บทที่ 3

การพัฒนาเครื่องโทรศัพท์อินเทอร์เนตต้นแบบ

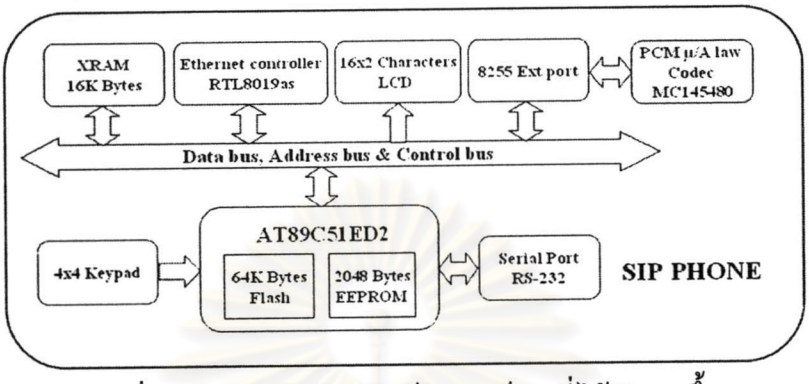
3.1 การเลือกอุปกรณ์สำหรับการพัฒนาเครื่องโทรศัพท์อินเทอร์เนตต้นแบบ

โดยทั่วไปเครื่องโทรศัพท์อินเทอร์เนตจะประกอบไปด้วยส่วนประมวลผล, หน่วยความจำ, ส่วนเข้ารหัสเสียง, ส่วนติดต่อกับอีเทอร์เนต, และส่วนติดต่อกับผู้ใช้งาน ซึ่งในการพัฒนาเครื่องโทรศัพท์อินเทอร์เนตต้นแบบในวิทยานิพนท์นี้ได้ใช้เกณฑ์การเลือกอุปกรณ์ต่าง ๆ จากทั้งในเรื่องราคา, ความสามารถในการทำงาน, ความง่ายในการหาซื้อ และความสะดวกต่อการนำมาพัฒนา

ในส่วนประมวลผลนั้นต้องสามารถติดต่อกับอุปกรณ์อื่นได้โดยง่าย, ง่ายต่อการพัฒนาทางซอฟท์แวร์, และสามารถทำงานได้อย่างรวดเร็วเนื่องจากเครื่องโทรศัพท์ต้องรับส่งข้อมูลเสียงแบบเวลาจริง ในการพัฒนาจึงได้เลือกใช้ชิพ AT89C51ED2 ซึ่งเป็นไมโครคอนโทรลเลอร์ตระกูล MCS-51 ของบริษัท ATMEL สามารถทำงานได้เร็วกว่า MCS-51 พื้นฐาน 2 เท่า ภายในมีหน่วยความจำโปรแกรมแบบแฟลชขนาด 64 กิโลไบต์ และสามารถโหลดโปรแกรมแบบ ISP (In-System Programming) ลงตัวชิพโดยผ่านพอร์ตอนุกรมได้โดยตรง ทำให้การพัฒนาสามารถทำได้สะดวก นอกจากนั้นยังมี EEPROM ขนาด 2,048 ไบต์ และหน่วยความจำข้อมูลภายนอกเพิ่มเติมขนาด 1,792 ไบต์ แต่เนื่องจากการทำงานของเครื่องโทรศัพท์อินเทอร์เนตนั้นจำเป็นต้องใช้หน่วยความจำข้อมูลมากกว่า 1,792 ไบต์ ดังนั้นจึงต้องใช้ชิพหน่วยความจำข้อมูลภายนอกเพิ่มขนาด 32 กิโลไบต์ ในส่วนอุปกรณ์เข้ารหัสเสียงได้ใช้ชิพ MC145480 ของบริษัท MOTOROLA เนื่องจากสามารถหาซื้อได้ภายในประเทศ และรองรับการเข้ารหัสเสียง PCM ทั้ง μ-law และ A-law ซึ่งเป็นการเข้ารหัสเสียงพื้นฐานที่เครื่องโทรศัพท์อินเทอร์เนตต้องสามารถรองรับได้ แต่เนื่องจากชิพ MC145480 นั้นรับส่งข้อมูล PCM เป็นแบบอนุกรม ดังนั้นเพื่อให้ง่ายต่อการรับส่งข้อมูลกับไมโครคอนโทรลเลอร์จึงได้ใช้ชิพ S/P, P/S และชิพ 82c55 แปลงข้อมูลให้เป็นแบบขนาน และสามารถทำ memory map เพื่อรับส่งข้อมูลกับไมโครคอนโทรลเลอร์ได้โดยง่าย ซึ่งจะอธิบายรายละเอียดในหัวข้อ 3.6 ต่อไป ในส่วนติดต่อกับอีเทอร์เนตนั้นได้ใช้ชิพควบคุมอีเทอร์เนต RTL8019as ของบริษัท REALTEK เนื่องจากชิพ RTL8019as มีรีจิสเตอร์ควบคุมตามมาตรฐาน NE2000 และสามารถรับส่งติดต่อกับไมโครคอนโทรลเลอร์ MCS-51 ผ่านระบบบัสข้อมูลขนาด 8 บิตได้โดยตรง ในส่วนติดต่อกับผู้ใช้งานนั้น ส่วนรับข้อมูลได้ใช้คีย์แพด 4x4 เนื่องจากเครื่องโทรศัพท์ต้องรับข้อมูลตัวเลข, ตัวอักขระ และการควบคุมเซสชันจากผู้ใช้งาน ในส่วนแสดงผลข้อมูลได้ใช้ LCD 16x2 ซึ่งสามารถแสดงได้ทั้งตัวอักขระ และตัวเลขเพื่อแสดงผลการทำงานให้กับผู้ใช้งานได้

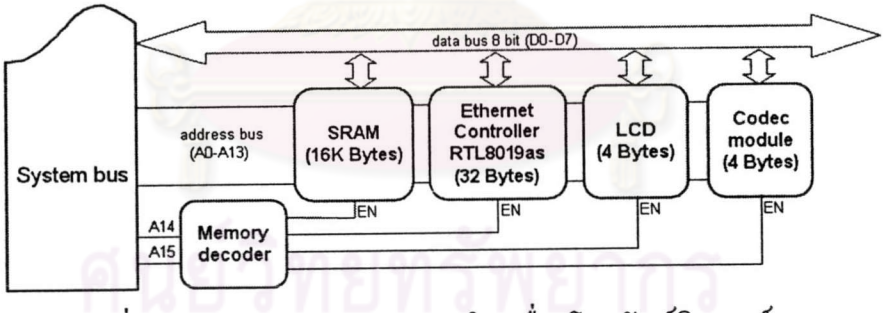
3.2 โครงสร้างเครื่องโทรศัพท์อินเทอร์เนตต้นแบบ

เครื่องโทรศัพท์อินเทอร์เนตต้นแบบประกอบไปด้วย AT89C51ED2 เป็นส่วนประมวลผลหลัก, หน่วยความจำภายนอก, ส่วนควบคุมอีเทอร์เนต, ส่วนเข้ารหัสเสียง และส่วนติดต่อกับผู้ใช้งานดังแสดงในรูปที่ 3.1



รูปที่ 3.1 โครงสร้างโทรศัพท์อินเทอร์เนตที่ได้พัฒนาขึ้น

ในการติดต่อสื่อสารข้อมูลระหว่างไมโครคอนโทรลเลอร์กับ ส่วนควบคุมอีเทอร์เนต, ส่วนแสดงผล LCD, และส่วนเข้ารหัสสัญญาณเสียงนั้นได้ต่อแบบ memory map โดยใช้สายสัญญาณข้อมูล, สายสัญญาณกำหนดตำแหน่งหน่วยความจำ, และสายสัญญาณควบคุม เช่นเดียวกับการติดต่อกับหน่วยความจำภายนอกตามที่ได้อธิบายข้างต้น และใช้ชิพเข้ารหัส 74HC135 เข้ารหัสตำแหน่งหน่วยความจำบิต A14, A15 เพื่อแยกตำแหน่งของอุปกรณ์แต่ละส่วนออกจากกันดังแสดงในรูปที่ 3.2 เมื่อทำ memory map แล้วอุปกรณ์แต่ละส่วนจะมีตำแหน่งดังแสดงในตารางที่ 3.1



รูปที่ 3.2 การทำ Memory map ภายในเครื่องโทรศัพท์อินเทอร์เนต

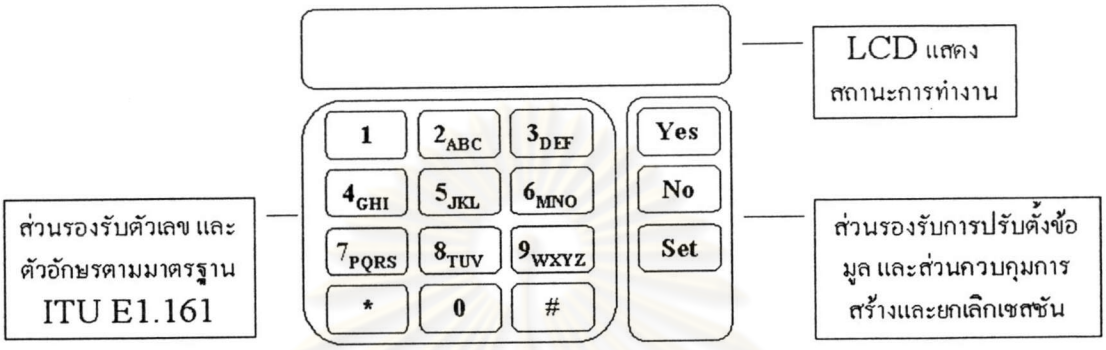
ตารางที่ 3.1 ตำแหน่งการทำ Memory map ภายในเครื่องโทรศัพท์อินเทอร์เนต

อุปกรณ์ภายนอก	จำนวนสายสัญญาณกำหนดตำแหน่งหน่วยความจำ (บิต)	ขนาด (ไบต์)	ตำแหน่ง
หน่วยความจำภายนอก SRAM	14	16,384	0x0000-0x3FFF
ส่วนควบคุมอีเทอร์เนต RTL8019as	5	32	0x4000-0x4031
ส่วนแสดงผล LCD	2	4	0x8000-0x8003
ส่วนเข้ารหัสเสียง PCM	2	4	0xC000-0xC003

ในการพัฒนาแบ่งออกได้เป็น 4 ส่วนหลัก ๆ คือ ส่วนติดต่อกับผู้ใช้งาน, ส่วนควบคุมการรับส่งข้อมูลผ่านชิพอีเทอร์เนต, ส่วนรองรับโพรโทคอล และส่วนควบคุมข้อมูลเสียง

3.3 ส่วนติดต่อกับผู้ใช้งาน

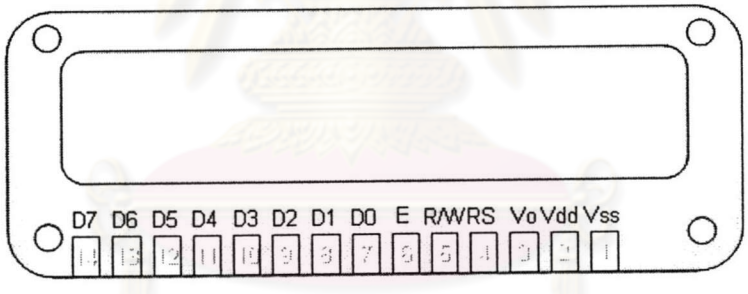
ส่วนติดต่อกับผู้ใช้งานประกอบไปด้วยส่วนแสดงสถานะการทำงานผ่าน LCD (Liquid Crystal Display) และส่วนรับข้อมูลจากผู้ใช้งานผ่านคีย์แพดตามมาตรฐาน ITU E.161 เพื่อรองรับข้อมูลอักขระ สำหรับการกำหนด SIP URL ของ เครื่องลูกข่ายปลายทาง และควบคุมการสร้างและยกเลิกเซสชัน แสดงในรูปที่ 3.3



รูปที่ 3.3 ส่วนติดต่อกับผู้ใช้งาน

3.3.1 ส่วนแสดงผลผ่าน LCD

ในส่วนแสดงผลได้ใช้ LCD ชนิด 16 อักขระ 2 บรรทัด มีขาเชื่อมต่อข้อมูล 8 ขา, ขาควบคุม 3 ขา และขาสำหรับไฟเลี้ยง 3 ขา ซึ่งเป็นมาตรฐานของ LCD ทั่วไป ดังแสดงในรูป ที่ 3.4



รูปที่ 3.4 ขาเชื่อมต่อตามมาตรฐานของ LCD

ขา Vdd และ Vss ใช้ต่อกับไฟเลี้ยง +5V และกราวด์ ส่วน Vo เป็นขาปรับแรงดันไฟเพื่อควบคุมความสว่างของหน้าจอ LCD

ขา RS (Register Select) เป็นขาอินพุตใช้สำหรับเลือกชนิดของข้อมูลที่ติดต่อสื่อสารว่าเป็นข้อมูลคำสั่ง หรือข้อมูลอักขระ โดยถ้า RS = 0 หมายความว่าข้อมูลที่ติดต่อสื่อสารนั้นเป็นข้อมูลคำสั่ง และถ้า RS = 1 หมายความว่าข้อมูลที่ติดต่อสื่อสารนั้นเป็นข้อมูลอักขระที่ต้องการให้แสดงผลออกทางหน้าจอ

ขา R/W (Read/Write) เป็นขาอินพุตใช้สำหรับเลือกการอ่าน หรือเขียนข้อมูล โดยถ้า R/W = 1 จะเป็นการอ่านข้อมูลจาก LCD และถ้า R/W = 0 จะเป็นการเขียนข้อมูลลงใน LCD

ขา E (Enable) เป็นขาอีนาเบิลให้ LCD ทำงาน โดยจะทำงานเมื่อได้รับพัลส์ความกว้างอย่างน้อย 450 nsec เข้ามา

ขา D0-D7 เป็นขารับส่งข้อมูลขนาด 8 บิตระหว่าง LCD กับอุปกรณ์อื่น

โครงสร้างของหน่วยความจำภายในโมดูล LCD ประกอบด้วย

รีจิสเตอร์คำสั่ง (Instruction Register: IR) เป็นรีจิสเตอร์ที่ใช้รับ และส่งข้อมูลคำสั่งจากอุปกรณ์ภายนอก เพื่อควบคุมการทำงาน และการแสดงผลของ LCD

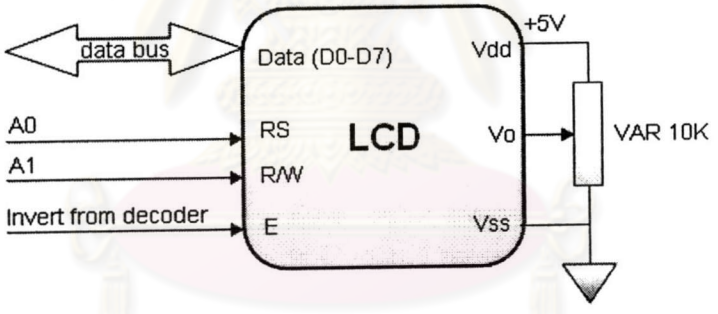
รีจิสเตอร์ข้อมูล (Data Register: DR) เป็นรีจิสเตอร์ที่ใช้รับ และส่งข้อมูลอักขระเพื่อนำไปแสดงผลออกหน้าจอ LCD หรืออ่านค่าอักขระที่ได้แสดงอยู่บนหน้าจอ LCD

แรมเก็บข้อมูลแสดงผล (Display Data RAM: DDRAM) เป็นหน่วยความจำเก็บค่าข้อมูลที่แสดงผลอยู่บนหน้าจอ ในแต่ละตำแหน่ง

รอมเก็บอักขระ (Character Generator ROM: CGROM) เป็นหน่วยความจำที่ใช้เก็บข้อมูลลักษณะของอักขระ และสัญลักษณ์ต่าง ๆ มีขนาด 7200 บิต โดยจะถูกอ่านด้วยค่าของ DDRAM

แรมเก็บอักขระ (Character Generator RAM: CGRAM) เป็นหน่วยความจำที่ใช้เก็บอักขระที่มีการสร้างเพิ่มเติมขึ้นใหม่ ในกรณีที่อักขระใน CGROM ไม่เพียงพอ มีขนาด 512 บิต

ในการทำ Memory map เพื่อต่อใช้งาน LCD นั้นได้ต่อขา RS และ R/W เข้ากับสายสัญญาณกำหนดตำแหน่งหน่วยความจำ A0 และ A1 ตามลำดับ ดังแสดงในรูปที่ 3.5ซึ่งทำให้ไมโครคอนโทรลเลอร์มอง LCD เป็นเสมือนหน่วยความจำขนาด 8 บิตจำนวน 4 ไบต์ดังแสดงในตารางที่ 3.2



รูปที่ 3.5 การทำ Memory map เพื่อใช้งาน LCD

ตารางที่ 3.2 การทำงานของ LCD เมื่อได้ทำ Memory map

ตำแหน่ง	R/W (A1)	RS (A0)	การทำงาน
0x8000	0	0	เขียนข้อมูลคำสั่งลงในรีจิสเตอร์ควบคุม LCD
0x8001	0	1	เขียนข้อมูลอักขระให้แสดงผลบน LCD
0x8002	1	0	อ่านค่ารีจิสเตอร์ควบคุมภายใน LCD
0x8003	1	1	อ่านข้อมูลอักขระในรีจิสเตอร์แสดงผลของ LCD

ตารางที่ 3.3 ข้อมูลคำสั่งของ LCD

คำสั่ง	Hex	D7	D6	D5	D4	D3	D2	D1	D0
เคลียร์หน้าจอแสดงผล	01	0	0	0	0	0	0	0	1
ให้เคอร์เซอร์มาอยู่ที่ตำแหน่งเริ่มต้น	02-03	0	0	0	0	0	0	1	x
เลือกโมดการป้อนอักขระ	04-07	0	0	0	0	0	1	1/D	S
ควบคุมการแสดงผล	08-0F	0	0	0	0	1	D	U	B
ควบคุมการเลื่อนเคอร์เซอร์และอักขระ	10-1F	0	0	0	1	D/C	R/L	x	x
ตั้งค่าการทำงานของ LCD	20-3F	0	0	1	8/4	2/1	10/7	x	x
x = don't care									

คำสั่งเคลียร์หน้าจอแสดงผล (Clear display) มีข้อมูลคำสั่งเป็น 01H เมื่อ LCD ได้รับคำสั่งนี้ส่วนควบคุมจะเขียนข้อมูลช่องว่างลงไปใน DDRAM ทุกตำแหน่ง และกำหนดตำแหน่งชี้ข้อมูลของ DDRAM ให้เป็น 0 ซึ่งจะเป็นการลบหน้าจอทั้งหมด และเลื่อนเคอร์เซอร์กลับมาอยู่ที่ตำแหน่งแรกสุดของหน้าจอแสดงผล

คำสั่งให้เคอร์เซอร์มาอยู่ที่ตำแหน่งเริ่มต้น (Return home) มีข้อมูลคำสั่งเป็น 02H หรือ 03H เมื่อ LCD ได้รับคำสั่งนี้ส่วนควบคุมจะกำหนดตำแหน่งชี้ข้อมูลของ DDRAM เป็น 0 ซึ่งจะทำให้เคอร์เซอร์กลับมาอยู่ที่ตำแหน่งแรกสุดของหน้าจอแสดงผล โดยไม่เปลี่ยนแปลงข้อมูลใน DDRAM

คำสั่งตั้งโมดการป้อนอักขระ (Entry mode set) มีข้อมูลคำสั่งเป็น 04H-07H ใช้กำหนดการกระทำต่อตำแหน่งชี้ข้อมูลของ DDRAM เมื่อมีการป้อนอักขระเพื่อแสดงผลบน LCDโดยบิต S เป็นบิตที่ใช้ในการกำหนดลักษณะของการแสดงผล เมื่อมีการป้อนข้อมูล โดยถ้า S = 1 เมื่อมีการป้อนข้อมูลใหม่ลงบน LCD เคอร์เซอร์จะอยู่กับที่ และดันข้อมูลอักขระเก่าไปด้านข้าง แต่ถ้า S = 0 เมื่อมีการป้อนข้อมูลใหม่ลงบน LCD อักขระเก่าจะอยู่กับที่ แต่เคอร์เซอร์จะเลื่อนไปด้านข้าง ส่วนบิต I/D นั้นเป็นบิตที่ใช้ในการกำหนดทิศทางในการเลื่อน เมื่อมีการป้อนข้อมูลใหม่ลง LCD โดยถ้า I/D = 0 เมื่อมีการป้อนอักขระใหม่ลงบน LCD ตำแหน่งชี้ข้อมูล DDRAM จะเพิ่มขึ้น แต่ถ้า I/D = 1 ตำแหน่งชี้ข้อมูล DDRAM จะลดลง คำสั่งตั้งโมดการป้อนอักขระที่ใช้บ่อยคือ 06H คือเมื่อมีการป้อนข้อมูลอักขระใหม่ให้กับ LCD อักขระเก่าจะอยู่กับที่ และเคอร์เซอร์เลื่อนไปทางขวามือ

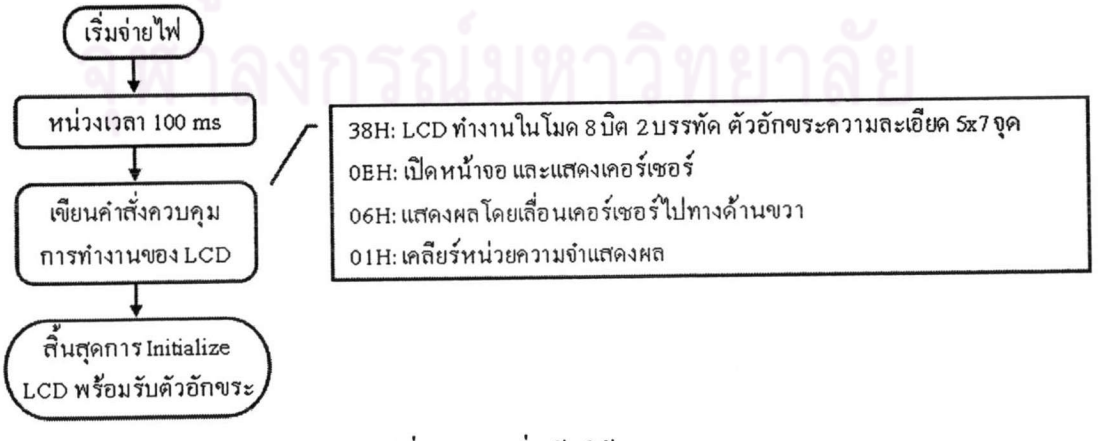
คำสั่งควบคุมการแสดงผล (Display ON/OFF Control) มีข้อมูลคำสั่งเป็น 08H-0FH ใช้ในการควบคุมหน้าจอแสดงผล และเคอร์เซอร์ โดยบิต D ใช้ควบคุมการเปิดปิดหน้าจอแสดงผล โดยถ้าบิต D = 1 จะเป็นการสั่งให้เปิดหน้าจอ และถ้า D = 0 จะเป็นการสั่งปิดหน้าจอ บิต C ใช้ควบคุมการเปิดปิดเคอร์เซอร์ โดยถ้า C = 1 จะเป็นการเปิดเคอร์เซอร์ และถ้า C = 0 จะเป็นการปิดเคอร์เซอร์หรือไม่แสดงเคอร์เซอร์ ส่วนบิต B ใช้ควบคุมการกระพริบของเคอร์เซอร์ โดยถ้าบิต B = 1 เคอร์เซอร์จะกระพริบ

คำสั่งควบคุมการเลื่อนเคอร์เซอร์และอักขระ (Display/Cursor Shift) การควบคุมนั้นขึ้นอยู่กับการกำหนดบิต S/C และ R/L โดยบิต S/C เป็นเป็นบิตเลือกอักขระ และเคอร์เซอร์ และบิต R/L เป็นบิตเลือกทิศทางในการเลื่อน ถ้าบิต S/C = 0 จะเป็นการกระทำกับเคอร์เซอร์ และถ้า S/C = 1 จะเป็นการกระทำกับอักขระ ส่วนบิต R/L = 0 จะเป็นการเลื่อนไปทางซ้าย และถ้าบิต R/L = 1 จะเป็นการเลื่อนไปทางขวา

คำสั่งตั้งค่าการทำงานของของ LCD (Function Set) บิต DL ใช้กำหนดจำนวนบิตที่ใช้ติดต่อส่งผ่านข้อมูล โดยถ้า DL = 0 จะเป็นการติดต่อแบบ 4 บิต และถ้า DL = 1 จะเป็นการติดต่อแบบ 8 บิต บิต N ใช้กำหนดจำนวนบรรทัดของการแสดงผลโดยถ้า N = 0 จะเป็นการตั้งค่าสำหรับการแสดงผลแบบ 1 บรรทัด และถ้า DL = 1 จะเป็นการตั้งค่าสำหรับการแสดงผลแบบ 2 บรรทัด ส่วนบิต F นั้นใช้เลือกความละเอียดของอักขระในการแสดงผล โดยถ้า F = 0 เป็นการตั้งค่าสำหรับการแสดงผลความละเอียดแบบ 5x7 จุดต่ออักขระและถ้า F = 1 จะเป็นการตั้งค่าสำหรับการแสดงผลแบบ 5x10 จุดต่ออักขระ

การเขียนข้อมูลคำสั่ง หรือข้อมูลอักขระลง LCD ทุกครั้ง จำเป็นต้องตรวจสอบสถานะของ LCD ก่อนว่า LCD นั้นพร้อมจะรับข้อมูลหรือไม่ โดยการอ่านบิต BUSY ซึ่งเป็นบิต D7 ของรีจิสเตอร์ควบคุมภายใน LCD โดยถ้าบิต BUSY นี้มีลอจิกเป็น “1” แสดงว่า LCD กำลังประมวลผลภายในอยู่ ไม่สามารถรับข้อมูลได้ในขณะนั้น แต่ถ้าบิต BUSY นี้มีลอจิกเป็น “0” แสดงว่า LCD อยู่ในสถานะว่าง และสามารถรับข้อมูลจากอุปกรณ์ภายนอกได้

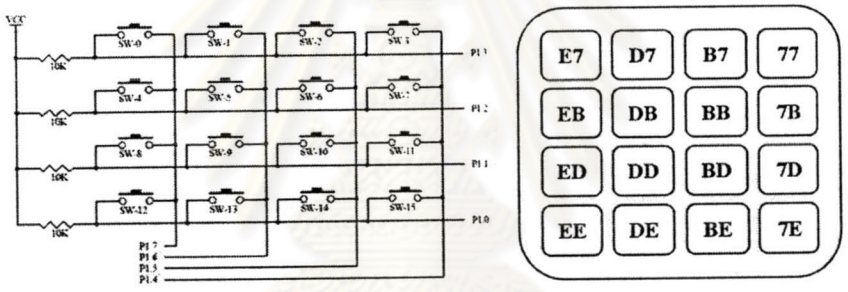
ในการใช้งาน LCD เริ่มแรกต้องเขียนข้อมูลคำสั่งควบคุมดังแสดงในตารางที่ 3.3 เพื่อกำหนดค่าเริ่มต้นต่าง ๆ ที่จำเป็นต่อการทำงานให้กับ LCD ก่อน ถึงจะสามารถเขียนข้อมูลอักขระให้ LCD แสดงผลได้ และเนื่องจากการทำงานของ LCD นั้นช้ากว่าไมโครคอนโทรลเลอร์มาก ดังนั้นในการใช้งานจริงเมื่อเริ่มจ่ายไฟให้กับไมโครคอนโทรลเลอร์ และ LCD แล้ว ต้องหน่วงเวลาไมโครคอนโทรลเลอร์ประมาณ 15 ms เพื่อรอให้ LCD อยู่ในสถานะพร้อมทำงาน ถึงจะสามารถเขียนข้อมูลคำสั่งเพื่อกำหนดค่าเริ่มต้นต่างๆให้กับ LCD ได้ ดังแสดงในรูปที่ 3.6



รูปที่ 3.6 การเริ่มต้นใช้งาน LCD

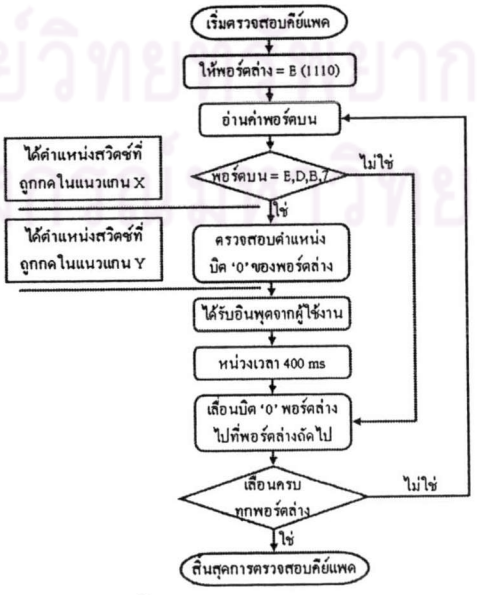
3.3.2 ส่วนรับข้อมูลจากผู้ใช้งานผ่านคีย์แพด

ในส่วนรับข้อมูลจากผู้ใช้งาน ได้ใช้สวิตช์จำนวน 16 ตัวมาต่อเรียงกันเป็นคีย์แพดขนาด 4x4 แบบเมตริกซ์และต่อเข้ากับพอร์ต 1 ของไมโครคอนโทรลเลอร์ดังแสดงในรูปที่ 3.7(a) การต่อแบบนี้สามารถใช้พอร์ตเพียง 8 บิต ซึ่งน้อยกว่าการต่อสวิตช์เข้ากับพอร์ตโดยตรงที่ต้องใช้พอร์ตถึง 16 บิต แต่ไมโครคอนโทรลเลอร์ต้องทำงานซับซ้อนกว่าดังแสดงในรูปที่ 3.8 พอร์ตบน (P1.4-P1.7) ทำหน้าที่เป็นอินพุต และพอร์ตล่าง (P1.0-P1.3) ทำหน้าที่เป็นเอาต์พุตโดยมีตัวต้านทานขนาด 10 กิโลโอห์มต่อพูลอัปกับแรงดันไฟฟ้า +5โวลต์ไว้ ในการตรวจสอบการกดคีย์ไมโครคอนโทรลเลอร์จะไล่ส่งลอจิก “0” ออกไปทางพอร์ตล่างครั้งละ 1 บิตอย่างรวดเร็ว และวนซ้ำเรื่อย ๆ ในขณะเดียวกันก็จะอ่านค่าจากพอร์ตบนไปด้วยถ้าค่าที่อ่านได้จากพอร์ตบนเป็นลอจิก ‘1’ หมดแสดงว่าในแถวที่พอร์ตล่างส่งลอจิก “0” อยู่นั้นไม่มีการกดคีย์ แต่ถ้าค่าที่อ่านมาไม่เป็นลอจิก ‘1’ หมดแสดงว่ามีการกดคีย์ที่แถวนั้นเกิดขึ้น ค่าที่อ่านได้จากพอร์ตบนนี้จะเป็นตัวระบุตำแหน่งของสวิตช์ที่ถูกกดในแนวแกน X และตำแหน่งของลอจิก “0” ที่พอร์ตล่างจะเป็นตัวระบุตำแหน่งของสวิตช์ที่ถูกกดในแนวแกน Y โดยมีความสัมพันธ์กันดังแสดงในรูปที่ 3.7 (b)



(a) (b)

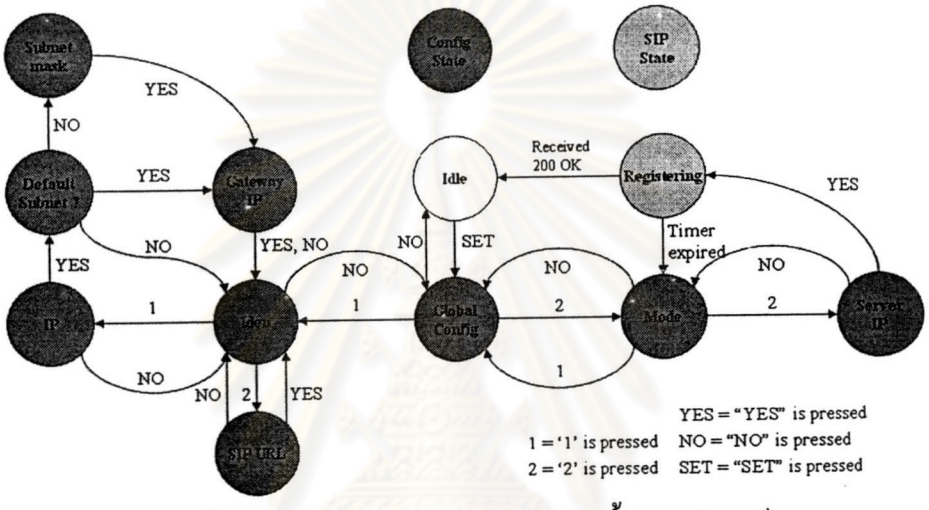
รูปที่ 3.7 (a) การต่อคีย์แพด 4x4 เพื่อรับอินพุตจากผู้ใช้งาน
(b) รหัสของคีย์แพดแบบเมตริกซ์



รูปที่ 3.8 ขั้นตอนการตรวจสอบคีย์แพด

คีย์แพดเป็นส่วนสำคัญที่ใช้ในการปรับตั้งค่าพารามิเตอร์ที่สำคัญต่อการทำงานของเครื่องโทรศัพท์ ได้แก่ไอพีแอดเดรส, หมายเลขซับเน็ต, SIP URL ของตัวเครื่องโทรศัพท์, ไอพีแอดเดรสของเกตเวย์, และไอพีแอดเดรสของเครื่องแม่ข่าย รวมไปถึงการปรับตั้งโมดการทำงานของตัวเครื่องโทรศัพท์ด้วย

ในการปรับตั้งพารามิเตอร์ต่าง ๆ บนเครื่องโทรศัพท์นั้น ทำได้โดยกดปุ่ม Set ในขณะที่เครื่องโทรศัพท์อยู่ในสภาวะ Idle เมื่อกดปุ่ม Set แล้ว LCD จะแสดงสถานะแจ้งให้ทราบว่าเครื่องโทรศัพท์ได้เข้าสู่โมดปรับตั้งพารามิเตอร์ และสามารถปรับตั้งพารามิเตอร์ต่าง ๆ ได้ดังแสดงในรูปที่ 3.9



รูปที่ 3.9 สถานะการทำงานในโมดปรับตั้งพารามิเตอร์

จะเห็นได้ว่าการปรับตั้งพารามิเตอร์นั้นสามารถแบ่งออกได้เป็น 2 ส่วนใหญ่ ๆ คือ พารามิเตอร์ที่ใช้ระบุเครื่องโทรศัพท์ และพารามิเตอร์ที่ใช้บ่งบอกโมดการทำงานของเครื่องโทรศัพท์ พารามิเตอร์ที่ใช้ระบุเครื่องโทรศัพท์ได้แก่ SIP URL, ไอพีแอดเดรสของเครื่องโทรศัพท์, ซับเน็ต, และไอพีแอดเดรสของเกตเวย์ ส่วนพารามิเตอร์ที่ใช้บ่งบอกโมดการทำงานของเครื่องโทรศัพท์ได้แก่ การเลือกทำงานร่วมกับเครื่องแม่ข่าย หรือทำงานโดยไม่อาศัยเครื่องแม่ข่าย ซึ่งในการทำงานร่วมกับเครื่องแม่ข่ายนั้น จำเป็นต้องใช้ไอพีแอดเดรสของเครื่องแม่ข่ายด้วย เมื่อกำหนดค่าไอพีแอดเดรสของเครื่องแม่ข่ายเสร็จแล้ว เครื่องโทรศัพท์อินเทอร์เนตจะลงทะเบียนกับเครื่องแม่ข่ายทันที ในกรณีที่ไม่สามารถลงทะเบียนได้ เครื่องโทรศัพท์จะกลับเข้าสู่โมดปรับตั้งการทำงานของเครื่องโทรศัพท์อีกครั้ง

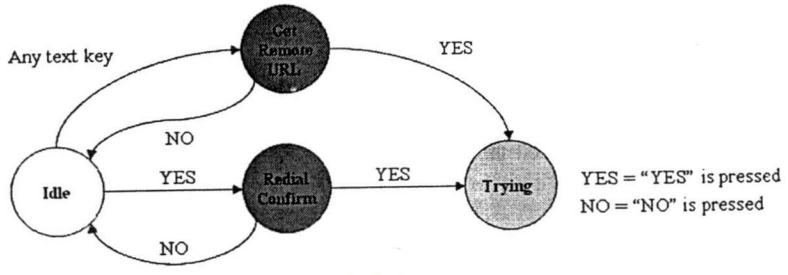
ในการป้อนค่าไอพีแอดเดรสให้ไมโครคอนโทรลเลอร์ผ่านคีย์แพดนั้น สวิตช์แต่ละตัวใช้ระบุแทนตัวเลขโดยตรงเหมือนคีย์แพดของโทรศัพท์ทั่วไป และใช้สวิตช์ในตำแหน่ง '*' แทนเครื่องหมาย '.' เพื่อแบ่งแยกค่าไอพีแอดเดรสในแต่ละไบต์ และสวิตช์ในตำแหน่ง '#' แทนปุ่มลบ (Delete) เมื่อต้องการลบสิ่งที่ได้พิมพ์ผ่านคีย์แพดไป เนื่องจากการป้อนข้อมูลไอพีแอดเดรสผ่านคีย์แพดนั้น ไมโครคอนโทรลเลอร์จะได้รับข้อมูลจากผู้ใช้ครั้งละดิจิต ดังนั้นในขณะพิมพ์ไอพี

แอดเดรส ไมโครคอนโทรลเลอร์จะเก็บข้อมูลที่พิมพ์ในรูปแบบของตัวอักขระครั้งละตัวไว้ในบัฟเฟอร์ชั่วคราวก่อน และในขณะที่ป้อนคีย์แต่ละครั้ง ไมโครคอนโทรลเลอร์จะตรวจสอบรูปแบบของอักขระให้เป็นไปตามรูปแบบของหมายเลขไอพีแอดเดรสที่ถูกต้อง จนเมื่อป้อนไอพีแอดเดรสครบ และกดปุ่ม "Yes" ไมโครคอนโทรลเลอร์จึงจะนำข้อมูลในบัฟเฟอร์ชั่วคราวนั้นไปแปลงเป็นค่าไอพีแอดเดรสในแต่ละไบต์ และเขียนค่านั้นลงหน่วยความจำข้อมูล RAM และ EEPROM ต่อไป ในกรณีที่ไอพีแอดเดรสที่ปรับตั้งเป็นของเกตเวย์ ไมโครคอนโทรลเลอร์จะตรวจสอบไอพีแอดเดรสที่ได้จากผู้ใช้งาน ว่าอยู่ในโครงข่ายพื้นที่ท้องถิ่นเดียวกับตัวเครื่องโทรศัพท์หรือไม่ โดยการนำไอพีแอดเดรสนั้นมาแอนด์ (AND) กับซับเนตมาสค์ ถ้าเนตเวิร์กแอดเดรสที่ได้ตรงกับ เนตเวิร์กแอดเดรสของเครื่องโทรศัพท์แสดงว่าอยู่ในโครงข่ายพื้นที่ท้องถิ่นเดียวกัน แต่ถ้าเนตเวิร์กแอดเดรสที่ได้ไม่ตรงกันแสดงว่าอยู่คนละโครงข่าย ถ้าไม่ได้อยู่ในโครงข่ายเดียวกันเครื่องโทรศัพท์จะกลับเข้าสู่โมดการตั้งค่าไอพีแอดเดรสของเกตเวย์ใหม่อีกครั้ง

ในการพัฒนาส่วนควบคุมเพื่อรับการป้อนค่า SIP URL ซึ่งเป็นอักขระ ผ่านคีย์แพด 4x4 นั้น ได้พัฒนาตามมาตรฐาน ITU E1.161 ซึ่งสวิตช์แต่ละตัวจะต้องรองรับอักขระหลายตัว มีขั้นตอนการทำงานดังนี้ เมื่อผู้ใช้งานกดสวิตช์เพื่อป้อนอักขระ ไมโครคอนโทรลเลอร์จะรับอักขระแรกที่สวิตช์ตำแหน่งนั้นรองรับเข้ามาเก็บไว้ในหน่วยความจำชั่วคราว และแสดงออกทาง LCD พร้อมกับเปิดไทเมอร์เพื่อจับเวลาที่ใช้ในการเลื่อนตำแหน่งเคอร์เซอร์ ถ้าอักขระที่แสดงไม่ใช่อักขระที่ผู้ใช้งานต้องการ ผู้ใช้งานสามารถเปลี่ยนอักขระได้โดยการกดสวิตช์ตำแหน่งเดิมซ้ำ จนกว่าจะถึงอักขระที่ผู้ใช้งานต้องการ โดยทุกครั้งที่ผู้ใช้งานกดสวิตช์ไมโครคอนโทรลเลอร์จะรีเซตไทเมอร์ และเปิดไทเมอร์เพื่อจับเวลาใหม่ จนเมื่อสวิตช์ไม่มีการกดเป็นเวลานานประมาณ 1 วินาที ไมโครคอนโทรลเลอร์จึงจะเขียนอักขระนั้นลงในบัฟเฟอร์ชั่วคราว พร้อมสั่งให้เคอร์เซอร์ LCD เลื่อนไปเพื่อรอรับอักขระถัดไป เมื่อผู้ใช้งานป้อน SIP URL จนเสร็จ และกดปุ่ม "Yes" ไมโครคอนโทรลเลอร์จะนำข้อมูลในบัฟเฟอร์ชั่วคราวนั้นเขียนลงในหน่วยความจำข้อมูล RAM และ EEPROM ต่อไป

นอกจากคีย์แพดจะใช้สำหรับการปรับตั้งพารามิเตอร์ต่าง ๆ แล้ว ยังใช้ในการสร้าง และยกเลิกเซสชันกับเครื่องลูกข่ายอื่นด้วยดังแสดงในรูปที่ 3.10 เมื่อผู้ใช้งานต้องการสร้างเซสชันกับเครื่องลูกข่ายอื่นสามารถทำได้โดยการกดสวิตช์สำหรับป้อนอักขระในขณะที่เครื่องอยู่ในสถานะ Idle เมื่อสวิตช์ถูกกดแล้ว เครื่องโทรศัพท์จะเปลี่ยนสถานะพร้อมกับแสดงผลผ่าน LCD เพื่อรอรับ SIP URL ปลายทางจากผู้ใช้งาน เมื่อผู้ใช้งานป้อน SIP URL ปลายทางเสร็จและกดปุ่ม "YES" เครื่องโทรศัพท์จะเก็บข้อมูล SIP URL นั้นไว้ในหน่วยความจำข้อมูล RAM และส่งข้อมูลเพื่อขอสร้างเซสชันกับปลายทางต่อไป ในกรณีที่ได้สร้างเซสชันและยกเลิกเซสชันจนสำเร็จแล้ว หรือในการสร้างเซสชันไม่สำเร็จ และผู้ใช้ต้องการสร้างเซสชันใหม่ไปยังปลายทางเดิม (Redial) สามารถทำได้โดยกดสวิตช์ "YES" ในขณะที่เครื่องอยู่ในสถานะ Idle เครื่องโทรศัพท์จะเปลี่ยนสถานะและ

แสดงผลผ่าน LCD เพื่อให้ผู้ใช้งานยืนยันการสร้างเซสชันไปยังปลายทางเดิม เมื่อผู้ใช้งานกดปุ่ม "YES" อีกครั้ง เครื่องโทรศัพท์จะส่งข้อมูลเพื่อขอสร้างเซสชันกับปลายทางเดิมต่อไป



รูปที่ 3.10 การใช้คีย์แพดในการสร้างเซสชัน

3.4 ส่วนติดต่อรับส่งข้อมูลผ่านชิพอีเทอร์เนต RTL8019as

3.4.1 โครงสร้าง และคุณลักษณะของชิพอีเทอร์เนต RTL8019as

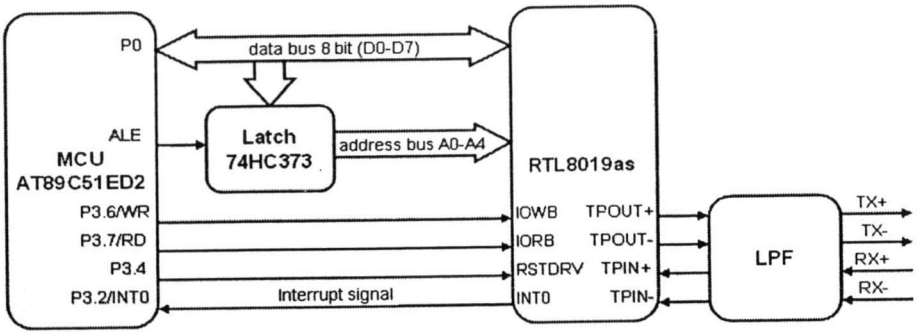
ชิพอีเทอร์เนต RTL8019as เป็นชิพที่รองรับการรับส่งข้อมูลตามมาตรฐาน 10BaseT มีรีจิสเตอร์ควบคุมการทำงานตามมาตรฐาน NE2000 ซึ่งเป็นมาตรฐานสำหรับอุตสาหกรรมการออกแบบชิพอีเทอร์เนต ดังแสดงในตารางที่ 3.4 ตัวชิพประกอบไปด้วยรีจิสเตอร์ควบคุม 4 หน้า, หน่วยความจำ SRAM สำหรับบัฟเฟอร์ข้อมูลขนาด 16K ไบต์ ฟังก์ชันการคำนวณ CRC (Cyclic Redundancy Check), วงจรเข้ารหัสข้อมูลแบบแมนเชสเตอร์ (Manchester Coding), และวงจรรับส่งข้อมูลผ่านอีเทอร์เนต

ตารางที่ 3.4 โครงสร้างรีจิสเตอร์ควบคุมตามมาตรฐาน NE2000 ภายในชิพ RTL8019as

No (Hex)	Page 1		Page 2	Page 3	Page 4	
	[R]	[W]	[R/W]	[R]	[R]	[W]
00	CR	CR	CR	CR	CR	CR
01	CLDA0	PSTART	PAR0	PSTART	*9346CR*	*9346CR*
02	CLDA1	PSTOP	PAR1	PSTOP	*BPAGE*	*BPAGE1*
03	BNRY	BNRY	PAR2	-	*CONFIG0*	-
04	TSR	TPSR	PAR3	TPSR	*CONFIG1*	*CONFIG1*
05	NCR	TBSR0	PAR4	-	*CONFIG2*	*CONFIG2*
06	FIFO	TBCR1	PAR5	-	*CONFIG3*	*CONFIG3*
07	ISR	ISR	CURR	-	-	*TEST*
08	CRDA0	RSAR0	MAR0	-	*CSNSAV*	-
09	CRDA1	RSAR1	MAR1	-	-	*HLTCLK*
0A	*8019ID0*	RBCR0	MAR2	-	-	-
0B	*8019ID1*	RBCR1	MAR3	-	*INTR*	-
0C	RSR	RCR	MAR4	RCR	-	*FMWP*
0D	CNTR0	TCR	MAR5	TCR	*CONFIG4*	-
0E	CNTR1	DCR	MAR6	DCR	-	-
0F	CNTR2	IMR	MAR7	IMR	-	-
10-17	Remote DMA Port					
18-1F	Reset Port					

* รีจิสเตอร์ที่เป็นตัวเอียงทึบ เป็นรีจิสเตอร์ควบคุมเพิ่มเติมของชิพ RTL8019as นอกเหนือจากมาตรฐาน NE2000

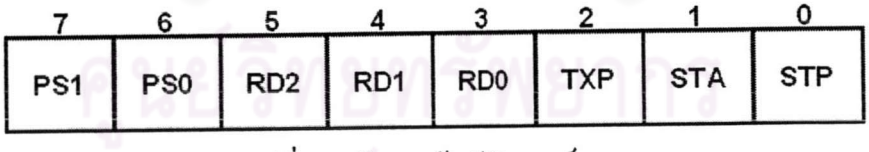
ในการพัฒนาได้ต่อไมโครคอนโทรลเลอร์กับชิพ RTL8019as แบบ Memory map โดยให้ชิพ RTL8019as เป็นหน่วยความจำอยู่ในตำแหน่ง 0x4000 ถึง 0x4031 ซึ่งในการต่อนั้นนอกจากจะใช้สายสัญญาณข้อมูล D0-D7, สายสัญญาณกำหนดตำแหน่งหน่วยความจำ A0-A4, และสายสัญญาณควบคุม WR, RD แล้ว ยังต่อพอร์ต 3.4 เข้ากับขา RSTDRV เพื่อควบคุมการรีเซตการทำงานของชิพ RTL8019as และต่อพอร์ต 3.2 เข้ากับขา INT0 เพื่อรับสัญญาณอินเตอร์รัปต์จากชิพ RTL8019as ด้วยดังแสดงในรูปที่ 3.11



รูปที่ 3.11 การต่อไมโครคอนโทรลเลอร์ กับชิพอีเทอร์เนต RTL8019as

ในการควบคุมชิพ RTL8019as เพื่อให้รับส่งข้อมูลนั้นสามารถทำได้โดยการอ่าน และเขียนข้อมูลลงในรีจิสเตอร์ควบคุมภายในชิพ RTL8019as ซึ่งมี 4 หน้า และมีรีจิสเตอร์ที่สำคัญดังนี้

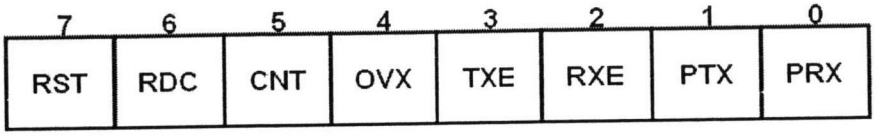
รีจิสเตอร์ CR (Command Register) อยู่ที่ตำแหน่ง 0x01 ของทุกหน้า สามารถกระทำการอ่าน และเขียนได้เป็นรีจิสเตอร์ที่ใช้ในการเลือกหน้ารีจิสเตอร์ และเป็นรีจิสเตอร์ที่ใช้ควบคุมพอร์ต DMA ซึ่งเป็นพอร์ตที่ใช้ในการรับส่งข้อมูลกับอุปกรณ์ภายนอก มีโครงสร้างดังแสดงในรูปที่ 3.12 บิต PS1, PS0 เป็นบิตที่ใช้ในการเลือกหน้ารีจิสเตอร์ โดยถ้า PS1, PS0 = “00” เป็นการใช้งานหน้า 0, “01” เป็นการใช้งานหน้า 1, “10” เป็นการใช้งานหน้า 2 และ “11” เป็นการใช้งานหน้า 3 ตามลำดับ RD2-0 เป็นบิตที่ใช้ในการควบคุมการรับส่งข้อมูลผ่านพอร์ต DMA โดยถ้า RD2-0 = “001” เป็นการควบคุมให้พอร์ต DMA ทำงานในโมดส่งข้อมูล, “010” ทำงานในโมดรับข้อมูล, “011” เป็นการควบคุมให้ชิพอีเทอร์เนตทำงานส่งแพ็กเกตออกไป และ “1**” เป็นตัวแสดงว่าการรับส่งข้อมูลผ่านพอร์ต DMA นั้นได้เสร็จสิ้นแล้ว บิต TXP เป็นบิตควบคุมการส่งข้อมูลออก โดยก่อนส่งต้องปรับให้บิต TXP นี้เป็น ‘1’ ก่อน เมื่อข้อมูลได้ส่งออกไปสำเร็จแล้วบิต TXP นี้จะเป็น ‘0’ โดยอัตโนมัติ ส่วนบิต STA, STP ใช้ควบคุมการเปิดปิดการรับส่งข้อมูลโดยถ้า STA, STP = “10” ชิพจะสามารถรับส่งข้อมูลได้ แต่ถ้า STA, STP = “01” ชิพจะไม่สามารถรับส่งข้อมูลได้



รูปที่ 3.12โครงสร้างรีจิสเตอร์ CR

รีจิสเตอร์ ISR (Interrupt Status Register) อยู่ที่ตำแหน่ง 0x07 ของหน้า 0 สามารถกระทำการอ่าน และเขียนได้ ใช้ในการบ่งบอกสาเหตุการเกิดสัญญาณอินเตอร์รัปต์ภายในชิพอีเทอร์เนต มีโครงสร้างดังแสดงในรูปที่ 3.13 บิต RST จะเป็น ‘1’ เมื่อชิพถูกรีเซต สามารถเคลียร์เองได้ เมื่อมีการเขียนค่ารีจิสเตอร์ CR ให้อยู่ในโมดสามารถรับส่งข้อมูลได้ (STA, STP = “10”) บิต RDC จะเป็น ‘1’ เมื่อมีการรับส่งค่าผ่านพอร์ต DMA สำเร็จ บิต OVW จะเป็น ‘1’ เมื่อบัฟเฟอร์ภาครับล้น บิต TXE เป็น ‘1’ เมื่อส่งข้อมูลแล้วเกิดการชนขึ้น บิต RXE เป็น ‘1’ เมื่อข้อมูลที่ได้รับมามีความผิดพลาด เช่นตรวจสอบ CRC ไม่ผ่าน หรือซิงโครไนซ์ข้อมูลไม่ได้ เป็นต้น PTX เป็น ‘1’ เมื่อชิพ

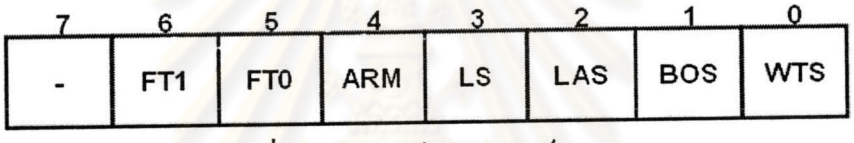
อีเทอร์เนตได้ส่งข้อมูลออกไปโดยไม่มีความผิดพลาดเกิดขึ้น และ PRX เป็น '1' เมื่อชิพอีเทอร์เนตได้รับข้อมูลซึ่งไม่มีความผิดพลาดเข้ามา



รูปที่ 3.13โครงสร้างรีจิสเตอร์ ISR

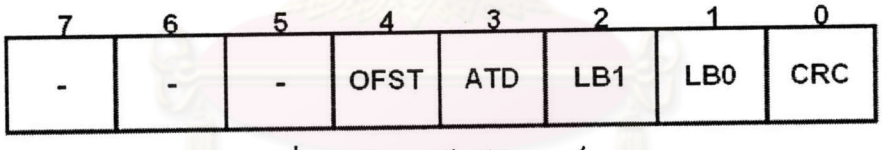
รีจิสเตอร์ IMR (Interrupt Mask Register) อยู่ที่ตำแหน่ง 0xFH ในการเขียนอยู่ในหน้า 0 ส่วนในการอ่านอยู่ในหน้า 2 ใช้ในการเลือกใช้งานอินเตอร์รัปต์ โดยมีตำแหน่งสอดคล้องกับรีจิสเตอร์ ISR ซึ่งในการเลือกใช้งานอินเตอร์รัปต์นั้น ๆ สามารถทำได้โดยการเซตบิตตำแหน่งนั้นให้เป็น '1'

รีจิสเตอร์ DCR (Data Configuration Register) อยู่ที่ตำแหน่ง 0x0E ในการเขียนอยู่ในหน้า 0 ส่วนในการอ่านอยู่ในหน้า 2 ใช้สำหรับระบุโมดการทำงานของชิพอีเทอร์เนต มีโครงสร้างดังแสดงในรูปที่ 3.14



รูปที่ 3.14โครงสร้างรีจิสเตอร์ DCR

รีจิสเตอร์ TCR (Transmit Configuration Register) อยู่ที่ตำแหน่ง 0x0D ในการเขียนอยู่ในหน้า 0 ส่วนการอ่านอยู่ในหน้า 2 มีโครงสร้างดังแสดงในรูปที่ 3.15 ใช้กำหนดโมดการส่งข้อมูล



รูปที่ 3.15 โครงสร้างรีจิสเตอร์ TCR

รีจิสเตอร์ RCR (Receive Configuration Register) อยู่ที่ตำแหน่ง 0x0C ในการเขียนอยู่ในหน้า 0 ส่วนการอ่านอยู่ในหน้า 2 มีโครงสร้างดังแสดงในรูปที่ 3.16 ใช้สำหรับระบุโมดการรับข้อมูล โดยถ้าบิต MON เป็น '1' ชิพอีเทอร์เนตจะทำงานในโมดมอนิเตอร์ โดยจะตรวจเช็คฮาร์ดแวร์แอดเดรสปลายทาง, CRC, การซิงโครไนซ์เฟรม แต่ไม่เก็บเฟรมที่ได้รับมานั้นเข้ามาในบัฟเฟอร์ แต่ถ้าเป็น '0' ชิพอีเทอร์เนตจะเก็บเฟรมนั้นไว้ในบัฟเฟอร์เพื่อรอการเรียกใช้งานต่อไป บิต PRO เป็นบิตที่ใช้ตั้งการเลือกรับข้อมูลโดยถ้าเป็น '1' ชิพอีเทอร์เนตจะรับเฟรมที่เข้ามาหมดโดยไม่สนใจฮาร์ดแวร์แอดเดรสปลายทาง แต่ถ้าเป็น '0' ชิพอีเทอร์เนตจะเลือกรับเฉพาะเฟรมที่มีฮาร์ดแวร์แอดเดรสตรงกับตัวเองเท่านั้น บิต AM เป็นบิตที่ใช้ตั้งการเลือกรับเฟรมแบบมัลติคาสต์ (multicast) โดยถ้า AM เป็น '1' ชิพอีเทอร์เนตจะรับเฟรมที่มีฮาร์ดแวร์แอดเดรสมัลติคาสต์ แต่ถ้าเป็น '0' ชิพอีเทอร์เนตจะไม่รับเฟรมที่มีฮาร์ดแวร์แอดเดรสแบบมัลติคาสต์ บิต AB เป็นบิตที่ใช้ตั้งการเลือกรับข้อมูลแบบแพร่สัญญาณโดยถ้า AB เป็น '1' ชิพอีเทอร์เนตจะรับเฟรมที่มีฮาร์ดแวร์แอดเดรส

แบบมัลติคาส์ท แต่ถ้าเป็น ‘0’ ชิพอีเทอร์เนตจะไม่รับเฟรมที่มีฮาร์ดแวร์แอดเดรสแบบแพร่สัญญาณ บิต AR เป็นบิตที่ใช้ตั้งการเลือกรับตามขนาดของเฟรมโดยถ้า AR เป็น ‘1’ ชิพอีเทอร์เนตจะรับเฟรมที่มีขนาดเล็กกว่า 64 ไบต์ แต่ถ้าเป็น ‘0’ ชิพอีเทอร์เนตจะไม่รับเฟรมที่มีขนาดเล็กกว่า 64 ไบต์ และบิต SEP เป็นบิตที่ใช้ตั้งการเลือกรับแพ็กเกตที่มีความผิดพลาดโดยถ้า SEP เป็น ‘1’ ชิพอีเทอร์เนตจะรับเฟรมที่มีความผิดพลาด แต่ถ้าเป็น ‘0’ ชิพอีเทอร์เนตจะไม่รับเฟรมที่มีความผิดพลาด

7	6	5	4	3	2	1	0
-	-	MON	PRO	AM	AB	AR	SEP

รูปที่ 3.16 โครงสร้างรีจิสเตอร์ RCR

รีจิสเตอร์ PAR0-5 (Physical Address Register) อยู่ที่ตำแหน่งที่ 0x01 ถึง 0x06 ของหน้าที่ 1 สามารถกระทำได้ทั้งการอ่าน และเขียน ใช้เก็บฮาร์ดแวร์แอดเดรสของโฮสต์

รีจิสเตอร์ PSTART (Page Start Register) อยู่ที่ตำแหน่ง 0x01 ในการเขียนอยู่ในหน้า 0 ส่วนในการอ่านอยู่ในหน้า 2 เป็นรีจิสเตอร์ที่ใช้กำหนดตำแหน่งจุดเริ่มต้นของวงบัฟเฟอร์ (buffer ring) ข้อมูลภาครับ

รีจิสเตอร์ PSTOP (Page Stop Register) อยู่ที่ตำแหน่ง 0x02 ในการเขียนอยู่ในหน้า 0 ส่วนในการอ่านอยู่ในหน้า 2 ใช้กำหนดตำแหน่งจุดสิ้นสุดของวงบัฟเฟอร์ข้อมูลภาครับ

รีจิสเตอร์ BNRY (Boundary Register) อยู่ที่ตำแหน่ง 0x03 ของหน้า 0 เป็นรีจิสเตอร์ที่ใช้ป้องกันการซ้อนทับของข้อมูล โดยจะเป็นตัวชี้ตำแหน่งหน้าของบัฟเฟอร์ข้อมูลที่ได้อ่านออกไป

รีจิสเตอร์ TPSR (Transmit Page Start Register) อยู่ที่ตำแหน่ง 0x03 ในการเขียนอยู่ในหน้า 0 ส่วนในการอ่านอยู่ในหน้า 2

รีจิสเตอร์ CURR (Current Page Register) อยู่ที่ตำแหน่ง 0x07 ของหน้าที่ 1 เป็นรีจิสเตอร์ชี้ตำแหน่งหน้าของบัฟเฟอร์ข้อมูลที่ได้รับเข้ามา

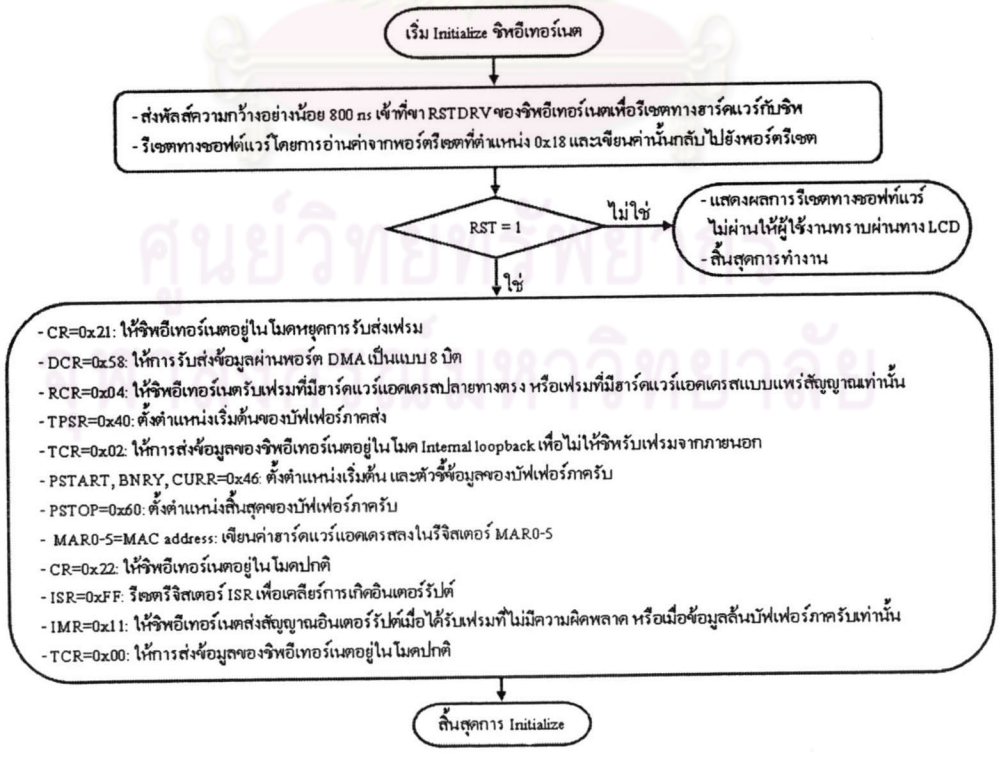
รีจิสเตอร์ TBCR0, 1 (Transmit Byte Count Register) อยู่ที่ตำแหน่ง 0x05 และ 0x06 ของหน้าที่ 0 ใช้กำหนดขนาดข้อมูลที่จะส่งออกไป

รีจิสเตอร์ RBCR0, 1 (Remote Byte Count Register) อยู่ที่ตำแหน่ง 0x0A และ 0x0B ของหน้าที่ 0 ใช้กำหนดขนาดข้อมูลที่จะอ่าน หรือเขียนผ่านพอร์ต DMA

บัฟเฟอร์ภายในชิพ RTL8019as นั้นมีโครงสร้างเป็นแบบวงแบ่งออกเป็นหน้า ๆ แต่ละหน้ามีขนาด 256 ไบต์ ในกรณีที่เฟรมมีขนาดเล็กกว่า 256 ไบต์ หรือเป็นเศษไม่เต็ม ชิพอีเทอร์เนตจะเพิ่มข้อมูลแพดจนเต็ม 256 ไบต์และเก็บข้อมูลเฟรมใหม่ไว้ในหน้าต่อไป ในการใช้งานสามารถเข้าถึงได้โดยการเขียน และอ่านข้อมูลผ่านพอร์ต DMA ที่ตำแหน่ง 10H-17H ของตัวชิพ โดยการทำงานในภาครับนั้นถูกควบคุมด้วยรีจิสเตอร์ PSTART, PSTOP, BNRY, และ CURR ตามลำดับ

3.4.2 การเริ่มต้นใช้งานชิพอีเทอร์เนต RTL8019as

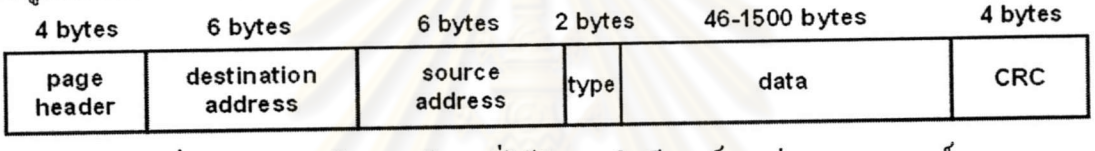
ในการใช้งานชิพอีเทอร์เนตดังแสดงในรูปที่ 3.17 เริ่มต้นต้องรีเซต และตั้งค่ารีจิสเตอร์ควบคุมต่าง ๆ ภายในตัวชิพอีเทอร์เนตเพื่อกำหนดรูปแบบการทำงานเสียก่อน การรีเซตตัวชิพอีเทอร์เนตควรรีเซตทั้งในระดับฮาร์ดแวร์ และซอฟต์แวร์ โดยการรีเซตในระดับฮาร์ดแวร์สามารถทำได้โดยป้อนพัลส์สัญญาณที่มีความกว้างมากกว่า 800 ns เข้าที่ขา RSTDRV ส่วนการรีเซตในระดับซอฟต์แวร์สามารถทำได้โดยการเขียนค่าที่อ่านได้จากพอร์ตรีเซตนั้นกลับไปที่พอร์ตรีเซตของตัวชิพเอง เมื่อชิพเข้าไปอยู่ในสถานะรีเซตทางซอฟต์แวร์ ไมโครคอนโทรลเลอร์จะตรวจสอบบิต RST ภายในรีจิสเตอร์ ISR โดยถ้ามีค่าเป็น '1' แสดงว่าชิพเข้าสู่สถานะรีเซต และพร้อมที่จะรับการตั้งค่าควบคุมต่าง ๆ แล้ว แต่ถ้าบิต RST เป็น '0' แสดงว่าชิพยังไม่ได้เข้าสู่โมดรีเซต และอาจมีความผิดพลาดทางฮาร์ดแวร์เกิดขึ้น เมื่อได้รีเซตทางซอฟต์แวร์แล้ว ในการเซตค่ารีจิสเตอร์ควบคุมการทำงานต่าง ๆ ควรเขียนตั้งค่าบิต STA, STP ภายในรีจิสเตอร์ CR เป็น '0' และ '1' ตามลำดับก่อน เพื่อให้ชิพอยู่ในโมดหยุดการทำงาน จากนั้นจึงค่อยเขียนค่าลงรีจิสเตอร์ควบคุมต่าง ๆ เพื่อกำหนดการทำงานของชิพอีเทอร์เนต ได้แก่รูปแบบการรับส่งข้อมูลผ่านพอร์ต DMA, การเลือกรับเฟรมเข้ามาเก็บในบัฟเฟอร์, ตำแหน่งเริ่มต้น และสิ้นสุดของบัฟเฟอร์ภาครับ และภาคส่ง, การทำ CRC บนตัวชิพ, ฮาร์ดแวร์แอดเดรสของอุปกรณ์, และการควบคุมอินเตอร์รัปต์ ดังแสดงในรูปที่ 3.17 เมื่อตั้งโมดการทำงานเรียบร้อยแล้ว จึงค่อยตั้งค่าบิต STA, STP ให้เป็น '1' และ '0' ตามลำดับ เพื่อให้ชิพเริ่มทำงาน และรีเซตรีจิสเตอร์ ISR เพื่อเคลียร์บิตแสดงอินเตอร์รัปต์จากการรีเซตทางซอฟต์แวร์ให้ชิพพร้อมทำงานต่อไป



รูปที่ 3.17 การ Initialize ชิพอีเทอร์เนต

3.4.2 การรับข้อมูลผ่านชิพอีเทอร์เนต RTL8019as

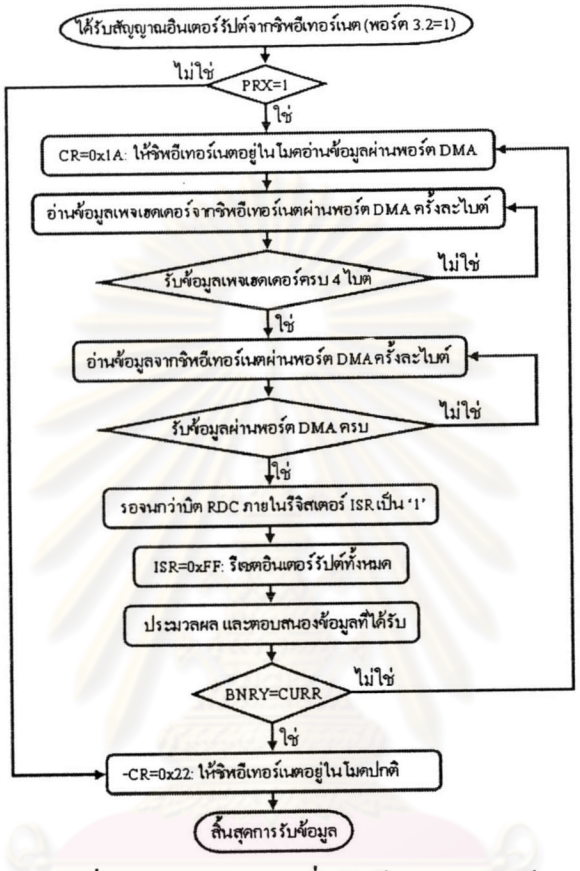
จากการตั้งโมดการทำงานของชิพข้างต้น เมื่อมีข้อมูลเข้ามา ชิพอีเทอร์เนตจะตรวจสอบการซิงโครไนซ์เฟรม, CRC และฮาร์ดแวร์แอดเดรสปลายทาง ภายในเฮดเดอร์อีเทอร์เนต โดยถ้าข้อมูลที่ได้รับมานั้นมีการซิงโครไนซ์เฟรมที่ถูกต้อง, ไม่มีข้อมูลผิดพลาด, และมีฮาร์ดแวร์แอดเดรสปลายทางตรงกับค่าที่เก็บในรีจิสเตอร์ PAR0-PAR5 หรือมีฮาร์ดแวร์แอดเดรสเป็นแอดเดรสแพร่สัญญาณ ชิพอีเทอร์เนตจะเก็บข้อมูลที่ได้รับมาไว้ในบัฟเฟอร์ พร้อมกับส่งสัญญาณอินเตอร์รัปต์ไปหาไมโครคอนโทรลเลอร์ผ่านทางพอร์ต INT0 และปรับเปลี่ยนบิต PRX ภายในรีจิสเตอร์ ISR ให้เป็น '1' เมื่อไมโครคอนโทรลเลอร์ได้รับสัญญาณอินเตอร์รัปต์จากชิพอีเทอร์เนต ไมโครคอนโทรลเลอร์จะอ่านค่ารีจิสเตอร์ ISR เพื่อตรวจสอบชนิดของอินเตอร์รัปต์ที่เกิดขึ้น ถ้าตรวจพบว่าบิต PRX เป็น '1' ไมโครคอนโทรลเลอร์จะเซตบิต RD2, RD1, RD0 ภายในรีจิสเตอร์ CR ให้เป็น "001" ตามลำดับ เพื่อให้สามารถรับข้อมูลจากพอร์ต DMA ของชิพอีเทอร์เนตได้ โดยข้อมูลที่รับได้จากชิพอีเทอร์เนตจะมีโครงสร้างดังแสดงในรูปที่ 3.18



รูปที่ 3.18 โครงสร้างของข้อมูลที่ได้รับจากชิพอีเทอร์เนตผ่าน DMA พอร์ต

จะเห็นได้ว่าในส่วนต้นของโครงสร้างข้อมูลที่ได้รับจากชิพอีเทอร์เนตผ่าน DMA พอร์ตนั้นไม่เหมือนกับโครงสร้างเฟรมอีเทอร์เนตมาตรฐานทั่วไป ทั้งนี้เนื่องจากข้อมูลส่วน preamble นั้นใช้ในการซิงโครไนซ์เฟรมเท่านั้น ชิพอีเทอร์เนตจึงไม่นำส่วน preamble นี้เข้ามาเก็บในบัฟเฟอร์ภาครับด้วย แต่จะเพิ่มข้อมูล page header ซึ่งเป็นข้อมูลขนาด 4 ไบต์ใช้บ่งบอกพารามิเตอร์ภายในบัฟเฟอร์ภาครับไว้เพื่อให้ไมโครคอนโทรลเลอร์สามารถควบคุม และรับข้อมูลได้อย่างถูกต้องแทนข้อมูลภายใน page header ไบต์แรกบ่งบอกถึงสถานะของเฟรมข้อมูลที่รับมาได้ ไบต์ที่สองบ่งบอกตำแหน่งหน้าบัฟเฟอร์ภาครับถัดไป และไบต์ที่สามและสี่บ่งบอกขนาดข้อมูลในเฟรม ดังนั้นในการรับข้อมูลจากชิพอีเทอร์เนตผ่านพอร์ต DMA ไมโครคอนโทรลเลอร์จะอ่านข้อมูล page header และตรวจสอบขนาดทั้งหมดของเฟรมข้อมูลภายในบัฟเฟอร์ก่อน แล้วจึงค่อยอ่านค่าข้อมูลจากชิพอีเทอร์เนตนั้นมาเก็บไว้ในหน่วยความจำภายนอกครั้งละไบต์จนครบตามที่ระบุไว้ใน page header จากนั้นไมโครคอนโทรลเลอร์จะตรวจสอบบิต RDC ภายในรีจิสเตอร์ ISR โดยจะรอจนกว่าบิต RDC นี้มีค่าเป็น '1' เพื่อยืนยันว่าข้อมูลเฟรมนั้นได้ถูกอ่านออกจากพอร์ต DMA ครบและสำเร็จแล้ว เมื่อบิต RDC นี้มีค่าเป็น '1' แล้ว ไมโครคอนโทรลเลอร์จะรีเซตอินเตอร์รัปต์ทั้งหมด โดยการเขียนค่า '1' ลงในรีจิสเตอร์ ISR ทุกบิต จากนั้นจะเปรียบเทียบรีจิสเตอร์ CURR และ BNRY ซึ่งเป็นรีจิสเตอร์ชี้ตำแหน่งหน้าการอ่าน และรับข้อมูลของบัฟเฟอร์ภาครับตามลำดับ โดยถ้ารีจิสเตอร์ CURR และ BNRY มีค่าเท่ากันแสดงว่าไม่มีข้อมูลภายในบัฟเฟอร์ภาครับแล้วไมโครคอนโทรลเลอร์จะรีเซตอินเตอร์รัปต์ และปรับบิต RD2 ภายในรีจิสเตอร์ CR ให้มีค่าเป็น '1' เพื่อให้ชิพออกจากโมดการ

อ่านข้อมูลจากพอร์ต DMA และเป็นการสิ้นสุดกระบวนการรับข้อมูลจากชิพอีเทอร์เนต แต่ถ้ารีจิสเตอร์ CURR และ BNRY มีค่าไม่เท่ากันแสดงว่ายังมีเฟรมข้อมูลหลงเหลืออยู่ในบัฟเฟอร์ภาครับอยู่ ไมโครคอนโทรลเลอร์จะรับข้อมูลในบัฟเฟอร์นั้นโดยใช้กระบวนการดังที่กล่าวไว้ข้างต้นรับข้อมูลจากบัฟเฟอร์ภาครับจนกว่าจะหมด ดังแสดงในรูปที่ 3.19

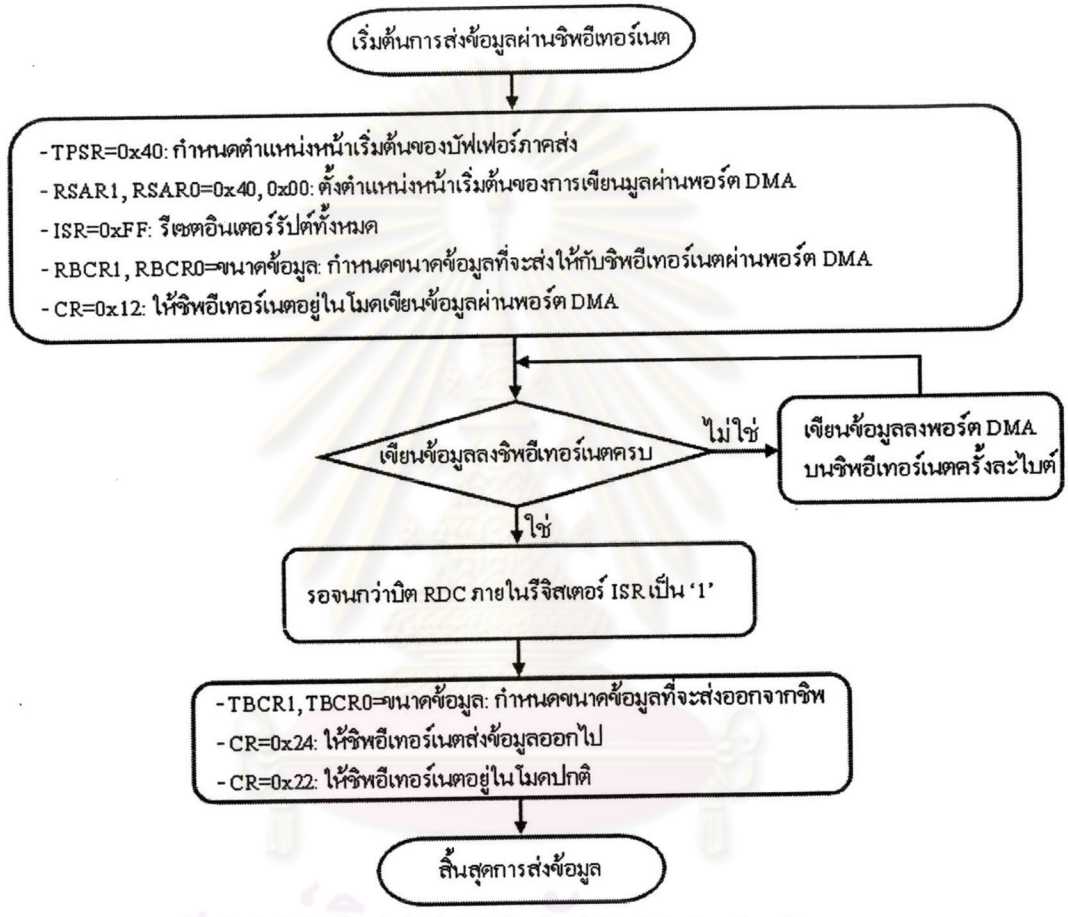


รูปที่ 3.19 การควบคุมเพื่อรับข้อมูลจากชิพอีเทอร์เนต

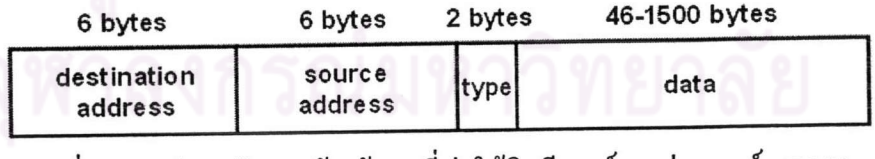
3.4.3 การส่งข้อมูลผ่านชิพอีเทอร์เนต RTL8019as

ในการส่งข้อมูลผ่านชิพอีเทอร์เนตนั้นมีความซับซ้อนน้อยกว่าการรับข้อมูลมาก ดังแสดงในรูปที่ 3.20 เริ่มแรกไมโครคอนโทรลเลอร์จะตั้งค่ารีจิสเตอร์ควบคุม TPSR เพื่อกำหนดตำแหน่งหน้าเริ่มต้นของบัฟเฟอร์ภาคส่ง และตั้งค่ารีจิสเตอร์ RSAR0 และ RSAR1 เพื่อระบุตำแหน่งการเขียนข้อมูลผ่านพอร์ต DMA ให้เริ่มเขียนที่ตำแหน่งหน้าเริ่มต้นของบัฟเฟอร์ภาคส่ง และตั้งค่า RBCR1 และ RBCR0 ซึ่งใช้ในการกำหนดขนาดของข้อมูลที่จะส่งเข้าพอร์ต DMA จากนั้นไมโครคอนโทรลเลอร์จะรีเซตรีจิสเตอร์ ISR และปรับบิต RD0, RD1, RD2 ภายในรีจิสเตอร์ CR ให้เป็น "010" ตามลำดับ เพื่อให้พอร์ต DMA ทำงานในโมดรับข้อมูล จากนั้นจึงเขียนข้อมูลลงพอร์ต DMA ครั้งละไบต์โดยข้อมูลที่เขียนนั้นต้องมีโครงสร้างดังแสดงในรูปที่ 3.21 เมื่อเขียนข้อมูลลงพอร์ต DMA ครบแล้ว ไมโครคอนโทรลเลอร์จะตรวจสอบบิต RDC ภายในรีจิสเตอร์ ISR และรอจนกว่าบิต RDC นี้มีค่าเป็น '1' เพื่อรับการยืนยันจากชิพอีเทอร์เนตว่ากระบวนการส่งข้อมูลผ่านพอร์ต DMA นั้นได้สำเร็จแล้ว เมื่อถึงขั้นตอนนี้ข้อมูลจะยังคงอยู่ในบัฟเฟอร์ภาคส่งของชิพอีเทอร์

เนต และยังไม่ถูกส่งออกไป ในการส่งข้อมูลออกจากชิพอีเทอร์เนตทำได้โดยการกำหนดค่ารีจิสเตอร์ควบคุม TBCR1, TBCR0 เพื่อกำหนดขนาดของข้อมูลที่ต้องการส่ง และตั้งบิต TXP ภายในรีจิสเตอร์ CR ให้เป็น '1' ชิพอีเทอร์เนตจะคำนวณ CRC สำหรับข้อมูลในบัฟเฟอร์ภาคส่งนั้น และเพิ่มข้อมูล CRC ขนาด 4 ไบต์ไว้ตอนท้ายของเฟรม และส่ง preamble ขนาด 8 ไบต์ออกไป ตามด้วยข้อมูลภายในบัฟเฟอร์ภาคส่งตามลำดับ เพื่อให้ได้เฟรมข้อมูลตามมาตรฐานของอีเทอร์เนต และเป็นการสิ้นสุดการควบคุมการส่งข้อมูล



รูปที่ 3.20 การควบคุมการส่งข้อมูลผ่านชิพอีเทอร์เนต



รูปที่ 3.21 รูปแบบโครงสร้างข้อมูลที่ส่งให้ชิพอีเทอร์เนตผ่านพอร์ต DMA

3.5 ส่วนรองรับโพรโทคอล

ในส่วนรองรับโพรโทคอลทำหน้าที่ประมวลผลข้อมูล, สร้างข้อมูล, และควบคุมการทำงานให้เป็นไปตามมาตรฐานโพรโทคอล ARP (Address Resolution Protocol), IP (Internet Protocol), UDP (User Datagram Protocol), RTP (Real Time Protocol), และ SIP (Session Initiation Protocol) ตามลำดับ

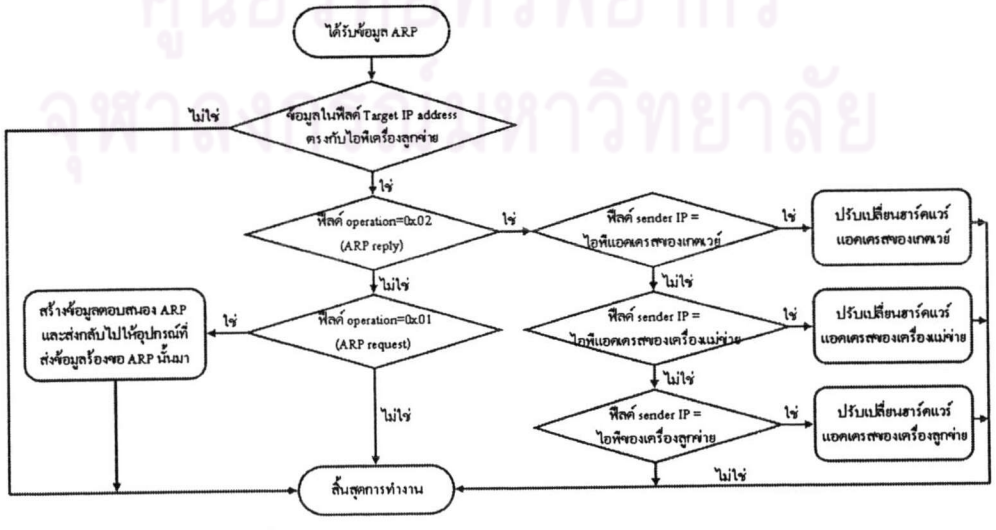
3.5.1 การทำงานรองรับข้อมูลที่ได้รับ

ในภาครับนั้น เมื่อได้รับข้อมูลจากชิพอีเทอร์เนตแล้ว ไมโครคอนโทรลเลอร์จะตรวจสอบและแยกแยะข้อมูลที่ได้รับว่าเป็นข้อมูลประเภทใด โดยตรวจสอบจากฟิลด์ Type ในอีเทอร์เนตเฮดเดอร์, ฟิลด์ Protocol ในเฮดเดอร์ไอพี, และฟิลด์ UDP destination port number ใน UDP เฮดเดอร์ ซึ่งจะสามารถแบ่งแยกข้อมูลออกได้เป็น ข้อมูลร้องขอ ARP, ข้อมูลตอบสนอง ARP, ข้อมูล ICMP, ข้อมูล SIP, ข้อมูลเสียง, และข้อมูลอื่น ๆ ที่ไม่สนใจ ดังแสดงในตารางที่ 3.5

ตารางที่ 3.5 การจำแนกข้อมูลของไมโครคอนโทรลเลอร์ในส่วนภาครับ

Type of data	Type (Ethernet header)	Protocol (IP header)	UDP destination port
ARP	0x0806 (ARP)	-	-
ICMP	0x0800 (IP)	0x01 (ICMP)	-
SIP	0x0800 (IP)	0x11 (UDP)	5060
Voice	0x0800 (IP)	0x11 (UDP)	49170

ในกรณีที่ข้อมูลที่ได้รับมานั้นเป็นข้อมูล ARP ไมโครคอนโทรลเลอร์จะตรวจสอบฟิลด์ target IP address และเปรียบเทียบกับไอพีแอดเดรสของเครื่องโทรศัพท์ ถ้ามีค่าไม่ตรงกัน ไมโครคอนโทรลเลอร์จะไม่สนใจ และจบการทำงานรองรับข้อมูลนั้น แต่ถ้ามีค่าตรงกัน ไมโครคอนโทรลเลอร์จะตรวจสอบฟิลด์ operation เพื่อแยกแยะว่าข้อมูล ARP นี้เป็นข้อมูลร้องขอ ARP หรือข้อมูลตอบสนอง ARP ถ้าตรวจสอบแล้วเป็นข้อมูลร้องขอ ARP ไมโครคอนโทรลเลอร์จะสร้างข้อมูลตอบสนอง ARP ภายในหน่วยความจำภายนอก และส่งให้ชิพอีเทอร์เนตเพื่อตอบสนองการร้องขอนั้น แต่ถ้าเป็นข้อมูล ARP ตอบสนอง ไมโครคอนโทรลเลอร์จะตรวจสอบฟิลด์ Sender IP address ว่าเป็นไอพีแอดเดรสของเครื่องลูกข่ายปลายทาง, เครื่องแม่ข่าย หรือเกตเวย์ เมื่อสามารถจำแนกได้แล้วก็จะใช้ค่าภายในฟิลด์ sender hardware address ปรับเปลี่ยนค่าฮาร์ดแวร์แอดเดรสของอุปกรณ์นั้นใหม่ดังแสดงในรูปที่ 3.22



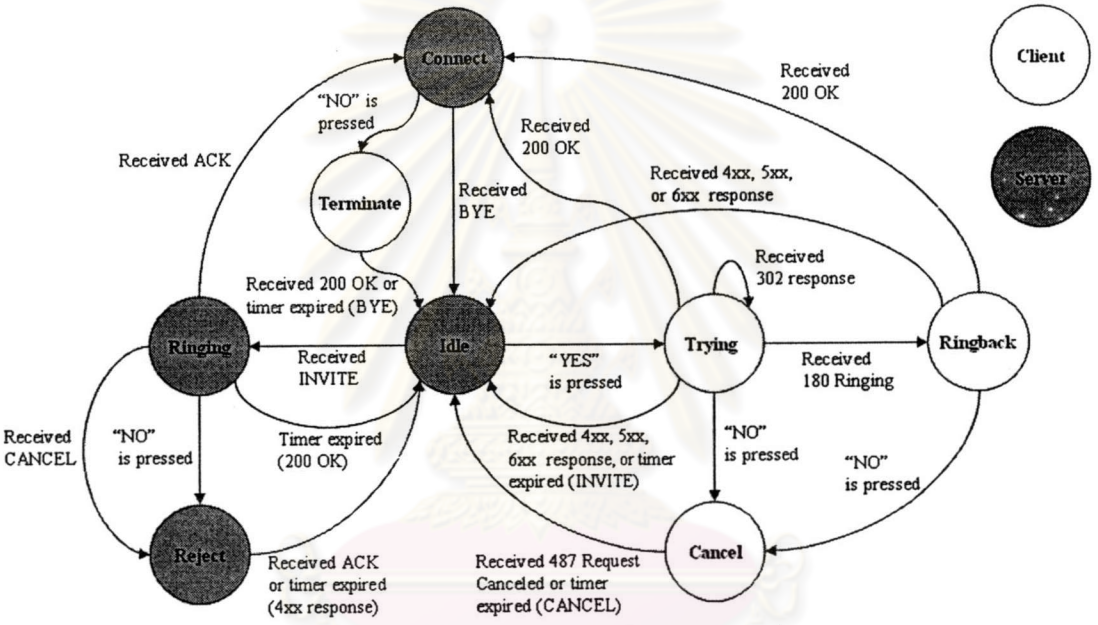
รูปที่ 3.22 การรองรับโพรโทคอล ARP ในภาครับ

ในการส่งข้อมูลร้องขอ ARP เพื่อหาฮาร์ดแวร์แอดเดรสของเกตเวย์นั้น จะเกิดขึ้นทุกครั้งที่ไมโครคอนโทรลเลอร์ได้รับข้อมูลไอพีแอดเดรสของอุปกรณ์ภายนอกอื่นมาใหม่ และไอพีแอดเดรสนั้นอยู่ในโครงข่ายพื้นที่ท้องถิ่นเดียวกับไอพีแอดเดรสของเครื่องโทรศัพท์ ซึ่งมีอยู่ 3 ช่วงด้วยกัน คือช่วงเริ่มเปิดเครื่องโทรศัพท์, ช่วงที่ได้จัดรูปลักษณ์ (configure) ไอพีแอดเดรสให้กับอุปกรณ์อื่นใหม่ และช่วงเริ่มสร้างเซสชันกับเครื่องลูกข่ายปลายทาง ในช่วงเริ่มเปิดเครื่องโทรศัพท์เมื่อไมโครคอนโทรลเลอร์โหลดไอพีแอดเดรส ของอุปกรณ์ภายนอก จาก EEPROM ลงในหน่วยความจำเรียบร้อยแล้ว เครื่องโทรศัพท์จะส่งข้อมูลร้องขอ ARP เพื่อค้นหาฮาร์ดแวร์แอดเดรสของเกตเวย์ เมื่อได้รับหมายเลขฮาร์ดแวร์แอดเดรสของเกตเวย์แล้ว ในกรณีที่เครื่องโทรศัพท์ทำงานในโมดทำงานร่วมกับเครื่องแม่ข่าย เครื่องโทรศัพท์จะตรวจสอบว่าเครื่องแม่ข่ายอยู่ในโครงข่ายพื้นที่ท้องถิ่นวงเดียวกับเครื่องโทรศัพท์หรือไม่ ถ้าเครื่องแม่ข่ายอยู่ในโครงข่ายเดียวกัน เครื่องโทรศัพท์จะส่งข้อมูลร้องขอ ARP เพื่อค้นหาฮาร์ดแวร์แอดเดรสของเครื่องแม่ข่ายต่อไป ในช่วงที่ได้จัดรูปลักษณ์ไอพีแอดเดรสให้กับเกตเวย์ และเครื่องแม่ข่ายก็เช่นกัน เมื่อจัดรูปลักษณ์เรียบร้อยแล้วถ้าไอพีแอดเดรสนั้นอยู่ในโครงข่ายเดียวกับเครื่องโทรศัพท์ ไมโครคอนโทรลเลอร์จะส่งข้อมูลร้องขอ ARP เพื่อค้นหาฮาร์ดแวร์แอดเดรสของอุปกรณ์ภายนอกใหม่ ส่วนช่วงเริ่มสร้างเซสชันกับเครื่องลูกข่ายปลายทาง ในกรณีที่เครื่องโทรศัพท์ทำงานในโมดติดต่อโดยตรง เมื่อผู้ใช้งานต้องการสร้างเซสชันกับเครื่องลูกข่ายปลายทางที่อยู่ภายในโครงข่ายเดียวกัน และได้ป้อน SIP URL ปลายทางเรียบร้อยแล้ว เครื่องโทรศัพท์จะส่งข้อมูลร้องขอ ARP เพื่อค้นหาฮาร์ดแวร์แอดเดรสของเครื่องลูกข่ายปลายทางก่อนทุกครั้ง ส่วนในกรณีที่เครื่องโทรศัพท์ทำงานในโมดทำงานร่วมกับเครื่องแม่ข่าย หรือเป็นฝ่ายถูกร้องขอการสร้างเซสชัน เมื่อเครื่องโทรศัพท์ได้รับข้อความ SIP ที่บ่งบอกถึงไอพีแอดเดรสของเครื่องลูกข่ายปลายทางที่อยู่ภายในโครงข่ายเดียวกัน เครื่องโทรศัพท์จะส่งข้อมูลร้องขอ ARP เพื่อค้นหาฮาร์ดแวร์แอดเดรสของเครื่องลูกข่ายทุกครั้งเช่นกัน

ในกรณีที่ข้อมูลที่ได้รับมามีในอีเทอร์เนตเฮดเดอร์มีฟิลด์ type เป็น 0x0800 แสดงว่าข้อมูลในชั้นบนเป็นโพรโทคอล IP ไมโครคอนโทรลเลอร์จะเปรียบเทียบฟิลด์ destination IP address กับไอพีแอดเดรสของตัวเครื่อง ถ้าไม่ตรงกันไมโครคอนโทรลเลอร์จะไม่สนใจข้อมูลนั้น แต่ถ้าตรงกันไมโครคอนโทรลเลอร์จะตรวจสอบข้อมูลในฟิลด์ protocol ในไอพีเฮดเดอร์เพื่อแยกแยะระหว่างข้อมูล ICMP และ UDP โดยถ้าข้อมูลที่ได้รับเป็นข้อมูล ICMP ไมโครคอนโทรลเลอร์จะสร้างข้อมูลตอบสนอง ICMP และส่งกลับไปยังปลายทาง แต่ถ้าข้อมูลที่ได้รับเป็นข้อมูล UDP ไมโครคอนโทรลเลอร์จะตรวจสอบฟิลด์ destination port ภายใน UDP เฮดเดอร์ ถ้ามีค่าเป็น 49170 ซึ่งเป็นพอร์ตที่เครื่องโทรศัพท์นั้นตกลงกับเครื่องลูกข่ายปลายทางให้เป็นพอร์ตที่ใช้ในการรับข้อมูลเสียงไว้ก่อนแล้ว ไมโครคอนโทรลเลอร์จะนำข้อมูลเสียงนั้นเขียนลงในบัฟเฟอร์ข้อมูลเสียงภาครับเพื่อรอการแสดงผล (Play Back) ออกไป แต่ถ้าข้อมูลมีพอร์ตปลายทางเป็น 5060 ซึ่งเป็น

พอร์ตสำหรับใช้ในการส่งข้อความ SIP ไมโครคอนโทรลเลอร์จะตรวจสอบข้อความ SIP เพื่อตีความข้อความ SIP นั้นต่อไป

ในการสร้างและ ยกเลิกเซสชันด้วยโพรโทคอล SIP นั้น เครื่องโทรศัพท์ต้องสามารถสามารถทำงานเป็นได้ทั้งเครื่องลูกข่าย และเครื่องแม่ข่าย และต้องทำงานแบบมีสถานะ โดยมีสถานะ และการทำงานคร่าว ๆ ดังแสดงในรูปที่ 3.23 จะเห็นได้ว่าการสร้างเซสชันนั้นสามารถแบ่งออกได้เป็น 2 แบบ แบบแรกเครื่องโทรศัพท์ที่ทำหน้าที่เป็นเครื่องลูกข่ายร้องขอเพื่อขอสร้างเซสชัน และแบบที่ 2 เครื่องโทรศัพท์ที่เป็นเครื่องแม่ข่ายถูกร้องขอการสร้างเซสชัน ในการยกเลิกเซสชันก็เช่นกันเครื่องโทรศัพท์อาจเป็นเครื่องลูกข่ายขอยกเลิกเซสชันเองก็ได้ หรือเป็นเครื่องแม่ข่ายถูกร้องขอให้ยกเลิกเซสชันก็ได้



รูปที่ 3.23 แผนผังสถานะการทำงานเพื่อรองรับโพรโทคอล SIP

การเปลี่ยนสถานะของเครื่องโทรศัพท์นั้นขึ้นอยู่กับ การกดคีย์แพดบนเครื่องโทรศัพท์ และข้อมูล SIP ที่ได้รับจากภายนอก ในกรณีเครื่องโทรศัพท์ทำงานเป็นเครื่องลูกข่ายขอสร้างเซสชันกับเครื่องปลายทาง เมื่อผู้ใช้ได้ป้อนข้อมูล SIP URL ในขณะที่เครื่องโทรศัพท์อยู่ในสถานะ Idle และกดปุ่ม “YES” เพื่อเริ่มการขอสร้างเซสชัน เครื่องโทรศัพท์จะเปลี่ยนสถานะจาก Idle เป็น Trying พร้อมกับส่งข้อความ INVITE ออกไป ถ้าเครื่องโทรศัพท์ได้รับข้อมูลตอบสนอง 180 จากปลายทาง เครื่องโทรศัพท์จะเปลี่ยนสถานะจาก Trying เป็น Ringback ทันที และเมื่อได้รับข้อมูลตอบสนอง 200 OKในกรณีที่ผู้ใช้ปลายทางตกลงรับการสร้างเซสชัน เครื่องโทรศัพท์จะเปลี่ยนสถานะเป็น Connect และถือเป็นการสิ้นสุดการสร้างเซสชัน แต่ในกรณีที่ผู้ใช้งานปลายทางปฏิเสธการขอสร้างเซสชัน หรือมีปัญหาไม่สามารถสร้างเซสชันได้เครื่องโทรศัพท์จะได้รับข้อมูลตอบสนอง 4xx, 5xx, หรือ 6xx และจะกลับมาอยู่ในสถานะ Idle อีกครั้ง และในกรณีที่ผู้ใช้งานเกิดเปลี่ยนใจต้องการยกเลิกการสร้างเซสชันโดยกดปุ่ม “NO” ในขณะที่เครื่องโทรศัพท์อยู่ในสถานะ Trying หรือ

Ringback เครื่องโทรศัพท์จะเปลี่ยนสถานะเป็น Cancel พร้อมกับส่งข้อความ CANCEL ไปยังปลายทาง เมื่อได้รับข้อความตอบสนอง 487 ซึ่งเป็นการตอบรับการยกเลิกเซสชัน เครื่องโทรศัพท์จะเปลี่ยนสถานะกลับเป็น Idle

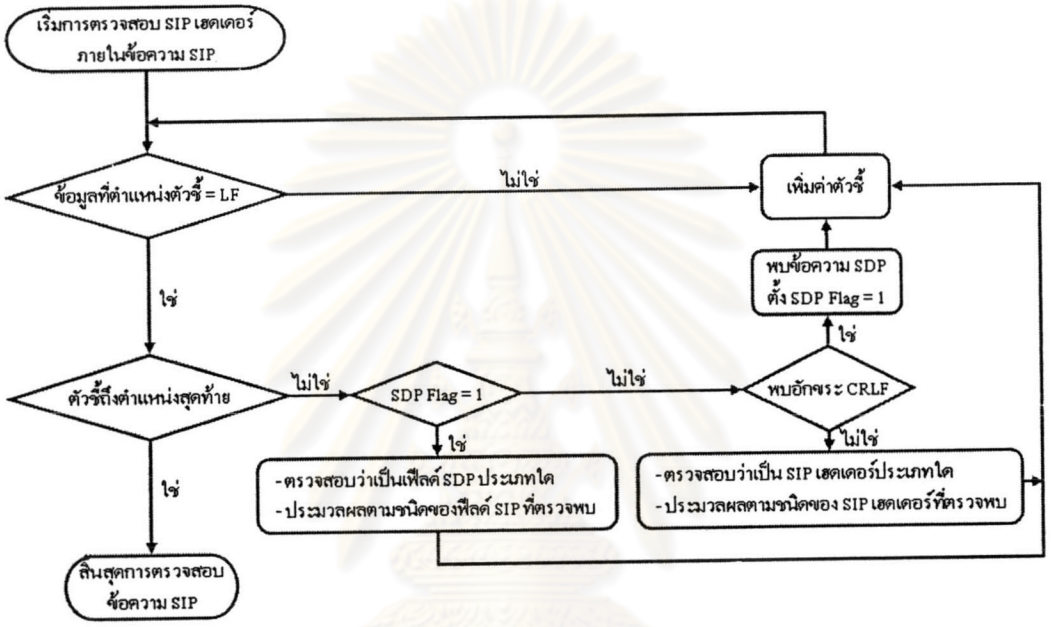
ในการสร้างเซสชันในกรณีเครื่องโทรศัพท์ทำงานเป็นเครื่องแม่ข่าย เมื่อเครื่องโทรศัพท์ได้รับข้อความร้องขอ INVITE จะเปลี่ยนสถานะจาก Idle เป็น Ringing พร้อมกับส่งข้อความตอบสนอง 180 Ringing กลับไป เมื่อผู้ใช้กดปุ่ม "YES" เพื่อรับการร้องขอการสร้างเซสชันนั้น เครื่องโทรศัพท์จะส่ง 200 OK กลับไปและรอจนกว่าจะได้รับข้อความ ACK ซึ่งเป็นการตอบรับการสร้างเซสชัน เครื่องโทรศัพท์จะเปลี่ยนสถานะเป็น Connect และเป็นการสิ้นสุดการสร้างเซสชัน แต่ในกรณีที่ผู้ใช้งานปฏิเสธการสร้างเซสชันโดยกดปุ่ม "NO" หรือปลายทางต้องการยกเลิกการสร้างเซสชันซึ่งเครื่องโทรศัพท์จะได้รับข้อความ CANCEL ในขณะที่เครื่องอยู่ในสถานะ Ringing เครื่องโทรศัพท์จะส่งข้อความตอบสนองกลับไป และเปลี่ยนสถานะกลับไปอยู่ในสถานะ Idle เมื่อเครื่องโทรศัพท์ได้รับข้อความ ACK ซึ่งเป็นการตอบรับการตอบสนองนั้น

ในการปิดเซสชัน กรณีที่เครื่องโทรศัพท์เป็นผู้ร้องขอการปิดเซสชัน เมื่อผู้ใช้งานกดปุ่ม "NO" ในขณะที่เครื่องอยู่ในสถานะ Connect เครื่องโทรศัพท์จะเปลี่ยนสถานะเป็น Terminate พร้อมกับส่งข้อความร้องขอ BYE ไปยังเครื่องปลายทาง และเมื่อได้รับข้อความตอบสนอง 200 OK เครื่องโทรศัพท์จึงจะกลับมาอยู่ในสถานะ Idle และเป็นการสิ้นสุดการขอปิดเซสชัน ส่วนในกรณีที่เครื่องโทรศัพท์เป็นผู้ถูกร้องขอการปิดเซสชัน เมื่อเครื่องโทรศัพท์ได้รับข้อความร้องขอ BYE จากเครื่องปลายทาง เครื่องโทรศัพท์จะเปลี่ยนสถานะกลับไปยังสถานะ Idle พร้อมกับส่งข้อความตอบสนอง 200 OK กลับไป และเป็นการสิ้นสุดการขอปิดเซสชัน เช่นกัน

จะเห็นได้ว่าเครื่องโทรศัพท์ใช้ Method ภายใน Request-Line และ Status-Code ภายใน Status-Line ของข้อความ SIP ที่ได้รับมาเป็นส่วนสำคัญที่ใช้ควบคุมสถานะของเครื่องโทรศัพท์ แต่ในการสร้างเซสชันเครื่องโทรศัพท์จำเป็นต้องทราบพารามิเตอร์อื่น ๆ ที่สำคัญเช่น เซสชันไอดี, SIP URL ปลายทาง, ไอพีแอดเดรสเครื่องลูกข่ายปลายทาง, พอร์ตที่ใช้ในการรับข้อมูลเสียง, ชนิดการเข้ารหัสเสียงที่จะใช้ในเซสชัน เป็นต้น ดังนั้นในบางกรณีไมโครคอนโทรลเลอร์จะเข้าไปตรวจสอบภายในเฮดเดอร์ และรวมไปถึงข้อความ SDP (Session Description Protocol) ด้วย

การตรวจสอบ SIP เฮดเดอร์ และฟิลด์ SDP ภายในข้อความ SIP แสดงในรูปที่ 3.24 ไมโครคอนโทรลเลอร์จะไล่หาจุดเริ่มต้นของ SIP เฮดเดอร์ โดยการไล่ตรวจสอบอักขระครั้งละตัวเริ่มตั้งแต่ในส่วน Request-Line หรือ Status-Line ไปจนกว่าจะพบอักขระพิเศษ CRLF (Carriage Return/Line Feed) ซึ่งเป็นตัวระบุจุดสิ้นสุดบรรทัด เมื่อพบแล้วไมโครคอนโทรลเลอร์จะตรวจสอบอักขระในบรรทัดถัดไปเพื่อแยกแยะว่าเป็น SIP เฮดเดอร์ประเภทไหน แล้วจึงค่อยประมวลผลข้อความภายในบรรทัดนั้นตาม SIP เฮดเดอร์ที่ตรวจพบ เมื่อประมวลผลข้อความเสร็จไมโครคอนโทรลเลอร์จะไล่หาอักขระพิเศษ CRLF เพื่อตรวจสอบ SIP เฮดเดอร์บรรทัดต่อไป

เรื่อย ๆ จนกว่าจะประมวลผลครบทุกเฮดเดอร์ ในกรณีที่มีข้อความ SDP ต่อท้าย เมื่อไมโครคอนโทรลเลอร์ประมวลผล SIP เฮดเดอร์จนครบหมด และได้ไล่หาอักขระพิเศษ CRLF ที่ใช้ระบุจุดสิ้นสุดบรรทัดสุดท้ายของ SIP เฮดเดอร์แล้ว ในการตรวจสอบอักขระบรรทัดต่อไปจะพบอักขระ CRLF ซึ่งเป็นการเว้นบรรทัดระหว่าง SIP เฮดเดอร์กับข้อความ SDP ไมโครคอนโทรลเลอร์จะปรับเปลี่ยนการประมวลผลข้อความจาก SIP เฮดเดอร์ เป็นฟิลด์ SDP และจะไล่ตรวจสอบฟิลด์ SDP นั้นครั้งละบรรทัดโดยใช้หลักการเดียวกับที่ได้ตรวจ SIP เฮดเดอร์จนกว่าจะจบบรรทัดสุดท้าย เป็นการสิ้นสุดการประมวลผลข้อความ SIP



รูปที่ 3.24 ขั้นตอนการตรวจสอบ SIP เฮดเดอร์ และ SDP ภายในข้อความ SIP

ในการพัฒนาโพรโทคอล SIP ในวิทยานิพนธ์นี้ ไมโครคอนโทรลเลอร์สามารถประมวลผล SIP เฮดเดอร์ ได้ 6 ชนิดคือ Via, From, To, Cseq, Call-ID, และ Contact และสามารถประมวลผลฟิลด์ SDP ได้ 2 ชนิดคือฟิลด์ c และ m ดังแสดงในตารางที่ 3.6 จะเห็นได้ว่าวัตถุประสงค์ของการประมวลผล SIP เฮดเดอร์ และฟิลด์ SDP นั้นเพื่อเก็บข้อมูลต่าง ๆ ที่จำเป็นต่อการสร้างเซสชัน ซึ่งได้แก่พอร์ตที่ใช้ในการส่งข้อความ SIP, SIP URL ของผู้ใช้ปลายทาง, หมายเลข Cseq, หมายเลข Call-ID, ไอพีแอดเดรสปลายทาง, Tag, พอร์ตที่ใช้ในการส่งข้อมูลเสียง, การเลือกใช้วิธีการเข้ารหัสเสียง รวมไปถึงการตรวจสอบ SIP URL ที่ผู้ใช้ปลายทางต้องการสร้างเซสชัน ในกรณีที่ตัวเครื่องโทรศัพท์เป็นผู้ขอสร้างเซสชัน ตัวเครื่องโทรศัพท์จะทราบข้อมูลส่วนใหญ่อยู่แล้ว มีเพียงข้อมูลที่ระบุการส่งข้อมูลเสียง และไอพีแอดเดรสของเครื่องโทรศัพท์ปลายทางในกรณีที่เครื่องทำงานในโมดทำงานร่วมกับเครื่องแม่ข่ายเท่านั้นที่เครื่องต้องตรวจสอบจาก SIP เฮดเดอร์ และฟิลด์ SDP ของข้อความตอบสนองที่ได้รับ แต่ในกรณีที่เครื่องโทรศัพท์เป็นผู้ถูกร้องขอการสร้างเซสชัน เครื่องจะไม่มีข้อมูลเลย และต้องตรวจสอบจาก SIP เฮดเดอร์ และฟิลด์ SDP ของข้อความร้องขอ INVITE ที่ได้รับทั้งหมด

ตารางที่ 3.6 ข้อมูลที่ได้จากการประมวลผล SIP เฮดเดอร์ และฟิลด์ SDP

		INVITE	200 OK	302 moved temporarily
SIP เฮดเดอร์	Via	พอร์ต SIP ปลายทาง	-	-
	From	SIP URL ปลายทาง	-	-
	To	*Tag, ตรวจสอบ SIP URL	-	-
	Cseq	Cseq	-	-
	Call-ID	Call-ID	-	-
	Contact	-	-	*ไอพีแอดเดรส ปลายทาง
ฟิลด์ SDP	c	ไอพีแอดเดรสปลายทาง	*ไอพีแอดเดรสปลายทาง	-
	m	พารามิเตอร์ของข้อมูลเสียง	พารามิเตอร์ของข้อมูลเสียง	-

* เครื่องโทรศัพท์ทำงานในโมด ทำงานร่วมกับเครื่องแม่ข่าย

3.5.3 การสร้างข้อมูลตามมาตรฐานโพรโทคอลต่าง ๆ

ในภาคส่งนั้น เครื่องโทรศัพท์สามารถสร้างข้อความร้องขอได้ 5 ชนิดคือ INVITE, CANCEL, ACK, BYE, และ REGISTER ข้อความตอบสนองได้ 4 ชนิดคือ 180 Ringing, 200 OK, 486 Busy Here, และ 487 Request Canceled และในส่วน SIP เฮดเดอร์ นั้นเครื่องโทรศัพท์สามารถสร้างได้ 7 ชนิดคือ Via, From, To, Call-ID, Cseq, Content-Length, และ Contact และในส่วนฟิลด์ SDP สามารถสร้างได้ 6 ชนิดคือ v, o, s, c, t, และ m โดยในการสร้างข้อความ SIP นั้น ในกรณีที่มีโพรโทคอล SDP อยู่ในข้อความด้วย ไมโครคอนโทรลเลอร์จะสร้างข้อความ SDP เก็บไว้ในหน่วยความจำภายนอกก่อน เพื่อให้ทราบขนาดทั้งหมดของข้อความ SDP เพื่อนำไปใช้ในเฮดเดอร์ Content-Length จากนั้นจึงค่อยสร้างข้อความ SIP ครั้งละบรรทัดโดยแต่ละบรรทัดคั่นด้วยอักขระพิเศษ CRLF เริ่มจาก Request-line หรือ Status-line และตามด้วย SIP เฮดเดอร์ โดยในการสร้างเฮดเดอร์ Via ไมโครคอนโทรลเลอร์จะตรวจสอบสถานะของเครื่องขณะนั้น ว่าเป็นผู้ร้องขอ หรือผู้ถูกร้องขอ โดยในกรณีที่ทำงานเป็นผู้ร้องขอไมโครคอนโทรลเลอร์จะใช้ไอพีแอดเดรสของตัวเครื่องเอง และพอร์ต 5060 เป็นข้อมูลสำหรับการสร้างเฮดเดอร์ แต่ในกรณีที่เครื่องทำงานเป็นผู้ถูกร้องขอ ไมโครคอนโทรลเลอร์จะตรวจสอบว่าข้อความตอบสนองที่จะสร้างนี้ ตอบสนองข้อความร้องขอที่ผ่านเครื่องแม่ข่ายมาหรือไม่ ถ้าเป็นข้อความร้องขอที่ไม่ผ่านเครื่องแม่ข่าย ไมโครคอนโทรลเลอร์จะใช้ไอพีแอดเดรสของเครื่องปลายทาง และพอร์ตรับข้อความ SIP ปลายทางที่ได้จากการตรวจสอบ SIP เฮดเดอร์ในข้อความ INVITE มาเป็นข้อมูลสำหรับสร้างเฮดเดอร์ แต่ถ้าเป็นข้อความร้องขอที่ผ่านเครื่องแม่ข่ายมา ไมโครคอนโทรลเลอร์จะใช้ไอพีแอดเดรสของเครื่องแม่ข่ายแทน ในการสร้างเฮดเดอร์ From ซึ่งเป็นเฮดเดอร์ที่ระบุ SIP URL ของผู้ร้องขอ ดังนั้นในกรณีที่เครื่องโทรศัพท์อยู่ในสถานะเป็นผู้ร้องขอ ไมโครคอนโทรลเลอร์จะใช้ข้อมูล SIP URL ของตัวเครื่องเอง เป็นข้อมูลในการสร้างเฮดเดอร์ แต่ในกรณีที่เครื่องอยู่ในสถานะเป็นผู้ถูกร้องขอไมโครคอนโทรลเลอร์จะใช้ข้อมูล SIP URL ของผู้ใช้ปลายทางเป็นข้อมูลในการสร้างเฮดเดอร์ ในการสร้างเฮดเดอร์ To ซึ่งเป็น

เฮดเดอร์ที่ระบุ SIP URL ของผู้ใช้งานปลายทางซึ่งเป็นผู้ถูกร้องขอ ดังนั้นในกรณีที่เครื่องอยู่ในสถานะ เป็นผู้ร้องขอ ไมโครคอนโทรลเลอร์จะนำ SIP URL ปลายทางที่ได้จากการป้อนข้อมูลของผู้ใช้งานผ่านคีย์แพดมาสร้างเฮดเดอร์ และในกรณีที่เครื่องอยู่ในสถานะเป็นผู้ถูกร้องขอ ไมโครคอนโทรลเลอร์จะใช้ SIP URL ของตัวเครื่องเองเป็นข้อมูลในการสร้างเฮดเดอร์ ในการสร้างเฮดเดอร์ Call-ID ซึ่งเป็นเฮดเดอร์ที่ใช้ในการระบุเซสชัน มีรูปแบบเป็น *หมายเลขสุ่ม@ไอพีแอดเดรสของเครื่องที่ร้องขอสร้างเซสชัน* ดังนั้นในกรณีที่เครื่องทำงานเป็นผู้ร้องขอ ไมโครคอนโทรลเลอร์จะใช้หมายเลขสุ่ม ที่ได้จากการสุ่มก่อนร้องขอ และไอพีแอดเดรสของเครื่องมาเป็นข้อมูลในการสร้างเฮดเดอร์ และในกรณีที่เครื่องทำงานในสถานะเป็นผู้ถูกร้องขอ ไมโครคอนโทรลเลอร์จะนำอักขระซึ่งระบุถึง Call-ID ภายในเฮดเดอร์ Call-ID ของข้อความร้องขอมาเป็นข้อมูลในการสร้างเฮดเดอร์ ในการสร้างเฮดเดอร์ Cseq ซึ่งเป็นเฮดเดอร์ที่ระบุจำนวนครั้งและชนิดของการร้องขอ โดยในกรณีที่เครื่องทำงานในสถานะเป็นผู้ร้องขอ ไมโครคอนโทรลเลอร์จะใช้ข้อมูลที่ทำหน้าที่นับการร้องขอภายในหน่วยความจำ และชนิดการร้องขอนั้นมาเป็นข้อมูลในการสร้างเฮดเดอร์ และในกรณีที่เครื่องทำงานในสถานะเป็นผู้ถูกร้องขอ ไมโครคอนโทรลเลอร์จะนำข้อมูลที่ได้จากเฮดเดอร์ Cseq ของข้อความร้องขอมาเป็นข้อมูลในการสร้างเฮดเดอร์ ในการสร้างเฮดเดอร์ Contact ซึ่งเป็นเฮดเดอร์ที่ใช้ระบุหมายเลขไอพีแอดเดรสเครื่อง ดังนั้นไมโครคอนโทรลเลอร์จะใช้ไอพีแอดเดรสของเครื่องเป็นข้อมูลในการสร้างเฮดเดอร์ ทั้งในกรณีที่เครื่องทำงานในสถานะเป็นผู้ร้องขอ และเป็นผู้ถูกร้องขอ ในการสร้างเฮดเดอร์ Content-Type ซึ่งเป็นเฮดเดอร์ที่ระบุชนิดของข้อมูลซึ่งบรรจุอยู่ในข้อความ SIP ซึ่งในการพัฒนาโทรศัพท์ SIP นี้มีเพียงข้อมูล SDP เท่านั้น ดังนั้นในการสร้างเฮดเดอร์ Content-Type จึงเป็นข้อมูลตายตัวระบุถึงข้อมูล SDP เป็นข้อมูลภายในข้อความ SIP ในการสร้างเฮดเดอร์ Content-Length ซึ่งใช้ระบุขนาดของข้อมูลที่บรรจุอยู่ในข้อความ SIP ซึ่งก็คือขนาดของข้อมูล SDP นั่นเอง ไมโครคอนโทรลเลอร์จะใช้ข้อมูลที่ได้จากการนับขนาดของข้อมูล SDP ที่ได้สร้างไว้ก่อนมาเป็นข้อมูลในการสร้างเฮดเดอร์นี้

ในการสร้างเฮดเดอร์ RTP สำหรับรองรับข้อมูลเสียงไมโครคอนโทรลเลอร์จะสร้างฟิลด์ V ให้มีค่าเป็น 2 เพื่อเป็นการระบุการใช้โพรโทคอล RTP เวอร์ชัน 2 ซึ่งเป็นเวอร์ชันปัจจุบัน ส่วนฟิลด์ P, X, CC, และ M นั้นจะกำหนดค่าให้เป็น 0 ทั้งหมดเพื่อระบุว่าไม่มีการเพิ่มข้อมูลแพดในตอนท้ายของแพ็กเกต, ไม่มีเฮดเดอร์เพิ่มเติม, มีแหล่งกำเนิดข้อมูลเดียว, และไม่มีการกระทำพิเศษเกิดขึ้นบนแพ็กเกตนั้น, ส่วนในฟิลด์ payload type ไมโครคอนโทรลเลอร์จะกำหนดค่าเป็น 0 ในกรณีชิพเข้ารหัสเสียงทำงานในโมดเข้ารหัส PCM μ-law หรือกำหนดค่าเป็น 8 ในกรณีที่ชิพเข้ารหัสเสียงทำงานในโมดเข้ารหัส PCM A-law ตามลำดับ ในฟิลด์ sequence number และ time stamp เมื่อเริ่มเปิดเซสชันไมโครคอนโทรลเลอร์จะสุ่มค่าเริ่มต้นสำหรับฟิลด์ และจะเพิ่มค่าภายในฟิลด์ทุก ๆ ครั้งที่ส่งแพ็กเก็ตข้อมูลเสียงออกไป โดยในฟิลด์ sequence number ไมโครคอนโทรลเลอร์จะเพิ่มค่าที

ละ 1 เพื่อระบุลำดับแพ็กเก็ตเสียง ส่วนในฟิลด์ time stamp ไมโครคอนโทรลเลอร์จะเพิ่มค่าภายในฟิลด์ทีละ 160 เพื่อบ่งบอกจำนวนการชักตัวอย่างข้อมูลเสียงในแพ็กเก็ตนั้น ในฟิลด์ SSRCI ซึ่งใช้ระบุต้นทางผู้สร้างแพ็กเกต เมื่อเริ่มเปิดเซสชันไมโครคอนโทรลเลอร์จะสุ่มค่าภายในฟิลด์ และใช้ค่าที่สุ่มขึ้นนั้นตลอดจนกว่าจะปิดเซสชัน ส่วนในฟิลด์ CSRC ไมโครคอนโทรลเลอร์จะละทิ้ง เนื่องจากมีแหล่งกำเนิดข้อมูลเสียงเพียงแหล่งเดียว

ในการสร้างเฮดเดอร์ UDP ซึ่งประกอบด้วยฟิลด์ Source Port, Destination Port, UDP length, และ Checksum ไมโครคอนโทรลเลอร์จะกำหนดฟิลด์ Source Port เป็น 50000 เสมอ ส่วนฟิลด์ Destination Port ในกรณีที่ข้อมูลเป็นข้อความร้องขอ SIP ไมโครคอนโทรลเลอร์จะกำหนดให้เป็น 5060 ซึ่งเป็นพอร์ต Default สำหรับโพรโทคอล SIP แต่ในกรณีที่ข้อมูลเป็นข้อความตอบสนอง SIP ไมโครคอนโทรลเลอร์จะใช้ค่าพอร์ตที่ได้จากเฮดเดอร์ Via ภายในข้อความร้องขอ SIP มาเป็นข้อมูลในการกำหนดพอร์ตปลายทาง และกรณีที่เป็นข้อมูลเสียงไมโครคอนโทรลเลอร์จะใช้ค่าพอร์ตที่ได้จากฟิลด์ m ภายในข้อความ SDP มาเป็นข้อมูลในการกำหนดพอร์ตปลายทางตามลำดับ ในการสร้างฟิลด์ Length ซึ่งเป็นฟิลด์ที่ระบุถึงขนาดของ UDP ดาตาแกรมนั้น ในกรณีที่ข้อมูลเป็นข้อความ SIP ไมโครคอนโทรลเลอร์จะตรวจสอบขนาดของข้อความ SIP ที่ได้สร้างขึ้น และใช้ข้อมูลจากการตรวจสอบนี้มารวมกับเฮดเดอร์ UDP ซึ่งมีขนาด 8 ไบต์เพื่อให้ได้ขนาดของ UDP ดาตาแกรม และในกรณีที่ข้อมูลเป็นข้อมูลเสียงไมโครคอนโทรลเลอร์จะกำหนดให้ฟิลด์ Length นี้มีค่าเป็น 180 เสมอ เนื่องจากข้อมูลเสียงทุกแพ็กเกตมีขนาดคงที่ 160 ไบต์ รวมกับเฮดเดอร์ RTP ขนาด 12 ไบต์และเฮดเดอร์ UDP ขนาด 8 ไบต์ตามลำดับ ในส่วนการสร้างฟิลด์ UDP Checksum นั้น ไมโครคอนโทรลเลอร์จะละไว้ ไม่คำนวณและใส่ค่าเป็น '0' หมดทั้งฟิลด์

ในการสร้าง IP เฮดเดอร์ดังแสดงในตารางที่ 3.7 ไมโครคอนโทรลเลอร์จะกำหนดฟิลด์ version, header length, TOS, flag, fragment offset, และ time to live เป็น 4, 5, 0x10, 0, 0, และ 64 ตามลำดับ เพื่อให้โพรโทคอลไอพีที่ใช้เป็นเวอร์ชัน 4, ขนาดของเฮดเดอร์คงที่ 20 ไบต์, ให้เราเตอร์ส่งผ่านไอพีดาตาแกรมบนเส้นทางที่มีการประวิงเวลาต่ำที่สุด, และอนุญาตให้ แบ่ง (fragment) แพ็กเกตได้ตามลำดับ ส่วนฟิลด์ total length นั้นไมโครคอนโทรลเลอร์จะคำนวณจากขนาดของข้อความ SIP รวมกับ UDP เฮดเดอร์ 8 ไบต์ และ IP เฮดเดอร์ 20 ไบต์ตามลำดับ หรือถ้าเป็นข้อมูลเสียงจะมีค่าเป็น 200 คงที่ตลอดซึ่งได้มาจากข้อมูลเสียงขนาด 160 ไบต์รวมกับ RTP เฮดเดอร์ 12 ไบต์, UDP เฮดเดอร์ 8 ไบต์, และ IP เฮดเดอร์ 20 ไบต์ตามลำดับ ในฟิลด์ identification ไมโครคอนโทรลเลอร์จะเพิ่มค่าในฟิลด์นี้ทุกครั้งที่ส่งไอพีแพ็กเกตออกไป ส่วนในฟิลด์ protocol ซึ่งใช้บ่งบอกโพรโทคอลในชั้นบนนั้น ไมโครคอนโทรลเลอร์จะกำหนดให้ฟิลด์นี้มีค่าเป็น 17 ในกรณีที่ข้อมูลที่ส่งเป็นข้อมูล SIP หรือข้อมูลเสียง ซึ่งมีโพรโทคอล UDP เป็นโพรโทคอลในชั้นเคลื่อนย้ายทั้งคู่ และกำหนดให้มีค่าเป็น 1 ในกรณีที่ข้อมูลที่ต้องการส่งเป็นข้อมูล ICMP ส่วนฟิลด์ Source IP และ Destination IP ไมโครคอนโทรลเลอร์จะนำค่าไอพีแอดเดรสของเครื่อง และไอพีแอดเดรส

ปลายทางเขียนลงในฟิลด์ตามลำดับ จากนั้นไมโครคอนโทรลเลอร์จะคำนวณ checksum จากIP เฮดเดอร์ที่สร้างขึ้น โดยนำข้อมูลภายในเฮดเดอร์มาบวกแบบ 1 complement 16 บิตจนครบทั้งเฮดเดอร์ แล้วค่อยนำค่าที่คำนวณได้นั้นมาใส่ในฟิลด์ checksum เป็นอันสิ้นสุดการสร้าง IP เฮดเดอร์

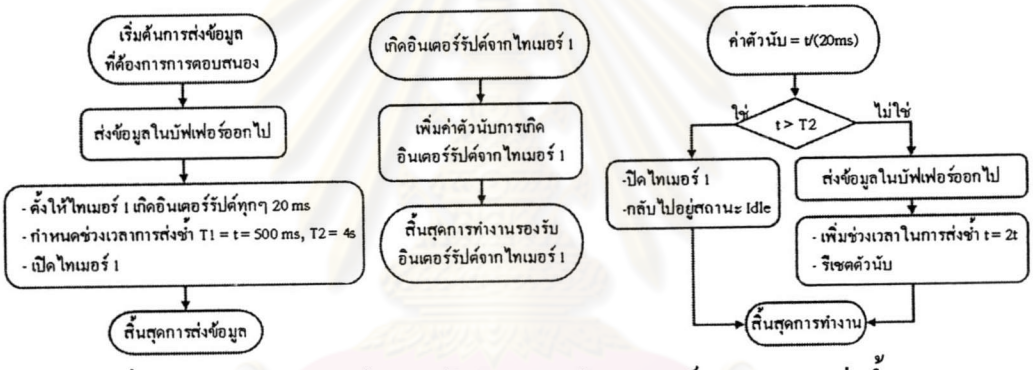
ตารางที่ 3.7 การสร้างฟิลด์ต่าง ๆ ใน IP เฮดเดอร์

ฟิลด์	ขนาด	ค่าที่เขียนลงในฟิลด์	หมายเหตุ
version	4 บิต	4	โพรโคคอล IP เวอร์ชัน 4
header length	4 บิต	5	IP เฮดเดอร์ขนาด 20 ไบต์
type of service	1 ไบต์	0x01	เพื่อการประวิงทางเวลาน้อยที่สุด
total length	2 ไบต์	ตามขนาดของ SIP, Voice	-
identification	2 ไบต์	เพิ่มขึ้นตามจำนวนแพ็กเกตที่ส่ง	-
flag	3 บิต	0	ให้มีการทำ Fragment ได้
fragment offset	13 บิต	0	-
time to live	1 ไบต์	64	-
protocol	1 ไบต์	ICMP=1, UDP=17	-
header checksum	2 ไบต์	ค่าที่ได้จากการคำนวณ checksum	-
source IP	4 ไบต์	ไอพีแอดเดรสของเครื่อง	-
destination IP	4 ไบต์	ไอพีแอดเดรสปลายทาง	-

การสร้างอีเทอร์เนตเฮดเดอร์ ในส่วนของฟิลด์ Destination address ไมโครคอนโทรลเลอร์จะตรวจสอบค่าไอพีแอดเดรสปลายทางซึ่งอยู่ในฟิลด์ Destination IP address ของไอพีเฮดเดอร์ว่าไอพีแอดเดรสของโฮสต์ปลายทางอยู่ในโครงข่ายพื้นที่ท้องถิ่นเดียวกับตัวเครื่องหรือไม่ ในกรณีที่เครื่องโฮสต์ปลายทางอยู่ในโครงข่ายเดียวกัน ไมโครคอนโทรลเลอร์จะใช้ฮาร์ดแวร์แอดเดรสของเครื่องปลายทางใส่ลงในฟิลด์ แต่ในกรณีที่อยู่คนละโครงข่าย ไมโครคอนโทรลเลอร์จะใช้ฮาร์ดแวร์แอดเดรสของเกตเวย์ใส่ลงในฟิลด์แทน ส่วนการสร้างฟิลด์ Source address ไมโครคอนโทรลเลอร์จะใช้ฮาร์ดแวร์แอดเดรสของตัวเครื่องใส่ลงในฟิลด์เท่านั้น และการสร้างฟิลด์ Type นั้นในกรณีที่ข้อมูลที่สร้างเป็น ARP ไมโครคอนโทรลเลอร์จะใช้ค่า 0x80 แต่กรณีที่ข้อมูลที่สร้างเป็น IP ไมโครคอนโทรลเลอร์จะใช้ค่า 0x86 ตามลำดับ

การส่งข้อมูลผ่านโครงข่ายอินเทอร์เนต ข้อมูลอาจสูญหายระหว่างทางได้ ดังนั้นเพื่อเพิ่มความเชื่อถือได้ให้กับข้อมูลที่ส่ง ในการส่งข้อมูลที่ต้องการการตอบสนอง (ข้อมูลร้องขอ ARP, ข้อความ INVITE, CANCEL, BYE, REGISTER, 200 OK, 486 Busy Here, และ 487 Request Canceled) จึงใช้อินเตอร์รัปต์จากไทเมอร์ 1 (Timer 1) จับเวลา และควบคุมการส่งซ้ำด้วยวิธี exponential backoff โดยสามารถแบ่งการควบคุมการส่งซ้ำออกเป็น 3 ส่วนได้แก่ส่วนเริ่มต้นการส่งซ้ำ, ส่วนรองรับการเกิดอินเตอร์รัปต์จากไทเมอร์ 1, และส่วนรองรับการเกิดไทเมอร์ expired ดังแสดงในรูปที่ 3.25 เมื่อมีการส่งข้อมูลที่ต้องการการตอบสนองออกไป ส่วนเริ่มต้นการส่งซ้ำจะตั้งค่าเริ่มต้นให้รีจิสเตอร์ TH1 และ TL1 เป็น 0x0F, 0xFF ตามลำดับ เพื่อกำหนดให้เกิดอินเตอร์รัปต์

จากไทเมอร์ 1 ทุก ๆ 20 ms จากนั้นจึงเปิดไทเมอร์ 1 เพื่อเริ่มจับเวลา ในขณะที่ไทเมอร์ 1 ทำงาน ไมโครคอนโทรลเลอร์สามารถทำงานอื่นได้ตามปกติ แต่ทุกครั้งที่เกิดอินเตอร์รัปต์จากไทเมอร์ 1 ขึ้น ส่วนรองรับการเกิดอินเตอร์รัปต์จากไทเมอร์ 1 จะเพิ่มค่าตัวนับการเกิดอินเตอร์รัปต์ขึ้นทันที และเมื่อทำงานรองรับการเกิดอินเตอร์รัปต์เสร็จแล้ว ไมโครคอนโทรลเลอร์จะกลับไปทำงานอื่นตามปกติ รวมไปถึงการวนตรวจสอบตัวนับจำนวนการเกิดอินเตอร์รัปต์ โดยถ้าค่าตัวนับการเกิดอินเตอร์รัปต์มากกว่าค่าเปรียบเทียบตัวนับที่ตั้งไว้ซึ่งมีค่าเริ่มต้นเป็น 25 เพื่อให้ T1 = 500 ms ส่วนรองรับการเกิดไทเมอร์ expire จะส่งข้อมูลเดิมไปหาปลายทางอีกครั้ง พร้อมกับรีเซตค่าตัวนับ และเพิ่มตัวเปรียบเทียบตัวนับเป็น 2 เท่าจากเดิม และถ้ายังไม่ได้รับการตอบสนองภายในช่วงเวลาที่ตั้งไว้ใหม่อีก ก็จะส่งซ้ำและเพิ่มตัวเปรียบเทียบตัวนับอย่างนี้ไปเรื่อย ๆ จนกว่าจะได้รับการตอบสนอง หรือช่วงเวลาในการส่งซ้ำถูกเพิ่มจนมีค่ามากกว่า T2 = 4 s ซึ่งจะถือว่าไม่สามารถติดต่อกับปลายทางได้ ไมโครคอนโทรลเลอร์ถึงจะปิดไทเมอร์ 1 พร้อมกับยกเลิกการทำงานและกลับมาอยู่ในสถานะ Idle

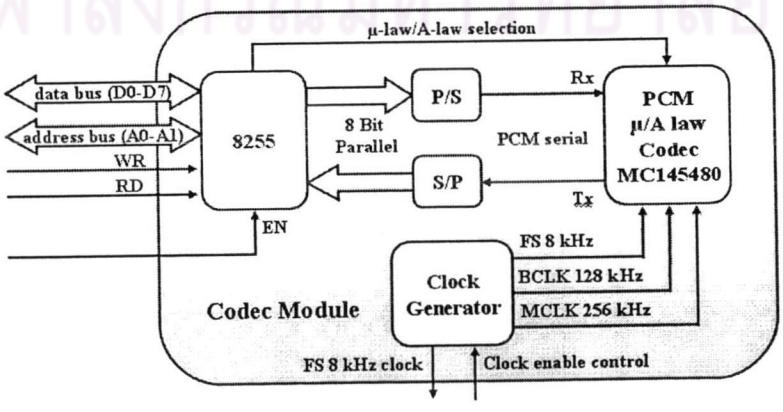


รูปที่ 3.25 การใช้ไทเมอร์ภายในไมโครคอนโทรลเลอร์ควบคุมการส่งซ้ำ

3.6 ส่วนควบคุมข้อมูลเสียง และติดต่อกับชิพเข้ารหัสเสียง PCM

3.6.1 โครงสร้างพื้นฐานของมอดูลเข้ารหัสเสียง

มอดูลเข้ารหัสสัญญาณเสียง PCM ประกอบด้วยวงจรกำเนิดสัญญาณนาฬิกา, ชิพเข้ารหัสสัญญาณเสียง PCM, วงจรแปลงการส่งข้อมูลอนุกรมเป็นขนาน (S/P), วงจรแปลงการส่งข้อมูลขนานเป็นอนุกรม (P/S), และชิพขยายพอร์ต 8255 ดังแสดงในรูปที่ 3.26



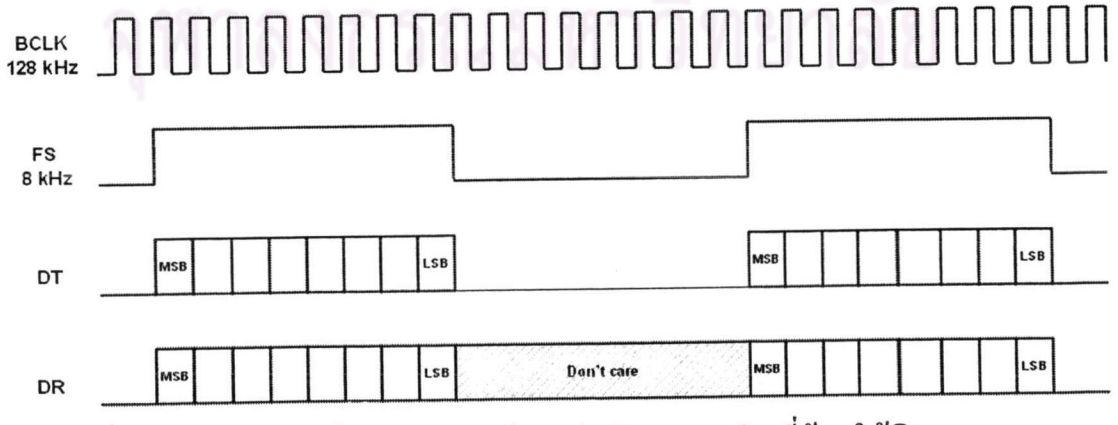
รูปที่ 3.26 โครงสร้างมอดูลเข้ารหัสสัญญาณเสียงแบบ PCM

ชิพเข้ารหัสสัญญาณเสียง MC145480 ของบริษัท Motorola สามารถเข้ารหัสสัญญาณเสียงแบบ PCM ได้ทั้ง μ-law และ A-law ภายในชิพมีวงจรขยายสัญญาณแอนะลอก และฟิลเตอร์กรองสัญญาณความถี่ต่ำในตัวทำให้สามารถต่อแฮนด์เซตได้โดยตรง ในส่วนดิจิทัลตัวชิพสามารถส่งและรับข้อมูลได้หลายรูปแบบขึ้นอยู่กับสัญญาณนาฬิกาที่ป้อนให้ มีรายละเอียดดังนี้

สัญญาณนาฬิกา Master Clock (MCLK) เป็นสัญญาณนาฬิกาหลักที่ป้อนเข้าในขา MCLK ของตัวชิพ เพื่อใช้ในกระบวนการ ADC (Analog to Digital Converter) และ DAC (Digital to Analog Converter) ในการใช้งานสามารถใช้สัญญาณนาฬิกาความถี่ 256, 512, 1536, 2048, 2560, หรือ 4096 kHz ก็ได้ ในกรณีที่ใช้สัญญาณนาฬิกาความถี่ 256 kHz และ 512 kHz นั้น สัญญาณนาฬิกา MCLK จำเป็นต้องซิงโครไนซ์กับสัญญาณนาฬิกา FS เพื่อให้ชิพทำงานได้อย่างมีประสิทธิภาพ ในการพัฒนาเครื่องโทรศัพท์อินเทอร์เน็ตได้เลือกใช้สัญญาณนาฬิกาความถี่ 256 kHz ซึ่งเป็นสัญญาณความถี่ต่ำสุดที่ชิพสามารถทำงานได้ ทั้งนี้เพื่อหลีกเลี่ยงปัญหาจากสัญญาณความถี่สูง ซึ่งอาจทำให้วงจรไม่เสถียรได้

สัญญาณนาฬิกา Bit Clock (BCLK) เป็นสัญญาณนาฬิกาที่ใช้ควบคุมอัตราการรับส่งข้อมูล PCM ของตัวชิพ โดยในการใช้งานนั้นตัวชิพสามารถรับสัญญาณนาฬิกา BCLK ความถี่ตั้งแต่ 64 kHz จนถึง 4.096 MHz ได้ แต่ในการพัฒนาไม่เลือกใช้สัญญาณนาฬิกา BCLK ความถี่ 64 kHz เนื่องจากในการรับข้อมูลเสียงจากไมโครคอนโทรลเลอร์มายังตัวชิพนั้น จำเป็นต้องผ่านกระบวนการแปลงข้อมูล P/S (Parallel to Serial) ก่อน ซึ่งจากสัญญาณนาฬิกาที่มีในวงจรนั้น ในการทำ P/S ในขั้นตอนการโหลดข้อมูลแบบขนานเข้ามาในชิฟต์รีจิสเตอร์นั้น จำเป็นต้องใช้เวลาช่วงหนึ่งต่างหาก จากเวลาที่ใช้ในการส่งข้อมูลอนุกรมออกไป จึงทำให้ไม่สามารถสร้างสัญญาณ 64 kbps ที่มีอัตราข้อมูลต่อเนื่องมาให้กับชิพเข้ารหัสสัญญาณเสียงได้ ดังนั้นเพื่อให้เป็นการง่าย และไม่จำเป็นต้องเพิ่มอุปกรณ์จึงเลือกใช้สัญญาณนาฬิกา BCLK ที่ความถี่ 128 kHz แทน

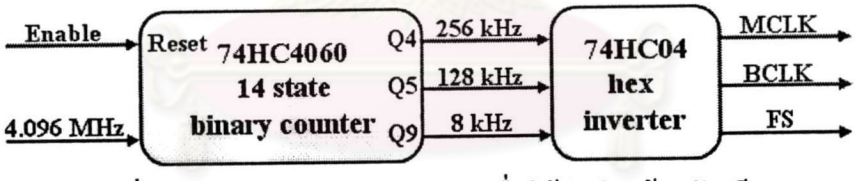
สัญญาณนาฬิกา Frame Sync (FS) เป็นสัญญาณนาฬิกาความถี่ 8 kHz ใช้ในการซิงโครไนซ์เฟรมข้อมูลในแต่ละเฟรม โดยตัวชิพ MC145480 จะรับส่งเฟรมข้อมูล 8 บิตทุก ๆ คาบเมื่อสัญญาณนาฬิกา FS เป็นลอจิก '1' ดังแสดงในรูปที่ 3.27



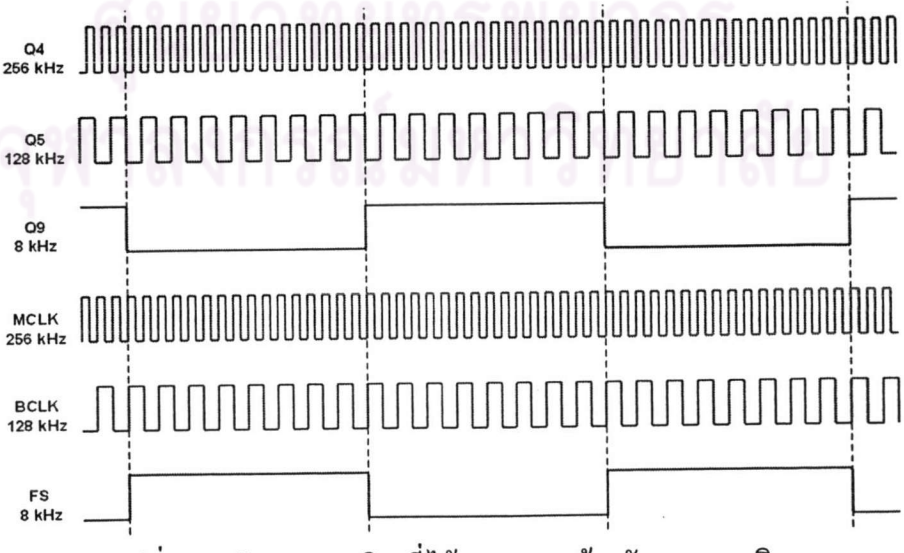
รูปที่ 3.27 ความสัมพันธ์ของการรับส่งข้อมูลกับสัญญาณนาฬิกาที่ป้อนให้ชิพ MC145480

เนื่องจากข้อจำกัดทางด้านความเร็วในการทำงานของไมโครคอนโทรลเลอร์ จึงทำให้ไมโครคอนโทรลเลอร์ไม่สามารถรับส่งข้อมูลแบบอนุกรมกับชิพ MC145480 ได้โดยตรง ดังนั้นจึงต้องใช้เทคนิคการแปลง S/P และ P/S เพื่อให้ไมโครคอนโทรลเลอร์รับส่งข้อมูลแบบขนาน 8 บิตต่อทิศทางแทน ซึ่งต้องใช้พอร์ตถึง 16 พอร์ตแยกต่างหากจากสายสัญญาณข้อมูล, สายสัญญาณกำหนดตำแหน่งหน่วยความจำ, และสายสัญญาณควบคุม ซึ่งพอร์ตของไมโครคอนโทรลเลอร์นั้นไม่เพียงพอ จึงต้องใช้ชิพ 8255 ซึ่งเป็นชิพขยายพอร์ต I/O เข้ามาเพิ่มด้วย

วงจรกำเนิดสัญญาณนาฬิกาภายในมอดูลเข้ารหัสสัญญาณเสียงทำหน้าที่สร้างสัญญาณนาฬิกาความถี่ 8, 128, และ 256 kHz ใช้ในการทำงานของชิพเข้ารหัสสัญญาณเสียง, กระบวนการแปลง S/P, กระบวนการแปลง P/S, และซิงโครไนซ์เพื่อรับส่งข้อมูลกับไมโครคอนโทรลเลอร์ โดยในการสร้างสัญญาณนาฬิกาได้ใช้ชิพ 74HC4060 ซึ่งเป็นชิพวงจรนับแบบไบนารี 14 บิต (14 state binary counter) ภายในชิพมีวงจรออสซิลเลเตอร์ในตัว ซึ่งทำให้สามารถต่อกับคริสตอลได้โดยตรง โดยในการพัฒนาได้เลือกใช้คริสตอลความถี่ 4.096 MHz เพื่อให้สามารถหารความถี่ที่ต้องการได้อย่างลงตัว โดยจะได้สัญญาณนาฬิกา 256 kHz จากเอาต์พุต Q4 (หาร 16), สัญญาณนาฬิกา 128 kHz จากเอาต์พุต Q5 (หาร 32), และสัญญาณนาฬิกา 8 kHz จากเอาต์พุต Q9 (หาร 512) ตามลำดับ แต่เนื่องจากสัญญาณนาฬิกาที่ได้มีการซิงโครไนซ์ที่ขอบขาลงของสัญญาณนาฬิกาที่มีความถี่สูงกว่า แต่ชิพเข้ารหัสเสียงต้องใช้สัญญาณนาฬิกาที่มีการซิงโครไนซ์ ช่วงขอบขาขึ้น ดังนั้นจึงต้องนำสัญญาณที่ได้มาผ่านชิพ 74HC04 ซึ่งเป็นอินเวอร์เตอร์กลับบิตสัญญาณดังแสดงในรูปที่ 3.28 เพื่อให้ได้สัญญาณนาฬิกาที่มีรูปแบบการซิงโครไนซ์ที่ถูกต้องดังแสดงในรูปที่ 3.29

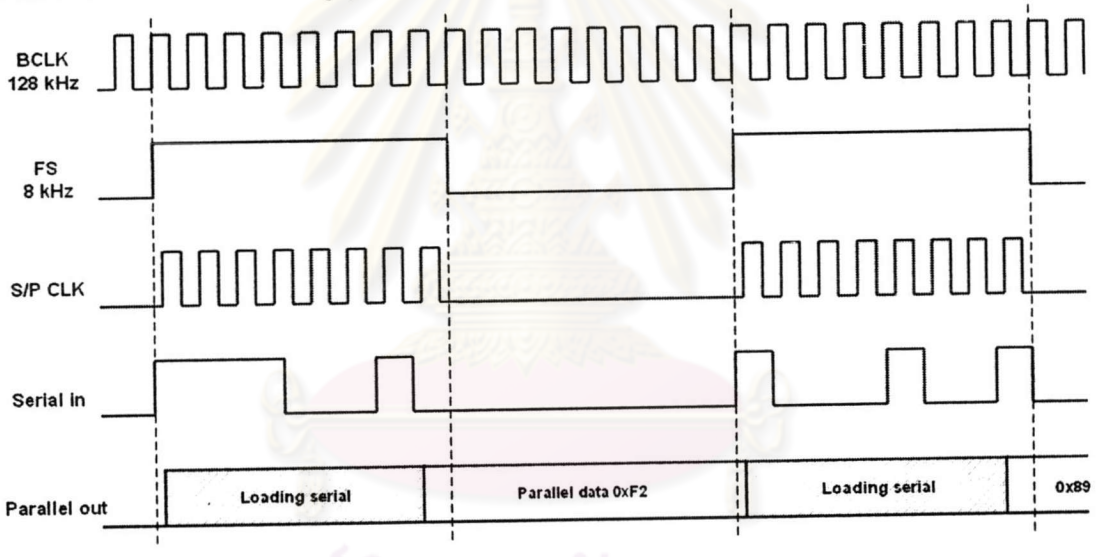


รูปที่ 3.28 การสร้างสัญญาณนาฬิกาเพื่อใช้กับชิพเข้ารหัสเสียง



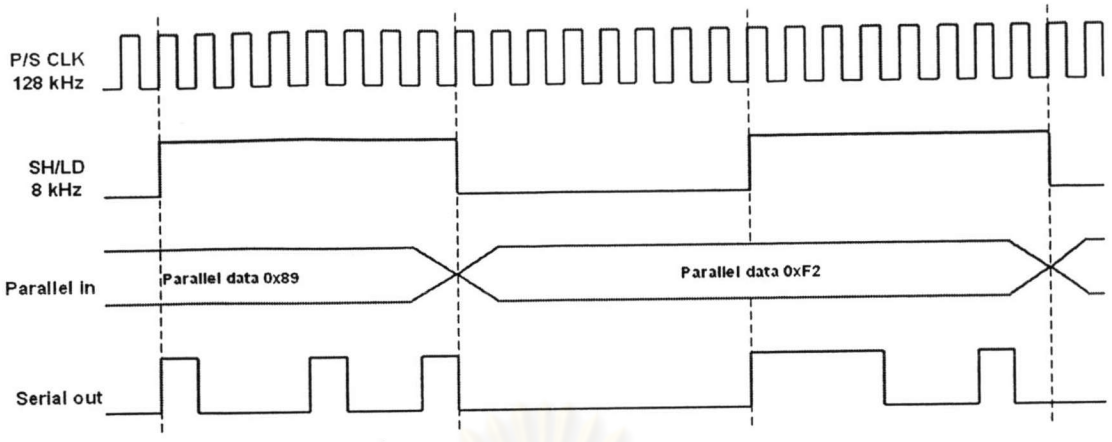
รูปที่ 3.29 สัญญาณนาฬิกาที่ได้จากวงจรสร้างสัญญาณนาฬิกา

การแปลง S/P ในการพัฒนาได้ใช้ชิพ 74HC164 ตัวชิพประกอบด้วยพอร์ตอินพุตแบบอนุกรม, พอร์ตเอาต์พุตแบบขนาน 8 บิต, พอร์ตรับสัญญาณนาฬิกา, และพอร์ตรับสัญญาณควบคุม ในการใช้งานต้องป้อนสัญญาณนาฬิกาเพื่อควบคุมอัตราการรับข้อมูลแบบอนุกรม โดยตัวชิพจะเก็บข้อมูลที่ได้รับจากพอร์ตอินพุตอนุกรมเข้ามาเก็บในชิฟต์รีจิสเตอร์พร้อมกับเลื่อนข้อมูลเก่าไปยังรีจิสเตอร์ถัดไปครั้งละบิต ทุก ๆ ขอบขาขึ้นของสัญญาณนาฬิกา ดังนั้นในการแปลงข้อมูลอนุกรมที่ได้จากชิพเข้ารหัสเสียงให้อยู่ในรูปของข้อมูลขนานนั้น จึงต้องป้อนสัญญาณนาฬิกาความถี่ 128 kHz ซึ่งเท่ากับความถี่ของสัญญาณนาฬิกา BCLK เพื่อให้ชิพ 74HC164 รับข้อมูลอนุกรมเข้ามาเก็บในชิฟต์รีจิสเตอร์ครั้งละบิตได้อย่างถูกต้อง และเพื่อเป็นการสะดวกในการอ่านค่าข้อมูลแบบขนาน 8 บิตจากชิพ 74HC164 ผ่านชิพ 8255 จึงจ่ายสัญญาณนาฬิกาให้กับชิพ 74HC164 เฉพาะช่วงที่สัญญาณ FS มีลอจิกเป็น '1' เพื่อให้ตัวชิพ 74HC164 นั้นค้างค่าข้อมูลออกทางพอร์ตขนานในช่วงเวลาที่สัญญาณ FS มีลอจิกเป็น '0' ไว้ดังแสดงในรูปที่ 3.30 ซึ่งสัญญาณนาฬิกา S/P CLK สามารถสร้างได้จากการนำสัญญาณ BCLK มาแอนด์ (AND) กับสัญญาณ FS นั่นเอง



รูปที่ 3.30 สัญญาณบนชิพ S/P 74HC164

การแปลง P/S ในการพัฒนาได้ใช้ชิพ 74HC165 ตัวชิพประกอบด้วยพอร์ตอินพุตแบบขนาน 8 บิต, พอร์ตเอาต์พุตอนุกรม, พอร์ตรับสัญญาณนาฬิกา, และพอร์ตรับสัญญาณควบคุม ในการใช้เมื่อระดับสัญญาณที่พอร์ต SH/LD (Shift/Load) มีลอจิกเป็น '0' ชิพจะอ่านค่าจากพอร์ตขนาน 8 บิตเข้ามาเก็บในชิฟต์รีจิสเตอร์ และเมื่อสัญญาณ SH/LD มีระดับลอจิกเป็น '1' ชิพจะส่งข้อมูลในชิฟต์รีจิสเตอร์นั้นออกไปทางพอร์ตอนุกรมครั้งละบิต ทุก ๆ ขอบขาขึ้นของสัญญาณนาฬิกาที่ป้อนให้กับชิพนั้น ดังนั้นในการใช้งานเพื่อให้ชิพ 74HC165 ส่งข้อมูลให้กับชิพเข้ารหัสเสียงได้อย่างถูกต้อง จึงต้องใช้สัญญาณนาฬิกา BCLK ความถี่ 128 kHz ป้อนเป็นสัญญาณนาฬิกาให้กับตัวชิพ 74HC164 และใช้สัญญาณ FS 8 kHz เป็นสัญญาณควบคุม SH/LD ดังแสดงในรูปที่ 3.31



รูปที่ 3.31 สัญญาณบนชิพ P/S 74HC165

ชิพ 8255 เป็นชิพที่ใช้ในการขยายพอร์ต I/O โดยตัวชิพจะมีพอร์ต I/O ขนาด 8 บิต 3 พอร์ต สามารถต่อกับไมโครคอนโทรลเลอร์แบบ Memory map ได้ ซึ่งในการพัฒนาได้ทำ Memory map ให้ชิพ 8255 อยู่ที่ตำแหน่ง C000H-C003H และมีรายละเอียดดังแสดงในตารางที่ 3.8

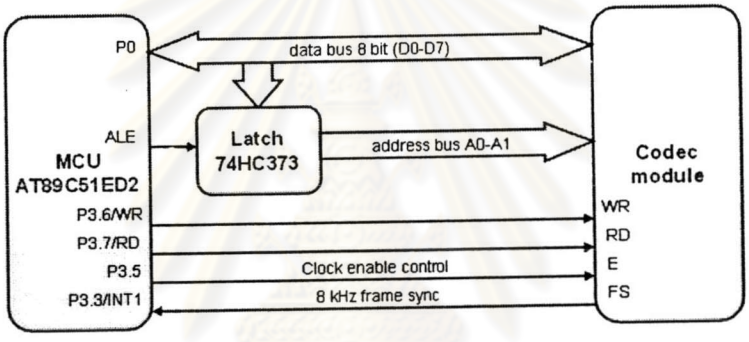
ตารางที่ 3.8 ตำแหน่งการทำ Memory map บนชิพ 8255 และการใช้งาน

A1	A0	ตำแหน่ง (A14 = 1, A15 = 1)	การใช้งาน
0	0	C000H	พอร์ต A
0	1	C001H	พอร์ต B
1	0	C002H	พอร์ต C
1	1	C003H	รีจิสเตอร์ควบคุมการทำงานของ 8255

ชิพ 8255 นั้นสามารถทำงานได้ 3 โมด โดยสามารถเลือกโมดการทำงานได้จากการตั้งค่าในรีจิสเตอร์ควบคุมการทำงานภายในชิพ 8255 ดังแสดงในตารางที่ 3.9 ในโมด 0 ทุกพอร์ตของ 8255 เป็นอินพุตหรือเอาต์พุตโดยต้องกำหนดว่าพอร์ตไหนเป็นอินพุต หรือเอาต์พุต และไม่สามารถกำหนดให้พอร์ตหนึ่งเป็นทั้งอินพุต และเอาต์พุตในเวลาเดียวกันได้ ในโมด 2 พอร์ต A และพอร์ต B เป็นอินพุต หรือเอาต์พุตโดยใช้ไบต์ต่ำของพอร์ต C และไบต์สูงของพอร์ต C เป็น Handshaking ให้กับพอร์ต A และพอร์ต B ตามลำดับ ส่วนในโมด 2 พอร์ต A เป็นทั้งอินพุต และเอาต์พุตแบบสองทิศทาง โดยใช้พอร์ต B กับพอร์ต C ในการตรวจสอบการทำงาน ซึ่งในการพัฒนานั้นได้กำหนดค่าภายในรีจิสเตอร์ควบคุมเป็น 0x82 นั่นคือให้ชิพ 8255 ทำงานในโมด 0 เพื่อใช้งานทั้ง 3 พอร์ตโดยให้พอร์ต A เป็นเอาต์พุตเพื่อส่งข้อมูลเสียง PCM แบบขนาน 8 บิตไปยังวงจร P/S และพอร์ต B เป็นเอาต์พุตเพื่อรับข้อมูลเสียง PCM แบบขนาน 8 บิตจากวงจร S/P ตามลำดับ ส่วนพอร์ต C บิต 0 ต่อกับพอร์ตเลือกรูปแบบการเข้ารหัสของชิพ MC145480 โดยถ้าเป็นลอจิก '1' MC145480 จะเข้ารหัสแบบ μ-law และถ้าเป็นลอจิก '0' จะเข้ารหัสแบบ A-law ส่วนบิต 1 ของพอร์ต C ใช้สำหรับควบคุม Buzzer เพื่อสร้างเสียงกระดิ่งในกรณีที่มีการร้องขอการเปิดเซสชันจากเครื่องลูกข่ายอื่น

ตารางที่ 3.9 การกำหนดค่ารีจิสเตอร์ควบคุมเพื่อควบคุมการทำงานของ 8255

ตำแหน่งบิต	การทำงาน
0	กำหนดการทำงานของพอร์ต C ล่างโดย '0' = เอาต์พุต และ '1' = อินพุต
1	กำหนดการทำงานของพอร์ต B โดย '0' = เอาต์พุต และ '1' = อินพุต
2	กำหนดโมดการทำงานของพอร์ต B โดย '0' = โมด 0 และ '1' = โมด 1
3	กำหนดการทำงานของพอร์ต C บนโดย '0' = เอาต์พุต และ '1' = อินพุต
4	กำหนดการทำงานของพอร์ต A โดย '0' = เอาต์พุต และ '1' = อินพุต
5	ใช้กำหนดโมดการทำงานของพอร์ต A โดย
6	"00" = โมด 0, "01" = โมด 1, "10", หรือ "11" = โมด 2
7	กำหนดการทำงานของพอร์ตทุกพอร์ตโดย '0' = ไม่ทำงาน และ '1' = ทำงาน



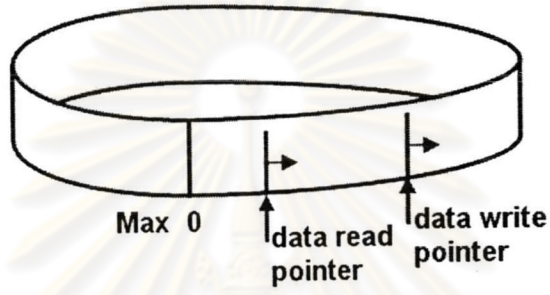
รูปที่ 3.32 การต่อไมโครคอนโทรลเลอร์เข้ากับมอดูลเข้ารหัสสัญญาณเสียง

ในการต่อกับมอดูลเข้ารหัสสัญญาณเสียงนั้น นอกจากใช้สายสัญญาณข้อมูล, สายสัญญาณกำหนดตำแหน่งหน่วยความจำ, และสายสัญญาณควบคุมแล้ว ยังใช้พอร์ต 3.5 ควบคุมการเปิดปิดวงจรสร้างสัญญาณนาฬิกา และพอร์ต 3.3 รับสัญญาณซิงโครไนซ์เฟรมจากมอดูลเข้ารหัสสัญญาณเสียงด้วย ดังแสดงในรูปที่ 3.32

3.6.2 การควบคุมข้อมูลเสียงภายในบัฟเฟอร์ภาคส่ง และภาครับ

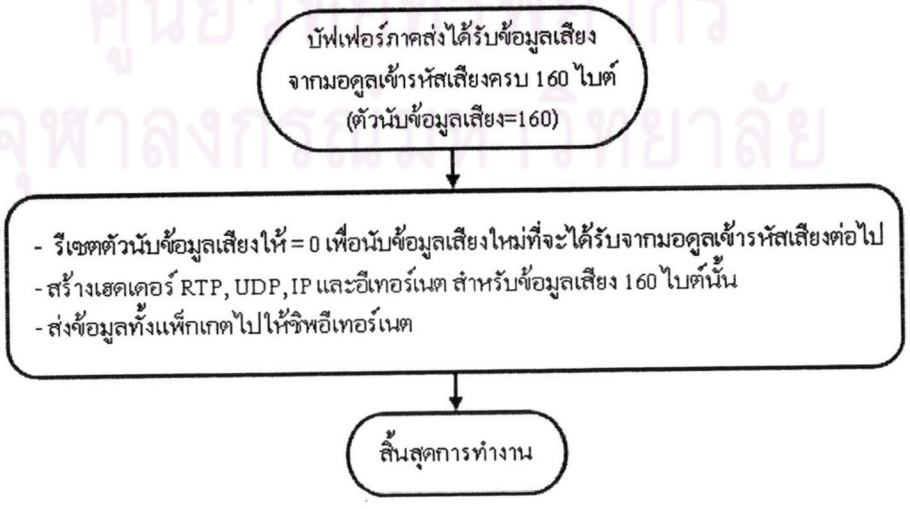
ในการรับส่งข้อมูลเสียงภายในเครื่องโทรศัพท์นั้น มอดูลเข้ารหัสเสียงจะไม่ได้รับส่งข้อมูลกับชิพอีเทอร์เนตโดยตรง ในภาคส่งไมโครคอนโทรลเลอร์จะนำข้อมูลเสียงที่ได้จากมอดูลเข้ารหัสสัญญาณเสียงมาเก็บภายในหน่วยความจำภายนอกเพื่อรอให้ครบ 160 ไบต์ก่อน จากนั้นจึงนำไปเข้าแพ็กเกต UDP/IP แล้วจึงส่งไปให้ชิพอีเทอร์เนต และเช่นเดียวกันในภาครับ ไมโครคอนโทรลเลอร์จะนำข้อมูลเสียงที่ได้จากชิพอีเทอร์เนตมาเก็บในบัฟเฟอร์ภายในหน่วยความจำภายนอกเพื่อลดผลของ Jitter ก่อนแล้วจึงค่อยส่งข้อมูลเสียงนั้นไปให้มอดูลถอดรหัสเสียงต่อไป ดังนั้นการควบคุมการรับส่งข้อมูลเสียงภายในเครื่องโทรศัพท์สามารถแบ่งออกเป็น 2 ส่วนได้แก่ การรับส่งข้อมูลเสียงระหว่างชิพอีเทอร์เนต RTL8019as กับบัฟเฟอร์หน่วยความจำข้อมูลภายนอก และการรับส่งข้อมูลเสียงระหว่างหน่วยความจำข้อมูลภายนอกกับมอดูลเข้ารหัสเสียง

เนื่องจากข้อมูลเสียงนั้นเป็นข้อมูลที่มีการรับส่งอย่างต่อเนื่อง ในการพัฒนาจึงได้ใช้การบัฟเฟอร์แบบเป็นวง เพื่อให้การควบคุมการอ่าน และการเขียนข้อมูลเสียงภายในบัฟเฟอร์ทำได้สะดวก และต่อเนื่อง ในการเขียนข้อมูลไมโครคอนโทรลเลอร์จะเขียนข้อมูลลงบัฟเฟอร์พร้อมกับปรับค่าตัวชี้ตำแหน่งที่ได้เขียนไล่ไปเรื่อย ๆ จนเต็ม แล้ววนกลับมาที่ตำแหน่งเริ่มต้นใหม่อีกครั้งซ้ำดังนี้ไปเรื่อย ๆ ในการอ่านก็เช่นกัน ไมโครคอนโทรลเลอร์จะอ่านข้อมูลจากบัฟเฟอร์พร้อมกับปรับค่าตัวชี้ตำแหน่งที่ได้อ่านไล่ตามการเขียนวนไปเรื่อยดังแสดงในรูปที่ 3.33 ในการพัฒนาได้ใช้บัฟเฟอร์รับข้อมูลเสียงขนาด 640 ไบต์ และบัฟเฟอร์ส่งข้อมูลเสียงขนาด 320 ไบต์ตามลำดับ



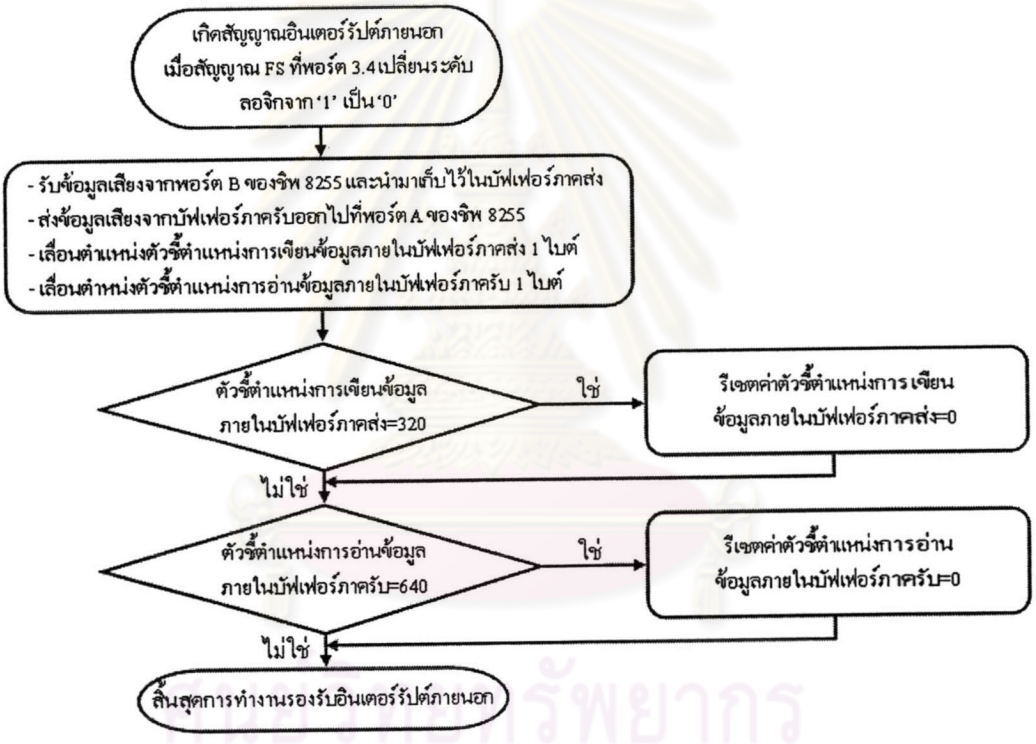
รูปที่ 3.33 การบัฟเฟอร์ข้อมูลเสียงภายในหน่วยความจำภายนอก

ในการควบคุมการรับส่งข้อมูลเสียงระหว่างหน่วยความจำภายนอก กับชิพอีเทอร์เนต ในภาคส่งเมื่อไมโครคอนโทรลเลอร์นำข้อมูลจากมอดูลเข้ารหัสเสียง มาเก็บในบัฟเฟอร์หน่วยความจำข้อมูลภายนอกจนครบทุก ๆ 160 ไบต์ ไมโครคอนโทรลเลอร์จะสร้าง RTP เฮดเดอร์ และ UDP/IP เฮดเดอร์สำหรับข้อมูลเสียงนั้นแล้วจึงค่อยส่งข้อมูลทั้งแพ็กเกตนั้นต่อไปยังชิพอีเทอร์เนตดังแสดงในรูปที่ 3.34 ส่วนในภาครับเมื่อได้รับข้อมูลจากชิพอีเทอร์เนต ไมโครคอนโทรลเลอร์จะตรวจสอบฟิลด์ destination port ภายใน UDP เฮดเดอร์ โดยถ้ามีค่าเป็น 49170 ซึ่งเป็นพอร์ตที่มีการตกลงไว้ก่อนเปิดเซสชันให้เป็นพอร์ตสำหรับรับข้อมูลเสียง ไมโครคอนโทรลเลอร์จะนำข้อมูลเสียงภายในแพ็กเกตนั้นเขียนลงในบัฟเฟอร์ภาครับ เพื่อรอการอ่านไปให้ชุดมอดูลเข้ารหัสเสียงต่อไป



รูปที่ 3.34 การควบคุมการส่งข้อมูลเสียงให้ชิพอีเทอร์เนต

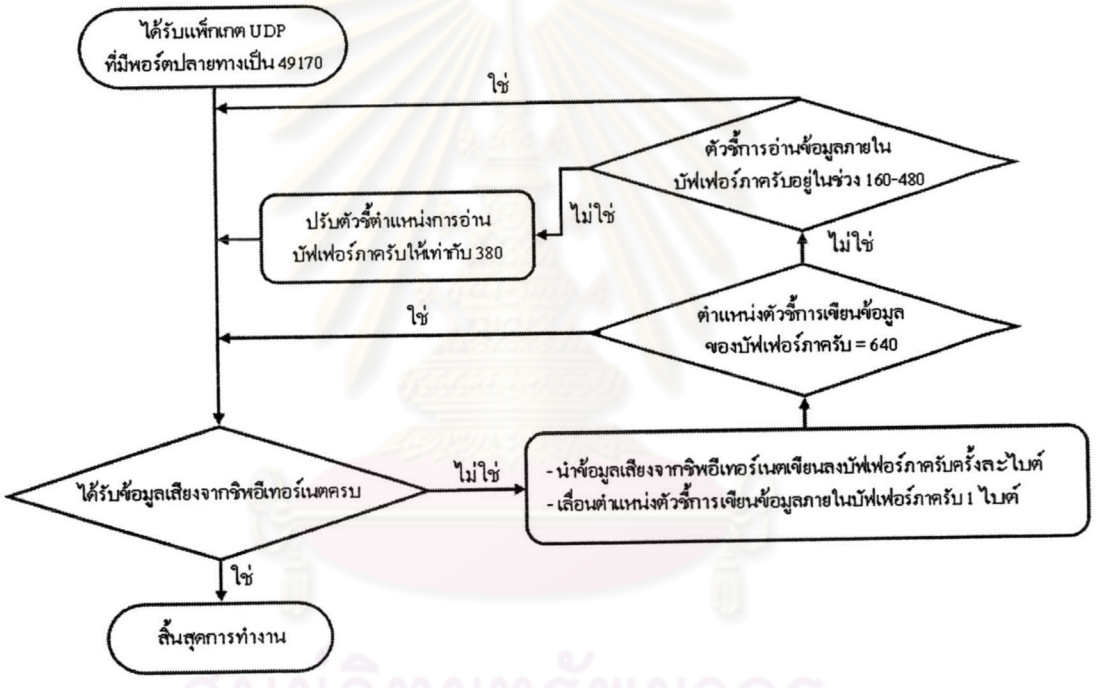
ในการควบคุมการรับส่งข้อมูลระหว่างหน่วยความจำภายนอก กับมอดูลเข้ารหัสเสียง เนื่องจากการทำงานภายในมอดูลเข้ารหัสเสียงเป็นแบบซิงโครไนซ์กันทั้งหมด ดังนั้น ไมโครคอนโทรลเลอร์จึงต้องควบคุมการรับส่งข้อมูลระหว่างหน่วยความจำภายนอก กับชิพ 8255 เป็นจังหวะซิงโครไนซ์กับตัวมอดูลด้วย ซึ่งในการซิงโครไนซ์นี้ ไมโครคอนโทรลเลอร์จะใช้ขอบขาลงของสัญญาณนาฬิกา FS เป็นสัญญาณอินเตอร์รัปต์จากภายนอก เพื่อควบคุมจังหวะการอ่านและเขียนข้อมูลลงพอร์ต B และ A ของชิพ 8255 ตามลำดับ ทำให้ในภาคส่งข้อมูลเสียงที่ได้รับจากพอร์ต B ของชิพ 8255 จะถูกเขียนลงในวงบัฟเฟอร์ขาส่งครั้งละไบต์ อย่างสม่ำเสมอด้วยอัตราข้อมูล 8000 ไบต์/วินาที ส่วนในภาครับข้อมูลเสียงจะถูกอ่านจากวงบัฟเฟอร์ขารับ เพื่อส่งออกไปที่พอร์ต A ของชิพ 8255 ครั้งละไบต์ อย่างสม่ำเสมอด้วยอัตราข้อมูล 8000 ไบต์/วินาทีเช่นกันดังแสดงในรูปที่ 3.35



รูปที่ 3.35 การควบคุมการรับส่งข้อมูลระหว่างหน่วยความจำ กับมอดูลเข้ารหัสเสียง

เนื่องจากการส่งข้อมูลเสียงผ่านโครงข่ายอินเทอร์เนตนั้น แพ็กเกตแต่ละแพ็กเกตจะใช้เวลาในการเดินทางผ่านโครงข่ายไม่เท่ากัน ทำให้ข้อมูลเสียงที่ได้รับที่ปลายทางมี Jitter และเกิดเสียงที่เล่นออกไปแบบไม่ต่อเนื่อง ในทางปฏิบัติสามารถลด Jitter ที่เกิดขึ้นนี้ได้โดยการบัฟเฟอร์ข้อมูลที่ได้รับมาในช่วงเวลาหนึ่งก่อน แล้วจึงค่อยเล่นข้อมูลนั้นออกไป โดยถ้ายิ่งมีบัฟเฟอร์ข้อมูลมากขึ้นเพียงไรก็ยิ่งสามารถลด Jitter ได้มากขึ้นเพียงนั้น แต่จะมีผลเสียคือเกิดการประวิงทางเวลาเกิดขึ้น ในการพัฒนาเครื่องโทรศัพท์ ไมโครคอนโทรลเลอร์จะบัฟเฟอร์ข้อมูลที่ได้รับจากชิพอีเทอร์เนตก่อน 320 ไบต์แล้วจึงค่อยส่งข้อมูลนั้นออกไปให้มอดูลเข้ารหัสสัญญาณเสียง ซึ่งในการบัฟเฟอร์ข้อมูลนี้ทำให้เกิดการประวิงเวลาขึ้น 40 ms

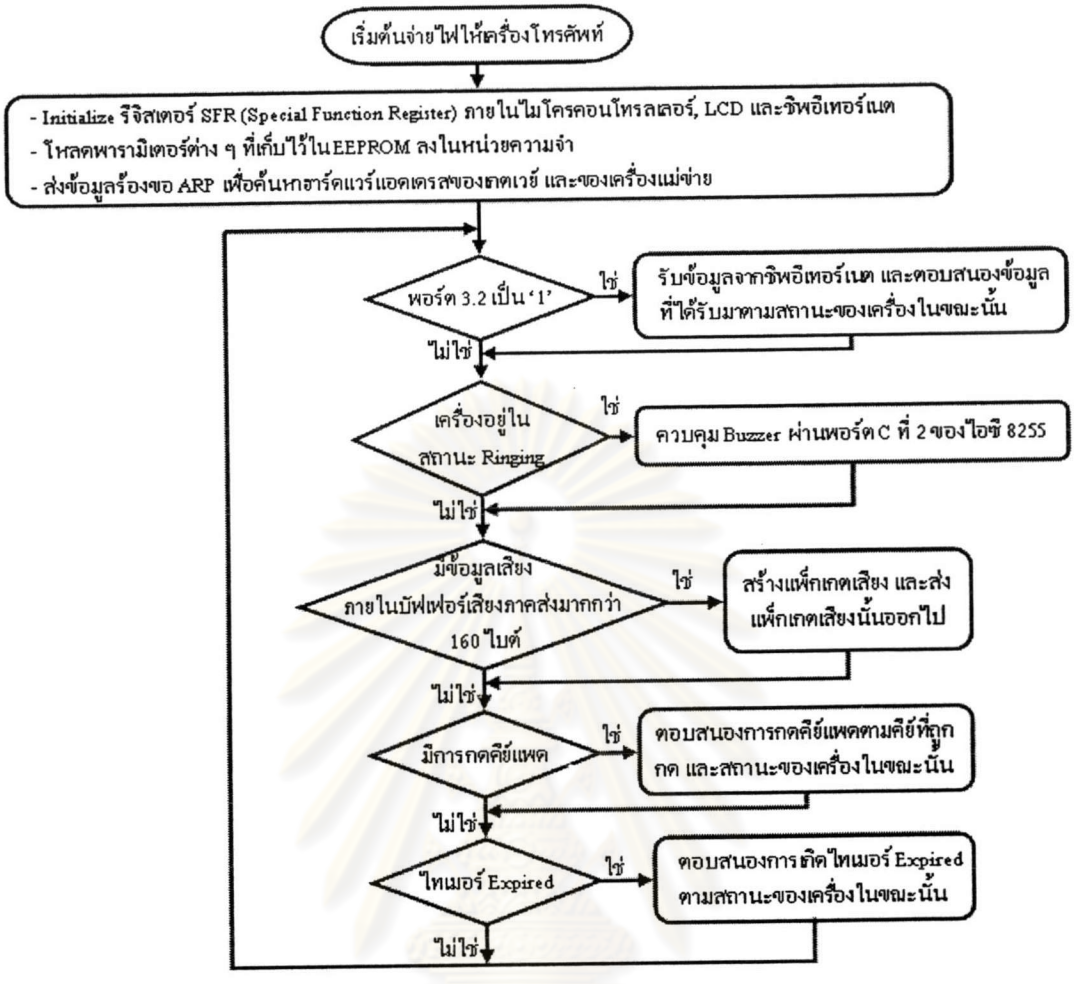
ในการรับส่งข้อมูลแบบเวลาจริง นอกจากปัญหา Jitter แล้ว ยังเกิดปัญหาที่เกิดจากการขยับเลื่อนของสัญญาณนาฬิกา (Clock drift) ระหว่างต้นทาง กับปลายทาง ซึ่งทำให้อัตราข้อมูลเสียงที่ได้รับกับอัตราการเล่นข้อมูลเสียงไม่เท่ากัน มีผลให้การอ่าน และการเขียนข้อมูลในวงบัฟเฟอร์เกิดการซ้อนทับกันได้ ซึ่งในการพัฒนาได้แก้ปัญหาโดยเมื่อได้รับข้อมูลเสียงจากชิพอีเทอร์เนตจนครบรอบวงบัฟเฟอร์ทุกรอบ ไมโครคอนโทรลเลอร์จะตรวจสอบตำแหน่งตัวชี้ข้อมูลเสียงที่ได้อ่านไปว่ายังอยู่ในช่วงที่ยอมรับได้ (160-480) หรือไม่ ถ้าตัวชี้ตำแหน่งข้อมูลเสียงที่ได้อ่านไปไม่อยู่ในช่วงที่ยอมรับได้ แสดงว่าเกิดการขยับเลื่อนของสัญญาณนาฬิกามาก และอาจเกิดการซ้อนทับของข้อมูลภายในวงบัฟเฟอร์ ไมโครคอนโทรลเลอร์จะปรับตัวชี้ข้อมูลที่อ่านให้อยู่ในตำแหน่งที่ไบต์ที่ 320 ซึ่งเป็นตำแหน่งที่ถูกต้องอีกครั้งดังแสดงในรูปที่ 3.36



รูปที่ 3.36 การควบคุมการรับข้อมูลเสียงจากชิพอีเทอร์เนต

3.7 การควบคุมการทำงานโดยรวม

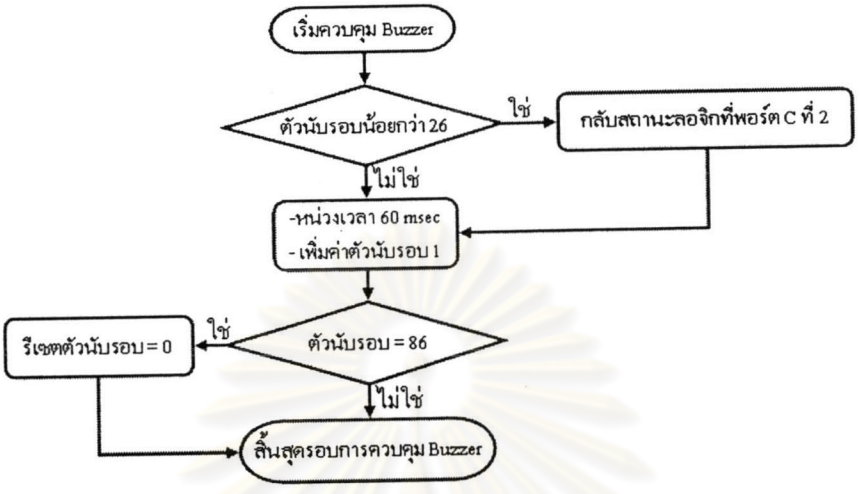
ในการควบคุมโดยรวมไมโครคอนโทรลเลอร์จะวนตรวจสอบ อุปกรณ์ภายนอก และตัวแปรที่ใช้ระบุสถานการณ์การทำงานของเครื่องโทรศัพท์ พร้อมกับตอบสนองผลที่ได้จากการตรวจสอบนั้น ๆ ไปเรื่อย ๆ และเมื่อเกิดอินเตอร์รัปต์จากภายนอก หรืออินเตอร์รัปต์จากไทเมอร์ ไมโครคอนโทรลเลอร์จะไปทำงานรองรับการเกิดอินเตอร์รัปต์นั้นก่อนจนเสร็จ แล้วจึงกลับมาวนตรวจสอบอุปกรณ์ภายนอก และตัวแปรที่ใช้ระบุการทำงานต่อไป



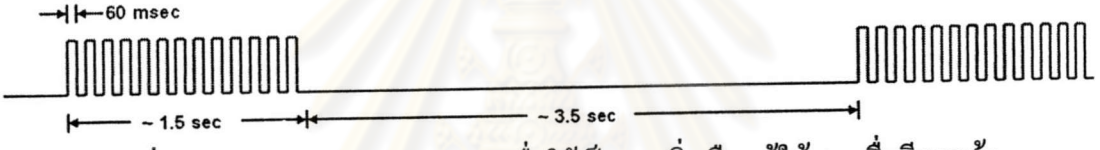
รูปที่ 3.37 การควบคุมการทำงานโดยรวม

เมื่อเริ่มจ่ายไฟให้กับเครื่องโทรศัพท์ดังแสดงในรูปที่ 3.37 ไมโครคอนโทรลเลอร์จะ Initialize รีจิสเตอร์ SFR, LCD, และชิพอีเทอร์เนตให้พร้อมใช้งาน จากนั้นจึงโหลดพารามิเตอร์ต่าง ๆ ที่เก็บไว้ใน EEPROM ลงในหน่วยความจำ แล้วค่อยส่งข้อมูลร้องขอ ARP เพื่อค้นหาฮาร์แวร์แอดเดรสของเกตเวย์ และของเครื่องแม่ข่าย เมื่อ Initialize ฮาร์ดแวร์ และโหลดข้อมูลเรียบร้อยแล้ว ไมโครคอนโทรลเลอร์จะตรวจสอบพอร์ต 3.2 ซึ่งต่ออยู่กับขาสัญญาณอินเตอร์รัปต์ของชิพอีเทอร์เนตนั้น เพื่อตรวจจับการรับข้อมูลจากภายนอกผ่านชิพอีเทอร์เนต โดยถ้าพอร์ต 3.2 มีลอจิกเป็น '1' ไมโครคอนโทรลเลอร์จะรับข้อมูลจากชิพอีเทอร์เนต และนำข้อมูลที่ได้รับมาจำแนก และประมวลผลเพื่อตอบสนองข้อมูลนั้นต่อไป จากนั้นไมโครคอนโทรลเลอร์จะตรวจสอบสถานะของเครื่อง เพื่อควบคุมพอร์ต C ที่ 2 ของชิพ 8255 ซึ่งต่อเข้ากับ Buzzer และใช้เป็นกระดิ่งสำหรับเตือนผู้ใช้งานเมื่อมีสายเรียกเข้า โดยในกรณีที่เครื่องอยู่ในสถานะ Ringing ดังแสดงในรูปที่ 3.38 ไมโครคอนโทรลเลอร์จะควบคุม Buzzer ให้เป็นเสียงกระดิ่งดัง 1.5 วินาที หยุด 3.5 วินาทีสลับกันไปเรื่อย ๆ โดยเสียงกระดิ่งที่ดังแต่ละครั้งจะประกอบไปด้วยเสียงดัง และหยุดสลับกันไปทุก ๆ 60 msec ดังแสดงในรูปที่ 3.39 ในการควบคุมแต่ละครั้งไมโครคอนโทรลเลอร์จะใช้เวลาเพียงรอบละ

60 msec ควบคุม Buzzer ซึ่งเท่ากับช่วงเวลาเสียงดังหยุดสลับกัน แล้วไปตรวจสอบอย่างอื่นต่อทั้งนี้ เนื่องจากเมื่อเครื่องอยู่ในสถานะ Ringing เครื่องโทรศัพท์อาจได้รับข้อมูลจากชิพอีเทอร์เนต หรือผู้ใช้งานอาจกดคีย์แพดซึ่งมีผลต่อการควบคุมเซสชันได้



รูปที่ 3.38 การควบคุม Buzzer



รูปที่ 3.39 สัญญาณควบคุม Buzzer เพื่อใช้เป็นกระดิ่งเตือนผู้ใช้งานเมื่อมีสายเข้า

เมื่อตรวจสอบสถานะของเครื่องเรียบร้อยแล้ว ไมโครคอนโทรลเลอร์จะตรวจสอบจำนวนข้อมูลเสียงภายในบัฟเฟอร์ข้อมูลเสียงภาคส่ง ในกรณีที่มีการเขียนข้อมูลเสียงใหม่ครบ 160 ไบต์ ไมโครคอนโทรลเลอร์จะย้ายข้อมูลเสียงทั้ง 160 ไบต์จากบัฟเฟอร์ข้อมูลเสียงมาใส่ลงในบัฟเฟอร์ภาคส่ง แล้วเพิ่มเฮดเดอร์ RTP, UDP, IP, และ Ethernet แล้วค่อยส่งข้อมูลไปให้ชิพอีเทอร์เนต และควบคุมให้ชิพอีเทอร์เนตส่งข้อมูลนั้นออกไป จากนั้นไมโครคอนโทรลเลอร์จะตรวจสอบคีย์แพด โดยถ้ามีการกดคีย์ ไมโครคอนโทรลเลอร์จะตอบสนองการกดคีย์นั้นตามคีย์ที่ได้ถูกกด และสถานะของเครื่องในขณะนั้น เมื่อตรวจสอบการกดคีย์เรียบร้อยแล้วไมโครคอนโทรลเลอร์จะตรวจสอบตัวแปรซึ่งใช้นับจำนวนการเกิดอินเตอร์รัปต์จากไทเมอร์ 1 ซึ่งใช้เป็นไทเมอร์สำหรับการส่งซ้ำ โดยถ้าตัวแปรที่ใช้นับจำนวนการเกิดอินเตอร์รัปต์มีค่าเกินค่าที่กำหนดไว้ตามกฎ exponential backoff ไมโครคอนโทรลเลอร์จะส่งข้อมูลภายในบัฟเฟอร์ภาคส่งไปให้ชิพอีเทอร์เนต และควบคุมให้ชิพอีเทอร์เนตส่งข้อมูลนั้นซ้ำไปยังปลายทางอีกครั้ง เมื่อตรวจสอบตัวแปรซึ่งใช้นับจำนวนการเกิดอินเตอร์รัปต์เรียบร้อยแล้ว ไมโครคอนโทรลเลอร์จะกลับไปตรวจสอบพอร์ต 3.2 และไล่วนตรวจสอบสถานะของเครื่อง, จำนวนข้อมูลเสียงภายในบัฟเฟอร์ข้อมูลเสียงภาคส่ง, คีย์แพด, และตัวแปรซึ่งใช้นับจำนวนการเกิดอินเตอร์รัปต์จากไทเมอร์ 1 วนซ้ำไปเรื่อย ๆ ต่อไป

3.8 ต้นทุนการพัฒนาเครื่องโทรศัพท์อินเทอร์เนตต้นแบบ

การพัฒนาเครื่องโทรศัพท์อินเทอร์เนตต้นแบบ ได้ใช้อุปกรณ์ที่หาซื้อได้จากภายในประเทศ มีรายละเอียดดังแสดงในตารางที่ 3.10 การพัฒนาเครื่องโทรศัพท์อินเทอร์เนตต้นแบบมีต้นทุนอยู่ที่ประมาณ 2,500 บาทต่อเครื่อง แต่ถ้าผลิตจำนวนมากสามารถลดต้นทุนการทำ PCB (Print Circuit Board) และต้นทุนอุปกรณ์ได้

ตารางที่ 3.10 รายการอุปกรณ์ที่ใช้ในการพัฒนาเครื่องโทรศัพท์อินเทอร์เนตต้นแบบ

Electronic parts	Part No.	Description	Supplier	Unit Price (Baht)	Quantity	Price (Baht)
MCU, Memory &	AT89C51ED2	MCU	ES	230	1	230
	61C256AH	32 kbytes SRAM	ES	65	1	65
	82C55	4 Extended Port IC	ES	67	1	67
	74HC373	Latch IC	ES	6	1	6
	74HC139	Memory Decoder	ES	4	1	4
	MAX232	RS-232 Interface	ES	33	1	33
	X'tal 18.432 MHz	For MCU	ES	7	1	7
Ethernet Interface	RTL8019as	Ethernet Controller	ETT	310	1	310
	20F001N	10BaseT Low Past Filter	ETT	80	1	80
	RJ-45 Connector	Ethernet Interface	ES	12	1	12
	X'tal 20 Mhz	For Ethernet Controller	ES	7	1	7
Codec Module	MC145480	5V PCM μ/A Law Codec	WesTech	90	1	90
	74HC4060	14 State Binary Counter	ES	7	1	7
	74HC165	8 bit P/S	ES	7	1	7
	74HC164	8 bit S/P	ES	7	1	7
	74HC02	Nor-Gate x 4	ES	3	1	3
	74HC04	Inverter x 6	ES	4	1	4
	74HC08	And-Gate x 4	ES	4	1	4
	RJ-11 Connector	Handset interface	ES	6	1	6
	X'tal 4.096 MHz	For Clock Module	ES	7	1	7
User Interface	LCD 16x2	User Display	ES	284	1	284
	5mm Switch	User Input (Keypad)	ES	1	16	16
	Connector 14 pin	For LCD Connection	ES	3	1	3
	Connector 10 pin	For Keypad Connection	ES	2	1	2
Power Supply	LM7805	5V-1A Regulator	ES	6	1	6
	Bridge Diode	Full Wave Rectifier	ES	4	1	4
Miscellaneous (Resistor, Capacitor, LED, etc)			ES			20
Print Circuit Board (4x5 inch)			WARA	1,230	1	1,230
Total Price (B)						2,521

ในการสั่งทำ PCB ต้นแบบนั้นต้องเสียค่าใช้จ่ายในการ SETUP และค่าฟิล์มทำให้ต้นทุนในการผลิต PCB ต้นแบบสูง แต่ในการผลิตเพิ่มเป็นจำนวนมากนั้นมีค่าใช้จ่ายเพียงค่าแผ่น PCB และค่าทำ PCB สำเร็จเท่านั้นซึ่งมีราคาต้นทุนต่อตารางนิ้วละประมาณ 6 บาท ทำให้ต้นทุนการทำ PCB ต่อเครื่องลดลงเหลือประมาณ 120 บาท ส่วนอุปกรณ์อื่น ๆ มีต้นทุนประมาณ 1,200 บาท แต่ในกรณีที่สั่งซื้อเป็นจำนวนมากสามารถลดต้นทุนอุปกรณ์ต่อหน่วยได้ประมาณ 10-30% (อ้างอิงราคาจาก Electronic Source) ทำให้ต้นทุนอุปกรณ์ต่าง ๆ ลดลงเหลือประมาณ 1,000 บาทต่อเครื่อง รวมต้นทุนPCB และอุปกรณ์ต่อเครื่องประมาณ 1,000-1,200 บาท