง เช่นการส่งข้อมูลเสียง หรือวิดีโอแบบเวลาจริงเป็นต้น โครงสร้างดาตาแกรมของโพรโทคอล UDP แสดงดังรูปที่ 2.19



รูปที่ 2.19 โครงสร้างดาตาแกรมของโพรโทคอล UDP

ฟิลด์ UDP source port number มีขนาด 16 บิต ใช้ในการระบุหมายเลขพอร์ตต้นทาง

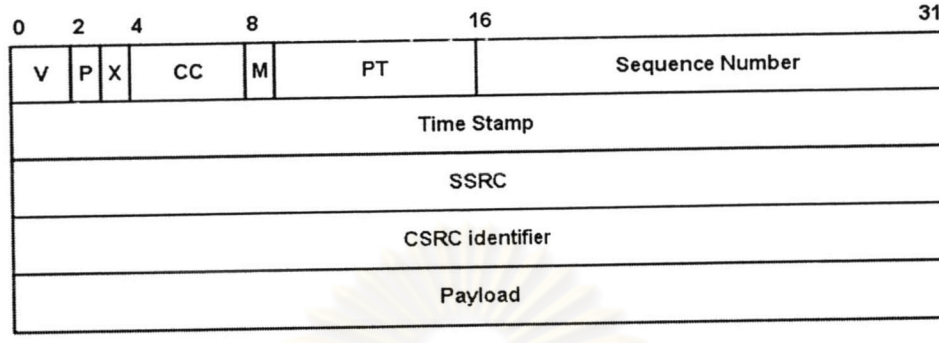
ฟิลด์ UDP destination port number มีขนาด 16 บิต ใช้ในการระบุหมายเลขพอร์ตปลายทาง และเป็นการระบุชนิดของแอปพลิเคชันที่สถานีต้นทางต้องการติดต่อกับสถานีปลายทางด้วย

ฟิลด์ UDP length มีขนาด 16 บิต ใช้ในการระบุขนาดของ UDP ดาตาแกรมโดยมีหน่วยเป็นไบต์

ฟิลด์ UDP checksum มีขนาด 16 บิต ใช้ในการตรวจวัดความถูกต้องของ UDP ดาตาแกรมโดยการคำนวณใช้วิธีการบวกแบบ one's complement ครั้งละ 16 บิตเช่นเดียวกับการคำนวณ checksum ของเฮดเดอร์ IP แต่เนื่องจากฟิลด์ UDP checksum นี้เป็นการตรวจสอบ UDP ดาตาแกรมทั้งหมด ดังนั้นจึงต้องบวกรวม UDP ดาตาแกรมทั้งหมด อย่างไรก็ตามการคำนวณ checksum ของโพรโทคอล UDP นี้ไม่ได้เป็นข้อบังคับว่าผู้พัฒนาต้องคำนวณเสมอไป ในกรณีที่ผู้พัฒนาเลือกที่จะไม่ตรวจวัดความผิดพลาดก็สามารถทำได้โดยกำหนดค่าภายในฟิลด์ checksum ทั้ง 16 บิตให้เป็น 0 ทั้งหมด

2.2.5 โพรโทคอล RTP (Real Time Protocol)

โพรโทคอล RTP เป็นโพรโทคอลที่ใช้ในการส่งข้อมูลผ่านโครงข่าย แบบเวลาจริงเช่นการส่งเสียง และวิดีโอมีโครงสร้างดังแสดงในรูปที่ 2.20



รูปที่ 2.20 โครงสร้างดาตาแกรมของโพรโทคอล RTP

ฟิลด์ version (V) มีขนาด 2 บิต ใช้ระบุเวอร์ชันของโพรโทคอล RTP โพรโทคอล RTP ที่ใช้งานอยู่ปัจจุบันเป็นเวอร์ชัน 2

ฟิลด์ padding (P) มีขนาด 1 บิต โดยถ้าบิต P นี้ถูกตั้งให้เป็น '1' แสดงว่าภายในแพ็กเกตมีการเพิ่มข้อมูลแพดไว้ในตอนท้าย นอกเหนือจากข้อมูล payload

ฟิลด์ extension (X) มีขนาด 1 บิต ถ้าเป็น '1' แสดงว่ามีเฮดเดอร์เพิ่มเติมนอกเหนือจากเฮดเดอร์ RTP ที่มีขนาดคงที่

ฟิลด์ CSRC count (CC) มีขนาด 4 บิต ใช้บ่งบอกจำนวนแหล่งกำเนิดข้อมูล ในกรณีที่มีแหล่งกำเนิดเดียวค่าภายในฟิลด์จะเป็น '0' หมด

ฟิลด์ marker bit (M) มีขนาด 1 บิต ใช้บ่งบอกว่ามีการกระทำพิเศษเกิดขึ้นบนแพ็กเกตนั้น

ฟิลด์ payload type (PT) มีขนาด 7 บิต ใช้บ่งบอกชนิดของ payload ที่บรรจุอยู่ในแพ็กเกต RTP มีรายละเอียดดังแสดงในตารางที่ 2.3

ฟิลด์ sequence number มีขนาด 16 บิต ค่าในฟิลด์นี้จะเพิ่มขึ้นครั้งละ 1 ทุก ๆ ครั้งที่มีการส่งแพ็กเกต RTP ออกไป โดยค่าเริ่มต้นได้จากการสุ่ม ผู้รับจะใช้ค่าในฟิลด์นี้ตรวจสอบการสูญหายของข้อมูล และแก้ไขลำดับของข้อมูลที่ได้รับมา

ฟิลด์ time stamp มีขนาด 32 บิต ค่าในฟิลด์นี้จะถูกสุ่มเลือกก่อนการสร้างเซสชัน และจะมีค่าเพิ่มขึ้นทุกครั้งที่มีการส่งแพ็กเกต RTP ออกไป ใช้บ่งบอกจุดเวลาในการชักตัวอย่าง (sampling) ของข้อมูลไบต์แรกภายใน RTP payload เพื่อให้ภาครับใช้ในการซิงโครไนซ์ข้อมูล และนำไปคำนวณหา Jitter

ฟิลด์ SSRC มีขนาด 32 บิต ใช้ระบุต้นทางผู้สร้างแพ็กเกต RTP โดยค่าในฟิลด์นี้ผู้สร้างแพ็กเกต RTP จะสุ่มเลือกก่อนการสร้างเซสชัน เพื่อหลีกเลี่ยงการซ้ำกับผู้สร้างแพ็กเกต RTP อื่น

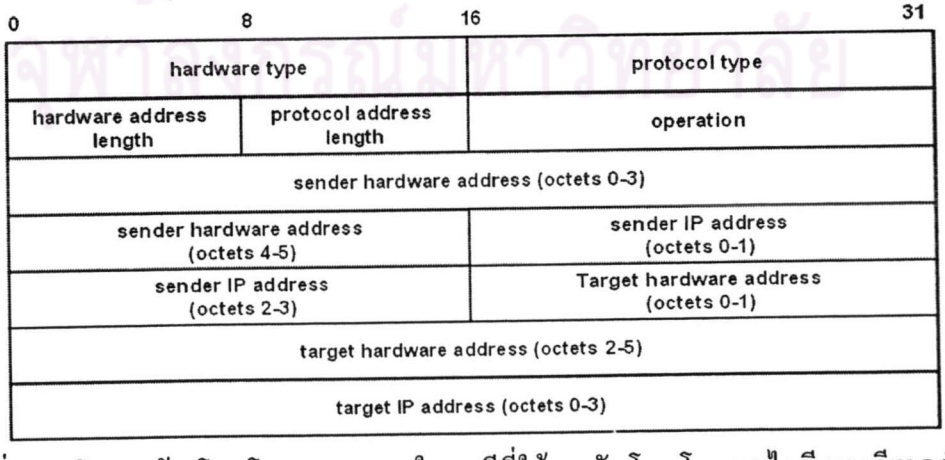
ฟิลด์ CSRC มีขนาด 32 บิต ใช้ระบุแหล่งกำเนิดข้อมูล ในกรณีที่มีแหล่งกำเนิดเดียว ฟิลด์นี้จะถูกละทิ้ง

ตารางที่ 2.3 ค่าในฟิลด์ PT ซึ่งบ่งบอกการเข้ารหัสเสียง และภาพของข้อมูล

Payload Type	Codec	Description
0	PCMU	ITU G.711 μ-Law Audio 64 kbps
1	1016	CELP Audio 4.8 kbps
2	G.721	ITU G.721 ADPCM Audio 32 kbps
3	GSM	European GSM Audio 32 kbps
5	DVI4	DVI ADPCM Audio 32 kbps
6	DVI4	DVI ADPCM 64 kbps
7	LPC	Experimental LPC Audio
8	PCMA	ITU G.711 PCM A-Law Audio 64 kbps
9	G.722	ITU G.722 Audio
10	L16	Linear 16 bit Audio 705.6 kbps
11	L16	Linear 16 bit Stereo Audio 1411.2 kbps
14	MPA	MPEG-I or MPEG-II Audio only
15	G.728	ITU G.728 Audio 16 kbps
31	H.261	ITU H.261 Video
32	MPV	MPEG-I and MPEG-II Video
33	MP2T	MPEG-II transport stream Video

2.2.6 โพรโทคอล ARP (Address Resolution Protocol)

โพรโทคอล ARP ทำงานอยู่ในระหว่างชั้นโครงข่าย และชั้นเชื่อมโยงข้อมูลใช้ในการค้นหาฮาร์ดแวร์แอดเดรสของอุปกรณ์ภายในโครงข่ายพื้นที่ท้องถิ่น จากไอพีแอดเดรสที่ทราบมีโครงสร้างดังแสดงในรูปที่ 2.21



รูปที่ 2.21 โครงสร้างโพรโทรคอล ARP ในกรณีที่ใช้รองรับโพรโทคอลไอพี และอีเทอร์เนต

ฟิลด์ hardware type มีขนาด 2 ไบต์ ใช้บ่งบอกชนิดของโพรโทคอลชั้นเชื่อมโยงข้อมูล ที่โพรโทคอล ARP นี้ทำงานเพื่อรองรับอยู่ โดยในกรณีที่เป็นอีเทอร์เนต ฟิลด์ hardware type จะถูกกำหนดให้มีค่าเป็น 1

ฟิลด์ protocol type มีขนาด 2 ไบต์ ใช้บ่งบอกชนิดของโพรโทคอลชั้นโครงข่ายที่โพรโทคอล ARP นี้ทำงานเพื่อรองรับอยู่ โดยในกรณีที่เป็นโพรโทคอลไอพี ฟิลด์ protocol type จะถูกกำหนดให้มีค่าเป็น 0x0800

ฟิลด์ hardware address length มีขนาด 1 ไบต์ ใช้บ่งบอกความยาวของฮาร์แวร์แอดเดรสที่โพรโทคอล ARP นี้ทำงานรองรับอยู่ ในกรณีที่เป็นอีเทอร์เนต ฟิลด์ hardware address length นี้จะถูกกำหนดให้มีค่าเป็น 6

ฟิลด์ protocol address length มีขนาด 1 ไบต์ ใช้บ่งบอกความยาวของแอดเดรส ที่ใช้ในโพรโทคอลชั้นบน ในกรณีที่โพรโทคอลชั้นบนเป็นโพรโทคอลไอพี ฟิลด์ protocol address length นี้จะถูกกำหนดให้มีค่าเป็น 4

ฟิลด์ operation มีขนาด 2 ไบต์ ใช้ในการแยกแยะชนิดของโพรโทคอล ARP โดยถ้าเป็นข้อมูลร้องขอ ARP ฟิลด์ operation นี้จะถูกกำหนดให้มีค่าเป็น 1 และในกรณีที่เป็นข้อมูลตอบสนอง ARP จะถูกกำหนดให้มีค่าเป็น 2

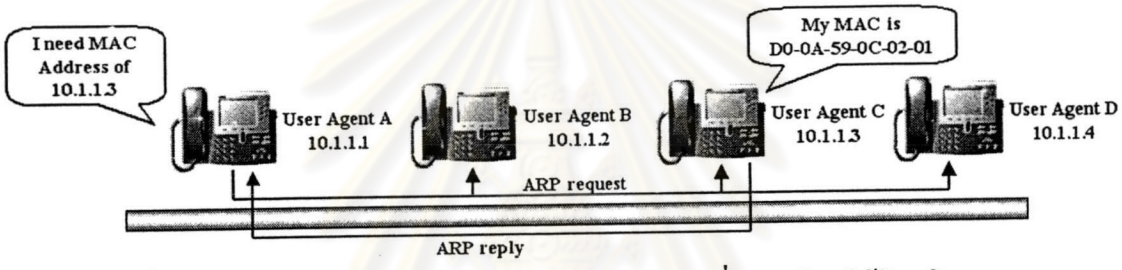
ฟิลด์ sender hardware address ใช้ในการบ่งบอกฮาร์ดแวร์แอดเดรสของผู้ส่ง มีความยาวสัมพันธ์กับค่าภายในฟิลด์ hardware address length ในกรณีใช้งานเพื่อรองรับโพรโทคอลอีเทอร์เนตจะมีขนาด 6 ไบต์

ฟิลด์ sender IP address ใช้ในการบ่งบอกไอพีแอดเดรสของผู้ส่ง มีความยาวสัมพันธ์กับค่าภายในฟิลด์ protocol address length ในกรณีใช้งานเพื่อรองรับโพรโทคอลไอพีจะมีขนาด 4 ไบต์

ฟิลด์ target hardware address ใช้ในการบ่งบอกฮาร์ดแวร์แอดเดรสปลายทาง มีความยาวสัมพันธ์กับค่าภายในฟิลด์ hardware address length ในกรณีใช้งานเพื่อรองรับโพรโทคอลอีเทอร์เนตจะมีขนาด 6 ไบต์ ในการส่งข้อความร้องขอ ARP จะไม่มีการใส่ค่าลงในฟิลด์นี้ เนื่องจากเป็นค่าที่โฮสต์ต้องการทราบจากปลายทางนั่นเอง

ฟิลด์ target IP address มีขนาด 4 ไบต์ ใช้ในการบ่งบอกไอพีแอดเดรสปลายทาง โดยจะมีความยาวสัมพันธ์กับค่าภายในฟิลด์ protocol address length ในกรณีใช้งานเพื่อรองรับโพรโทคอลไอพีจะมีขนาด 4 ไบต์

ในการส่งข้อมูลไปยังอุปกรณ์ปลายทางผ่านโครงข่ายอินเทอร์เนต นอกจากการใช้ไอพีแอดเดรสเป็นตัวระบุที่อยู่ของอุปกรณ์ปลายทางแล้ว ยังต้องอาศัยโครงสร้างทางกายภาพ หรือฮาร์ดแวร์แอดเดรสของอุปกรณ์ปลายทางนั้นด้วย ดังแสดงในรูปที่ 2.22 เครื่องลูกข่าย A ต้องการสร้างการติดต่อกับเครื่องลูกข่าย C ซึ่งมีไอพีแอดเดรส 10.1.1.3 เครื่องลูกข่าย A จะประกาศผ่านเฟรมชนิดแพร่สัญญาณ (broadcasting frame) เพื่อถามหาฮาร์ดแวร์แอดเดรส ของเครื่องลูกข่าย C เมื่อเครื่องลูกข่าย C ได้รับเฟรมแพร่สัญญาณนี้ ก็จะตรวจสอบไอพีแอดเดรสที่ประกาศไว้ และเทียบกับไอพีแอดเดรสของตัวเอง ถ้าตรงกันก็จะส่งเฟรมที่ระบุฮาร์ดแวร์แอดเดรสของตัวเองกลับไปให้เครื่องลูกข่าย A โดยตรง เมื่อเครื่องลูกข่าย A ได้รับเฟรมตอบสนองนี้แล้วก็จะเก็บคู่ไอพีแอดเดรสกับฮาร์ดแวร์แอดเดรสที่ทราบแล้วไว้ในแคช (cache) ของตัวเอง เพื่อให้ในกรณีที่เครื่องลูกข่าย A ต้องการส่งข้อมูลหาเครื่องลูกข่าย C อีกก็จะสามารถส่งข้อมูลได้โดยตรง โดยไม่ต้องอาศัยโพรโทคอล ARP อีก



รูปที่ 2.22 การหาฮาร์ดแวร์แอดเดรสจากไอพีแอดเดรสที่ทราบโดยใช้โพรโทคอล ARP

2.2.7 โพรโทคอล SIP (Session Initiation Protocol)

SIP เป็นโพรโทคอลในชั้นประยุกต์ใช้งาน ที่พัฒนาขึ้นโดยคณะทำงาน IETF (Internet Engineering Task Force) ใช้ในการสร้าง สิ้นสุด และเปลี่ยนแปลงแก้ไขมัลติมีเดียเซสชันบนโครงข่ายอินเทอร์เนต โพรโทคอล SIP ได้รับการออกแบบมาให้ง่ายต่อการพัฒนาใช้งาน สะดวกต่อการเพิ่มเติมคุณลักษณะของโพรโทคอล และสามารถใช้กับโครงข่ายขนาดใหญ่ได้ดี เราสามารถแบ่งการทำงานของ SIP ออกได้เป็น 5 ส่วนหลัก ๆ ดังนี้

1. User location: ค้นหา และระบุที่อยู่ของผู้ใช้ปลายทาง
2. User capability: บ่งบอกพารามิเตอร์ของเซสชันที่ผู้ใช้งานสามารถรองรับได้
3. User availability: รองรับความใช้สอยได้ของผู้ถูกเรียกที่จะเข้าร่วมการสื่อสาร
4. Call setup: สร้างเซสชัน และกำหนดพารามิเตอร์ของเซสชัน
5. Call handling: ควบคุมการโอนสาย และการสิ้นสุดการเรียก

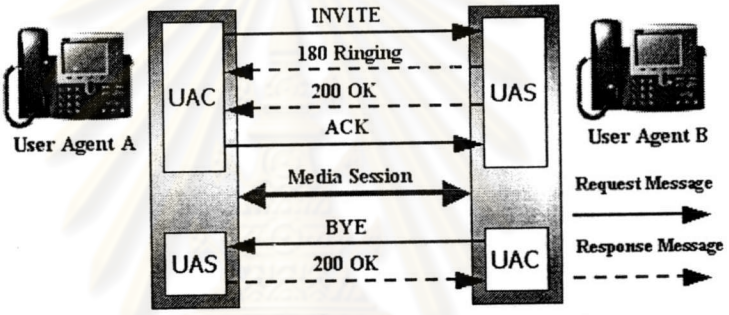
โพรโทคอล SIP รับการออกแบบให้มีลักษณะคล้ายคลึงกับโพรโทคอลมาตรฐานอื่น ๆ บนอินเทอร์เนตเช่น HTTP (Hypertext Transfer Protocol) และ SMTP (Simple Mail Transfer Protocol)

SIP ได้กำหนดลักษณะที่สำคัญของการติดต่อไว้ 2 ส่วนคือ ส่วนของเครื่องลูกข่าย (User Agent) และส่วนของเครื่องแม่ข่าย (Network Server)

เครื่องลูกข่าย คือ ระบบปลายทาง ซึ่งเป็นได้ทั้งผู้เริ่มต้นการเรียกสาย และผู้ถูกเรียกสาย สามารถแบ่งรูปแบบการทำงานของเครื่องลูกข่ายออกเป็น 2 ส่วนคือ

1. User Agent Client (UAC): คือเครื่องลูกข่ายที่ร้องขอ และรอการตอบสนองการร้องขอนั้น
2. User Agent Server (UAS): คือเครื่องลูกข่ายที่รับการร้องขอ และตอบสนองการร้องขอนั้นกลับไป

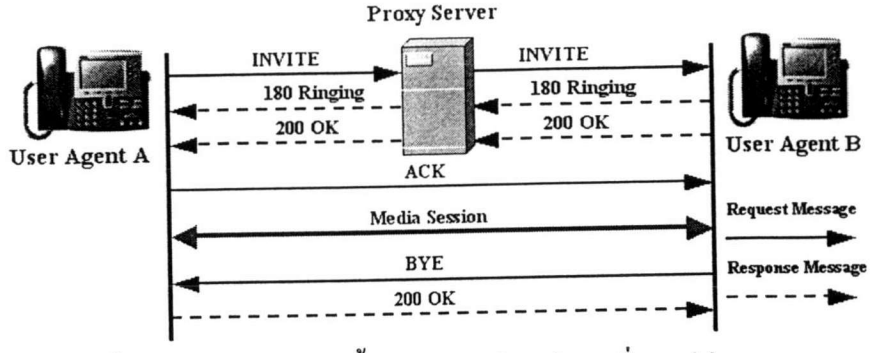
การติดต่อเพื่อสร้าง และสิ้นสุดเซสชันระหว่างเครื่องลูกข่ายซึ่งเป็นการติดต่อแบบพื้นฐานที่สุดแสดงในรูปที่ 2.23 จะเห็นได้ว่าเครื่องลูกข่ายแต่ละเครื่องจะต้องสามารถร้องขอ และตอบสนองการเรียกได้ นั่นก็คือเครื่องลูกข่ายแต่ละเครื่องจะต้องสามารถเป็นได้ทั้ง UAC และ UAS



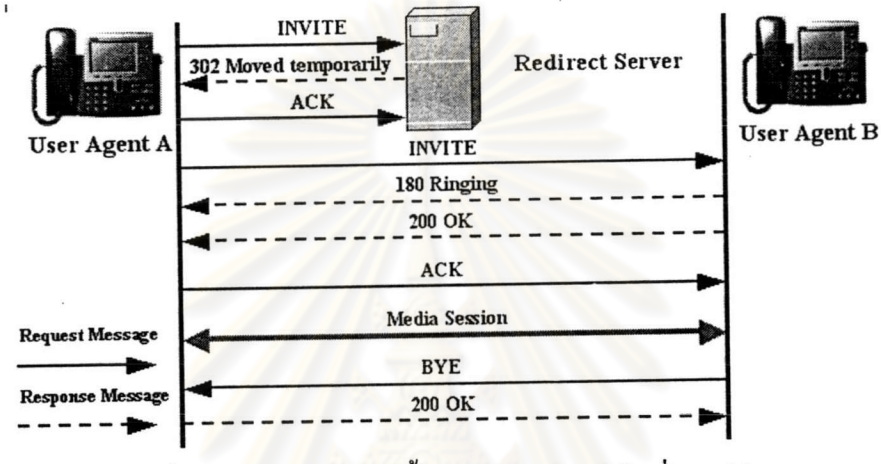
รูปที่ 2.23 การสร้าง และสิ้นสุดเซสชันระหว่างเครื่องลูกข่าย

ในส่วนของเครื่องแม่ข่ายนั้นมีทำหน้าที่จัดการกับข้อความที่ได้รับจาก เครื่องลูกข่าย หรือเครื่องแม่ข่ายอื่น ๆ สามารถแบ่งเครื่องแม่ข่ายออกได้เป็น 2 ชนิด คือ

1. SIP Proxy Server: ทำหน้าที่ระบุที่อยู่ และส่งข้อความร้องขอการเปิดเซสชัน (INVITE Message) ที่ได้รับต่อไปยังเครื่องลูกข่าย หรือเครื่องแม่ข่ายถัดไป และส่งต่อข้อความตอบสนองการร้องขอการเปิดเซสชันนั้นกลับไปทางเส้นทางเดิม ดังแสดงในรูปที่ 2.24
2. Redirect Server: มีหน้าที่ระบุที่อยู่ และส่งข้อความตอบสนองการเปิดเซสชันที่ระบุที่อยู่ของเครื่องลูกข่ายปลายทาง หรือเครื่องแม่ข่ายถัดไป กลับไปให้เครื่องลูกข่ายที่ร้องขอการเปิดเซสชันมา เพื่อให้เครื่องลูกข่ายนั้นส่งข้อความร้องขอการเปิดเซสชันไปยังเครื่องลูกข่ายปลายทาง หรือเครื่องแม่ข่ายถัดไปโดยตรง ดังแสดงในรูปที่ 2.25
3. Registrar Server: มีหน้าที่รับ Register Message จากเครื่องลูกข่าย และเก็บข้อมูลที่จำเป็นของผู้ใช้งานเช่น SIP URL, IP address และพอร์ตของผู้ใช้งาน เพื่อเป็นข้อมูลให้ Proxy Server และ Redirect Server ใช้ในการทำงาน



รูปที่ 2.24 การสร้าง และสิ้นสุดเซสชันโดยผ่านเครื่องแม่ข่าย Proxy



รูปที่ 2.25 การสร้าง และสิ้นสุดเซสชันโดยใช้เครื่องแม่ข่าย Redirect

SIP เป็นโพรโทคอลที่มีรูปแบบข้อความเป็นตัวอักขระคล้ายกับมาตรฐาน HTTP/1.1 สามารถแบ่งชนิดของข้อความออกเป็น 2 ชนิดคือ ข้อความร้องขอ (SIP Request) ใช้ร้องขอจากเครื่องลูกข่ายไปยังเครื่องแม่ข่าย และข้อความตอบสนอง (SIP Response) ใช้ตอบรับ หรือปฏิเสธการร้องขอจากเครื่องแม่ข่ายกลับไปยังเครื่องลูกข่าย ในการส่งข้อความร้องขอ และข้อความตอบสนอง จะอยู่ในรูปแบบของตัวอักขระ ISO 10646 ที่เข้ารหัสแบบ UTF-8 [10] และใช้อักขระพิเศษ CRLF (Carriage Return/Line Feed) ในการระบุการขึ้นบรรทัดใหม่ ภายในข้อความ SIP ประกอบไปด้วยบรรทัดเริ่มต้น (Start-line) ตามด้วยส่วนเฮดเดอร์ฟิลด์ (Header field) และอาจมี Message body โดยใช้ บรรทัดว่างเปล่า คั่นระหว่างส่วนเฮดเดอร์ฟิลด์กับส่วน Message body สามารถแสดงโครงสร้างของ ข้อความ SIP ได้ดังต่อไปนี้

generic-message = start-line
*message-header fields
CRLF
[message-body]

สำหรับข้อความร้องขอ SIP จะมีบรรทัดเริ่มต้นเป็น Request-Line ซึ่งมีรูปแบบเป็น

Method space ***Request-URI*** space ***SIP-Version*** CRLF

- SIP-Version คือ เวอร์ชันของ SIP ที่ใช้ในการควบคุมเซสชันนั้น เช่น SIP/2.0
- Request-URI (Uniform-Resource-Identifiers) คือ SIP URL ใช้ในการระบุผู้ส่งข้อความร้องขอนั้น
- Method คือชนิดของข้อความร้องขอ โดยโพรโทคอล SIP ได้แบ่งข้อความร้องขอออกเป็น 6 ชนิดดังนี้

1. INVITE: คือข้อความร้องขอที่ใช้ในการเชิญผู้ใช้งานปลายทางให้เข้าร่วมเซสชัน โดยจะมี Message body แสดงถึงค่าพารามิเตอร์ต่าง ๆ ของเซสชันที่ผู้เรียกต้องการใช้ในการติดต่อ เมื่อผู้รับการเรียกได้รับข้อความร้องขอ INVITE แล้ว จะส่งข้อความตอบสนองกลับไปเพื่อตอบรับ หรือปฏิเสธการเรียกนั้น ในกรณีที่ตอบรับการเรียกผู้รับจะระบุค่าพารามิเตอร์ต่าง ๆ ที่ผู้รับต้องการไว้ใน Message body ของข้อความตอบสนองนั้น
2. ACK: คือข้อความร้องขอที่ใช้ในการยืนยันการต่อถึงกัน ใช้ในกรณีที่ UAC ได้รับข้อความตอบสนองสุดท้ายของการร้องขอ INVITE จากเครื่องแม่ข่ายเรียบร้อยแล้ว โดยอาจมี Message body เพื่อแสดงค่าพารามิเตอร์ที่ใช้ หากไม่มี Message body เซสชันที่สร้างขึ้นจะใช้ค่าพารามิเตอร์ที่กำหนดในข้อความร้องขอ INVITE
3. BYE: คือข้อความร้องขอที่ใช้ในการขอยกเลิกการติดต่อ โดยทั้งผู้เรียก และผู้รับการเรียก สามารถส่งข้อความร้องขอ BYE นี้ได้
4. CANCEL: คือข้อความร้องขอที่ใช้สำหรับบอกผู้รับปลายทางว่าผู้ส่งต้องการยกเลิกข้อความร้องขอที่ได้ส่งไปก่อนหน้านี้ และยังไม่ได้รับการตอบรับกลับมา
5. OPTIONS: คือข้อความร้องขอที่ใช้บอกข้อมูลเพิ่มเติมเกี่ยวกับความสามารถ (capability information) ของเครื่องลูกข่ายให้เครื่องแม่ข่ายทราบ และใช้ถามข้อมูลเกี่ยวกับเครื่องแม่ข่าย
6. REGISTER: คือข้อความร้องขอที่ใช้สำหรับการขอลงทะเบียน ระหว่างเครื่องลูกข่ายกับเครื่องแม่ข่าย

หลังจากที่ผู้รับ (UAS หรือเครื่องแม่ข่าย) ได้รับข้อความร้องขอ และได้ตีความข้อความร้องขอนั้นแล้ว ผู้รับสามารถตอบรับได้ โดยการส่งข้อความตอบสนองที่เหมาะสมกลับไป ข้อความตอบสนองที่ส่งนั้นอาจบอกผู้ส่งถึงสถานะปัจจุบันของผู้รับ หรือบอกการตัดสินใจว่ารับ หรือปฏิเสธการเรียกนั้น

ข้อความตอบสนองของ SIP จะมีบรรทัดเริ่มต้นเป็น Status-Line ซึ่งมีรูปแบบเป็น

SIP-Version space ***Status-Code*** space ***Reason-Phrase*** CRLF

- Reason-Phrase เป็นข้อความสั้น ๆ ที่อธิบายความหมายของ Status-Code
- Status-Code เป็นเลขจำนวนเต็ม 3 หลักที่แสดงถึงการตอบสนองต่อข้อความร้องขอ ตัวเลขหลักแรกบ่งบอกถึงคลาสของข้อความตอบสนอง แบ่งออกได้เป็น 6 คลาส คือ
  1. 1xx Informational: เป็นข้อความตอบสนองใช้เมื่อปลายทางได้รับข้อความร้องขอแล้วและกำลังดำเนินการต่อข้อความร้องขอนั้นอยู่ เช่นการตอบสนอง 180 RINGING เมื่อผู้ถูกเรียกได้รับ INVITE และกำลังรอการตอบรับจากผู้ใช้ปลายทางเป็นต้น
  2. 2xx Success: เป็นข้อความตอบสนองใช้เพื่อตอบตกลงยอมรับการเรียก โดยอาจส่งค่าพารามิเตอร์ที่เหมาะสมไปกับ Message body ด้วย เช่น ส่ง 200 OK เพื่อตอบตกลงยอมรับข้อความร้องขอ INVITE เป็นต้น
  3. 3xx Redirection: เป็นข้อความตอบสนองใช้สำหรับแจ้งให้ผู้เรียกทราบว่าข้อความร้องขอนั้นต้องการการกระทำเพิ่มเติม เพื่อให้การร้องขอนั้นเป็นผลสำเร็จ เช่น การแจ้งเปลี่ยนที่อยู่ปลายทาง ผู้รับการเรียกจะส่งข้อความตอบสนอง 3xx กลับไปบอกให้ผู้เรียกส่งข้อความร้องขอใหม่ไปยังที่อยู่ใหม่ที่ปรากฏใน Contact header ของข้อความตอบสนอง 3xx และเริ่มต้นเรียกใหม่อีกครั้งหนึ่ง
  4. 4xx Client error: เป็นข้อความตอบสนองที่ใช้บ่งบอกว่าข้อความร้องขอนั้นมีข้อผิดพลาด และผู้รับไม่สามารถตอบสนองต่อการร้องขอนั้นได้
  5. 5xx Server error: เป็นข้อความตอบสนองที่ใช้บ่งบอกว่าเกิดข้อผิดพลาดขึ้นในส่วนของเครื่องแม่ข่าย ทำให้เครื่องแม่ข่ายไม่สามารถตอบสนองต่อการร้องขอนั้นได้
  6. 6xx Global failure: เป็นข้อความตอบสนองที่ใช้บ่งบอกว่าเครื่องแม่ข่ายทุกเครื่องไม่สามารถตอบสนองต่อการร้องขอนั้นได้

เครื่องลูกข่ายไม่จำเป็นต้องเข้าใจข้อความตอบสนองทุกชนิด แต่ต้องสามารถพิจารณาข้อความตอบสนองที่ไม่รู้จักว่าเป็นข้อความในคลาสไหนได้ เช่น เมื่อได้รับ 603 Decline ต้องสามารถตีความเป็น 600 Busy everywhere แทนได้

ส่วนเฮดเดอร์ฟิลด์ใช้ในการแสดงข้อมูลต่าง ๆ ที่จำเป็น เช่น ผู้เรียกสาย, ผู้ถูกเรียกสาย, ความยาวของข้อความ เป็นต้น โดยเครื่องลูกข่ายไม่จำเป็นต้องเข้าใจเฮดเดอร์ทั้งหมด สามารถละเลยเฮดเดอร์ที่ไม่เข้าใจได้

เฮดเดอร์ของ SIP มีทั้งหมด 37 เฮดเดอร์ มีเฮดเดอร์ที่สำคัญ และใช้บ่อยแสดงได้ดังต่อไปนี้

- To ใช้บ่งบอกถึงผู้รับข้อความร้องขอ
- From ใช้บ่งบอกถึงผู้ส่งข้อความร้องขอ
- Call-ID ใช้บ่งบอกหมายเลขของการติดต่อ โดยหมายเลขเหล่านี้จะไม่ซ้ำกันในการติดต่อแต่ละครั้ง
- Contact ใช้แสดงที่อยู่ปลายทาง ที่ผู้รับสามารถใช้ในการติดต่อได้โดยตรง
- Via ใช้แสดงเส้นทางในการส่งข้อความ
- CSeq ใช้บ่งบอกหมายเลขของข้อความร้องขอสำหรับ Call ID นั้น ๆ
- Content-Length ใช้บ่งบอกความยาวของ message body หน่วยเป็นไบต์

ส่วน Message Body ใช้ในการส่งข่าวสารเพื่อบ่งบอกลักษณะของเซสชันมัลติมีเดียที่จะสร้างขึ้น เช่น อัตราข้อมูล การเข้ารหัสสัญญาณเสียง การต่อถึงกัน ฯลฯ โพรโทคอล SIP ได้กำหนดรูปแบบของ Message Body ให้เป็นไปตามมาตรฐานโพรโทคอล SDP (Session Description Protocol) [9] ซึ่งถูกกำหนดอยู่ใน RFC 2327 ของคณะทำงาน IETF ดังแสดงตัวอย่างได้ดังนี้

```
v=0
o=bob 289084470 2890844730 IN IP4 host.example.com
s=Hello, How are you ?
c=IN IP4 host.example.com
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

- ฟิลด์ “v=” แสดงเวอร์ชันของโพรโทคอล SDP
- ฟิลด์ “o=” แสดงข้อมูลของผู้ส่งข้อความ SDP
- ฟิลด์ “s=” แสดงหัวข้อของการสนทนา
- ฟิลด์ “c=” แสดงข้อมูลของการต่อมัลติมีเดียเซสชัน
- ฟิลด์ “m=” แสดงพอร์ต และชนิดของการให้บริการมัลติมีเดีย
- ฟิลด์ “a=” แสดงรายละเอียดของการเข้ารหัสข้อมูลมัลติมีเดีย

ชนิดของเครื่องลูกข่ายที่ระบุตามมาตรฐาน SIP สามารถแบ่งตามความสามารถของเครื่องลูกข่ายได้เป็น ความสามารถขั้นต่ำ (Minimum), ความสามารถพื้นฐาน (Basic), ความสามารถโอนสาย (Redirect), ความสามารถในการทำงานร่วมกับไฟร์วอลล์ (Firewall friendly), ความสามารถในการต่อรองระหว่างเครื่องลูกข่าย (Negotiation), และความสามารถในการตรวจพิสูจน์ผู้ใช้ (Authentication) ดังแสดงในตารางที่ 2.4

ตารางที่ 2.4 เปรียบเทียบความสามารถของเครื่องลูกข่ายแต่ละชนิด

User agent type	Supports
Minimum	INVITE, ACK, SDP, response classes
Basic	Minimum + BYE
Redirection	Basic + Contact header
Firewall friendly	Redirection + Route, Record-Route, and default proxy server
Negotiation	Firewall + OPTION, Warning, 380 response
Authentication	Negotiation + 401 response, WWW-Authenticate, and Authorization header