บทที่ 3

โลดเดอร์มอนิเตอร์ที่ไม่กำหนดตำแหน่งหน่วยความจำ

การวิจัยนี้เป็นการศึกษาการทำงานของโลดเดอร์มอนิเตอร์ที่ไม่กำหนดตำแหน่งหน่วยความจำ ว่ามีการทำงานภายในตัวเองอย่างไรบ้าง และเอื้อต่อการจัดหน่วยความจำออกเป็นสองส่วน เพื่อให้สามารถใช้หน่วยความจำเก็บงานไว้ภายในได้ เพื่อให้หน่วยประมวลผลกลางใช้โปรแกรมหรือข้อมูลได้ในช่วงเวลาเดียวกัน

การออกแบบสร้างโลดเดอร์ใหม่นี้ จำเป็นที่จะต้องคำนึงถึงขีดความสามารถและขอบเขตดังต่อไปนี้

1. มาโครโปรแกรมของระบบโลดเดอร์มอนิเตอร์ที่ไม่กำหนดตำแหน่งหน่วยความจำ

ซึ่งมีชื่อทางระบบว่า FTLMC จากการศึกษาถึงโครงสร้างของโปรแกรมนี้พบว่ามีส่วนเกี่ยวข้องกับระบบที่จะนำมาใช้คือ

1.1 สามารถกำหนดขนาดของหน่วยความจำที่มีอยู่ในระบบได้ โดยแอดเดรสซิ่งโมด (Addressing Mode) ซึ่งมีให้ใช้ได้สองขนาดคือ แอดเดรสซิงโมด = 3 ขนาดหน่วยความสูงสุดเท่ากับ 32 กิโลคาร์แรคเตอร์ และแอดเดรสซิงโมด = 4 ขนาดหน่วยความจำสูงสุดจะเป็นถึง 262 กิโลคาร์แรคเตอร์

1.2 สามารถกำหนดอุปกรณ์ที่ใช้ในการติดต่อกับผู้ควบคุมระบบ (System-Operator) ได้ว่าจะใช้แผงควบคุมกลาง (Console Panel) หรือเครื่องพิมพ์ควบคุม (Console Typewriter) เพื่อเป็นเครื่องมือในการติดต่อสั่งงาน

1.3 สามารถใช้เครื่องอ่านบัตร (Card Reader) หรือใช้เครื่องอ่านแถบเทปกระดาษ (Paper Tape) เพื่อนำข้อมูลเข้าโลดเดอร์มอนิเตอร์

1.4 โลดเดอร์นี้เป็นเทปโลดเดอร์มอนิเตอร์ กล่าวคือใช้ในการโลดโปรแกรมจากแมคเนติคเทป (Magnetic Tape Unit) เพราะระบบที่ใช้อยู่เป็นระบบซึ่งเก็บโปรแกรมไว้บนแถบแม่เหล็ก

2. อุปกรณ์และคุณสมบัติของระบบเครื่อง

2.1 อุปกรณ์ในระบบเครื่องที่ใช้ในการติดต่อกับโลดเดอร์มอนิเตอร์ซึ่งในระบบปัจจุบัน มีแผงควบคุมกลางสำหรับการควบคุมระบบ และเครื่องอ่านบัตรเพื่อส่งข้อมูลให้กับโลดเดอร์มอนิเตอร์ อันได้แก่บัตรเรียกโปรแกรม (Console Call Card) เป็นต้น

2.2 คุณสมบัติของหน่วยประมวลผลกลางที่ใช้อยู่นี้สามารถที่จะจัดให้มีแอดเดรสซิ่งโมดได้ 3 ขนาดคือ

แอดเดรสซิงโมด = 2 มีหน่วยความจำสูงสุด 12 กิโลคาร์แรคเตอร์

แอดเดรสซิงโมด = 3 มีหน่วยความจำสูงสุด 32 กิโลคาร์แรคเตอร์

แอดเดรสซิงโมด = 4 มีหน่วยความจำสูงสุด 262 กิโลคาร์แรคเตอร์

นอกจากนี้หน่วยประมวลผลกลางยังมีความสามารถซึ่งเอื้อในการควบคุมการทำงานของหน่วยประมวลผลกลางคือ สามารถมีเอ็กเตอร์นอลอินเตอร์รัพ (External Interrupt) ซึ่งเกิดขึ้นได้โดย

เพอริเฟอราลอินเตอร์รัพ (Peripheral Interrupt) กล่าวคือ เพอริเฟอราล เช่นพวก แมคเนติคเทป เครื่องอ่านบัตร เป็นต้น สามารถที่จะขอเวลาในการส่งข้อมูลจากหน่วยประมวลผลกลางได้

อินเตอร์รัพจากผู้ควบคุมระบบ (Operator's Interrupt) โดยการกดปุ่มอินเตอร์รัพซึ่งอยู่บนแผงควบคุม

อินเตอร์รัพจากโปรแกรม (Program Instruction Interrupt) โดยการใช้คำสั่งมอนิเตอร์คอลล์ (Monitor Call-MC) ซึ่งเมื่อทำคำสั่งนี้แล้วหน่วยประมวลผลกลางจะเข้าสู่สภาวะอินเตอร์รัพทันที

จากคุณสมบัติที่กล่าวมาแล้วข้างต้นผู้วิจัยเห็นว่าสามารถที่จะสร้างโลดเดอร์มอนิเตอร์ที่ไม่กำหนดตำแหน่งหน่วยความจำขึ้นใช้ในระบบเครื่องที่มีอยู่ได้โดยมีคุณสมบัติคือ

1. เป็นเทปโลดเดอร์มอนิเตอร์
2. การติดต่อผ่านทางแผงควบคุมกลาง
3. อ่านข้อมูลผ่านทางเครื่องอ่านบัตร
4. ใช้แอดเดรสซึ่งโมดเท่ากับ 3

<u>การเตรียมงาน</u>

ลำดับขั้นตอนการเตรียมการสร้างโลดเดอร์นี้ ทำเป็นขั้น ๆ ดังนี้

ขั้นแรก จัดเตรียมทำ SPT-Library ขึ้น เพื่อให้สะดวกในการสร้างโลดเดอร์มอนิเตอร์ โดยใช้ระบบ SPT-Merge Call แยกเอาตัวมาโครที่จำเป็นต้องใช้ คือ FTLMC ออกมาจากห้องสมุดของระบบ (System Library) ในตอนนี้ได้นำเอามาโครอีกตัวที่อาจจะใช้ต่อไปออกมาด้วยคือ มาโครของอินเตอร์รับคอนโทรล ซึ่งมีชื่อว่า PINM ซึ่งใช้เมื่อต้องการสร้างระบบเพื่อควบคุมการอินเตอร์รัพ ดังได้กล่าวแล้วข้างต้น

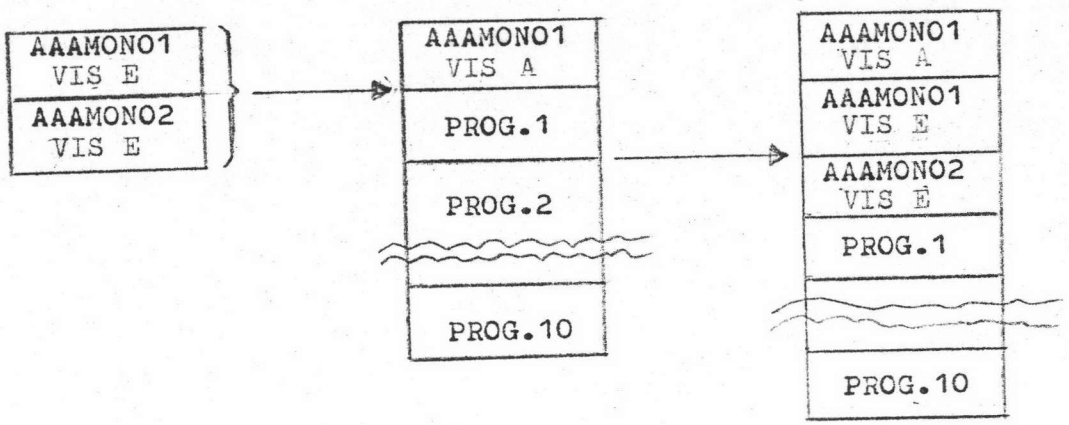
SPT เป็นชื่อย่อของ Symbolic Program Tape ซึ่งเป็นโครงสร้างของระบบห้องสมุดของนีแอค 2200 กล่าวคือจะเก็บเอาโปรแกรมไว้ในรูปของ Source-Program และ Object Program ในตัวเดียวกัน

ขั้นที่สอง สร้างตัวโปรแกรมโลดเดอร์มอนิเตอร์จากมาโคร โดยใช้ระบบโปรแกรมห้องสมุด (Library Processor) โดยระบุอักษรควบคุมเพื่อให้มีคุณลักษณะและขีดความสามารถที่กล่าวไว้ในตอนก่อน ในตอนนี้จะได้ตัวโปรแกรมของโลดเดอร์มอนิเตอร์ในรูปของภาษา Assembly ซึ่งเก็บไว้บนแถบแม่เหล็ก เหมือนกับการใช้บัตรคอมพิวเตอร์

ขั้นที่สาม ทำการแปลโปรแกรมที่ได้จากขั้นที่สอง โดยการ Assembly ซึ่งจะได้ตัวโปรแกรมของโลดเดอร์มอนิเตอร์ ซึ่งอยู่ในรูปแบบที่พร้อมที่จะนำเข้าไปไว้ในหน่วยความจำเพื่อทำงานได้ทันที (Binary Run Program หรือ Object Program)

โลดเดอร์มอนิเตอร์แบ่งออกเป็น 2 ส่วน (Segment) โดยให้ชื่อว่า AAAMONO1 และ AAAMONO2 โดยมีกลุ่มเป็น E (Visibility = E) เพื่อให้ไม่เหมือนกับโลดเดอร์มอนิเตอร์ที่ใช้อยู่เดิม ซึ่งมีกลุ่ม =A

ขั้นที่สี่ นำเอาโปรแกรมโลดเดอร์มอนิเตอร์ที่สร้างขึ้นใหม่นี้เข้าไปแทรกไว้ (Insert) ในเทปโปรแกรมที่ใช้ทำงาน (Binary Run Tape ; BRT) เพื่อใช้งานต่อไป โดยใช้การอัพเดทของระบบ (BRT UPDATE SYSTEM)



รูปที่ 3.1 รูปแสดงโครงสร้างเทปเพื่อใช้งาน

ในตอนนี้ก็จะได้เทปที่จะใช้ทำการทดลองหรือใช้งานได้ โดยสามารถใช้งานได้ทั้งโลดเดอร์มอนิเตอร์ตัวเดิมและตัวใหม่ ซึ่งจะกล่าวถึงในภาคการทดลองต่อไป

<u>การทำงานของโลดเดอร์มอนิเตอร์ที่ไม่กำหนดตำแหน่งหน่วยความจำ</u>

เพื่อให้ง่ายแก่การเข้าใจและอธิบายการทำงานภายในของโลดเดอร์มอนิเตอร์สามารถแบ่งออกเป็นส่วนๆดังต่อไปนี้

1. ส่วนข้อมูลร่วมกันเพื่อใช้ในระบบ (Communication Area) ส่วนนี้เป็นส่วนที่เก็บข้อมูลซึ่งจำเป็นต้องใช้ในระหว่างการทำงานของระบบเพื่อให้สามารถทราบถึงสภาวะการทำงาน เพื่อการประสานงานของการทำงานทั้งระบบ เช่นชื่อของ

โปรแกรมที่ต้องการโลดเข้าสู่หน่วยความจำ หรือส่วนที่เป็นตัวชี้ถึงแอดเดรสที่จำเป็นในการทำงาน เป็นต้น สำหรับรายละเอียดแสดงไว้ในโปรแกรมซึ่งอยู่ในภาคผนวก ก ของวิทยานิพนธ์นี้

2. ส่วนค้นหาโปรแกรมที่ต้องการ (Search) ส่วนนี้จะทำการค้นหาตัวโปรแกรมที่ต้องการ เนื่องจากในเทปที่ใช้งานหนึ่งๆ ประกอบด้วยโปรแกรมต่างๆมากมาย เพื่อจะได้นำโลดเข้าสู่หน่วยความจำต่อไป สำหรับข้อมูลที่จำเป็นในการค้นหาซึ่งเก็บไว้ในส่วนข้อมูลร่วม มีดังนี้

- ชื่อโปรแกรม (Program Name)
- ชื่อเซกเมนท์ (Segment Name)
- กลุ่ม (Visibility)
- ตำแหน่งสัมพัทธ์ (Relative Position)
- หน่วยที่เก็บโปรแกรม (Magnetic Tape Unit)

ในการใช้งานนั้นสามารถค้นหาโปรแกรมที่ต้องการจากหน่วยเก็บได้โดย

1. ชื่อโปรแกรมและเซกเมนท์
2. ชื่อโปรแกรม, เซกเมนท์และกลุ่ม
3. ตำแหน่งสัมพัทธ์

การทำงานของส่วนที่ทำได้โดยการเปรียบเทียบข้อมูลที่ได้จากเทปกับข้อมูลที่เก็บไว้ในส่วนข้อมูลร่วม ถ้าไม่ตรงกันตามข้อกำหนดข้างต้น ก็จะค้นหาต่อไป ซึ่งทำการหาได้ทั้งสองทิศทางคือไปข้างหน้า (Forward)หรือย้อนหลัง (Backward) ก็ได้ ถ้าค้นไม่พบก็จะหยุดการค้นหา โดยให้ข้อมูลกับผู้ควบคุมระบบเพื่อตรวจสอบว่าทำไมจึงไม่มีโปรแกรมที่ต้องการอยู่ในเทปนั้นๆ แต่ถ้าพบก็จะทำการโลดเข้าหน่วยความจำต่อไป

3. ส่วนโลดเข้าหน่วยความจำ (Loading) ส่วนนี้เป็นส่วนที่จะโลดเอาโปรแกรมเข้าสู่หน่วยความจำระบบนี้แอด 2200 นี้ โปรแกรมเก็บไว้บนเทปแม่เหล็กในลักษณะความยาวของเรคคอร์ดไม่คงที่ (Variable Record-

Length) และขนาดความยาวของแต่ละคำสั่งก็ไม่คงที่ (Instruction Length) ขึ้นอยู่กับโครงสร้าง (FORMAT) ของแต่ละคำสั่ง เพราะเครื่องนีแอค 2200 เป็นคอมพิวเตอร์แบบขนาดของฟิลด์ แต่ละฟิลด์ไม่คงที่ (Variable Field Length) ดังนั้นจึงจำเป็นต้องใช้อักษรควบคุม (Control Character) เป็นตัวบอกขนาดความยาวและเครื่องหมายขั้นตอน (Punctuation Mark) ซึ่งมี 3 อย่างคือ เครื่องหมายบอกเวิร์ด (WORD MARK) เครื่องหมายบอกไอเทม (Item Mark) และ เครื่องหมายบอกเรคคอร์ด (Record Mark) นอกจากนี้ยังมีอักษรควบคุมเพื่องานพิเศษ เช่น สร้างพื้นที่ของข้อมูลต่างๆ ที่จำเป็นในการทำงานของโปรแกรมอีกด้วย

ดังนั้นส่วนโลดเข้าหน่วยความจำจึงต้องโลดโปรแกรมหรือข้อมูลตามคุณลักษณะของอักษรควบคุม ซึ่งได้แต่ละเรคคอร์ดของเทปที่อ่านเข้ามา สำหรับรายละเอียดของอักษรควบคุมแสดงไว้ในภาคผนวก ค.

เมื่อทำการโลดโปรแกรมที่ต้องการหมดแล้ว ส่วนโลดจะตรวจสอบดูว่าการเริ่มต้นทำงานตามโปรแกรมที่ต้องการเริ่มต้นในลักษณะอย่างไร และที่แอดเดรสไหน (START MODE) ซึ่งมี 3 แบบคือ

1. แบบปรกติ (NORMAL START) คือจะเริ่มต้นทำงานตามที่บอกไว้ในเรคคอร์ด ที่โลดเข้ามานั้น

2. แบบเริ่มที่แอดเดรสที่กำหนดให้ (SPECIAL START) คือเริ่มต้นทำงานตามแอดเดรสที่กำหนดทางภายนอก

3. แบบกลับไปทำงานต่อจากงานเดิมก่อนโลด (Return to Call Unit) คือกลับไปทำงานยังโปรแกรมที่ต้องการให้โลด

4. ส่วนการเลื่อนแอดเดรส (Address Relocation) เนื่องจากโลดเดอร์ที่สร้างเป็นโลดเดอร์ที่ไม่กำหนดหน่วยความจำ ดังนั้นคำสั่งและแอดเดรสของตัวโลดเดอร์ (AAAMONO2) จึงจำเป็นจะต้องเปลี่ยนแปลงไปตามตำแหน่งที่

ต้องการ ดังนั้นในโลดเดอร์ที่เป็นฐาน (AAAMONO1) ซึ่งเป็นตัวโลดตัวโลดเดอร์ที่ไม่กำหนดหน่วยความจำ จึงจำเป็นที่ต้องการเลื่อนแอดเดรสให้กับโลดเดอร์ที่จะเป็นตัวทำงานในระบบ

ส่วนนี้มีหน้าที่เลื่อนแอดเดรสของคำสั่งดังนี้

1. แอดเดรสในส่วนข้อมูลร่วมกัน
2. แอดเดรสเริ่มต้น ของโปรแกรมที่โลด
3. แอดเดรสของคำสั่งของโปรแกรมที่โลด

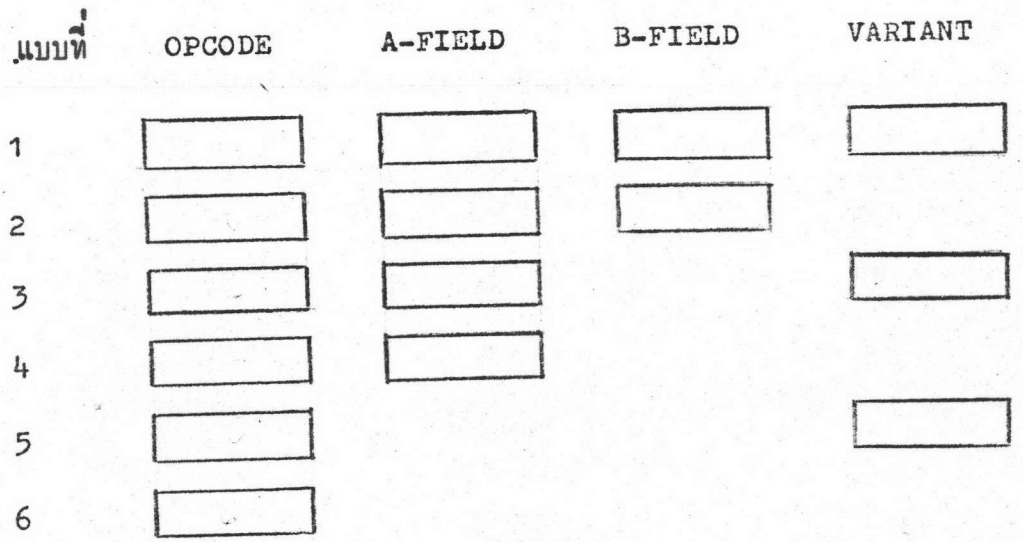
หลักการเลื่อนแอดเดรสทำโดย การเอาจำนวนแอดเดรสที่ต้องการเลื่อนไปบวกเข้ากับแอดเดรสเดิมที่ต้องการเลื่อนไป สำหรับจำนวนแอดเดรสที่จะต้องเลื่อนไปนั้นหาได้จากค่าแอดเดรสสูงสุดของแต่ละช่วงของหน่วยความจำ (Relocation-Indicator) ซึ่งต้องการให้โปรแกรมถูกโลดเข้าไป หักออกด้วยขนาดของโปรแกรมที่ต้องการโลด นั่นคือแอดเดรสสูงสุดของแต่ละช่วงเป็น $X7777_8$ ค่า X คือค่าของรีโลเกชันอินดิเกเตอร์ (Relocation Indicator) ซึ่งในระบบที่ใช้มีค่าตั้งแต่ 02 ถึง 07 คือช่วงแอดเดรส 27777_8 - 77777_8 (12K - 32K)

เช่นในการโลดเซกเมนท์ที่สองของโลดเดอร์ที่ไม่กำหนดตำแหน่งหน่วยความจำนี้ไปไว้ที่ช่วง 50000_8 - 57777_8 ค่ารีโลเกชันอินดิเกเตอร์ใช้ 05 และขนาดของโปรแกรมเท่ากับ 3657_8 ดังนั้นโปรแกรมจะโลดไว้ที่แอดเดรส 57777_8 - 03657_8 = 54120_8 ถึง 57777_8 ของหน่วยความจำเป็นต้น

หลังจากส่วนโลดได้ทำการโลดโปรแกรมไปไว้ในหน่วยความจำตามค่าของรีโลเกชันอิน ดิเกเตอร์ที่ต้องการแล้ว ขั้นต่อไปก็จะทำการเลื่อนแอดเดรสของแต่ละคำสั่ง ซึ่งมีทั้งคำสั่งของตัวแปลโปรแกรม(Assembler Instruction) เช่น DSA, DCW เป็นต้น และคำสั่งของโปรแกรม(Machine Instruction) เช่น SW,MCW เป็นต้น ดังนั้นโปรแกรมที่จะถูกเลื่อนตำแหน่งได้ควรมีโครงสร้างดังนี้

	DCW ADMODE	บอกคำแอดเดรสซึ่งโมดที่ใช้
	DSA (A)	บอกขนาดของ DSA ที่ใช้
	DSA (B)	บอกขนาดของคำสั่งที่มีในโปรแกรม
	DSA -----	DSA ของโปรแกรม
	DSA -----	DSA ของโปรแกรม
	.	
(A)	DSA	DSA สุดท้าย
	Instruction	คำสั่งแรก
	.	
	.	
(B)	Instruction	คำสั่งสุดท้าย

กล่าวคือส่วนเลื่อนตำแหน่งจะทำการเลื่อนตำแหน่งแอดเดรสใน DSA จนพบว่าค่าที่เก็บไว้ใน DSA แรกมีค่าเท่ากับแอดเดรสของ DSA สุดท้าย แต่เมื่อไม่เท่ากันก็จะทำการเลื่อนแอดเดรสของแต่ละคำสั่งจนหมดคำสั่ง DSA แต่ในการเลื่อนตำแหน่งของแอดเดรสของแต่ละคำสั่ง จำเป็นต้องตรวจสอบว่าเป็นคำสั่งชนิดใด มีโครงสร้างอย่างไร ซึ่งนีแอค 2200 มีโครงสร้างของคำสั่งได้ 6 รูปแบบคือ



รูปที่ 3.2 โครงสร้างของคำสั่ง 6 รูปแบบของนีแอค 2200

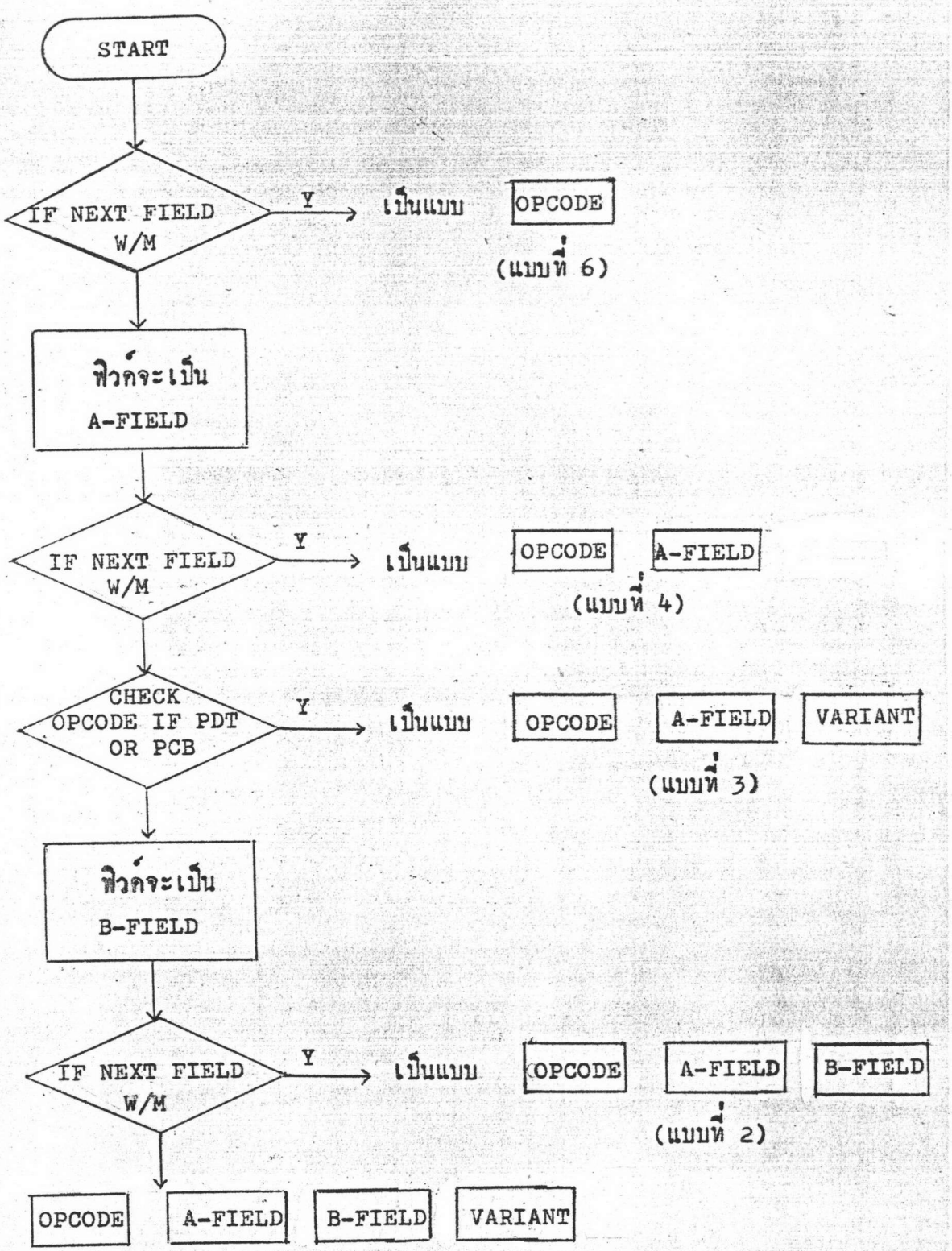
ซึ่งเฉพาะ A-FIELD และ B-FIELD เท่านั้นที่จำเป็นจะต้องเลื่อนตำแหน่งไปเพื่อสอดคล้องกับตำแหน่งใหม่ของโปรแกรมซึ่งโลดไว้ดังกล่าวในตอนแรก ส่วน OPCODE และ VARIANTก็คงรูปและตำแหน่งเดิมโดยจะข้ามไปเสีย สำหรับหลักการหา A-FIELD และ B-FIELD ของคำสั่ง ใช้วิธีการดังแสดงในแผนผังดังต่อไปนี้

เริ่มต้น 1. ตรวจดู OPCODE ว่าเป็น CAM หรือ HALT หรือไม่

-ถ้าเป็น CAM ซึ่งมีโครงสร้างเป็นแบบที่ 5 [OPCODE] [VARIANT] ไม่ต้องรีโลเกชัน เพียงแต่นำค่า VARIANT ซึ่งบอกค่าแอดเดรสชิงโมดไปเปลี่ยนค่าแอดเดรสชิงโมดที่ใช้อยู่เท่านั้น

-ถ้าเป็น HALT ซึ่งอาจมีรูปแบบได้คือ แบบที่ 6 [OPCODE] แบบที่ 4 [OPCODE] [A FIELD] และแบบที่ 2 [OPCODE] [A-FIELD] [B-FIELD] ซึ่งแบบแรกและสุดท้ายไม่ต้องรีโลเกชัน ดังนั้นจึงสามารถแยกได้โดยการทดสอบหาเวอร์ดมาร์ด (W/M) ของฟิลด์ที่ถัดจาก OPCODE ถ้ามีเวิร์ดมาร์ดก็เป็นรูปที่ 1 และทดสอบฟิลด์ที่ถัดจาก A-FIELD ถ้ามี W/M ก็เป็นแบบที่ 2 ถ้าไม่มี W/M ก็เป็นแบบที่ 3

2. เมื่อไม่ใช่ CAM และ HALT ก็เป็นคำสั่งอื่นๆ และคำสั่ง PDT และ PCB ซึ่งอาจมีรูปแบบได้ถึง 5 รูป ที่แสดงไว้ข้างต้น ยกเว้นแบบที่ 5 [OPCODE] [VARIANT] ซึ่งหา A-FIELD, B-FIELD, VARIANT ได้โดย



รูปที่ 3.3 แผนผังการตรวจหารูปแบบโครงสร้างของคำสั่ง

เมื่อได้แอดเดรสฟิลด์แล้วต้องมาศึกษาต่อว่าวิธีการแอดเดรส ซึ่งทำได้เป็น 3 วิธีคือ

1. ไดเรคแอดเดรสซิง (Direct Addressing)
2. อินเด็กซ์แอดเดรสซิง (Index Addressing)
3. อินไดเรคแอดเดรสซิง (Indirect Addressing)

ซึ่งจำเป็นต้องรีโลเกชันให้กลับแอดเดรสซิงฟิลด์ที่เป็นไดเรคและอินไดเรคแอดเดรสซิง เพราะไดเรคแอดเดรสใช้ตำแหน่งแอดเดรสโดยตรงของข้อมูลที่ใช้ และอินไดเรคแอดเดรสซิงใช้ตำแหน่งแอดเดรสของข้อมูลที่ได้จากตำแหน่งที่ระบุไว้ภายในแอดเดรสฟิลด์นั้น แต่การใช้ตำแหน่งของข้อมูลในแบบอินเดกซ์แอดเดรสซิง ทำได้โดยการใช้ค่าที่เอาเก็บไว้ก่อนในอินเดกซ์รีจิสเตอร์(Index Register) บวกเข้ากับค่าที่ระบุไว้ในแอดเดรสฟิลด์ จึงไม่จำเป็นต้องรีโลเกชันไป