บทที่ 5
การจัดการหน่วยความจำ

5.1 ความนำ

หน่วยความจำเป็นทรัพยากรที่มีความสำคัญในระบบคอมพิวเตอร์ที่จำเป็นต้องมีการจัดการอย่างระมัดระวัง ปัจจุบันหน่วยความจำในเครื่องคอมพิวเตอร์มีมากขึ้นเมื่อเทียบกับเครื่องคอมพิวเตอร์รุ่นก่อนๆ ทำให้สามารถเก็บข้อมูลหรือโปรแกรมที่มีขนาดใหญ่ขึ้นได้ และจากการที่หน่วยความจำทำหน้าที่เก็บข้อมูลและโปรแกรมที่ทำงานในคอมพิวเตอร์นี้เอง ทำให้ขนาดและปริมาณงานที่เครื่องคอมพิวเตอร์แต่ละเครื่องทำงานได้จึงมีขนาดที่แตกต่างกันออกไป เนื่องจากการที่โปรแกรมหรือข้อมูลจะถูกนำไปเก็บไว้ ในตำแหน่งใดของหน่วยความจำ หรือการเรียกข้อมูลจากหน่วยความจำตำแหน่งต่างๆ มาใช้งานต้องอยู่ภายใต้การควบคุมดูแล ของระบบปฏิบัติการเสมอซึ่งส่วนของระบบปฏิบัติการที่ทำหน้าที่ดังกล่าวเรียกว่า ผู้บริหารหน่วยความจำ (Memory Manager)

นอกจากนี้ ผู้บริหารหน่วยความจำยังทำหน้าที่ในการตรวจหาที่ว่างสำหรับการเก็บข้อมูล การย้ายข้อมูลออกจากหน่วยความจำ กรณีที่หน่วยความจำไม่มีที่ว่างสำหรับเก็บข้อมูลเป็นต้น ในการใช้งานโปรแกรมคอมพิวเตอร์มีหลักการอยู่ว่า โปรแกรมหรือข้อมูลที่นำมาใช้งานต้องอยู่ในหน่วยความจำเท่านั้น จึงสามารถทำงานได้ ในกรณีที่มีการเก็บข้อมูลหรือโปรแกรมในหน่วยความจำสำรอง เช่น ดิสก์ หรือ เทปแม่เหล็ก เมื่อจะใช้งานต้องทำการอ่านข้อมูลเหล่านั้นเข้าไปในหน่วยความจำก่อนจึงจะสามารถใช้งานได้ และลักษณะการจัดเก็บข้อมูลในหน่วยความจำในระบบปฏิบัติการแต่ละแบบก็มีความแตกต่างกัน รวมถึงการบริการการใช้งานหน่วยความจำก็มีความแตกต่างกันด้วยระบบการจัดการหน่วยความจำสามารถแบ่งออกได้เป็น 2 กลุ่มใหญ่ๆ คือ แบบที่เก็บ ข้อมูลไว้ในหน่วยความจำจนกว่าจะทำงานเสร็จ และแบบที่มีการย้ายข้อมูลจากหน่วยความจำออกไปเก็บไว้ที่หน่วยความจำสำรองเป็นการชั่วคราว แล้วจึงทำการย้ายกลับเข้ามาในหน่วยความจำใหม่เมื่อมีการอ้างอิงถึงข้อมูลนั้น ในบทนี้จะกล่าวถึงการจัดการหน่วยความจำในรูปแบบต่างๆ ตั้งแต่แบบธรรมดาที่ไม่มีความซับซ้อนมากนักจนถึงการจัดการหน่วยความจำขั้นสูงที่มีความยุ่งยากซับซ้อนในการทำงาน

5.2 ความหมายของหน่วยความจำ

หน่วยความจำ (Memory Unit) หมายถึง หน่วยหรืออุปกรณ์ที่ใช้ในการจดจำหรือจัดเก็บข้อมูลที่ใช้ในระบบคอมพิวเตอร์ ซึ่งหากคอมพิวเตอร์ปราศจากหน่วยความจำแล้ว คอมพิวเตอร์จะไม่สามารถทำงานได้เลย หน่วยความจำจึงถือเป็นหน่วยสำคัญหน่วยหนึ่งในระบบคอมพิวเตอร์ หน่วยความจำของคอมพิวเตอร์จะมีขนาดแตกต่างกัน ซึ่งในการวัดขนาดของหน่วยความจำคอมพิวเตอร์นั้นเราอธิบายโดยใช้จำนวนอักขระ ที่สามารถบรรจุภายในหน่วยความจำได้ ขนาดของหน่วยความจำนิยมพูดกันในเทอมของไบต์ (Byte) ซึ่งใน 1 ไบต์นั้นมีค่าเท่ากับ 1 อักขระ เช่น คอมพิวเตอร์ที่มีหน่วยความจำ 32 กิโลไบต์ หมายความว่าคอมพิวเตอร์เครื่องนั้นนั้นมีหน่วยความจำที่สามารถบรรจุอักขระได้ 32 x 1,024 อักขระ

5.3 ประเภทของหน่วยความจำ

หน่วยความจำในระบบคอมพิวเตอร์แบ่งออกเป็น 2 ประเภทหลัก คือ

5.3.1 หน่วยความจำหลัก (Main Memory)

หน่วยความจำหลัก คือ หน่วยความจำที่ทำหน้าที่ร่วมกับ CPU ในด้านต่างๆ โดยจะเป็นที่เก็บข้อมูลสำคัญๆ สำหรับ CPU หากจะใช้ระบบคอมพิวเตอร์ทำงานใดก็ตามภายในหน่วยความจำหลักของเครื่องคอมพิวเตอร์จะต้องมีข้อมูล และโปรแกรมที่เกี่ยวข้องกับการทำงานนั้นๆ เก็บไว้ก่อนจึงจะสามารถนำมาประมวลผลได้ ซึ่งข้อมูลและโปรแกรมเหล่านั้น สามารถที่จะเรียกใช้งานได้โดยองค์ประกอบอื่นๆ ของระบบ หน่วยความจำของคอมพิวเตอร์ใช้สำหรับเก็บชุดคำสั่งและเก็บข้อมูลบางส่วนหรือทั้งหมดที่จำเป็นต้องใช้ในการประมวลผล หน่วยความจำหลักแบ่งออกเป็น 2 ประเภท คือ

5.3.1.1 หน่วยความจำถาวร (Nonvolatile Memory)

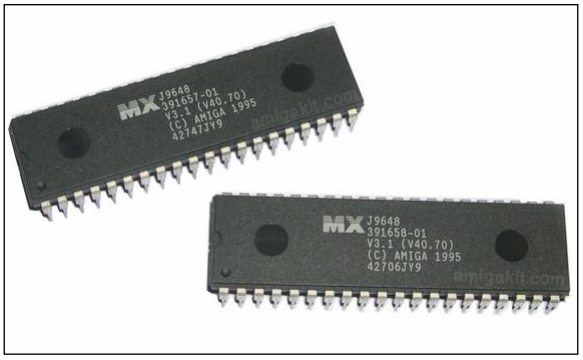
หน่วยความจำถาวรนี้ เป็นหน่วยความจำที่ทำหน้าที่ในการเก็บโปรแกรมควบคุมระบบหรือข่าวสารที่มีการใช้งานบ่อยๆ ซึ่งเมื่อโปรแกรมหรือข้อมูลเหล่านั้นถูกเขียนลงบนหน่วยความจำถาวรแล้วจะสามารถทำการอ่านได้เพียงอย่างเดียวไม่สามารถเข้าไปเขียนได้ ทั้งนี้เพื่อป้องกันความเสียหายอันเกิดขึ้นกับข้อมูลได้ ข้อมูลหรือโปรแกรมที่อยู่ในหน่วยความจำถาวรจะถูกอ่านออกมาเก็บไว้ในหน่วยความจำชั่วคราวเมื่อเริ่มเปิดเครื่องตัวอย่างของหน่วยความจำถาวร ได้แก่

รอม (ROM : Read Only Memory) ข้อมูลทั้งหมดที่อยู่ในรอม จะถูกโปรแกรมโดยผู้ผลิต และผู้ใช้ไม่สามารถ เปลี่ยนแปลงข้อมูลภายในรอมได้ โดยรอมแบ่งออกเป็น 3 ชนิด คือ

1. PROM (Programmable Read-Only Memory) ข้อมูลที่ต้องการโปรแกรมจะถูกโปรแกรมโดยผู้ใช้เอง โดยป้อนพัลส์ แรงดันสูง (HIGH VOLTAGE PULSED) ทำให้ METAL STRIPS หรือ POLYCRYSTALINE SILICON ที่อยู่ในตัว IC ขาดออกจากกัน เกิดเป็นลอจิก "1" หรือ "0" ตามตำแหน่ง ที่กำหนดในหน่วยความจำนั้นๆ เมื่อ PROM ถูกทำการโปรแกรมแล้ว ข้อมูลภายในจะไม่สามารถเปลี่ยนแปลงได้อีก หน่วยความจำชนิดนี้ จะใช้ในงานที่ใช้ความเร็วสูง ซึ่งความเร็วสูงกว่าหน่วยความจำ ที่โปรแกรมได้ชนิดอื่นๆ

2. EPROM (Erasable Programmable Read-Only Memory) ข้อมูลจะถูกโปรแกรม โดยผู้ใช้โดยการให้สัญญาณ ที่มีแรงดันสูง (HIGH VOLTAGE SIGNAL) ผ่านเข้าไปในตัว EPROM ซึ่งเป็นวิธีเดียวกับที่ใช้ใน PROM แต่ ข้อมูลที่อยู่ใน EPROM เปลี่ยนแปลงได้ โดยการลบข้อมูลเดิมที่อยู่ใน EPROM ออกก่อน แล้วค่อยโปรแกรมเข้าไปใหม่ การลบข้อมูลนี้ทำได้ด้วย การฉายแสง อุลตร้าไวโอเลตเข้าไปในตัว IC โดยผ่านทางกระจกใสที่อยู่บนตัว IC เมื่อฉายแสงครู่หนึ่ง (ประมาณ 5-10 นาที) ข้อมูลที่อยู่ภายในก็จะถูกลบทิ้ง ซึ่งช่วงเวลา ที่ฉายแสงนี้ สามารถดูได้จากข้อมูลที่กำหนดมากับตัว EPROM และ มีความเหมาะสมที่จะใช้ เมื่องานของระบบ มีโอกาสที่จะปรับปรุงแก้ไขข้อมูลใหม่

3. EAROM (Electrically Alterable Read-Only Memory) หรืออีกชื่อหนึ่งว่า EEPROM (Electrical Erasable EPROM) เนื่องจากมีการใช้ไฟฟ้าในการลบข้อมูลในรอมเพื่อเขียนใหม่ ซึ่งใช้เวลาสั้นกว่าของ EPROM การลบขึ้นอยู่กับพื้นฐานการใช้เทคโนโลยีที่แตกต่างกัน ดังนั้น EAROM จะอยู่บนพื้นฐานของเทคโนโลยีแบบ NMOS ข้อมูลจะถูกโปรแกรมโดยผู้ใช้เหมือนใน EPROM แต่สิ่งที่แตกต่างก็คือ ข้อมูลของ EAROM สามารถลบได้โดยทางไฟฟ้าไม่ใช่โดยการฉายแสงแบบ EPROM โดยทั่วไปจะใช้ EPROM เพราะเราสามารถหามาใช้และทดลองได้ง่าย มีราคาถูก วงจรต่อง่าย ไม่ยุ่งยาก และสามารถเปลี่ยนแปลงโปรแกรมได้นอกจากระบบ ที่ทำเป็นการค้าจำนวนมาก จึงจะใช้รอม ประเภทโปรแกรมสำเร็จ



ภาพที่ 5.1 แสดงตัวอย่างของหน่วยความจำถาวร

ที่มา (http://blogger.sanook.com/mk_melody/2008/12/, 2552)

5.3.1.2 หน่วยความจำชั่วคราว (Volatile Memory)

เป็นหน่วยความจำที่สามารถเข้าไปเขียน และอ่านได้ นอกจากนี้แล้วยังสามารถเลื่อนตำแหน่งต่างๆ ของการเขียน และอ่านข้อมูลในหน่วยความจำนี้โดยจะใช้เวลาเท่ากันในการอ่านหรือการเขียนข้อมูลลงในหน่วยความจำตำแหน่งที่แตกต่างกัน เราบอกขนาดหน่วยความจำของเครื่องคอมพิวเตอร์ด้วยขนาดของหน่วยความจำชั่วคราวนี้ เช่น เครื่องของไมโครคอมพิวเตอร์ขนาด 640 กิโลไบต์ ตัวเลข 640 กิโลไบต์จะบอกขนาดหน่วยความจำชั่วคราวในระบบคอมพิวเตอร์นั้นๆ หน่วยความจำชั่วคราวนี้จะทำงานได้ก็ต่อเมื่อมีกระแสไฟฟ้าเท่านั้น แต่ถ้าปิดเครื่องข้อมูลที่อยู่ในหน่วยความจำชั่วคราวจะสูญหายไปทันที ตัวอย่างของหน่วยความจำชั่วคราว ได้แก่ แรม (RAM : Random Access Memory) แรมเป็นหน่วยความจำหลักที่จำเป็นต้องมีในเครื่องคอมพิวเตอร์ทุกเครื่อง หน่วยความจำแรม ทำหน้าที่เก็บชุดคำสั่งและข้อมูลที่ระบบคอมพิวเตอร์กำลังทำงานอยู่ด้วย ไม่ว่าจะเป็นการนำเข้าข้อมูล (Input) หรือ การนำออกข้อมูล (Output) โดยที่เนื้อที่ของหน่วยความจำหลักแบบแรมนี้ถูกแบ่งออกเป็น 4 ส่วน คือ

1. Input Storage Area

เป็นส่วนที่เก็บข้อมูลนำเข้า ที่ได้รับมาจากหน่วยรับข้อมูล โดยข้อมูลนี้จะถูกนำไปใช้ในการประมวลผลต่อไป

2. Working Storage Area

เป็นส่วนที่เก็บข้อมูลที่อยู่ในระหว่างการประมวลผล

3. Output Storage Area

เป็นส่วนที่เก็บผลลัพธ์ที่ได้จากการประมวลผลตามความต้องการของผู้ใช้ เพื่อรอที่จะถูกส่งไปแสดงออก ยังหน่วยแสดงผลอื่นที่ผู้ใช้ต้องการ

4. Program Storage Area

เป็นส่วนที่ใช้เก็บชุดคำสั่งหรือโปรแกรมที่ผู้ใช้ต้องการ จะถูกส่งเข้ามาเพื่อให้คอมพิวเตอร์ปฏิบัติตามคำสั่ง ชุดดังกล่าว หน่วยควบคุมจะทำหน้าที่ดึงคำสั่งจากส่วนนี้ไปทีละคำสั่งเพื่อทำการแปลความหมายว่าคำสั่งนั้นสั่งให้ทำอะไร จากนั้นหน่วยควบคุม จะไปควบคุมฮาร์ดแวร์ที่ต้องการทำงานดังกล่าวให้ทำงานตามคำสั่งนั้นๆ

หน่วยความจำแรมที่เรานำมาใช้งานนั้นจะเป็นชิพตัวเล็กๆ ซึ่งส่วนที่เรานำมาใช้เป็นหน่วยความจำหลัก จะถูกบัดกรีติดอยู่บนแผงวงจร หรือ Printed Circuit Board ซึ่งมีโมดูล (Module) หลักๆ อยู่ 2 โมดูล คือ SIMM กับ DIMM

โดยโมดูล SIMM (Single In-line Memory Module) จะรองรับ Data Path 32 bit โดยทั้งสองด้านของ Circuit Board จะให้สัญญาณ เดียวกัน ในยุคต้นๆ ที่คอมพิวเตอร์เริ่มใช้งานกันอย่างแพร่หลายมากขึ้น ซึ่งส่วนมากมักเป็นคอมพิวเตอร์ ส่วนบุคคล (PC : Personal Computer) ใช้ ซีพียู 8088 หรือ 80286 หน่วยความจำถูกออกแบบให้ บรรจุอยู่ในแพคเกจแบบ DIP (Dual In-line Package) หรือที่เรียกว่าแบบตีนตะขาบเหมือนกับไอซีที่ใช้งานกันทั่วไป การใช้งานหน่วยความจำแบบนี้ จึงต้องมีการจัดสรรพื้นที่บนเมนบอร์ดมากพอสมควร การเพิ่มหน่วยความจำชนิดนี้ทำได้ง่าย เพียงแต่ซื้อแรม ตามขนาดความจุที่ต้องการมา เสียบลงในซ็อกเก็ต (Socket) ที่เตรียมไว้ และทำการติดตั้งจัมพ์เปอร์ (Jumper) อีกบางตัวหรือบางเครื่องอาจเพียงตั้งค่าใน ไบออส (BIOS : Basic Input Output Software) ของเครื่องก็สามารถใช้งานได้ทันที ครั้งเมื่อเวลาผ่านไปเทคโนโลยีก้าวหน้าขึ้น เทคนิคการแพคเกจชิพไอซีลงบนตัวถังทันสมัยมากขึ้น และเป็นที่รู้จักกันดีกับเทคโนโลยี อุปกรณ์ติดพื้นผิว ทำให้การติดตั้ง หน่วยความจำหรือเพิ่มหน่วยความจำ ทำได้ยากขึ้นและต้องมีเครื่องมือเฉพาะ จึงได้มีการคิดค้นวิธีการใหม่ โดยการนำเอาตัวไอซีของแรมแบบติดตั้งบนพื้นผิวไปติดบนแผงวงจรแผ่นเล็กๆ ก่อน แล้วจึงเดินลายทองแดงต่อขาจากตัวไอซีของแรมออกมา และแยกเป็นขาเชื่อมต่อเอาไว้เมื่อต้องการจะติดตั้งก็นำไปเสียบลงในซ็อกเก็ตที่เตรียมไว้บนเมนบอร์ดได้ทันที โมดูลหน่วยความจำแบบนี้มีชื่อเรียกว่า ซิพแรม (SIPRAM : Single In-line Package RAM) แรมชนิดนี้จะมี 30 ขา การพัฒนายังไม่หยุดเพียงเท่านี้ เพื่อความสะดวกในการ

ใช้งานมากขึ้น จึงได้มีการออกแบบซ็อกเก็ตสำหรับหน่วยความจำชั่วคราวแบบใหม่ โดยออกแบบในลักษณะคอนเน็กเตอร์ (Connector) ที่ส่วนของลายทองแดงบนแผ่นวงจรของซิพแรมโดยตรง ทำให้สามารถตัดขาที่ยื่นออกมาจากตัวโมดูลได้ ดังนั้นจึงได้มีการตั้งชื่อเรียกใหม่ว่า ซิมแรม (SIMM RAM : Single In-line Memory Module RAM) ซิมแรมมีขาต่อใช้งาน 30 ขา เช่นเดียวกับซิพแรม และสัญญาณที่ต่อใช้งานแต่ละขาก็เหมือนกันด้วย

สำหรับโมดูล DIMM (Dual In-line Memory Module) เพิ่งจะกำเนิดมาไม่นานนัก มี Datapath ถึง 64 บิต โดยทั้งสองด้านของ Circuit Board จะให้สัญญาณที่ต่างกันตั้งแต่ซีพียูตระกูล Pentium เป็นต้นมาได้มีการออกแบบให้ใช้งานกับ Data Path ที่มากกว่า 32 bit เพราะฉะนั้น เราจึงพบว่าเวลาจะใส่ซิมแรมบนสล๊อต (Slot) ของแรมจะต้องใส่เป็นคู่ใส่โดดๆ เพียงแผงเดียวไม่ได้

โมดูลของหน่วยความจำปัจจุบันมีอยู่ 3 รูปแบบคือ 30-pin, 72-pin, 168-pin ที่นิยมใช้ในเวลานี้คือ 168-pin ซึ่งหน่วยความจำแรมมีหลายประเภทดังนี้

1) DRAM (Dynamic Random Access Memory) จะทำการเก็บข้อมูลในตัวเก็บประจุ (Capacitor) ซึ่งจำเป็นต้องมีการรีเฟรช (Refresh) เพื่อเก็บข้อมูลให้คงอยู่โดยการรีเฟรชนี้ทำให้เกิดการหน่วงเวลาขึ้นในการเข้าถึงข้อมูล และก็เนื่องจากที่มันต้องรีเฟรชตัวเองอยู่ตลอดเวลานี้เองจึงเป็นเหตุให้ได้ชื่อว่า Dynamic RAM

2) SRAM (Static Random Access Memory) ต่างจาก DRAM ตรงที่ว่า DRAM ต้องทำการรีเฟรชข้อมูลอยู่ตลอดเวลา แต่ในขณะที่ SRAM จะเก็บข้อมูล นั้นๆ ไว้ และจะไม่ทำการรีเฟรชโดยอัตโนมัติ ซึ่งมันจะทำการรีเฟรชก็ต่อเมื่อ สั่งให้มันรีเฟรชเท่านั้น ซึ่งข้อดีของมันก็คือความเร็ว ซึ่งเร็วกว่า DRAM ปกติมาก แต่ก็ด้วยราคาที่สูงว่ามาก จึงเป็นข้อด้อยของแรมชนิดนี้

3) FPM DRAM (Fast Page Mode DRAM) ก็เหมือนกับ DRAM เพียงแต่ว่า ลดช่วงการหน่วงเวลาขณะเข้าถึงข้อมูลลง ทำให้มีความเร็วในการเข้าถึงข้อมูล สูงกว่า DRAM ปกติ ซึ่งโดยที่สัญญาณนาฬิกาในการเข้าถึงข้อมูล จะเป็น 6-3-3-3 หมายถึง Latency เริ่มต้นที่ 3 Clock พร้อมด้วย 3 Clock สำหรับการเข้าถึง Page และสำหรับ ระบบแบบ 32 bit จะมีอัตราการส่งถ่ายข้อมูลสูงสุด 100 MB ต่อวินาที ส่วนระบบแบ 64 bit จะมีอัตรา การส่งถ่ายข้อมูลที่ 200 MB

ต่อวินาที เช่นกัน ปัจจุบันแรมชนิดนี้แทบจะหมดไปจากตลาดแล้วแต่ ยังคงมีให้เห็นบ้าง และมักมีราคาที่ค่อนข้างแพงเมื่อเทียบกับแรม รุ่นใหม่ๆ เนื่องจากที่ว่าปริมาณในท้องตลาดมีน้อยมาก ทั้งๆ ที่ยังมีคนต้องการใช้แรมชนิดนี้อยู่

4) EDO DRAM (Extended-Data Output DRAM) หรือเรียกอีกชื่อหนึ่งก็คือ Hyper-Page Mode DRAM ซึ่งพัฒนาขึ้นอีกระดับหนึ่ง โดยการที่มันจะอ้างอิง ตำแหน่งที่อ่านข้อมูล จากครั้งก่อนไว้ด้วย ปกติแล้วการดึงข้อมูลจากแรม ณ ตำแหน่งใดๆ มักจะดึงข้อมูล ณ ตำแหน่งที่อยู่ใกล้ๆ จากการดึงก่อนหน้านี้ เพราะฉะนั้นถ้ามีการอ้างอิง ณ ตำแหน่งเก่าไว้ก่อน ก็จะทำให้ เสียเวลาในการเข้าถึงตำแหน่งน้อยลง และอีกทั้งมันยังลดช่วงเวลาของ CAS latency ลงด้วย และด้วยความสามารถนี้ทำให้การเข้าถึงข้อมูลดีขึ้นกว่าเดิมกว่า 40% และมีความสามารถโดยรวมสูงกว่า FPM กว่า 15% EDO จะทำงานได้ดีที่ 66 MHz ด้วย Timing 5-2-2-2 และก็ยังทำงานได้ดีเช่นกัน แม้จะใช้งานที่ 83 MHz ด้วย Timing นี้และหากว่า chip EDO นี้ มีความเร็วที่สูงมากว่า 50ns จะสามารถใช้งานได้ ณ 100 MHz ที่ Timing 6-3-3-3 อัตราการส่งถ่ายข้อมูลสูงสุด ของ DRAM ชนิดนี้อยู่ที่ 264 MB ต่อวินาที EDO RAM ในปัจจุบันนี้ไม่เป็นที่นิยมใช้แล้วBurst EDO (BEDO) DRAM BEDO ได้เพิ่มความสามารถขึ้นมาจาก EDO เดิม คือ Burst Mode โดยหลังจากที่ได้ Address ที่ต้องการ Address แรกแล้วนก็จะทำการ Generate อีก 3 Address ขึ้นทันที ภายใน 1 สัญญาณนาฬิกา ดังนั้นจึงตัดช่วงเวลาในการรับ Address ต่อไป เพราะฉะนั้น Timing จึงเป็น 5-1-1-1 ณ 66 MHz แรมแบบ BEDO ไม่เป็นที่แพร่หลายและได้รับความนิยมเพียงระยะเวลาสั้นๆ เนื่องจากว่าทาง Intel ตัดสินใจใช้ SDRAM แทน EDO และไม่ได้ใช้ BEDO เป็นส่วนประกอบในการพัฒนา Chipset ของตน ทำให้บริษัทผู้ผลิต ต่าง ๆ หันมาพัฒนา SDRAM แทน

5) SDRAM (Synchronous DRAM) จะต่างจาก DRAM เดิมคือจะทำงานสอดคล้องกับสัญญาณนาฬิกา สำหรับ DRAM เดิมจะทราบตำแหน่งที่อ่านก็ต่อเมื่อเกิดทั้ง RAS และ CAS ขึ้น แล้วจึงทำการไปอ่านข้อมูลโดยมีช่วงเวลาในการ เข้าถึงข้อมูล มักจะได้เห็นบนชิพของแรม เช่น -50, -60, -80 โดย -50 หมายถึง ช่วงเวลาเข้าถึง ใช้เวลา 50 นาโนวินาทีเป็นต้น แต่ว่า SDRAM จะใช้สัญญาณนาฬิกาเป็นตัวกำหนดการ ทำงานโดยจะใช้ความถี่ของสัญญาณเป็นตัวระบุ SDRAM จะทำงานตามสัญญาณนาฬิกาขาขึ้นเพื่อรอรับ ตำแหน่งข้อมูลที่ต้องการให้มันอ่านแล้วจากนั้นก็จะไปค้นหา และให้ผลลัพธ์ออกมาหลังจากได้รับ ตำแหน่งแล้ว เท่ากับค่าของ CAS เช่น CAS 2 ก็คือ หลังจากรับตำแหน่งที่อ่านแล้วจะให้ผลลัพธ์ออกมาภายใน 2 ลูกของสัญญาณ

นาฬิกา SDRAM จะมี Timing เป็น 5-1-1-1 ซึ่งมีความเร็วใกล้เคียงกับ BEDO RAM แต่สามารถทำงานได้ ณ 100 MHz หรือมากว่า และมีอัตราการส่งถ่าย ข้อมูลสูงสุดที่ 528 MB ต่อวินาที

6) DDR SDRAM (Double Data Rate SDRAM หรือ SDRAM II) เป็นชนิดของแรมที่แยกออกมาจาก SDRAM โดยจุดที่ต่างกันหลักๆ ของทั้งสองชนิดนี้คือ DDR SDRAM นี้สามารถที่จะใช้งานได้ทั้งขาขึ้น และขาลงของสัญญาณนาฬิกาเพื่อส่งถ่ายข้อมูล นั่นก็ทำให้อัตราส่งถ่าย เพิ่มขึ้นได้ถึงเท่าตัว ซึ่งมีอัตราการส่งถ่ายข้อมูลสูงสุดถึง 1Gbต่อวินาที มีการพัฒนาแรมชนิด RDRAM (Rambus DRAM) ชื่อของ RAMBUS เป็นเครื่องหมายการค้าของบริษัท RAMBUS Inc. โดยปัจจุบันได้เอาหลักการของ RAMBUS มาพัฒนาใหม่ โดยการลด pin รวม Static Buffer และทำการปรับแต่งทาง interface ใหม่ DRAM ชนิดนี้ จะสามารถ ทำงานได้ทั้งขอบขาขึ้น และลงของสัญญาณนาฬิกา และเพียงช่องสัญญาณเดียว ของหน่วยความจำแบบ RAMBUS นี้มี Performance มากกว่าเป็น 3 เท่า จาก SDRAM 100 MHz แล้ว และเพียงแค่ช่องสัญญาณเดียวนี้ก็มีอัตราการส่งถ่ายข้องมูลสูงถึง 1.6 Gb ต่อวินาที ถึงแม้ว่าเวลาในการเข้าถึง ข้อมูลแบบสุ่มของ RAM ชนิดนี้จะช้า แต่การเข้าถึงข้อมูลแบบต่อเนื่องจะเร็วมากๆ ซึ่ง RDRAM นี้มีการพัฒนา Interface และมี Printed Circuit Board รวมถึง Controller ของ Interfaceให้สามารถใช้งานได้ถึง 2 ช่องสัญญาณมีอัตราการส่งถ่ายข้อมูลเพิ่มเป็น 3.2 Gb ต่อวินาที และหากว่าสามารถใช้ได้ถึง 4 ช่องสัญญาณก็จะสามารถเพิ่มไปถึง 6.4 Gb ต่อวินาที

7) SGRAM (Synchronous Graphic RAM) เป็นชนิดของแรมที่แยกออกมาจาก SDRAM เช่นกันโดยถูกปรับแต่งมาสำหรับงานด้าน Graphics เป็นพิเศษแต่โดยโครงสร้างของ Hardware แล้ว แทบไม่มีอะไรต่างจาก SDRAM เลย เราจะเห็นจากบาง Graphic Card ที่เป็นรุ่นเดียวกัน แต่ใช้ SDRAM ก็มี เช่น Matrox G200 แต่จุดที่ต่างกัน ก็คือ ฟังก์ชัน ที่ใช้โดย Page Register ซึ่ง SGRAM สามารถทำการเขียนข้อมูลได้หลาย ๆ ตำแหน่งในสัญญาณนาฬิกาเดียว ในจุดนี้ทำให้ความเร็ว ในการแสดงผล และ Clear Screen ทำได้เร็วมาก และยังสามารถเขียนแค่ บาง bit ในการ Word ได้ คือไม่ต้องเขียนข้อมูลใหม่ทั้งหมดเขียนเพียงข้อมูลที่เปลี่ยนแปลงเท่านั้น โดยใช้ bitmask ในการเลือก bit ที่จะเขียนใหม่สำหรับงานโดยปกติแล้ว SGRAM แทบจะไม่ ให้ผลที่ต่างจาก SDRAM เลย ซึ่งเหมาะกับงานด้าน Graphics มากกว่า เพราะความสามารถที่แสดงผลเร็วและ Clear Screen ได้เร็วจึงเหมาะกับใช้บน Graphics Card มากกว่า ที่จะใช้บน System

8) VRAM (Video RAM) คือหน่วยความจำที่ทำงานเกี่ยวกับ Video เพราะถูก ออกแบบมาใช้บน Display Card โดย VRAM นี้ก็มีพื้นฐานมาจาก DRAM เช่นกัน แต่ที่ทำให้ต่างกันก็ด้วย กลไกการทำงานบางอย่าง ที่เพิ่มเข้ามา โดยที่ VRAM นั้น จะมี serial port พิเศษเพิ่มขึ้นมาอีก 1หรือ 2 port ทำให้เรามองว่าเป็น RAM แบบ พอร์ตคู่ (Dual-Port) หรือ ไตรพอร์ท (Triple-Port) ส่วน Parallel Port ซึ่งเป็น Standard Interface ของมัน จะถูกใช้ในการติดต่อกับ Host Processor เพื่อสั่งการให้ ทำการ refresh ภาพขึ้นมาใหม่ และ Serial Port ที่เพิ่มขึ้นมา จะใช้ในการส่งข้อมูลภาพออกสู่ Display

9) WRAM (Windows RAM) ถูกพัฒนาโดย บริษัท Matrox เพราะแทบจะเป็นผู้เดียวที่ใช้ RAM ชนิดนี้ บนการ์ดแสดงผลกราฟิก (Graphics Card) ของตน เฉพาะตระกูล Millennium และ Millennium II แต่ไม่รวม Millennium G200 ซึ่งเป็น ซึ่งใช้ SGRAM แรมชนิด WRAM นี้โดยรวมแล้วก็คล้ายคลึงกับ VRAM มีความแตกต่างคือรองรับ Bandwidth ที่สูงกว่า VRAM อีกทั้งยังใช้ระบบ Double-Buffer อีกด้วย จึงทำให้มีความเร็วมากกว่า VRAM อีกมากทีเดียว

5.3.2 หน่วยความจำสำรอง (Secondary Memory)

หน่วยความจำสำรองทำหน้าที่เก็บชุดคำสั่งหรือข้อมูลที่ไม่สามารถเก็บได้หมดในหน่วยความจำหลัก หรือใช้หน่วยความจำสำรองในการเก็บข้อมูลไว้เพื่อการเรียกใช้งานในภายหลังโดยไม่ต้องเสียเวลาป้อนข้อมูลหรือโปรแกรมนั้นใหม่ทุกครั้งที่มีการใช้งาน ซึ่งเปรียบเสมือนกับสมุดบันทึกสำหรับเก็บข้อมูลข่าวสารหรือโปรแกรมเอาไว้ใช้งานในโอกาสต่อไป ข้อดีของหน่วยความจำประเภทนี้ คือ ข้อมูลที่เก็บไว้ไม่สูญหายเมื่อปิดเครื่องและคงอยู่ตลอดไปตราบใดที่ไม่ลบข้อมูลทิ้ง หน่วยความจำหลักที่กล่าวมาแล้วนั้นแม้จะมีข้อดีในแง่ที่เป็นวงจรอิเล็กทรอนิกส์ ซึ่งทำให้การอ่านเขียนข้อมูลเป็นไปได้ด้วยความรวดเร็วก็ตาม แต่บางครั้งในการประมวลผลข้อมูลเรามีความจำเป็นจะต้องใช้ข้อมูลจำนวนมากหรือใช้โปรแกรมที่มีความยาวมากๆ จนไม่สามารถบรรจุในหน่วยความจำหลักได้เราจึงจำเป็นต้องใช้หน่วยความจำสำรองเข้าช่วย ซึ่งเหมาะสำหรับเก็บชุดคำสั่งและข้อมูลที่มีจำนวนมากหรือใช้บ่อยครั้ง สื่อที่ใช้เก็บข้อมูล ได้แก่ เทปแม่เหล็ก จานแม่เหล็ก การนำข้อมูลเข้าหรือออกจากสื่อเหล่านี้ จะต้องใช้อุปกรณ์ที่เหมาะสมกับสื่อนั้นๆ เช่น เครื่องขับจานแม่เหล็กใช้จะเพื่อการอ่านหรือบันทึกข้อมูลลงในจานแม่เหล็ก หรือเครื่องขับเทปแม่เหล็กจะใช้เพื่อการอ่านหรือบันทึกข้อมูลลงบนเทปแม่เหล็ก เป็นต้น

5.3.2.1 เทปแม่เหล็ก (Magnetic Tape)

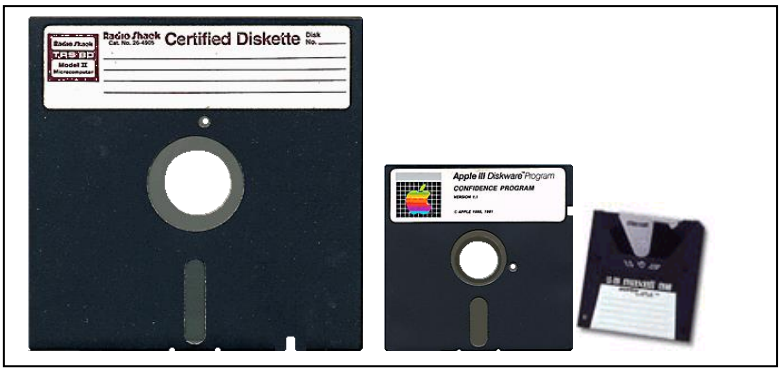
เทปแม่เหล็ก เป็นอุปกรณ์เก็บข้อมูลสำรองมีลักษณะคล้ายเทปเสียง มักจะใช้ในงานสำรองข้อมูล เผื่อไว้ในกรณีฉุกเฉิน โดยจะเข้าถึงข้อมูลแบบลำดับ (Sequence Access) นิยมใช้กับเครื่องคอมพิวเตอร์ขนาดกลางขึ้นไป ตัวเทปมีลักษณะเป็นสายเทปแบบม้วนเปลือย (Open Reel) หรือแบบตลับ (Cassette) ก็ได้ สายเทปทำด้วยพลาสติกชนิดพิเศษ เคลือบด้วยออกไซต์ของของเหล็ก (Iron Oxide) สารป้องกันการสึกหรอ และสารก่อเกิดจุดแม่เหล็ก (Magnetized Spot) โดยมีมาตรฐานความกว้างของสายเทป คือขนาด 1/2 , 3/4 และ 1 นิ้ว มีความยาวตั้งแต่ 600 - 3,600 ฟุต ม้วนเก็บในวงพลาสติก (Reel) ขนาดเส้นผ่าศูนย์กลางประมาณ 10 นิ้ว การอ่านเขียนจะอาศัยเครื่องอ่าน/เขียนเทป (Tape Drive) บรรจุเทป 2 ข้าง ทั้งด้านซ้ายด้านขวา โดยเครื่องจะหมุนเทปจากม้วนเทปเต็มด้านซ้าย เรียกว่า ม้วนเทปแฟ้มข้อมูล (File/Supply Reel) ไปยังม้วนเทปเปล่าด้านขวา เรียกว่า ม้วนเทปประจำเครื่อง (Machine/Take-up Reel) ม้วนเทปจะมีอุปกรณ์ป้องกันการเขียนหรือแก้ไขบริเวณส่วนกลางของม้วนเทป เรียกว่า วงแหวนป้องกันแฟ้มข้อมูล (File Protection Ring) การบันทึกข้อมูลจะอาศัยการหมุนสายเทปผ่านหัวอ่าน/เขียนข้อมูล ซึ่งจะทำให้สารแม่เหล็กที่เคลือบบนสายเทป รวมตัวกันเป็นจุดแม่เหล็ก ซึ่งมีความเข้มมากน้อยตามรหัสของข้อมูลที่ได้รับจากหัวเทป และสามารถบันทึกซ้ำได้ประมาณ 20,000 - 50,000 ครั้ง ขึ้นอยู่กับคุณภาพของเนื้อเทปและสารเคลือบ ความจุข้อมูลพิจารณาจากจำนวนข้อมูลต่อความยาวของสายเทป 1 นิ้ว (Bpi : Byte per Inch) โดยทั่วไปมีความจุ 5 - 28 MB และโดยทั่วไปจะมี 2 แบบ คือ แบบ 7 Track และ 9 Track โดยใช้รหัส BCD และ EBCDIC ตามลำดับ

ภาพที่ 5.2 แสดงตัวอย่างของเทปแม่เหล็ก

ที่มา (http://commons.wikimedia.org/wiki/File:Magnetic_tape_hg.jpg, 2552)

5.3.2.2 ฟลอบบี้ดิสก์เก็ต (Floppy Diskette)

ฟลอบบี้ดิสก์ หรือดิสก์เก็ต เป็นอุปกรณ์สำหรับเก็บข้อมูล สามารถเก็บบันทึกข้อมูล หรือ ลบข้อมูล และบันทึกใหม่ได้ มีลักษณะกลมบาง ทำจากสารไมล่าร์ (Mylar) ที่ฉาบด้วยสารแม่เหล็ก บรรจุในซองพลาสติกแข็ง เพื่อป้องกันฝุ่นละอองและการขูดขีด มีขนาด 8 นิ้ว ขนาด 5.25 นิ้ว และ ขนาด 3.25 นิ้ว ได้รับความนิยมสูง เพราะมีขนาดเล็ก และ สะดวกในการพกพา



ภาพที่ 5.3 แสดงตัวอย่างของฟลอบบี้ดิสก์เก็ต

ที่มา (http://oldcomputers.net/floppydisks.html, 2552)

5.3.2.3 ฮาร์ดดิสก์ (Hard Disk)

เป็นอุปกรณ์หน่วยความจำสำหรับเก็บข้อมูลขนาดใหญ่ มีความจุสูงถึงหน่วยเมกะไบต์ (Mega Byte) จนถึงกิกะไบต์ (GB : Giga Byte) และมีความเร็วสูงในการทำงาน และ การส่งผ่านข้อมูลมากกว่าหน่วยความจำรองทั่วไป ซึ่ง ฮาร์ดดิสก์จะประกอบไปด้วยจาน (Disk) หรือที่เรียกว่า แพลตเตอร์ (Platters) หลายๆ แผ่นมารวมกัน ซึ่งแต่ละด้านของแพลตเตอร์ จะถูกปกคลุมไปด้วยสารประกอบออกไซด์ (Oxide) เพื่อให้สามารถบันทึกข้อมูลได้ ฮาร์ดดิสก์ส่วนมากจะอยู่ภายในเครื่องคอมพิวเตอร์ ซึ่งไม่สะดวกในการเคลื่อนย้าย บางครั้งถูกเรียกว่าฟิกซ์ดิสก์ (Fixed Disk)

การทำงานของฮาร์ดดิสก์ มีลักษณะคล้ายๆกับแผ่นดิสก์ โดยก่อนที่จะทำการบันทึกข้อมูล จำเป็นจะต้อง Format เพื่อให้มีการกำหนด Track และ Cylinder ขึ้นมาก่อนเพื่อใช้ในการอ้างอิงตำแหน่ง นอกจากนี้แล้วมันยังสามารถจัดแบ่ง Partitions กล่าวคือ ฮาร์ดดิสก์ ตัวหนึ่งสามารถแบ่งได้หลาย Partition ขึ้นอยู่กับการแบ่ง Partition ก่อนการ Format นอกจากนี้ยังขึ้นกับเครื่องคอมพิวเตอร์ว่าใช้ระบบ PCI (Peripheral Component Interconnect) หรือไม่ ถ้าไม่ใช้ระบบ

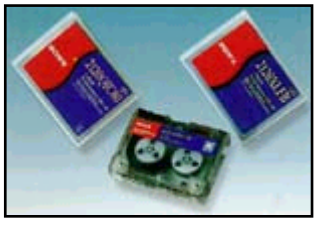
PCI ในเครื่องจะมองเห็นฮาร์ดดิสก์ขนาดสูงสุดเพียง 540 เมกะไบต์ แต่ถ้าเป็น PCI จะต้องตรวจสอบระบบปฏิบัติการว่าเป็นระบบใด เช่น หากเป็น Windows 95 จะสามารถมองเห็นเนื้อที่ฮาร์ดดิสก์ สูงสุดได้ที่ 1.27 กิกะไบต์ ต่อ 1 Partition ซึ่งถ้ามีฮาร์ดดิสก์ 1 ตัว แต่เป็น 2 กิกะไบต์ก็ต้องจัดแบ่งเป็น 2 Partition ถ้าเป็นระบบ Windows 95 OSR2 ก็จะสามารถมองเห็นเนื้อที่ฮาร์ดดิสก์ได้เกิน 2 กิกะไบต์ และถ้าเป็น ระบบปฏิบัติการ Linux จะสามารถมองเห็นฮาร์ดดิสก์ได้มากถึง 4 เทราไบต์ (TB : Tera Byte) เป็นต้น

ภาพที่ 5.4 แสดง ตัวอย่างของฮาร์ดดิสก์

ที่มา (http://www.electron.rmutphysics.com/science-news/, 2552)

5.3.2.4 ทปคาร์ทริดจ์ (Cartridge Tape)

เทปคาร์ทริดจ์ มีจุดเด่นตรงสามารถบันทึกข้อมูลซ้ำได้หลายครั้ง และมีความจุสูงถึงระดับกิกะไบต์ คือ ตั้งแต่ 1 กิกะไบต์ขึ้นไปสูงถึง 14 กิกะไบต์ มีลักษณะเทปคล้ายเทปคาสเซ็ทเป็นม้วนยาว 112 เมตร ใช้สำหรับการบันทึกข้อมูลที่มีจำนวนมาก เช่น การสำรองข้อมูลขององค์กรขนาดใหญ่ ใช้เป็นสื่อกลางในการบันทึกข้อมูลดาวเทียม



ภาพที่ 5.5 แสดง ตัวอย่างของเทปคาร์ทริดจ์

ที่มา (http://www.electron.rmutphysics.com/science-news/, 2552)

5.3.2.5 สื่อบันทึกข้อมูล ประเภทซีดี (CD : Compaq Disk)

เป็นอุปกรณ์ที่ใช้ลำแสงเลเซอร์ในการอ่านและเขียนข้อมูลมีทั้งชนิดอ่านได้อย่างเดียว ซึ่งเรียกว่า ซีดีรอม (CD-ROM : Compact Disk Read Only Memory) และชนิดที่สามารถอ่านและอ่านและเขียนได้ เรียกว่า ซีดีอาร์ (CD-R : Compact Disk Recordable) ปกติแล้วการบันทึกข้อมูลลงซีดีจะบันทึกได้เพียงครั้งเดียว ในความจุ 650 MB แต่มีเครื่องบันทึกซีดีที่ออกมารองรับการบันทึกข้อมูลได้มากกว่า 1 ครั้ง เรียกว่า ซีดีอาร์ดับเบิ้ลยู (CD-RW : Compact Disk Rewritable) สามารถลบข้อมูลในแผ่นและบันทึกใหม่ได้

ภาพที่ 5.6 แสดง ตัวอย่างของสื่อบันทึกข้อมูล ประเภท CD
ที่มา (http://www.electron.rmutphysics.com/science-news/, 2552)

5.3.2.6 Removable Disk อื่นๆ

มีลักษณะคล้ายดิสเก็ต แต่มีความจุสูงใกล้เคียงกับฮาร์ดดิสก์ สามารถติดตั้งได้ทั้งภายในและภายนอกเครื่อง เช่นแผ่นบันทึกข้อมูล Zip drive , Jaz drive, Syquest ปัจจุบันสามารถบันทึกข้อมูลได้สูงถึง กิกะไบต์

ภาพที่ 5.7 แสดงตัวอย่างลักษณะทางกายภาพของ สื่อบันทึกข้อมูลประเภท ดิสก์ชนิดอื่นๆ
ที่มา (http://www.electron.rmutphysics.com/science-news/, 2552)

นอกจากนี้ยังมีแฟลชไดร์ฟ (Flash Drive) ซึ่งมีชื่อเรียกได้หลายชื่อ เช่น ธัมไดร์ฟ (Thumb Drive) เนื่องจากในยุคแรกมีขนาดเท่ากับนิ้วหัวแม่มือ (Thumb) หรือ แฮนดี้ไดร์ฟ (Handy Drive) เนื่องจากสามารถใช้มือในการเสียบและดึงอุปกรณ์ออกจากเครื่องคอมพิวเตอร์ได้อย่างสะดวก แฟลชไดร์ฟทำงานเหมือนกับหน่วยความจำในเครื่องคอมพิวเตอร์ แต่สามารถจัดเก็บข้อมูลได้โดยไม่ต้องมีกระแสไฟฟ้าเลี้ยง แฟลชไดร์ฟเชื่อมต่อกับคอมพิวเตอร์ผ่านทางพอร์ตยูเอสบี (USB : Universal Serial Bus) โดยมีความจุตั้งแต่ 2GB 4GB 8GB 16GB และมีการพัฒนาเพิ่มความจุไปเรื่อยๆ

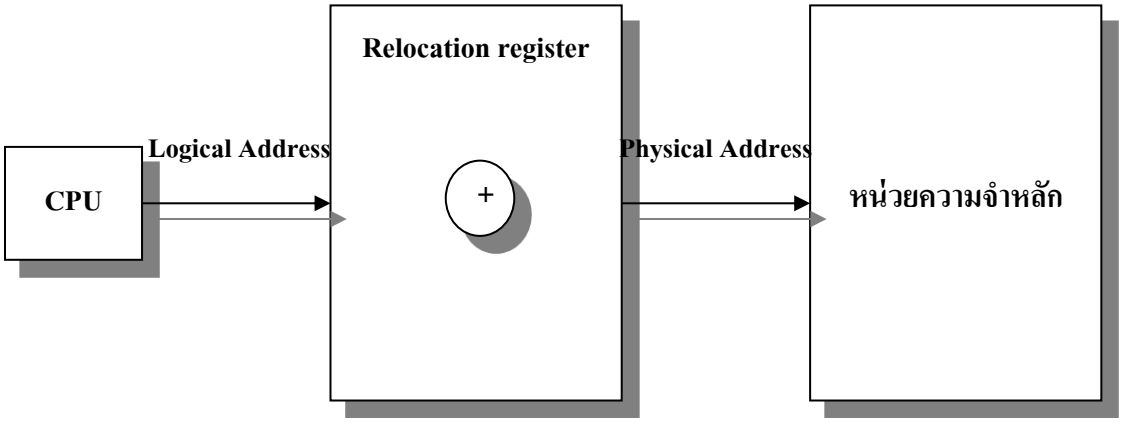
ภาพที่ 5.8 แสดงตัวอย่างลักษณะทางกายภาพของ แฟลชไดร์ฟ
ที่มา (http://www.electron.rmutphysics.com/science-news/, 2552)

5.4 เทคนิคการจัดการหน่วยความจำหลัก

ในหน่วยความจำหลัก จะประกอบไปด้วยที่เก็บข้อมูลย่อยที่มีขนาดเป็นไบต์ (Byte) ซึ่งโดยแต่ละไบต์ จะมีแอดเดรส (Address) บอกตำแหน่งของข้อมูลว่าอยู่ ณ ตำแหน่งใดของหน่วยความจำ แอดเดรส หรือ ตำแหน่งของหน่วยความจำ แบ่งออกเป็น 2 แบบ

1. ตำแหน่งทางกายภาพ (Physical Address) คือ ตำแหน่งจริงของอุปกรณ์หน่วยความจำ
2. ตำแหน่งทางตรรกะ (Logical Address) คือ ตำแหน่งที่ถูกสร้างโดยซีพียู

(พิรพร หมุนสนิท และคนอื่น ๆ, 2553)



ภาพที่ 5.9 แสดงการแปลงตำแหน่งของหน่วยความจำ

ที่มา (http://www.bncc.ac.th/childweb/OS/html/l5-3-1.html, 2552)

จากภาพที่ 5.9 การอ้างอิงตำแหน่งในหน่วยความจำ จะต้องนำค่าอ้างอิงที่เรียกว่าตำแหน่งทางตรรกะ มาบวกกับ ค่ารีจิสเตอร์พื้นฐาน (Base Register) เพื่อให้ได้ตำแหน่งทางกายภาพของหน่วยความจำ

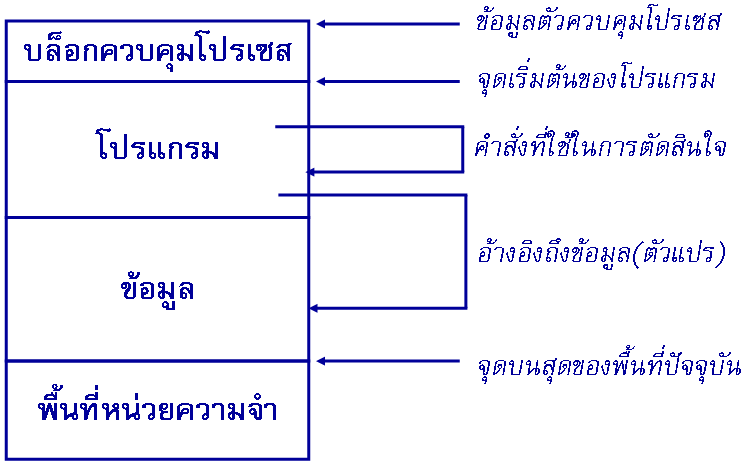
วิธีต่างๆ ที่ใช้ในการจัดการหน่วยความจำนั้น โดยทั่วไปแล้วต้องมีความสามารถพื้นฐานหรือมีกระบวนการพื้นฐาน 5 กระบวนการดังนี้ (นรีรัตน์ นิยมไทย, 2549)

1. การย้ายตำแหน่ง (Relocation)
2. การป้องกันพื้นที่ (Protection)
3. การใช้พื้นที่ร่วมกัน (Sharing)
4. การจัดการแบ่งทางตรรกะ (Logical Organization)
5. การจัดการแบ่งทางกายภาพ (Physical Organization)

1. การย้ายตำแหน่ง (Relocation)

ในการทำงานหลาย ๆ โปรแกรมพร้อมกัน หรือระบบมัลติโปรแกรมมิ่ง (Multiprogramming system) จะต้องมีการใช้ทรัพยากรต่างๆ ร่วมกัน ดังนั้นเมื่อมีการนำโปรเซสเข้าไปยังหน่วยความจำหลายๆ งานพร้อมกันจะต้องมีการจัดการที่ดี เพราะจะมีการนำงานเข้า และออกจากหน่วยความจำบ่อยครั้ง ซึ่งการนำโปรเซสเข้านั้นไม่สามารถคาดเดาได้ว่าโปรเซสนั้นจะใช้

หน่วยความจำไปเท่าใด เพราะบางครั้งหลังจากที่นำโปรเซสเข้าไปในหน่วยความจำแล้ว อาจจะมีการนำโปรเซสของโปรแกรมเดิมเข้ามาเพิ่มเติมอีก ดังนั้นจึงต้องมีตัวที่คอยจัดการว่า โปรเซสใดเข้ามาที่ตำแหน่งใด และโปรเซสใดที่ออกไปแต่ยังทำงานไม่เสร็จ และโปรเซสนั้นทำงานได้แค่ไหนแล้ว เพื่อที่เมื่อโปรเซสนั้นกลับมาทำงานอีกจะสามารถทำงานเดิมต่อไปได้ เพื่อแก้ปัญหาในการเปลี่ยนตำแหน่งของโปรเซสจึงได้มีการกำหนดแอดเดรสขึ้น



ภาพที่ 5.10 แสดงการกำหนดแอดเดรสให้หน่วยความจำ
ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

การกำหนดแอดเดรสนี้จะทำให้สามารถระบุได้ว่าโปรเซสนั้นๆ เริ่มต้นและสิ้นสุด ณ ตำแหน่งใด เช่นโปรเซส 1 เริ่มต้นที่แอดเดรส 0 และโปรเซสมีความยาว 520 ส่วนโปรเซสที่ 2 มีความยาว 300 จะมีการกำหนดว่าโปรเซสแรกมีจุดเริ่มต้นที่แอดเดรส 0 และสิ้นสุด ที่แอดเดรส 520 ส่วนโปรเซสที่ 2 มีจุดเริ่มต้นแรกที่แอดเดรส 521 และสิ้นสุดที่แอดเดรส 820 ถ้าต้องการเรียกข้อมูล ณ แอดเดรสที่ 100 ของโปรเซสที่ 2 เราจะใช้แอดเดรสเริ่มต้นของโปรเซสเป็นจุดอ้างอิงในการเรียก ซึ่งในกรณีนี้ต้องไปเรียกข้อมูลจากแอดเดรสที่ 620 (520+100)

2. การป้องกันพื้นที่ (Protection)

ในการทำงานที่มีหลายงานพร้อมกัน การจะนำโปรเซสเข้าไปในหน่วยความจำ จำเป็นต้องมีการตรวจสอบว่าพื้นที่ ที่จะนำงานเข้าไปนั้นว่างหรือไม่ ซึ่งก็เป็นการลำบากที่จะหาว่างานที่มีอยู่ก่อนหน้านั้นมีขนาด หรือจะสิ้นสุดที่ใด เพราะอาจจะมีการขอเนื้อที่

หน่วยความจำเพิ่มขึ้นอีกก็ได้ โดยเฉพาะการทำงานของอาร์เรย์ที่ไม่กำหนดขอบเขต หรือในกรณีที่อาจจะมีการเรียกใช้ ข้อมูลจากโปรเซสอื่น ซึ่งอาจจะส่งผลให้การทำงานของโปรเซสผิดพลาดได้ ดังนั้นจึงต้องมีการป้องกันพื้นที่ของแต่ละโปรเซสเพื่อห้าม หรือจำกัดสิทธิในการเข้ามาใช้หน่วยความจำ

การป้องกันพื้นที่หน่วยความจำนั้นจะเป็นหน้าที่ของโปรเซสเซอร์(ฮาร์ดแวร์) มากกว่าจะเป็นหน้าที่ของระบบปฏิบัติการ (ซอฟต์แวร์) ทั้งนี้เนื่องจากระบบปฏิบัติการไม่สามารถคาดเดาถึงแอดเดรสที่อ้างถึงทั้งหมดของโปรแกรมได้ ถึงแม้ว่าระบบจะสามารถทำการคาดเดาได้ กระบวนการดังกล่าวจะกินเวลาในการค้นหาแอดเดรสที่อ้างถึงของโปรแกรมแต่ละตัวมากเกินไป

3. การใช้พื้นที่ร่วมกัน (Sharing)

ถึงแม้จะมีการป้องกันพื้นที่ แต่ก็ยังมีงานบางอย่างที่อาจจะอนุญาตให้โปรเซสอื่นเข้ามาเรียกใช้ข้อมูลได้ เพราะในบางครั้งอาจมีโปรเซสหลายโปรเซสที่มีการทำงาน หรือมีหน้าที่คล้ายๆกัน ดังนั้นการที่จะดึงโปรเซสที่เหมือนกันมาเพียงโปรเซสเดียว แล้วให้โปรเซสที่จะเรียกใช้มาเรียกใช้จากโปรเซสนี้ร่วมกันจะทำให้ประหยัดเนื้อที่หน่วยความจำไปได้ แต่การเรียกใช้โปรเซสที่ใช้งานร่วมกัน ไม่ควรจะมีการเข้าไปแก้ไขโปรเซสนั้น เพราะจะส่งผลให้โปรเซสอื่นทำงานผิดพลาดได้

4. การจัดแบ่งทางตรรกะ (Logical Organization)

เป็นการแบ่งโปรแกรมออกเป็นโมดูลย่อยๆ ซึ่งหากโมดูลย่อยใดที่ไม่ถูกเรียกใช้ก็จะไม่ถูกบรรจุลงในหน่วยความจำ เมื่อต้องการเรียกใช้งานจึงค่อยตรวจสอบว่ามีโมดูลย่อยนั้นหรือยัง ถ้ายังไม่มีจึงค่อยดึงมาบรรจุลงในหน่วยความจำ วิธีนี้มีข้อดีคือ

1) หากโปรแกรมย่อยใดไม่ถูกใช้งานจะไม่ถูกนำไปไว้ในหน่วยความจำ ทำให้ประหยัดเนื้อที่ และใช้หน่วยความจำน้อยกว่าขนาดโปรแกรมทั้งหมด

2) โปรแกรมย่อยแต่ละตัวสามารถเรียกใช้และคอมไพล์แยกกันได้

3) โปรแกรมย่อยแต่ละตัวสามารถมีระดับการป้องกันที่แตกต่างกันได้

5. การจัดแบ่งทางกายภาพ (Physical Organization)

หน่วยความจำของระบบคอมพิวเตอร์มี 2 ระดับ คือหน่วยความจำหลัก และหน่วยความจำสำรอง หน่วยความจำหลักได้แก่แรม ซึ่งมีขนาดไม่ใหญ่ ราคาแพง เก็บข้อมูลได้

ชั่วคราวแต่มีความเร็วสูง ส่วนหน่วยความจำชั่วคราว ได้แก่ดิสก์ หรือฮาร์ดดิสก์ มีการอ่านข้อมูลได้ช้า แต่มีขนาดใหญ่ ราคาถูก และข้อมูลจะไม่ลบเลือน

ในระบบที่มีหน่วยความจำ 2 แบบดังกล่าว ระบบควรคำนึงถึงการจัดการ ในการเคลื่อนย้ายข้อมูลระหว่างหน่วยความจำทั้งสอง โดยอาจจะยกความรับผิดชอบดังกล่าวให้โปรแกรมหนึ่งๆ จัดการ แต่ผลลัพธ์ที่ได้ส่วนใหญ่จะไม่สามารถใช้งานได้หรือทำงานได้ไม่ตามที่ต้องการด้วยสาเหตุว่า

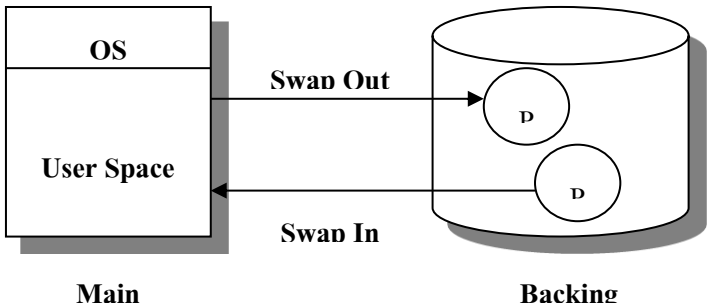
1) หน่วยความจำหลักอาจจะมีไม่พอเพียงสำหรับโปรแกรมนั้น รวมทั้งข้อมูลที่โปรแกรมใช้ ซึ่งจะทำให้โปรแกรมเมอร์นั้นจะต้องใช้วิธีการแบ่งส่วน (Overlay) โดยโปรแกรมและข้อมูลนั้นจะถูกจัดการแบ่งให้เป็นโปรแกรมย่อย ๆ ที่สามารถใช้งานพื้นที่ในหน่วยความจำเดียวกันได้ และหน่วยความจำหลักจะเป็นตัวที่ทำการสับเปลี่ยนโปรแกรมย่อยเข้าและออกเมื่อต้องการ

2) ในระบบมัลติโปรแกรมมิ่งที่สามารถทำงานโปรแกรมหลายๆ โปรแกรมพร้อมกันได้นั้น โปรแกรมเมอร์จะไม่สามารถทราบได้เลยว่าพื้นที่หน่วยความจำจะว่างเมื่อใด และอยู่ที่ไหน

จากวิธีการพื้นฐานทั้ง 5 กระบวนการที่กล่าวมา จึงมีการคิดค้นเทคนิคการจัดการหน่วยความจำขึ้นมาหลายเทคนิค เพื่อใช้จัดการกับหน่วยความจำที่มีอยู่อย่างจำกัดในคอมพิวเตอร์แต่ละเครื่องให้ถูกนำไปใช้ประโยชน์อย่างเต็มประสิทธิภาพ เทคนิคการจัดการหน่วยความจำที่ระบบปฏิบัติการต่างๆ นำไปประยุกต์ใช้มีดังนี้ (น.ท.ไพศาล โมลิสกุลมงคล และคนอื่น ๆ, 2545)

5.4.1 การสับเปลี่ยน (Swapping)

ทุกโปรเซสเมื่ออยู่ในสถานะทำงาน จำเป็นต้องอยู่ในหน่วยความจำเสมอแต่อาจจะถูกโยกย้ายไปเก็บไว้ชั่วคราวที่ หน่วยเก็บโปรแกรมชั่วคราว (Backing Store) ในระบบการจัดตารางงานให้กับซีพียู แบบวนรอบ เมื่อหมดช่วงเวลาในการประมวลผล ตัวจัดการหน่วยความจำจะทำการย้ายโปรเซสปัจจุบันออกไปเก็บไว้ที่หน่วยเก็บโปรแกรมชั่วคราว แล้วนำโปรเซสถัดไปที่อยู่ในคิวมาประมวลผล

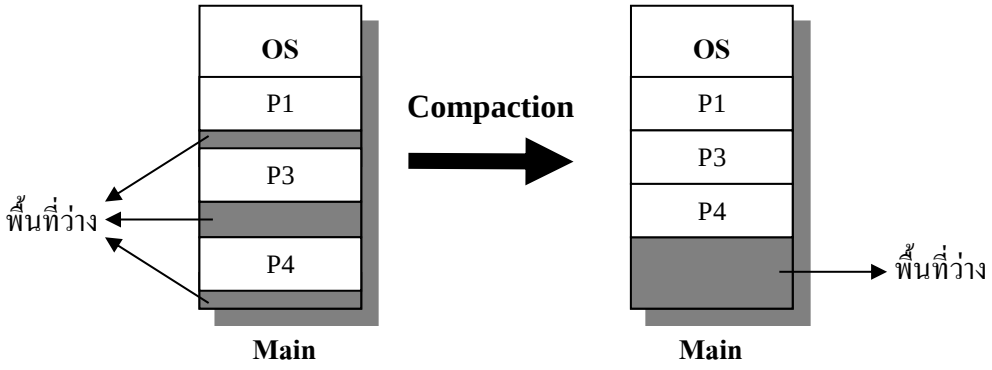


ภาพที่ 5.11 แสดงการจัดการหน่วยความจำแบบการสับเปลี่ยน
ที่มา (http://cptd.chandra.ac.th/selfstud/os1/Mp4.html, 2552)

วิธีการจัดการหน่วยความจำแบบการสับเปลี่ยน นี้มีใช้ในระบบคอมพิวเตอร์ UNIX ยุคต้นๆ ซึ่งเป็นยุคที่คอมพิวเตอร์มีหน่วยความจำอย่างจำกัด

5.4.2 การจัดสรรพื้นที่ที่อยู่ติดกัน (Contiguous Allocation)

ทุกโปรเซสเมื่ออยู่ในสถานะทำงานก็จะเข้ามาใช้เนื้อที่ของหน่วยความจำที่ยังว่างอยู่ หากโปรเซสใดมีขนาดใหญ่กว่าพื้นที่ว่างของหน่วยความจำ ก็มีความจำเป็นต้องรอให้หน่วยความจำมีพื้นที่ว่างพอที่จะให้โปรเซสเข้าไปใช้เนื้อที่ของของหน่วยความจำได้ และในหน่วยความจำมักเกิดพื้นที่ว่างเป็นช่วงๆ ซึ่งทำให้เกิดการสิ้นเปลืองเนื้อที่ของหน่วยความจำไปโดยเปล่าประโยชน์ ดังนั้นจำต้องทำการบีบอัด (Compaction) คือการโยกย้ายพื้นที่หน่วยความจำว่างๆ เหล่านั้นให้มาเป็นพื้นที่ว่างติดกันให้ได้ โดยอาศัยวิธีการสับเปลี่ยน เข้ามาช่วย

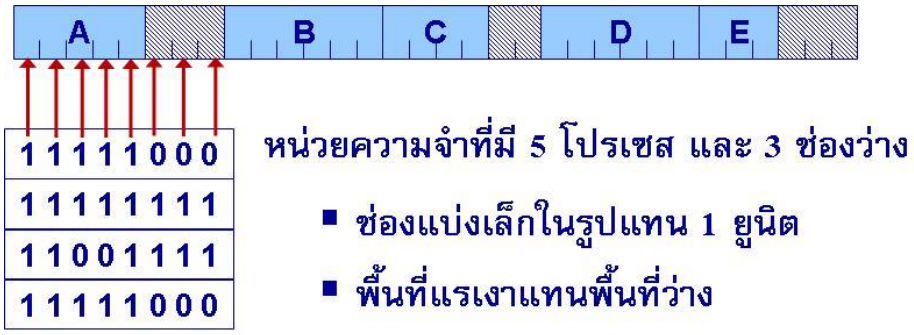


ภาพที่ 5.12 แสดงการจัดสรรพื้นที่ที่อยู่ติดกัน
ที่มา (http://csnon04.blogspot.com/, 2552)

ซึ่งการทำการบีบอัดนั้นโปรเซสทุกๆ Pโปรเซสจะต้องหยุดการทำงานชั่วคราว (Waiting) เพื่อทำการสับเปลี่ยน ทุกๆ โปรเซส ไปเก็บไว้ในอีกพื้นที่หนึ่ง จึงจะทำการบีบอัดได้ทำให้เสียเวลาในการทำงาน

5.4.3 การจัดการแบบบิตแมพ (Memory Management with Bitmap)

การจัดการแบบนี้จะทำการแบ่งหน่วยความจำออกเป็นหน่วย หรือยูนิต(Unit) ซึ่งแต่ละยูนิตอาจจะมีขนาดใหญ่ หรือเล็กก็ได้ ในแต่ละยูนิตนั้นจะแทนค่า 1 บิตเสมอ เช่น ถ้า หน่วยความจำมี 1000 ยูนิต ก็จะมี 1000 บิต เราเรียก บิต เหล่านี้ว่าระบบบิตแมพนั่นเอง ถ้ายูนิตใดมีข้อมูลอยู่ บิตของยูนิตนั้นจะมีค่าเป็น 1 ถ้ายูนิตนั้นว่าง บิตจะมีค่าเป็น 0



ภาพที่ 5.13 แสดง บิตแมพที่ได้จากหน่วยความจำ
ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

การกำหนดขนาดของยูนิต เป็นสิ่งสำคัญอย่างหนึ่งของการออกแบบในระบบบิตแมพ เพราะถ้ากำหนดยูนิตขนาดเล็ก ระบบจะมีบิตแมพขนาดใหญ่ แต่ถ้ากำหนดยูนิตขนาดใหญ่ บิตแมพก็จะมีขนาดเล็ก แต่อาจจะทำให้สูญเสียหน่วยความจำในยูนิตสุดท้ายได้ถ้าขนาดของโปรเซสมีขนาดไม่เท่ากับจำนวนของยูนิต

การจัดการแบบนี้จะตรวจสอบง่าย เพราะมีจำนวนบิตคงที่ แต่จะมีข้อเสียคือ

1. จะเกิดการสูญเสียพื้นที่ภายใน (Internal Fragmentation) ในยูนิตสุดท้ายของโปรเซส

2. จะเกิดการสูญเสียพื้นที่ภายนอก (External Fragmentation) คือในส่วนที่เป็น 0 ที่กระจายอยู่ในตารางบิตแมพ

3. การจะนำงานขนาด k ยูนิตมาใส่ จะต้องทำการค้นหาบิตที่เป็น 0 ติดต่อกัน k บิตให้ได้ก่อน ซึ่งจะทำให้เสียเวลาในการค้นหา

5.4.4 การจัดการแบบลิงค์ลิสต์ (Memory Management with Link List)

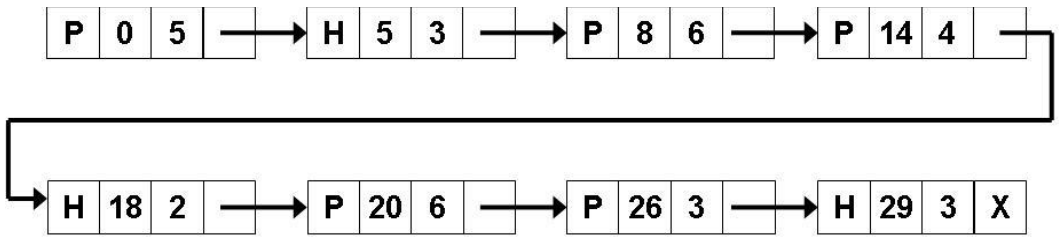
การจัดการแบบลิงค์ลิสต์จะมีการระบุการใช้พื้นที่หน่วยความจำโดยระบุว่ามีโปรเซสเริ่มต้นอยู่ที่แอดเดรสใด และมีความยาวของโปรเซสเท่าใด โดยมีรูปแบบการจัดเก็บข้อมูลดังนี้

1. ในส่วนแรก หากเป็นพื้นที่ว่างจะแสดงด้วยตัวอักษร H (Hold) ถ้าเป็นช่วงที่มีโปรเซสจะขึ้นต้นด้วย P (Process)

2. ในส่วนที่ 2 จะบอกแอดเดรสที่ส่วนนั้นเริ่มต้นของพื้นที่ว่าง หรือโปรเซส

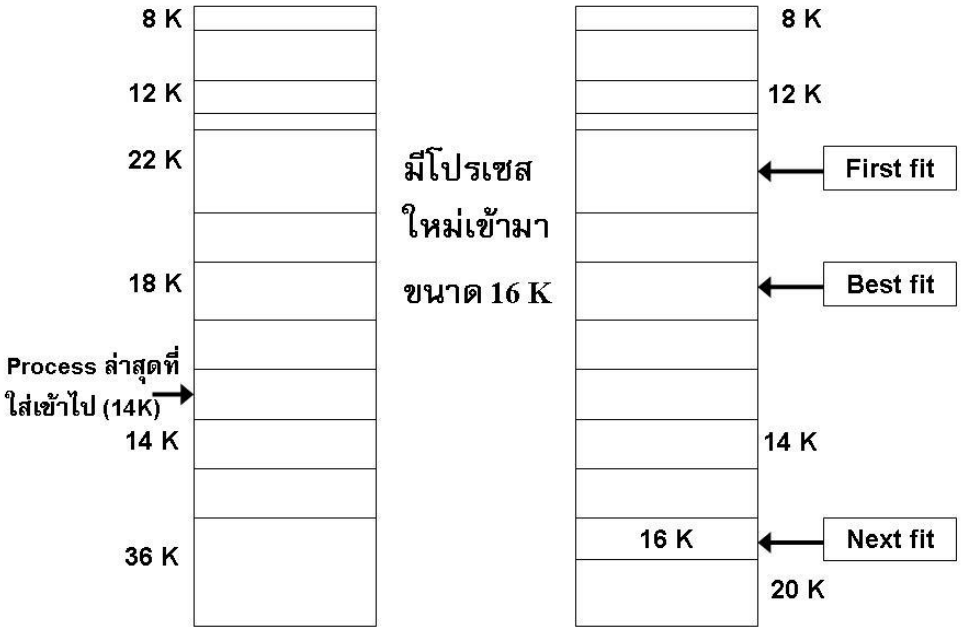
3. ในส่วนที่ 3 เป็นความยาวของพื้นที่

4. ในส่วนที่ 4 เป็นส่วนที่ชี้ไปยังข้อมูลของพื้นที่ส่วนต่อไป



ภาพที่ 5.14 แสดงตัวอย่างการจัดการแบบลิงค์ลิสต์
ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

เมื่อจะมีการนำโปรเซสเข้ามาในหน่วยความจำ จะมีการเลือกว่าโปรเซสใดควรจะเข้าไปอยู่ในพื้นที่ว่างส่วนใด โดยมีหลักในการคัดเลือกดังนี้



ภาพที่ 5.15 แสดง ตัวอย่างการนำโปรเซสเข้ามาใช้งานในหน่วยความจำตามวิธีต่างๆ
ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

วิธีต่างๆ ของการนำโปรเซสเข้ามาใช้งานในหน่วยความจำ

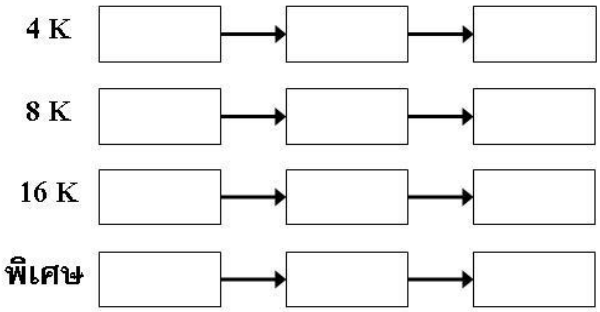
1. First Fit เป็นวิธีที่ง่ายที่สุด คือเลือกเอาพื้นที่ๆอยู่แรกสุดที่สามารถนำโปรเซสเข้าไปใส่ได้แต่แบบนี้จะไม่มีประสิทธิภาพมากนัก เพราะโปรเซสเล็กๆ อาจจะถูกเอาไปไว้ในพื้นที่ว่างที่มาก ทำให้สูญเสียหน่วยความจำไปมาก

2. Next Fit วิธีนี้ก็ไม่ได้ดีไปกว่าวิธีแรก เพราะเป็นการเลือกเอาพื้นที่ที่ใส่ได้อันที่อยู่ถัดจากการนำโปรเซสใส่ลงในหน่วยความจำครั้งล่าสุดเท่านั้น

3. Best Fit จะทำการค้นหาพื้นที่ว่างที่มีทั้งหมด แล้วเลือกเอาอันที่พอดีที่สุด วิธีนี้จะช่วยให้ใช้พื้นที่ได้มีประสิทธิภาพ แต่จะเสียเวลาในการค้นหาพื้นที่ทั้งหมดก่อนจึงจะทำการเลือกพื้นที่ๆ จะใส่ลงไป

4. Worst Fit จะทำการค้นหาพื้นที่ว่างที่มีทั้งหมด แล้วเลือกเอาส่วนที่มีพื้นที่ว่างมากที่สุดเพื่อหลีกเลี่ยงการเกิดปัญหาช่องว่างๆ เล็ก แต่ก็มีข้อเสียเหมือนแบบ Best Fit

5. Quick Fit เป็นการสร้างลิงค์ลิสต์ขึ้นมามากกว่า 1 ลิงค์ลิสต์ โดยแต่ละลิงค์ลิสต์จะมีขนาดโหนดแตกต่างกัน แต่ภายในลิงค์ลิสต์จะมีขนาดเท่ากัน



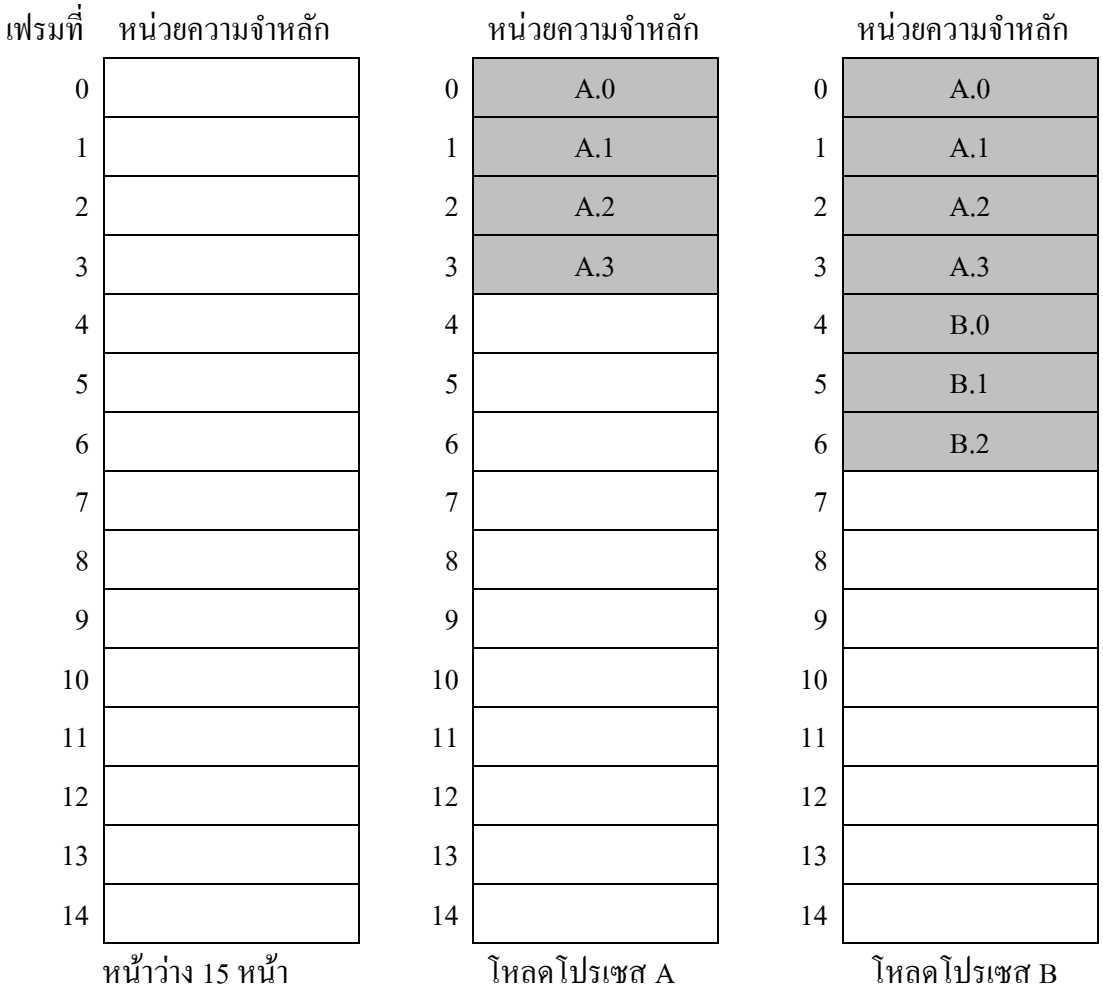
ภาพที่ 5.16 แสดงการทำงานของ Quick Fit
ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

โดยเมื่อมีโปรเซสเข้ามาก็จะดูว่าโปรเซสนั้นมีขนาดเท่าใด จากนั้นจึงนำโปรเซสนั้นไปบรรจุลงในลิงค์ลิสต์ที่มีขนาดโหนดเหมาะสมที่สุด

5.4.5 การแบ่งหน้า (Paging)

การจัดการหน่วยความจำแบบการแบ่งหน้า คือการแบ่งเนื้อที่ของหน่วยความจำออกเป็นส่วนเล็กๆ โดยมีขนาดเท่าๆ กัน ซึ่งเรียกว่า เฟรม (Frames) หรือ เพจเฟรม (Page Frames) โดยการจัดการหน่วยความจำแบบนี้จะมีการแบ่งโปรเซส ออกเป็นงานเล็กๆ เรียกว่า หน้า (Page) โดยทุกๆ หน้าจะมีขนาดเท่ากันหมด และมีเงื่อนไขคือ

1. ขนาดของ 1 หน้า จะต้องเท่ากับขนาดของ 1 เฟรม
2. ใน 1 เฟรม จะมีการนำเอาแค่ 1 หน้า มาบรรจุได้เท่านั้น เช่น หน่วยความจำหลักขนาด 1 MB และแบ่งออกเป็น 4 เฟรม ขนาดของเฟรมคือ 256 KB หน้าในดิสก์เท่ากับ 256 KB เหมือนกัน



ภาพที่ 5.17 แสดงการบรรจุหน้าของโปรเซสลงในเฟรม

ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

เฟรมที่	หน่วยความจำหลัก	เฟรมที่	หน่วยความจำหลัก	เฟรมที่	หน่วยความจำหลัก
0	A.0	0	A.0	0	A.0
1	A.1	1	A.1	1	A.1
2	A.2	2	A.2	2	A.2
3	A.3	3	A.3	3	A.3
4	B.0	4		4	D.0
5	B.1	5		5	D.1
6	B.2	6		6	D.2
7	C.0	7	C.0	7	C.0
8	C.1	8	C.1	8	C.1
9	C.2	9	C.2	9	C.2
10	C.3	10	C.3	10	C.3
11		11		11	D.3
12		12		12	D.4
13		13		13	
14		14		14	
	โหลดโปรเซส C		ถอนโปรเซส B ออก		โหลดโปรเซส D

ภาพที่ 5.18 แสดงการบรรจุหน้าของโปรเซสลงในเฟรม (ต่อ)

ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

เพื่อความสะดวกในการค้นหาหน้าจึงต้องมีการสร้างตารางหน้า (Page Table) ขึ้นมา ในตารางหน้าจะมีข้อมูลดังนี้

1. Page Frame Number เป็นฟิลด์ที่สำคัญที่สุด เพราะจะเก็บข้อมูลว่า หน้านี้ไปอยู่ที่เฟรมหมายเลขอะไรในหน่วยความจำหลัก

2. Present/Absent มีขนาด 1 บิต ใช้บอกว่าหน้าปัจจุบัน มีอยู่ (Present) หรือไม่มีอยู่ (Absent) ในหน่วยความจำหลัก ถ้ามีอยู่จะมีค่าเป็น 1 ถ้า ไม่มีอยู่จะมีค่าเป็น 0

3. Protection เก็บข้อมูลการป้องกัน เช่นกำหนดว่าให้อ่านได้อย่างเดียว อ่านและเขียนได้ หรือประมวลผลได้ (Execute) โดยจะใช้เพียง 3 บิต คือบิตการอ่าน บิตการเขียน และบิตการประมวลผล (R, W, E)

4. Modified เป็นการระบุว่าหน้านั้นๆ เคยถูกเรียกใช้ และมีการแก้ไขหรือไม่ ถ้าหน้าใดที่เคยถูกเรียกใช้และมีการแก้ไขหรือไม่ โดยกำหนดเป็น 0 ถ้าไม่มีการถูกบันทึกหรือปรับปรุงเปลี่ยนค่า และ บิตนี้จะมีค่าเป็น 1 หรือเรียกว่า Dirty Bit ถ้ามีการแก้ไขหน้านั้นจะต้องถูกเขียนกลับลงดิสก์

5. Referenced เป็นส่วนที่บอกว่าหน้านี้เคยถูกเรียกใช้บ้างหรือยัง ถ้าเคยถูกเรียกใช้แล้วจะมีค่าเป็น 1 บิตนี้มีความสำคัญในการระบุว่าหน้าใดเคยถูกเรียกใช้ ซึ่งมีความสำคัญในกระบวนการเปลี่ยนหน้า

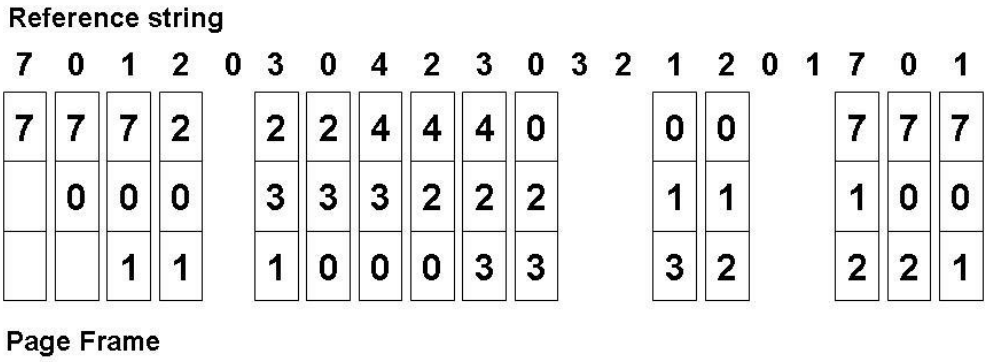
6. Caching Disable เป็นส่วนที่กำหนดว่าเราไม่ต้องทำการเก็บค่าแคชสำหรับหน้านั้น โดยทั่วไปแล้ว หน้าจะถูกโหลดเข้าไปในรีจิสเตอร์ของอุปกรณ์ที่จะติดต่อด้วย แต่ถ้าบิตนี้เป็น 1 จะไม่มีการโหลดหน้าเข้าไป

เมื่อมีหน้าของโปรเซสที่ต้องการจะนำเข้ามาในหน่วยความจำ แต่ภายในหน่วยความจำมีการพื้นที่ว่างไม่พอสำหรับหน้านั้น จึงต้องมีการสลับกับหน้าเดิมที่อยู่ในหน่วยความจำ ซึ่งในการสับเปลี่ยนหน้า (Page Replacement Algorithm) จะมีวิธีการตัดสินใจหลายแบบ ดังนี้

5.4.5.1 วิธีสับเปลี่ยนแบบมาก่อน-ออกก่อน (FIFO : First In-First Out Algorithm)

เป็นวิธีการสับเปลี่ยนหน้าที่ง่ายที่สุด โดยวิธีการนี้จะใช้เวลาที่หน้านั้นๆ ถูกนำเข้ามาในหน่วยความจำหลักเป็นเกณฑ์ในการตัดสินใจ เมื่อต้องการเลือกหน้าบางหน้าออก ก็ให้เลือกจากหน้าที่เข้ามานานที่สุดนั่นเอง ในทางปฏิบัติ เราอาจจะไม่ต้องบันทึกเวลาจริงๆ ที่หน้านั้น

เข้ามาก็ได้ เพียงแต่สร้างคิวแบบมาก่อน-ออกก่อน (FIFO Queue) สำหรับเก็บหมายเลขหน้าที่อยู่ในหน่วยความจำ เมื่อมีการนำหน้าใหม่เข้ามาให้เอาหมายเลขมาไว้ที่ปลายคิว แล้วเลือกหน้าออกจากหัวคิว



ภาพที่ 5.19 แสดงวิธีสับเปลี่ยนแบบมาก่อน-ออกก่อน
ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

จากรูปสมมติว่าเริ่มต้นในระบบมีแค่ 3 เฟรมและว่างอยู่

1. จากแถวของหน้าที่เข้ามาในเฟรมคือ (7, 0, 1) ซึ่งจะเกิดการผิดหน้าเนื่องจากหน้าทั้งสามยังไม่ได้อยู่ในเฟรม จึงมีการนำหน้าที่ต้องการเข้ามาในหน่วยความจำ

2. การเรียกหน้าต่อไปคือหน้า 2 จะถูกสับเปลี่ยนเข้ามาแทนหน้า 7 เพราะหน้า 7 เข้ามาก่อนเป็นหน้าแรก

3. ครั้งต่อไปเป็น หน้า 0 แต่หน้า 0 อยู่ในหน่วยความจำอยู่แล้ว จึงไม่เกิดการผิดหน้าและก็ไม่เกิดการสับเปลี่ยนหน้าด้วย

4. ครั้งต่อไป หน้า 3 จะถูกสับเปลี่ยนเข้ามาแทนหน้า 0

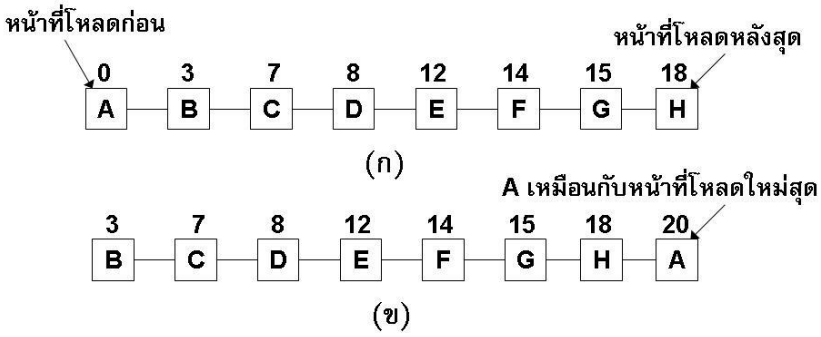
5. ครั้งต่อไปหน้า 0 จะถูกสับเปลี่ยนเข้ามาแทนหน้า 1 และต่อไปเรื่อย ๆ

การที่มีการสลับหน้าเกิดขึ้น เรียกว่าการผิดหน้า (Page Fault) ซึ่งจากตัวอย่างข้างบนจากตัวอย่างด้านบนแสดงให้เห็นว่าทุกๆ ครั้งที่เกิดการผิดหน้า หน้าอะไรที่จะถูกสับเปลี่ยนเข้ามาในเฟรม ซึ่งการผิดหน้าในตัวอย่างนี้มีทั้งหมด 15 ครั้ง การสับเปลี่ยนหน้าที่ไม่ดีจะเพิ่มอัตราการ ผิดหน้าได้ และทำให้การทำงานของโปรเซสช้าลง แต่จะไม่ทำให้ระบบทำงานผิดพลาด

5.4.5.2 วิธีสับเปลี่ยนแบบให้โอกาสครั้งที่สอง (Second Chance Page Replacement Algorithm)

จากวิธีสับเปลี่ยนแบบมาก่อน-ออกก่อน สามารถปรับปรุงได้โดยการป้องกันการเปลี่ยนหน้าที่ถูกเรียกใช้งานบ่อยออกไป ซึ่งสามารถทำได้โดยการเช็คที่บิต R (Referenced bit) ของหน้าที่เข้ามานานที่สุด ถ้าบิต R มีค่าเป็น 0 ก็แสดงว่าหน้านั้นเก่าและไม่ได้ถูกเรียกใช้งานเลย ระบบก็สามารถทำการสับเปลี่ยนได้ทันที แต่ถ้าบิต R มีค่าเท่ากับ 1 ก็ให้กำหนดบิต R นั้นเป็น 0 และนำหน้านั้นกลับไปเข้าแถวใหม่อีกครั้ง พร้อมกับทำการเปลี่ยนแปลงเวลาของหน้านั้นใหม่เหมือนดังหน้านั้นพึ่งเข้ามาในหน่วยความจำ จากนั้นก็ทำการค้นหน้าที่จะถูกสับเปลี่ยนต่อไป

วิธีนี้เรียกว่า การให้โอกาสครั้งที่สองกับหน้าที่เข้ามานานแต่ถูกเรียกใช้งานบ่อยนั่นเอง ในภาพ 5.18 (ก) จะเห็นได้ว่าหน้า A ถึง H นั้นจะถูกเก็บไว้ในลิงค์ลิสต์ และถูกจัดเรียงโดยเวลาที่หน้าเหล่านี้เข้ามาในหน่วยความจำ สมมติว่าการผิดหน้าเกิดขึ้นเมื่อเวลาเท่ากับ 20 หน้าที่อยู่มานานที่สุดคือ หน้า A ซึ่งเข้ามาตั้งแต่เวลาเท่ากับ 0 คือเมื่อโปรเซสเริ่มแรกทำงาน ถ้าบิต R ของหน้า A มีค่าเป็น 0 จะถูกสับเปลี่ยนออกไป แต่ถ้าบิต R เป็น 1 หน้า A จะถูกนำไปใส่ตอนท้ายของแถว และเวลาที่เข้ามาในลิสต์ก็ถูกเปลี่ยนเป็นเวลาปัจจุบันคือ 20 ดังภาพที่ 5.19 พร้อมกันนั้นบิต R ก็ถูกเปลี่ยนเป็น 0 หน้าที่จะถูกเช็คเพื่อสับเปลี่ยนต่อไปคือ หน้า B



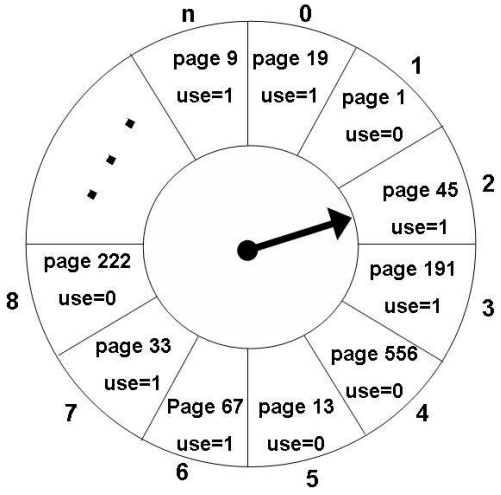
ภาพที่ 5.20 แสดงวิธีการสับเปลี่ยนหน้าแบบให้โอกาสครั้งที่สอง

ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

การทำงานของวิธีการนี้คือ การค้นหาหน้าที่เก่าที่สุดและไม่ได้ถูกอ้างอิงหรือเรียกใช้งาน แต่ถ้าทุกหน้าในระบบถูกเรียกใช้งานหมด วิธีการนี้ก็จะกลายเป็นวิธีสับเปลี่ยนหน้าแบบมาก่อน-ออกก่อน นั่นเอง

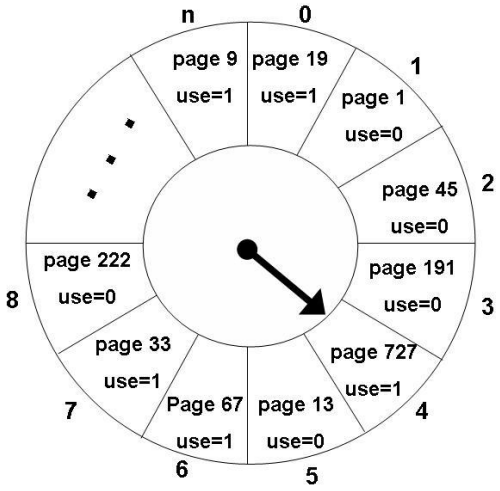
5.4.5.3 วิธีการสับเปลี่ยนแบบวงรอบนาฬิกา (Clock Page Replacement Algorithm)

ถึงแม้ว่าวิธีการแบบให้โอกาสครั้งที่สองจะเป็นวิธีการที่เป็นเหตุเป็นผลพอสมควร แต่ก็ยังเป็นวิธีที่ไม่มีประสิทธิภาพมากนัก เพราะการที่ระบบย้ายหน้าไปมาในลิสต์นั้นทำให้ระบบเสียเวลาพอสมควร วิธีการที่สามารถปรับปรุงวิธีการดังกล่าวได้ก็คือ ให้เรียงทุกเฟรมเป็นรูปวงกลมให้เหมือนรูปนาฬิกา ดังในภาพที่ 5.20 และมีเข็มนาฬิกาชี้ไปที่หน้าที่เก่าที่สุด



ภาพที่ 5.21 แสดงสถานะของบัฟเฟอร์ก่อนที่จะมีการสับเปลี่ยนหน้า

ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

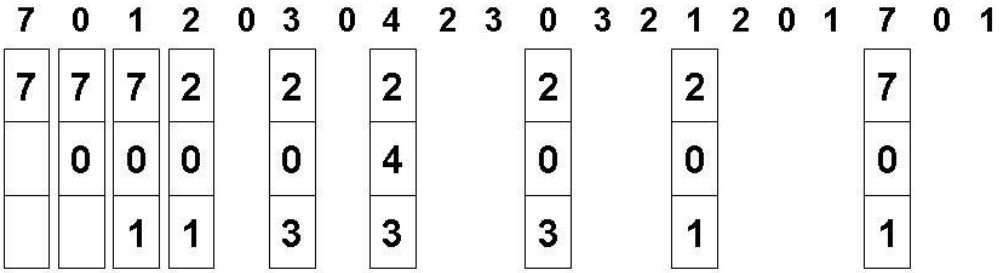


ภาพที่ 5.22 แสดง สถานะของบัฟเฟอร์หลังจากสับเปลี่ยนหน้าต่อไป
ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

เมื่อมีการผิดหน้าเกิดขึ้น หน้าที่มีเข็มนาฬิกาชี้จะถูกตรวจสอบ ถ้าบิต R มีค่าเป็น 0 หน้านั้นก็จะถูกสับเปลี่ยนออกไป และหน้าใหม่ก็จะถูกใส่เข้ามาในตำแหน่งเดิม พร้อมกันนั้นเข็มนาฬิกาจะทำการเลื่อนไปด้านหน้าอีก 1 ตำแหน่ง แต่ถ้าบิต R ถูกกำหนดเป็น 1 ก็ให้ลบค่าบิตนั้นเป็น 0 และเลื่อนเข็มไปหน้าถัดไป วิธีการดังกล่าวจะถูกทำซ้ำ จนกว่าเราจะได้หน้าที่มีบิต R เป็น 0 ซึ่งวิธีการนี้จะเหมือนวิธีการแบบให้โอกาสครั้งที่สอง เพียงแต่แตกต่างกันไปในวิธีการใช้งานเท่านั้น

5.4.5.4 วิธีสับเปลี่ยนแบบดีที่สุด (Optimal Page Replacement Algorithm)

วิธีสับเปลี่ยนหน้าแบบดีที่สุดนี้เรียกว่า OPT หรือ MIN ซึ่งมีวิธีการ คือให้เลือกสับเปลี่ยนหน้าที่จะไม่ถูกเรียกใช้งาน และมีระยะเวลาการรอเรียกใช้ที่นานที่สุด วิธีการนี้จะทำให้อัตราการผิดหน้าต่ำที่สุดสำหรับพื้นที่ที่มีจำนวนเฟรมหนึ่งๆ จากตัวอย่างหน้าที่เข้ามาใช้งานที่เหมือนกับหัวข้อวิธีสับเปลี่ยนแบบมาก่อน-ออกก่อน จะเห็นว่าวิธีที่ดีที่สุดนี ทำให้เกิดการผิดหน้าทั้งหมด 9 ครั้ง ดังในภาพที่ 5.22



ภาพที่ 5.23 แสดงวิธีสับเปลี่ยนหน้าแบบที่ดีที่สุด
ที่มา (http://csnon04.blogspot.com/2008/03/5.html, 2552)

ขั้นตอนการสับเปลี่ยนหน้าแบบดีที่สุด

1. การเรียก 3 หน้าแรกเข้ามาใช้งานทำให้เกิดการผิดหน้า 3 ครั้ง

2. การเรียกหน้า 2 จะสับหน้า 7 ออก เพราะหน้า 7 นั้น จะใช้งานอีกครั้งเป็นลำดับที่ 18, หน้า 0 จะใช้งานอีกเป็นลำดับที่ 5 และหน้า 1 อยู่ในลำดับที่ 14

3. การเรียกหน้า 3 ก็จะสับเปลี่ยนหน้า 1 ออก เพราะหน้า 1 เป็นหน้าสุดท้ายที่จะถูกเรียกใช้งาน ในขณะที่ หน้า 0 จะถูกเรียกเป็นลำดับที่ 1 และหน้า 2 เป็นลำดับที่ 3 เมื่อเทียบจากตำแหน่งที่หน้า 3 กำลังเข้าใช้งาน

ดังนั้นการจัดการหน่วยความจำแบบการแบ่งหน้า จึงสามารถทำให้โปรเซสอยู่ในหน่วยความจำได้โดยไม่ต้องเรียงต่อเนื่องกันตลอดทั้งโปรเซส จึงสามารถใช้เนื้อที่ว่างของหน่วยความจำที่ว่างกระจัดกระจายกันอยู่ได้โดยไม่ต้องมีการบีบอัดเนื้อที่ว่างของหน่วยความจำ การสับเปลี่ยนกระบวนการเข้าออกจากหน่วยความจำไปสู่ที่พักข้อมูลชั่วคราวจึงทำได้ง่ายขึ้น เนื่องจากทุกหน้ามีขนาดเท่ากันหมด แต่การเข้าถึงข้อมูลอาจมีความเร็วลดลงเล็กน้อยเนื่องจากงานย่อยของแต่ละโปรเซสอยู่กระจัดกระจายกัน วิธีการนี้เป็นที่นิยมในระบบปฏิบัติการหลายระบบ เพราะมีข้อดีกว่าวิธีการอื่นๆ ในขั้นต้น

5.4.6 การแบ่งเป็นตอน (Segmentation)

เป็นวิธีการจัดการหน่วยความจำหลักที่มีลักษณะการทำงานคล้ายกับการจัดการหน่วยความจำแบบการแบ่งหน้า แต่วิธีจัดการหน่วยความจำแบบการแบ่งเป็นตอนนั้นจะมีการแบ่งโปรเซส ออกเป็นงานย่อยๆ ที่ไม่จำเป็นต้องมีขนาดเท่ากันทุกงาน การแปลง Logical Address ให้เป็น Physical Address นั้นจะต้องใช้ข้อมูล 2 ส่วนคือ เลขที่ของเซ็กเมนต์ (Segment) และ ระยะเริ่มต้นจากเซ็กเมนต์นั้น (Offset)

การจัดการหน่วยความจำแบบการแบ่งเป็นตอน โปรแกรมเมอร์จะสามารถมองเห็นการทำงานของหน่วยความจำและมีส่วนร่วมในการจัดการกับโปรแกรมและข้อมูลของตนได้ โดยโปรแกรมเมอร์ หรือ คอมไพเลอร์ จะทำการแบ่งโปรแกรมออกเป็นส่วนๆ เอง เช่นการเขียนโปรแกรมเป็น โปรแกรมย่อย (Module) จากที่กล่าวมา การใช้เซ็กเมนต์มีผลดีต่อโปรแกรมเมอร์ดังนี้

1. การจัดการกับการเพิ่มโครงสร้างของข้อมูลในโปรแกรมสามารถทำได้ง่ายถ้าโปรแกรมเมอร์ไม่สามารถทราบได้ว่าโครงสร้างในอนาคตของข้อมูลจะมีขนาดเท่าใด ซึ่งการใช้เซ็กเมนต์นั้นจะช่วยให้ย่อหรือขยายโครงสร้างของข้อมูลให้เหมาะสมได้

2. วิธีนี้สามารถทำให้เราเปลี่ยนแปลง หรือคอมไพล์โปรแกรมแยกออกเป็นส่วนๆ ได้ โดยไม่จำเป็นที่จะต้องให้โปรแกรมย่อยๆ เหล่านั้นถูกเรียกขึ้นมาใหม่ หรือเชื่อมโยงกัน

3. การแบ่งเป็นเซ็กเมนต์จะอนุญาตให้มีการแบ่งปันข้อมูลระหว่างโปรเซสหลาย ๆ โปรเซส เช่นโปรแกรมเมอร์สามารถนำโปรแกรมยูทิลิตี้หรือข้อมูลที่เป็นประโยชน์ ใส่ลงไปในส่วนของเซ็กเมนต์หนึ่งๆ ที่สามารถ ถูกอ้างอิงโดยโปรเซสอื่นๆ ได้

4. การแบ่งพื้นที่แบบนี้จะอนุญาตให้มีการป้องกันข้อมูลด้วยตัวเองได้ เพราะส่วนของเซ็กเมนต์หนึ่งๆ นั้นจะถูกสร้างขึ้นมาเพื่อใส่โปรแกรมหรือข้อมูลที่ได้รับการนิยามมาอย่างดีแล้ว ซึ่งโปรแกรมเมอร์ที่เป็นผู้ดูแลระบบสามารถกำหนดสิทธิในการเข้าถึงข้อมูลในส่วนนั้นได้

การที่ขนาดเซ็กเมนต์สามารถเปลี่ยนแปลงได้ ทำให้ช่วยลดปัญหาการสูญเสียพื้นที่ภายในและการสูญเสียพื้นที่ภายนอกด้วย ผลของการแบ่งหน่วยความจำเป็นส่วนๆที่ไม่เท่ากันจึงทำให้ความสัมพันธ์ระหว่าง Logical Address และ Physical Address เป็นความสัมพันธ์ที่ค่อนข้างซับซ้อน โดยอาจใช้ Segmentation Table ในการเก็บรายละเอียดของแต่ละโปรเซส และเก็บที่อยู่เริ่มต้นของที่ว่างในหน่วยความจำ นอกจากนี้ยังต้องเก็บค่าความยาวของเซ็กเมนต์ไว้ด้วย เพื่อเป็นการยืนยันว่าระบบจะไม่ชี้อ่าน Address ของหน่วยความจำผิดตำแหน่ง

5.4.7 หน่วยความจำเสมือน (Virtual Memory)

เป็นการนำพื้นที่ว่างของฮาร์ดิสก์ มาทำเป็นหน่วยความจำหลัก โดยวิธีการนี้โปรเซสจะสามารถทำงานได้ถึงแม้ว่าโปรเซส นั้นๆ จะไม่ได้อยู่ในหน่วยความจำหลัก ทั้งหมดก็ตาม โดยระบบปฏิบัติการจะทำหน้าที่เก็บบางส่วนของโปรเซส ที่กำลังดำเนินการอยู่ไว้ที่หน่วยความจำหลัก และเก็บส่วนอื่นๆ ที่เหลือของโปรเซส นั้นไว้ในฮาร์ดิสก์ ตัวอย่างเช่น มีโปรเซส หนึ่งต้องการใช้พื้นที่ในหน่วยความจำ 100 KB สามารถทำงานบนเครื่องคอมพิวเตอร์ที่มีหน่วยความจำเพียง 64 KBได้ โดยระบบปฏิบัติการจะเลือกเอาบางส่วนของโปรเซส ขนาด 64 KB เข้ามาเก็บไว้ในหน่วยความจำหลัก และเก็บส่วนที่เหลืออีก 36 KB ในฮาร์ดิสก์ และจะทำการสลับเปลี่ยนไปมาเมื่อมีบางส่วนของโปรเซสต้องการทำงาน

5.4.8 หน่วยความจำแคช (Cache Memory)

แคช คือหน่วยความจำประเภทหนึ่งที่มีความเร็วในการทำงานสูงมาก การเข้าถึงข้อมูลในแคช สามารถทำได้รวดเร็วกว่าการเข้าถึงข้อมูลในฮาร์ดิสก์ จุดประสงค์ ของการใช้งานแคช คือการสำรองข้อมูลนั่นเอง โดยปกติข้อมูลจะถูกเก็บไว้ในสื่อบันทึกข้อมูล เมื่อต้องการเรียกใช้ข้อมูลใดๆ หากข้อมูลนั้นถูกคัดลอกและนำไปเก็บไว้ในสื่อบันทึกข้อมูล ที่มีความเร็วสูงกว่าก็คือแคช ระบบก็จะสามารถทำงานได้รวดเร็วกว่าการนำข้อมูลไป จัดเก็บไว้ในสื่อบันทึกข้อมูลที่มีความเร็วในการทำงานต่ำกว่านั่นเอง แต่เนื่องจากแคช มีราคาแพงกว่าหน่วยความจำแบบอื่นเช่น บัพเฟอร์ (Buffer) ดังนั้นการสำรองข้อมูลที่ต้องการพื้นที่มากๆ จึงหันมาใช้งานบัพเฟอร์แทน

5.4.9 หน่วยความจำบัพเฟอร์ (Buffer)

บัพเฟอร์ คือหน่วยความจำประเภทหนึ่งที่ใช้เก็บพักข้อมูลในการถ่ายโอนข้อมูลระหว่างอุปกรณ์ 2 ชนิด โดยการใช้งานบัพเฟอร์มีเหตุผลมาจาก

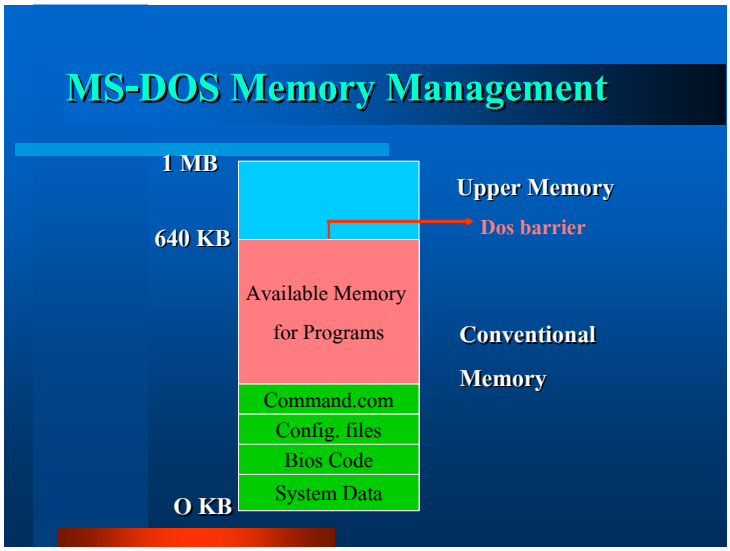
1. การใช้งานอุปกรณ์ 2 ชนิดที่มีความเร็วต่างกัน อุปกรณ์ที่มีความเร็วสูงกว่าไม่จำเป็นต้องรอคอยการทำงานจากอุปกรณ์อีกชิ้นหนึ่งที่มีความเร็วต่ำกว่า โดยระบบจะนำข้อมูลที่ต้องการถ่ายโอนระหว่างอุปกรณ์ 2 ชนิดนั้นไปพักเก็บไว้ในบัพเฟอร์ นั่นเอง ตัวอย่างเช่น การถ่ายโอนข้อมูลระหว่างโมเด็ม (MODEM) กับ ฮาร์ดิสก์ หรือระหว่าง คอมพิวเตอร์ กับเครื่องพิมพ์ นั่นเอง

2. ใช้งานบัพเฟอร์ เพื่อให้อุปกรณ์ที่มีอัตราเร็วในการถ่ายโอนข้อมูลในต่างกันให้สามารถใช้งานร่วมกันได้อย่างมีประสิทธิภาพ

3. ใช้งานบัพเฟอร์ เพื่อป้องกันการสูญหายของข้อมูลในระบบ

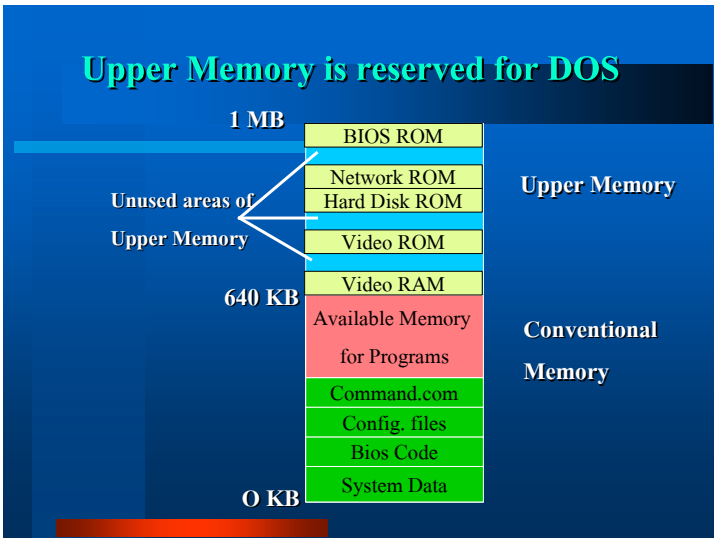
5.5 การจัดสรรหน่วยความจำหลักของระบบปฏิบัติการเอ็มเอสดอส (MS-DOS)

ระบบปฏิบัติการเอ็มเอสดอส เป็นระบบปฏิบัติการที่มีการทำงานเป็นแบบโปรแกรมเดี่ยว (Monoprogramming) คือ สามารถเรียกใช้งานโปรแกรมได้ทีละโปรแกรมเท่านั้น ดังนั้นการจัดสรรหน่วยความจำของระบบปฏิบัติการเอ็มเอสดอส จึงใช้หลักการง่ายๆ คือแบ่งหน่วยความจำออกเป็น 3 ส่วนหลักๆ โดยส่วนแรกกันไว้ให้ตัวระบบปฏิบัติการและโปรแกรมที่ต้องการเรียกใช้งาน (Conventional Memory) ส่วนที่สอง เป็นส่วนที่เก็บข้อมูลของแรมและรอม(Upper Memory) และส่วนที่สามเป็นพื้นที่ส่วนของผู้ใช้งานที่จะสามารถเรียกใช้งานแฟ้มข้อมูลต่างๆ (Extended Memory)



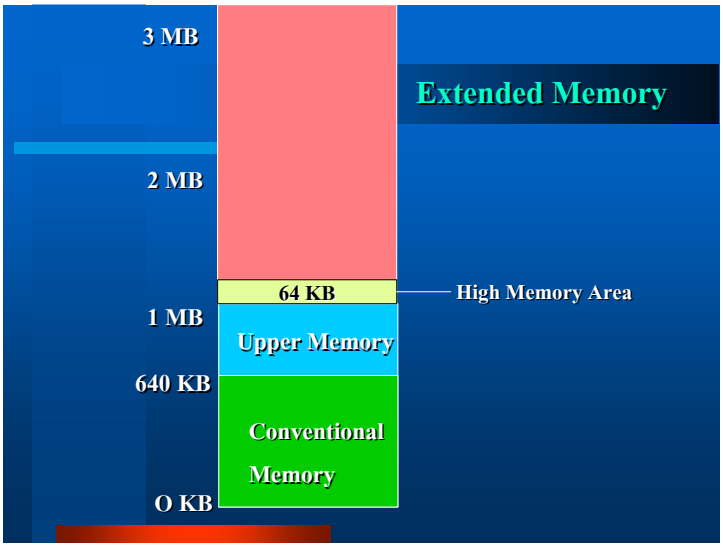
ภาพที่ 5.24 แสดงการจัดสรรหน่วยความจำหลักของระบบปฏิบัติการเอ็มเอสดอส
ที่มา (http://www.marinerthai.com/sara_it/view.php?No=it50033, 2522)

จากภาพที่ 5.24 ในส่วนของ Conventional Memory จะเก็บข้อมูลของระบบปฏิบัติการ ได้แก่ ไฟล์ Command.com, Config.sys เก็บข้อมูลของไบออส (BIOS Code) และข้อมูลของระบบ (System Data)



ภาพที่ 5.25 แสดงการจัดสรร Upper Memory ของระบบปฏิบัติการเอ็มเอสดอส
ที่มา (http://www.marinerthai.com/sara_it/view.php?No=it50033, 2522)

จากภาพที่ 5.25 ในส่วนของ Upper Memory มีการจัดสรรพื้นที่สำหรับเก็บข้อมูลของแรมและรอม ได้แก่ ไบออสรอม (BIOS ROM) เน็ตเวิร์กรอม (Network ROM) ฮาร์ดิสก์รอม (Hard Disk ROM) วิดิโอรอม (Video ROM) วิดิโอแรม (Video RAM) และยังมีพื้นที่ว่างเป็นส่วนที่ไม่ได้ถูกใช้งาน (Unused area of Upper Memory)

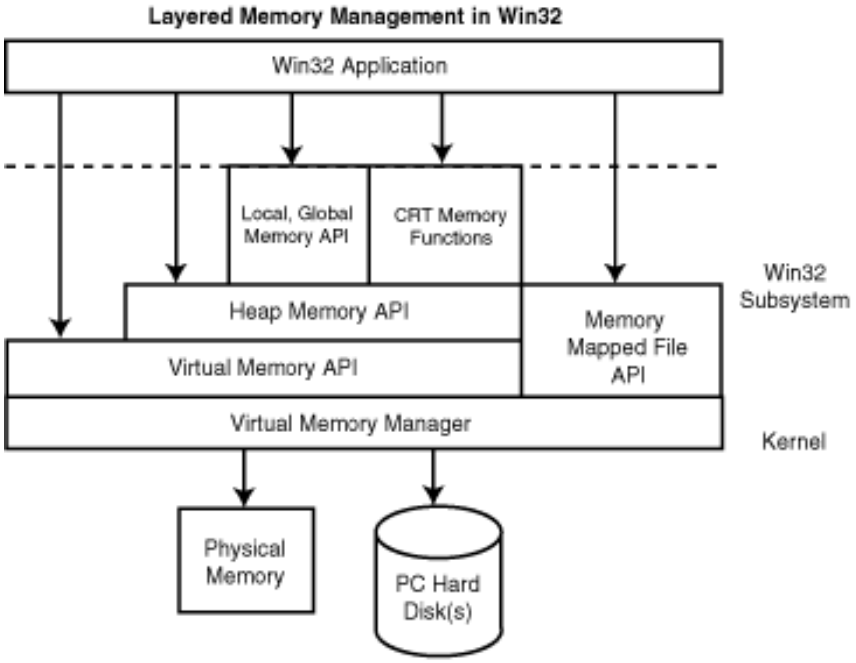


ภาพที่ 5.26 แสดงการจัดสรร Extended Memory ของระบบปฏิบัติการเอ็มเอสดอส
ที่มา (http://www.marinerthai.com/sara_it/view.php?No=it50033, 2522)

จากภาพที่ 5.26 หลังจากพื้นที่ในส่วนของ Upper Memory จะเป็นพื้นที่ที่เรียกว่า High Memory Area ซึ่งมีขนาด 64 KB และพื้นที่ต่อจาก High Memory Area ก็จะเป็นพื้นที่ของ Extended Memory

5.6 การจัดสรรหน่วยความจำหลักของระบบปฏิบัติการไมโครซอฟท์วินโดวส์ (Microsoft Windows)

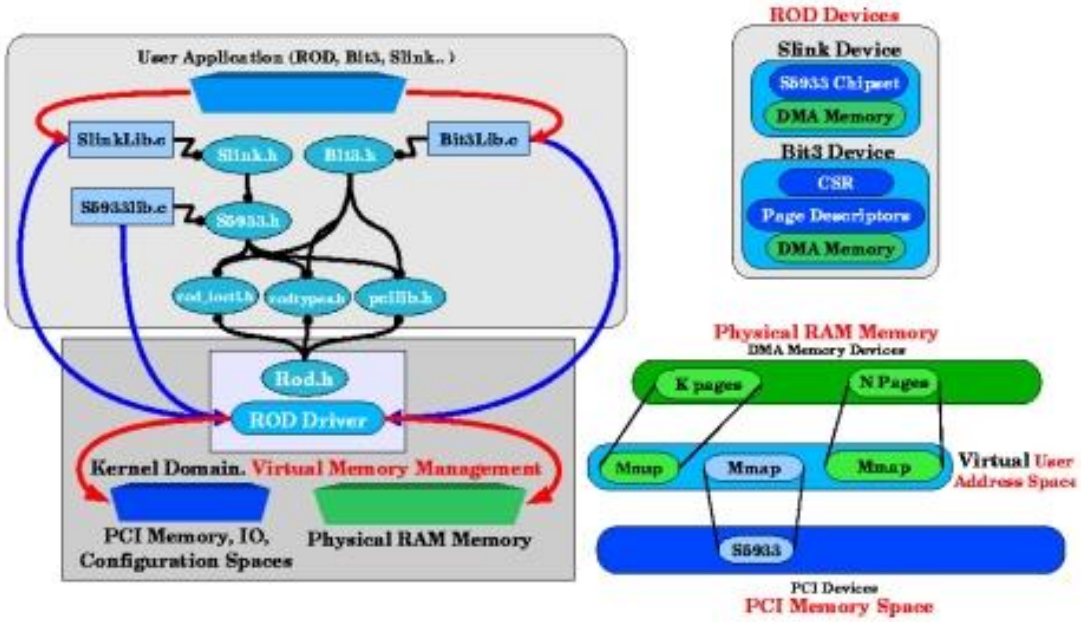
ระบบปฏิบัติการไมโครซอฟท์วินโดวส์ที่เป็นแบบ Win32 API มีวิธีการจัดสรรหน่วยความจำอยู่หลายวิธี เช่นการใช้ หน่วยความจำเสมือนในการทำงานของโปรเซสต่างๆ โดยใช้เทคนิคการจัดการหน่วยความจำแบบการแบ่งเป็นหน้า ใช้ฟังก์ชัน VirtualAlloc ในการสงวนหรือใช้งานหน่วยความจำเสมือน และใช้ฟังก์ชัน VirtualFree ในการยกเลิกการใช้งานหน่วยความจำเสมือน



ภาพที่ 5.27 แสดงการจัดสรรหน่วยความจำหลักของระบบปฏิบัติการวินโดวส์
ที่มา (http://docs.rinet.ru/Plugi/ch12.htm#Introduction, 2522)

5.7 การจัดสรรหน่วยความจำหลักของระบบปฏิบัติการลีนุกซ์ (Linux)

การจัดการหน่วยความจำของระบบปฏิบัติการก็อาศัยหน่วยความจำเสมือนในการทำงานของโปรเซสโดยจะทำหน้าที่จัดตำแหน่งหน่วยความจำทางกายภาพไปยังฮาร์ดดิสก์ โดยอาศัยเทคนิคการสับเปลี่ยน แต่ในลีนุกซ์เวอร์ชั่นใหม่ๆ จะใช้ เทคนิคการจัดการหน่วยความจำแบบการแบ่งเป็นหน้าแทน



ภาพที่ 5.28 แสดงการจัดสรรหน่วยความจำหลักของระบบปฏิบัติการลีนุกซ์
ที่มา (http://ific.uv.es/tical/rod_old/slink/slink.html, 2522)

บทสรุป

หน่วยความจำเป็นอุปกรณ์ที่ใช้ในการจดจำหรือจัดเก็บข้อมูลที่ใช้ในระบบคอมพิวเตอร์ ซึ่งหากคอมพิวเตอร์ปราศจากหน่วยความจำแล้ว คอมพิวเตอร์จะไม่สามารถทำงานได้เลย การจัดการหน่วยความจำจึงถือเป็นสิ่งสำคัญที่ระบบปฏิบัติการต่างๆ ใช้เทคนิคและวิธีการจัดการหน่วยความจำที่มีอยู่ในระบบคอมพิวเตอร์ให้ถูกใช้งานได้อย่างมีประสิทธิภาพ

คำถามท้ายบทที่ 5

1. หน่วยความจำในระบบคอมพิวเตอร์คืออะไร แบ่งเป็นกี่ชนิด อะไรบ้าง
2. จงอธิบายเทคนิคการจัดการหน่วยความจำแบบการแบ่งหน้า (Paging) มาพอสังเขป
3. จงอธิบายเทคนิคการจัดการหน่วยความจำแบบการแบ่งเป็นตอน (Segmentation) มาพอสังเขป
4. หน่วยความจำเสมือนคืออะไร อธิบายพอสังเขป
5. จงบอกส่วนประกอบของตารางหน้า (Page Table)
6. การผิดหน้า หมายความว่าอย่างไร
7. จงแสดงวิธีการสับเปลี่ยนหน้าแบบมาก่อน-ออกก่อน(First –in First-out Algorithm) และ การสับเปลี่ยนแบบที่ดีที่สุด (Optimal Page Replacement Algorithm) จากข้อมูลโปรเซสสมมุติดังนี้ 3 4 0 2 4 8 7 0 1 3 2 7 4 1 3 8 2
8. จงอธิบายการจัดสรรหน่วยความจำหลักของระบบปฏิบัติการเอ็มเอสดอส
9. จงอธิบายการจัดสรรหน่วยความจำหลักของระบบปฏิบัติการไมโครซอฟท์วินโดวส์
10. จงอธิบายการจัดสรรหน่วยความจำหลักของระบบปฏิบัติการลีนุกซ์

ปฏิบัติการ

ให้นักศึกษาทำปฏิบัติการที่ 3

เรื่อง ศึกษาการจัดการหน่วยความจำของระบบปฏิบัติการ Windows XP

โดยดาวน์โหลดแบบฟอร์มปฏิบัติการได้ที่ http://roongrote.crru.ac.th/CT2403.html

เอกสารอ้างอิง

น.ท.ไพศาล โมลิสกุลมงคล, ผศ.น.ท.ดร.ประสงค์ ปราณีตพลกรัง, น.ต.เมธา สุนทรศารทูล, น.ต.สุชาติ วีรกุลวัฒนา, ร.ท.อนุโชติ วุฒิพรพงษ์. (2545). **ระบบปฏิบัติการ (Operating Systems)**. กรุงเทพฯ : ไทยเจริญการพิมพ์.

นรีรัตน์ นิยมไทย. (2549). **ระบบปฏิบัติการ**. กรุงเทพฯ : ศูนย์ส่งเสริมวิชาการ.

พิรพร หมุนสนิท, สุธีพงศาสกุลชัย และอัจจิมา เลี้ยงอยู่. (2553). **ระบบปฏิบัติการ (Operating Systems)**. กรุงเทพฯ : เคทีพี.

(2552) [Online]. Available HTTP: http://blogger.sanook.com/mk_melody/2008/12/.

(2552) [Online]. Available HTTP: http://commons.wikimedia.org/wiki/File:Magnetic_tape_hg.jpg.

(2552) [Online]. Available HTTP: http://members.tripod.com/maesod_computer_club/abouts/ram.htm.

(2552) [Online]. Available HTTP: http://oldcomputers.net/floppydisks.html.

(2552) [Online]. Available HTTP: http://www.electron.rmutphysics.com/science-news/.

(2552) [Online]. Available HTTP: http://www.bncc.ac.th/childweb/OS/html/l5-3-1.html.

(2552) [Online]. Available HTTP: http://csnon04.blogspot.com/2008/03/5.html.

(2552) [Online]. Available HTTP: http://cptd.chandra.ac.th/selfstud/os1/Mp4.html.

(2552) [Online]. Available HTTP: http://csnon04.blogspot.com/.

(2552) [Online]. Available HTTP: http://csnon04.blogspot.com/2008/03/5.html.

(2552) [Online]. Available HTTP:

http://www.marinerthai.com/sara_it/view.php?No=it50033.

(2552) [Online]. Available HTTP:

http://docs.rinet.ru/Plugi/ch12.htm#Introduction.

(2552) [Online]. Available HTTP:

http://ific.uv.es/tical/rod_old/slink/slink.html.