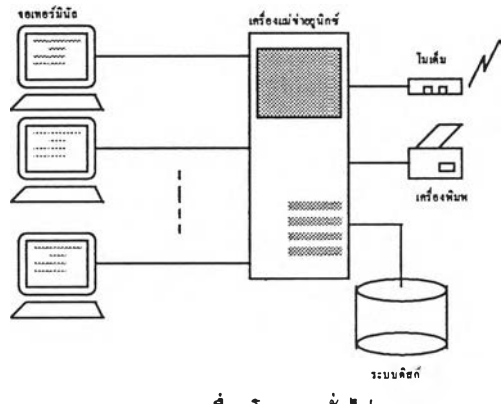
บทที่ 4

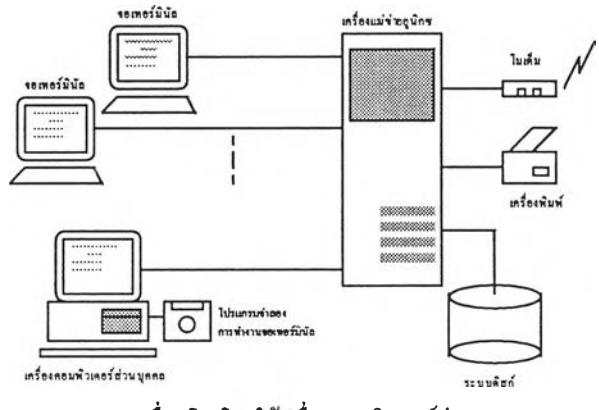
การพัฒนาระบบคอสเซลล์

4.1 การเชื่อมโยงระบบยูนิกซ์กับระบบคอสเซลล์

จากการที่เราได้ศึกษาถึงพื้นฐานในบทที่ผ่านมา จะเห็นได้ว่าระบบคอมพิวเตอร์ที่ใช้ระบบปฏิบัติการยูนิกซ์สามารถเชื่อมโยงกับจอเทอร์มินอลได้หลายแบบ รูปที่ 4.1 (ก) แสดงการเชื่อมโยงจอเทอร์มินอลในระบบยูนิกซ์โดยทั่วไป ในรูปที่ 4.1 (ข) แสดงให้เห็นการเชื่อมโยงกับเครื่องคอมพิวเตอร์ส่วนบุคคลโดยทำหน้าที่เป็นจอเทอร์มินอลเช่นเดียวกันโดยใช้โปรแกรมเลียนการทำงานของจอเทอร์มินอลสำหรับใช้งานโปรแกรมหลักขององค์กร



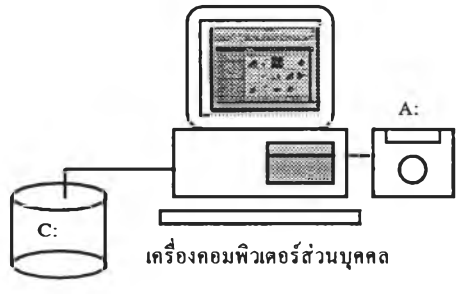
(ก) การเชื่อมโยงแบบทั่วไป



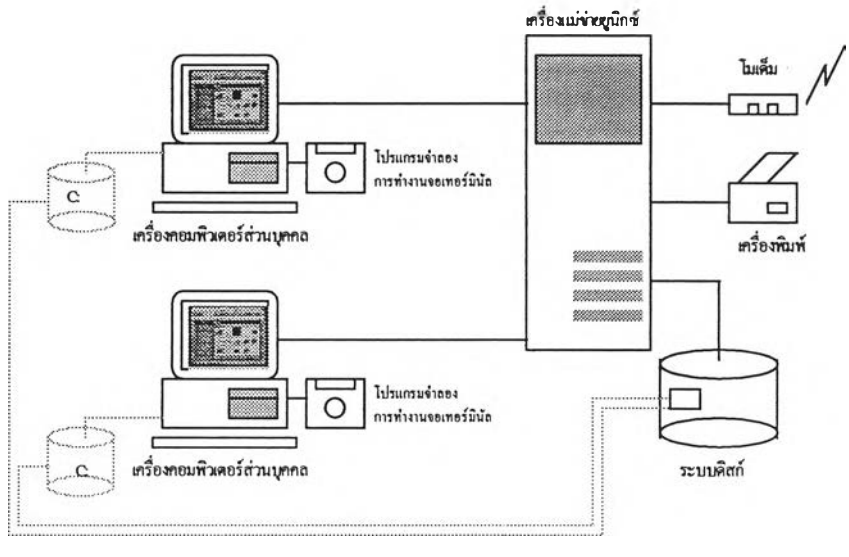
(ข) การเชื่อมโยงโดยใช้เครื่องคอมพิวเตอร์ส่วนบุคคล

รูปที่ 4.1 แสดงการเชื่อมโยงจอเทอร์มินอลแบบต่างๆ ในระบบยูนิกซ์

ในขณะเดียวกันเมื่อต้องการใช้โปรแกรมในระบบเอ็มเอสดอสก็สามารถออกจากโปรแกรมเลียนแบบและเรียกใช้โปรแกรมบนระบบปฏิบัติการเอ็มเอสดอสได้ทันที เนื่องจากการทำงานของเครื่องคอมพิวเตอร์ส่วนบุคคลใช้ระบบปฏิบัติการเอ็มเอสดอสอยู่แล้ว ดังรูปที่ 4.2 ซึ่งโดยปกติจะใช้โปรแกรมและข้อมูลที่เก็บไว้ในจานบันทึกข้อมูลชนิดแข็ง ดังนั้น ถ้าต้องการใช้โปรแกรมและระบบเชลล์ของระบบปฏิบัติการเอ็มเอสดอสเป็นส่วนที่ติดต่อกับผู้ใช้และใช้ทรัพยากร เช่น ระบบจานบันทึกข้อมูลของระบบยูนิกซ์และสามารถแลกเปลี่ยนข้อมูลระหว่างผู้ใช้ได้ ซึ่งรูปที่ 4.3 แสดงให้เห็นแนวความคิดนี้โดยการเชื่อมโยงเครื่องคอมพิวเตอร์ทั้งสองระบบปฏิบัติการเข้าด้วยกันและใช้ประโยชน์ดังกล่าว



รูปที่ 4.2 แสดงคอมพิวเตอร์ในระบบปฏิบัติการเอ็มเอสดอส

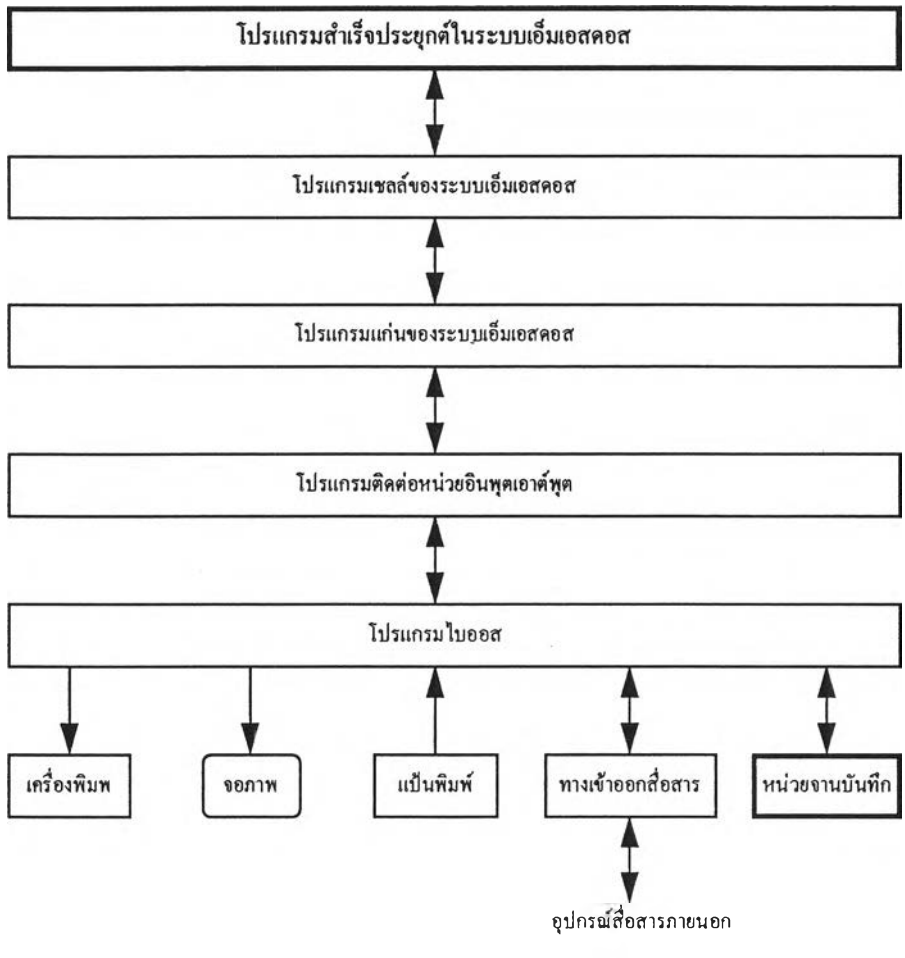


รูปที่ 4.3 แสดงแนวความคิดในการเชื่อมโยงระบบปฏิบัติการเอ็มเอสดอสและยูนิกซ์

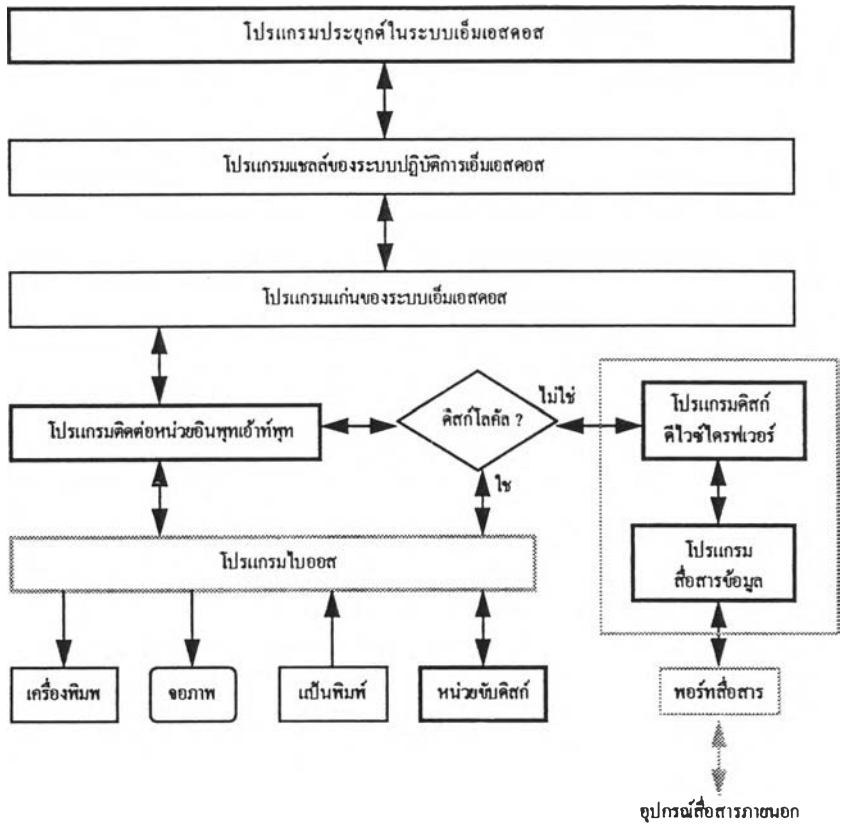
4.2 โปรแกรมในระบบคอสเชลล์

4.2.1 โปรแกรมในส่วนของระบบปฏิบัติการเอ็มเอสดอส

ในระบบปฏิบัติการเอ็มเอสดอสการทำงานปกติแสดงให้เห็นได้ดังรูปที่ 4.4 ในกรณีที่มีการติดต่อกับอุปกรณ์ต่าง ๆ จะถูกควบคุมโดยผ่านโปรแกรมไบออสทั้งหมด แต่ระบบปฏิบัติการเอ็มเอสดอสก็ยินยอมให้ผู้ใช้สามารถทำการติดต่อโดยตรงกับอุปกรณ์เหล่านี้ได้ ด้วยการเขียนโปรแกรมให้บริการโดยผ่านสัญญาณขัดจังหวะมาตรฐานที่ออกแบบไว้ ในกรณีของระบบคอสเชลล์ความต้องการ คือ การขอใช้ระบบจานบันทึกข้อมูลจากภายนอก จึงมีส่วนที่เกี่ยวข้องที่ต้องพัฒนาขึ้นมาซึ่งแสดงให้เห็นดังรูปที่ 4.5 โปรแกรมเหล่านี้ ได้แก่



รูปที่ 4.4 แสดงการทำงานของระบบปฏิบัติการเอ็มเอสดอส



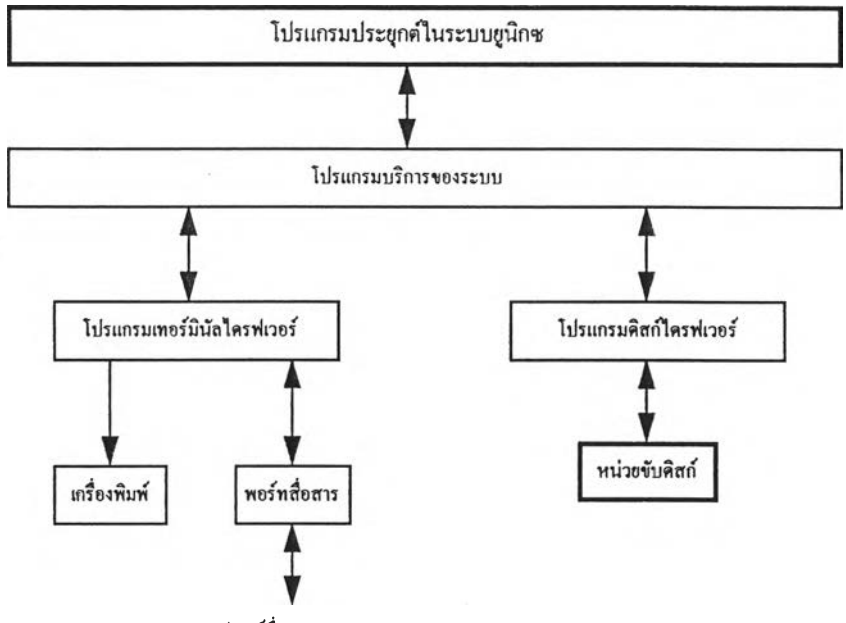
รูปที่ 4.5 แสดงโปรแกรมที่ต้องมีการพัฒนาในระบบปฏิบัติการเอ็มเอสดอส

4.2.1.1 *โปรแกรมย่อยอุปกรณ์จานบันทึกข้อมูล* เป็นโปรแกรมโปรแกรมย่อยอุปกรณ์ที่ทำหน้าที่ในการเปลี่ยนการทำงานของจานบันทึกข้อมูลปกติให้เป็นการทำงานแบบหน่วยตรรก และส่งการทำงานไปให้กับระบบยูนิกซ์ดำเนินการให้ต่อไป

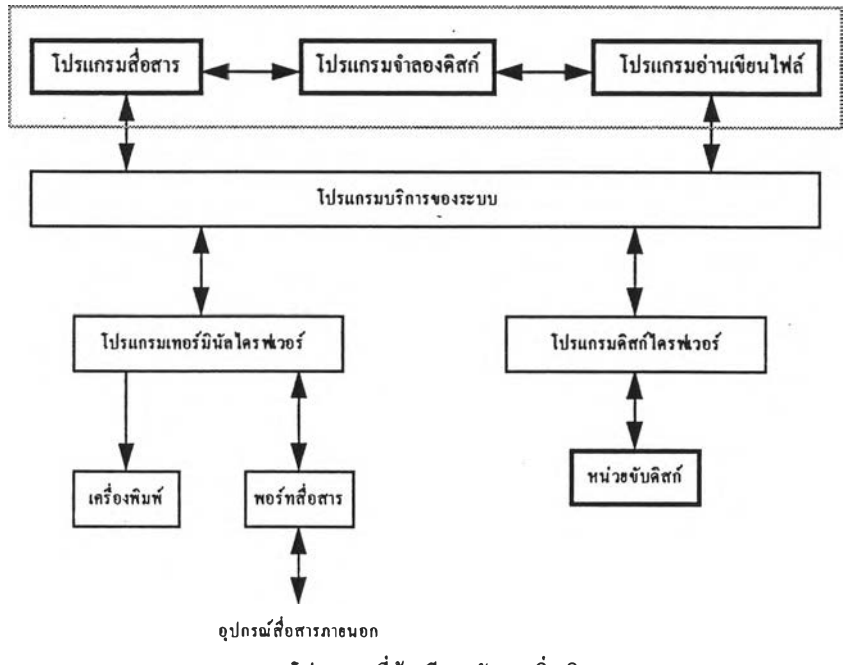
4.2.1.2 *โปรแกรมสื่อสารข้อมูล* ทำหน้าที่รับส่งข้อมูลระหว่างระบบปฏิบัติการทั้งสองโดยผ่านระบบฮาร์ดแวร์ที่มีอยู่แล้ว

4.2.2 โปรแกรมในส่วนของระบบปฏิบัติการยูนิกซ์

ในระบบปฏิบัติการยูนิกซ์การทำงานปกติแสดงให้เห็นได้ดังรูปที่ 4.6 (ก) ซึ่งการติดต่อกับอุปกรณ์ ต่าง ๆ จะถูกควบคุมผ่านโปรแกรมย่อยอุปกรณ์ของอุปกรณ์นั้น ๆ โดยระบบยูนิกซ์จะมองเห็นอุปกรณ์เหล่านี้ในลักษณะของแฟ้มเหมือนกันหมด การติดต่อใช้งานสามารถทำได้โดยการเรียกใช้ผ่านชื่อที่กำหนดไว้ในระบบโดยการเขียนโปรแกรม ในกรณีของระบบคอสเชลล์ระบบยูนิกซ์จะทำหน้าที่ให้บริการเก็บโปรแกรมหรือข้อมูลให้กับลูกข่ายที่เป็นเครื่องคอมพิวเตอร์ส่วนบุคคล ส่วนที่เกี่ยวข้องที่ต้องพัฒนาขึ้นมาแสดงให้เห็นได้ดังรูปที่ 4.6 (ข) ซึ่งได้แก่

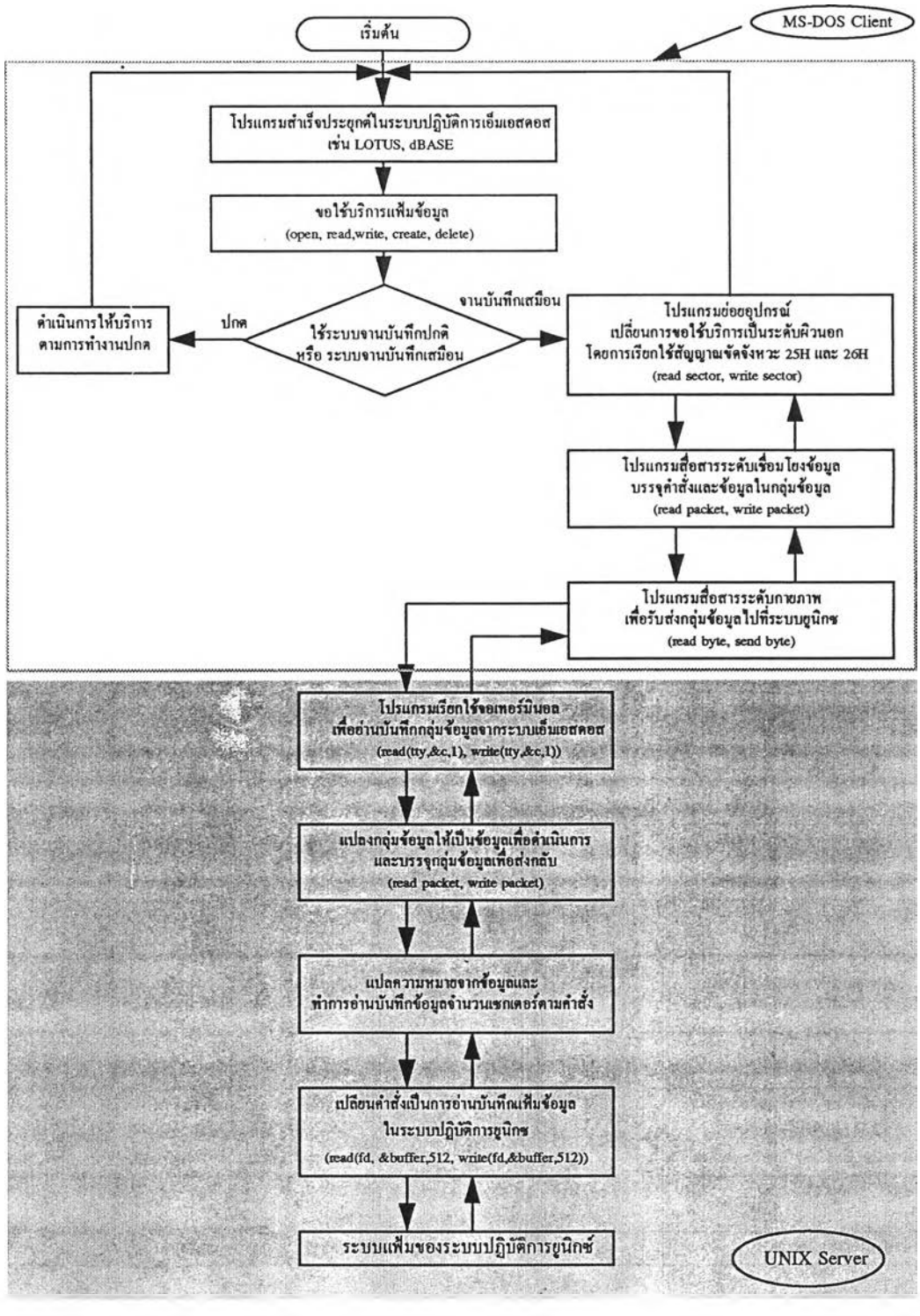


(ก) การทำงานในระบบยูนิกซ์เดิม



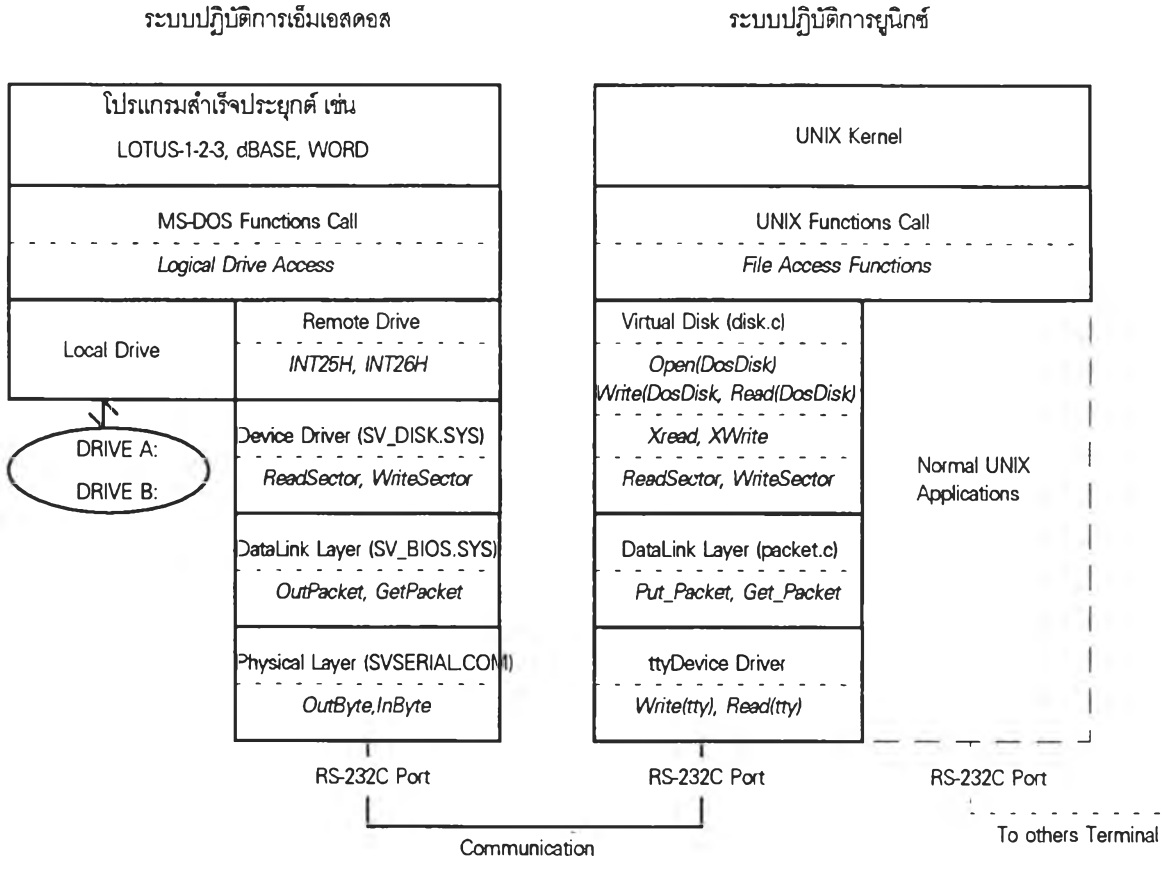
(ข) โปรแกรมที่ต้องมีการพัฒนาเพิ่มเติม

รูปที่ 4.6 แสดงโปรแกรมที่พัฒนาในระบบปฏิบัติการยูนิกซ์



รูปที่ 4.7 แสดงลำดับการทำงานของระบบคอสเซลล์

โครงสร้างของระบบดอสเชลล์



รูปที่ 4.8 แสดงโครงสร้างของระบบดอสเชลล์บนระบบปฏิบัติการยูนิกซ์

4.2.2.1 *โปรแกรมเลียนการทำงานจานบันทึกข้อมูลแบบระบบปฏิบัติการเอ็มเอสดอส* ทำหน้าที่ในการให้บริการอ่านและบันทึกข้อมูลลงในระบบจานบันทึกข้อมูลของระบบปฏิบัติการยูนิกซ์บนเครื่องแม่ข่าย ในรูปแบบการทำงานของหน่วยตรรกที่ใช้ในระบบปฏิบัติการเอ็มเอสดอส

4.2.2.2 *โปรแกรมสื่อสารข้อมูล* ทำหน้าที่รับส่งข้อมูลระหว่างระบบปฏิบัติการทั้งสองโดยผ่านโปรแกรมย่อยอุปกรณ์ควบคุมเทอร์มินอล

4.3 องค์ประกอบของระบบคอสเซลล์

ระบบคอสเซลล์บนระบบปฏิบัติการยูนิกซ์ที่ทำการพัฒนา สามารถแสดงหลักการการทำงานโดยภาพรวมให้เห็นในรูปที่ 4.7 และได้ออกแบบโดยมีองค์ประกอบแสดงให้เห็นดังในรูปที่ 4.8

4.4 โปรแกรมสื่อสารในระบบคอสเซลล์

การพัฒนาโปรแกรมสื่อสารระหว่างสองระบบปฏิบัติการ โดยอาศัยมาตรฐานที่กำหนดไว้ของ ISO[1] สามารถแบ่งได้เป็น 2 ระดับ คือ

4.4.1 *ระดับข่ายเชื่อมโยงข้อมูล (Datalink Layer)* ทำหน้าที่ในการรับส่งข้อมูลในลักษณะของกลุ่มข้อมูลสื่อสาร (Data packet) ซึ่งจะต้องมีการกำหนดรูปแบบของกลุ่มข้อมูลเพื่อให้สื่อความกันได้ทั้งสองด้านที่ติดต่อกันอยู่

4.4.2 *ระดับทางกายภาพ (Physical Layer)* ทำหน้าที่รับส่งข้อมูลในระดับเป็นหน่วยของไบต์

4.5 กลุ่มข้อมูลสื่อสาร

ในระบบคอสเซลล์บนระบบปฏิบัติการยูนิกซ์ ได้กำหนดรูปแบบของกลุ่มข้อมูลสื่อสารที่ใช้ในการสื่อสารระหว่างระบบคอมพิวเตอร์ทั้งสองด้าน โดยมีส่วนประกอบดังรูปที่ 4.9 ซึ่งได้แก่

4.5.1 *ส่วนหัว* เป็นข้อมูลขนาด 1 ไบต์ในที่นี้จะใช้รหัส SOH หรือ 01H ตาม รหัสแอสกี เพื่อเป็นการบอกให้รู้ว่าจุดเริ่มของชุดข้อมูลเริ่มจากตำแหน่งนี้ และข้อมูลไบต์ถัดไปจะเป็นรายละเอียดของข้อมูลในทางปฏิบัติทางผู้รับจะมีการส่งข้อมูลการตอบรับ (Acknowledge) เพื่อบอกให้ทางผู้ส่งรู้ว่าพร้อมที่จะรับข้อมูลแล้วให้ส่งข้อมูลได้

4.5.2 *ความยาวของกลุ่มข้อมูลสื่อสาร* เป็นข้อมูลถัดมาขนาด 2 ไบต์ เป็นส่วนที่บอกให้รู้ว่าจำนวนข้อมูลในส่วนของข้อมูลภายในชุดข้อมูลนั้นมีจำนวนกี่ไบต์ ซึ่งจำเป็นมากเพราะจะเป็นการบอกขอบเขตสำหรับการแยกข้อมูลต่าง ๆ ในกลุ่มข้อมูลสื่อสารออกมา

---

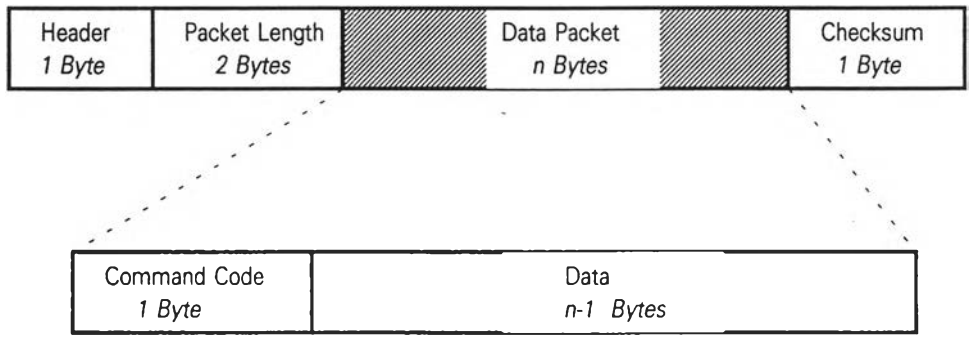
[1] William Stallings, Ph.D. Handbook of Computer-Communication standards. Vol.1. (Indianapolis : Macmillan, Inc., 1990), pp.6-13.

4.5.3 *ส่วนของข้อมูล* เป็นส่วนของข้อมูลทั้งหมดที่ต้องการส่งไปถึงผู้รับ ซึ่งจำนวนของข้อมูลสามารถหาได้จากการแยกส่วนควบคุมต่าง ๆ ออกไป ส่วนนี้เป็นส่วนที่สำคัญที่สุดเพราะการทำงานต่าง ๆ จะกำหนดจากส่วนนี้ทั้งหมด ในการพัฒนาระบบคอสเซลล์แบ่งส่วนนี้ออกเป็น 2 ส่วน คือ

4.5.3.1 *ส่วนของคำสั่ง* (Command Code) เป็นการกำหนดประเภทของการทำงานว่าจะให้ทำอะไร เช่นเขียนข้อมูล อ่านข้อมูล หรือ บอกสถานะของผู้รับ เป็นต้น ทั้งนี้ขึ้นอยู่กับการนำไปใช้ในระดับการประยุกต์ (Application Layer) ในการพัฒนาครั้งนี้กำหนดขนาดไว้ 1 ไบต์

4.5.3.2 *ส่วนของข้อมูล* เป็นข้อมูลที่ต้องการใช้งาน ซึ่งจำนวนไบต์ที่ใช้ขึ้นอยู่กับชนิดของคำสั่งที่ต้องการนำไปใช้ในระดับการประยุกต์ใช้งานต่อไป

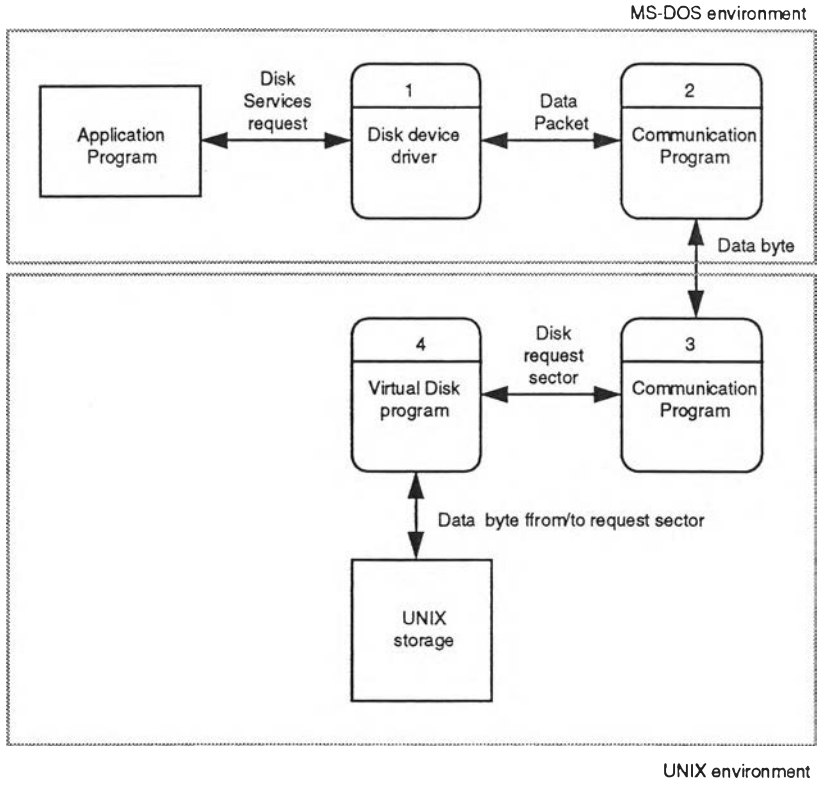
4.5.4 *ผลรวมการตรวจสอบ* (Checksum) เป็นข้อมูลขนาด 1 ไบต์ใช้เก็บค่าสำหรับตรวจสอบความผิดพลาดในการรับส่งข้อมูล วิธีการทำงานจะทำการบวกข้อมูลทั้งหมดแล้วเก็บเฉพาะไบต์ต่ำเป็นค่าผลรวมการตรวจสอบส่งไปพร้อมกับข้อมูล ตัวอย่างเช่น ชุดข้อมูลประกอบไปด้วยข้อมูล 5 ไบต์ คือ D1H, 1FH, 22H, 0DH, 31H เมื่อรวมกันแล้วเท่ากับ 0150H ดังนั้นค่าผลรวมการตรวจสอบจะเท่ากับ 50H



รูปที่ 4.9 แสดงส่วนประกอบของกลุ่มสื่อสารข้อมูลที่ใช้ในระดับข่ายเชื่อมโยงข้อมูล

4.6 การพัฒนาโปรแกรมคอสเซลล์บนระบบปฏิบัติการยูนิกซ์

จากองค์ประกอบที่กล่าวมาแล้วระบบโดยรวมของระบบคอสเซลล์สามารถเขียนเป็นผังกระแสข้อมูล (data flow diagram) ได้ดังรูปที่ 4.10

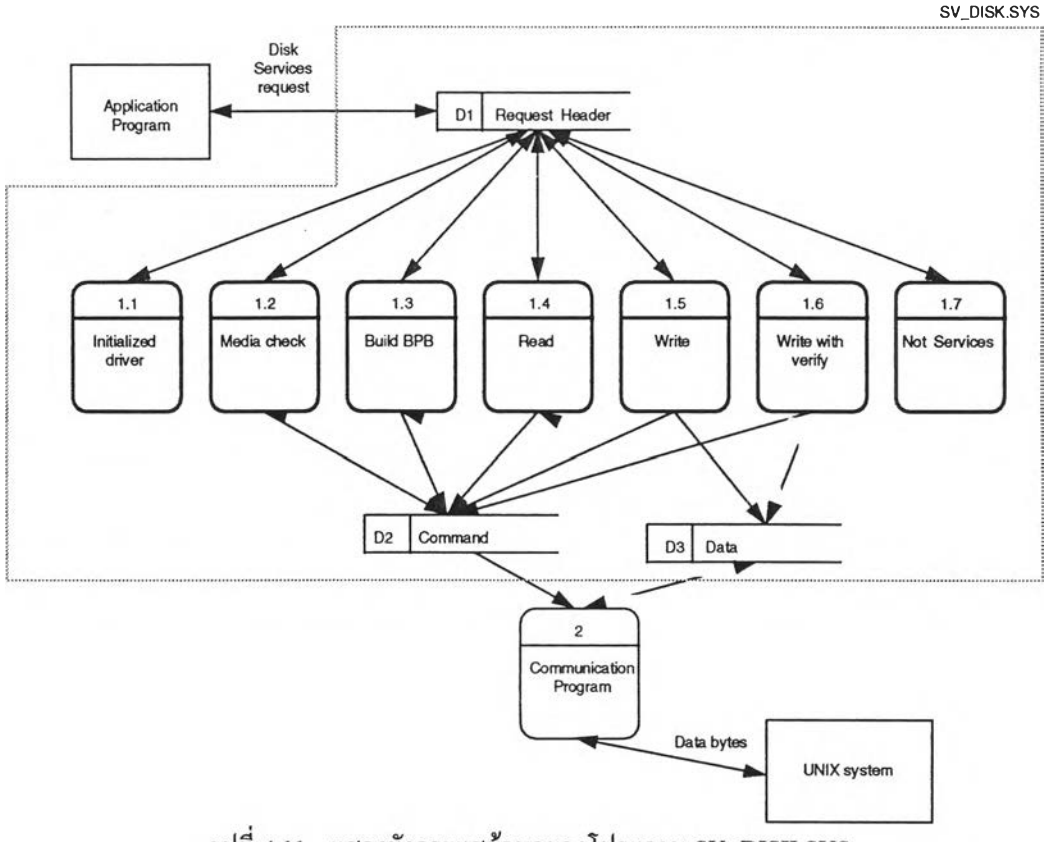


รูปที่ 4.10 ผังกระแสข้อมูลของระบบคอสเชลล์บนระบบปฏิบัติการยูนิกซ์

ซึ่งจะเห็นได้ว่าจะประกอบไปด้วยโปรแกรม 4 ส่วนสำคัญ คือ ส่วนโปรแกรมสื่อสารบนระบบปฏิบัติการเอ็มเอสดอสและยูนิกซ์ โปรแกรมดิสก์โปรแกรมย่อยอุปกรณ์บนระบบปฏิบัติการเอ็มเอสดอสและโปรแกรมเลียนแบบจานบันทึกข้อมูลบนระบบยูนิกซ์

4.6.1 โปรแกรมจานบันทึกข้อมูลโปรแกรมย่อยอุปกรณ์

เป็นโปรแกรมในชั้นการประยุกต์โดยอาศัยหลักการของโปรแกรมย่อยอุปกรณ์ ซึ่งจะทำหน้าที่ในการแปลงการขอใช้บริการระบบจานบันทึกข้อมูลของโปรแกรมสำเร็จประยุกต์ให้เป็นการทำงานในระดับกายภาพ คือ การอ่านหรือบันทึกข้อมูลเป็นเซกเตอร์ โปรแกรมที่พัฒนาไว้เป็นโปรแกรมโปรแกรมย่อยอุปกรณ์ใช้ชื่อว่า SV_DISK.SYS คล้ายกับโปรแกรมจานบันทึกข้อมูลบนระบบเครือข่าย สามารถแสดงผังกระแสข้อมูลการทำงานของโปรแกรมนี้ได้ดังรูปที่ 4.11



รูปที่ 4.11 แสดงผังกระแสข้อมูลของโปรแกรม SV_DISK.SYS

ตารางที่ 4.1 แสดงฟังก์ชันการทำงานของโปรแกรม SV_DISK.SYS

CODE	COMMAND FUNCTION
0	INIT
1	MEDIA CHECK
2	BUILD BIOS parameter block(BPB)
3	Bad (No service)
4	READ
5	Bad (No service)
6	Bad (No service)
7	Bad (No service)
8	WRITE
9	WRITE with VERIFY

โปรแกรมนี้ถูกออกแบบให้มีฟังก์ชันการทำงานตามมาตรฐานของโปรแกรมย่อยอุปกรณ์ ซึ่งการทำงานที่สำคัญและจำเป็นแสดงไว้ในตารางที่ 4.1 โดยมีรายละเอียดดังนี้

4.6.1.1 *ฟังก์ชันการทำงานที่ 00H* เป็นโปรแกรมส่วนของโปรแกรมย่อยอุปกรณ์ที่ทำหน้าที่ติดตั้งและส่งค่าเริ่มต้นต่าง ๆ ที่ติดต่อกับระบบปฏิบัติการเพื่อการเรียกโปรแกรมย่อยนี้ต่อไป ส่วนนี้จะมีการเรียกใช้เพียงครั้งแรกครั้งเดียวเท่านั้น ซึ่งรายละเอียดได้กล่าวไว้ในบทต้น ๆ แล้ว

4.6.1.2 *ฟังก์ชันการทำงานที่ 01H* การทำงานจะส่งคำสั่งไปให้กับโปรแกรมเลียนแบบจานบันทึกข้อมูลบนระบบยูนิกซ์ เพื่อตรวจสอบว่าระบบจานบันทึกข้อมูลมีการเปลี่ยนหรือไม่ ถ้ามีจะได้ดำเนินการตั้งค่าตัวแปรอุปกรณ์ให้ถูกต้องสอดคล้องกับจานบันทึกข้อมูลใหม่ เพื่อไม่ให้เกิดปัญหาในการใช้งานต่อไป รูปแบบของคำสั่งที่ใช้ในโปรแกรมนี้แสดงให้เห็นได้ในรูปที่ 4.12 และผังกระแสข้อมูลการทำงาน ดังรูปที่ 4.13

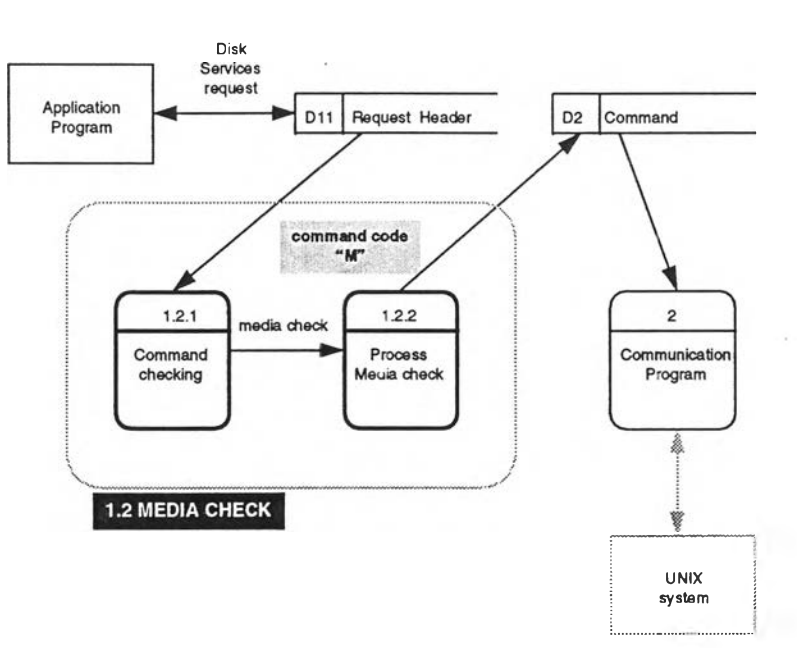
COMMAND CODE	DRIVE NO.
1 BYTE	*1 BYTE*
M	XX

(1) รูปแบบคำสั่งของการทำงานที่ 02H

COMMAND CODE	DATA
1 BYTE	*1 BYTE*
D	XX

(2) รูปแบบข้อมูลที่ได้รับกลับมา

รูปที่ 4.12 แสดงรูปแบบคำสั่งในฟังก์ชันการทำงาน 01H

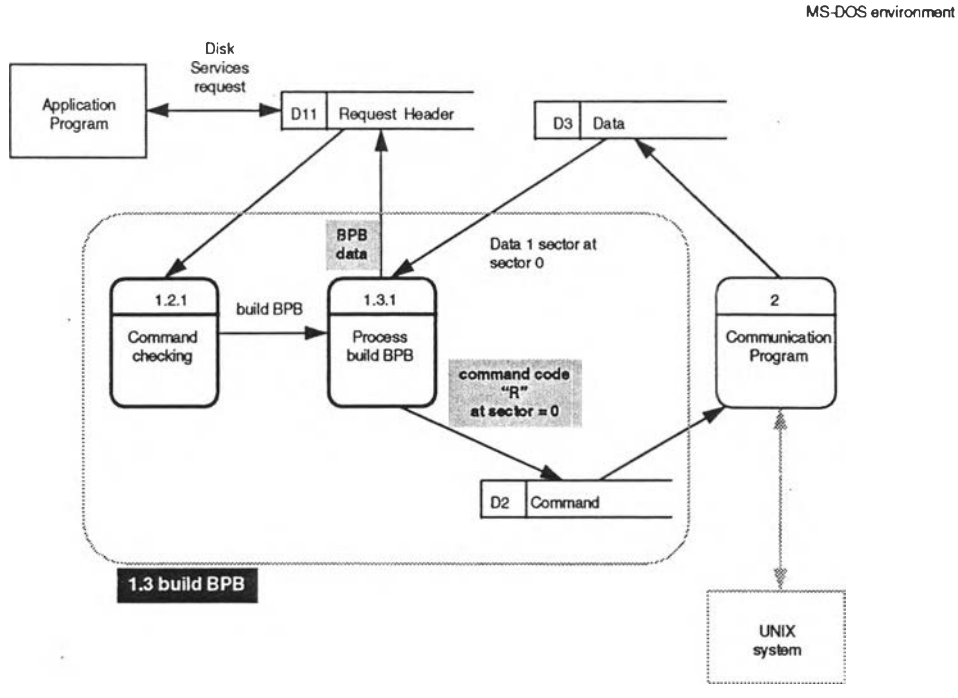


รูปที่ 4.13 แสดงผังกระแสข้อมูลการทำงานของฟังก์ชัน 01H ในโปรแกรม SV_DISK.SYS

4.6.1.3 *ฟังก์ชันการทำงานที่ 02H* การทำงานจะทำการอ่านค่าตัวแปรที่แสดงคุณลักษณะของหน่วยตรรก คือ ค่าบีพีบี ที่ติดตั้งไว้ให้กับระบบปฏิบัติการเอ็มเอสดอสเพื่อใช้ในการคำนวณหาตำแหน่งของข้อมูลที่แท้จริงต่อไป ในระบบดอสเซลล์หน่วยตรรกนี้ คือ ระบบจานบันทึกข้อมูลจำลองบนระบบยูนิกซ์ ซึ่งปกติระบบจานบันทึกข้อมูลของเอ็มเอสดอสจะเก็บค่านี้ไว้ที่เซคเตอร์ 0 ดังนั้นการขอข้อมูลนี้สามารถทำได้โดยการอ่านข้อมูลจากจานบันทึกข้อมูลจำลองที่ตำแหน่งเซคเตอร์ที่ 0 หรือบูตเซคเตอร์นั่นเอง การทำงานนี้จึงฟังก์ชันการอ่านข้อมูลโดยกำหนดค่าตำแหน่งของเซกเตอร์ที่ต้องการอ่านไว้ที่ 0 แทน ซึ่งสามารถแสดงผังกระแสข้อมูลได้ดังรูปที่ 4.14

4.6.1.4 *ฟังก์ชันการทำงานที่ 04H* เป็นการทำงานโดยการอ่านข้อมูลจากระบบจานบันทึกข้อมูลเสมือน ที่ตำแหน่งเซกเตอร์ที่กำหนดไว้จำนวน 1 เซคเตอร์ โดยมีรูปแบบของคำสั่งแสดงได้ดัง

ในรูปที่ 4.15 ค่าที่ได้จะเป็นข้อมูล 1 เซกเตอร์ คือ 512 ไบต์ ซึ่งจะถ่ายลงในบัฟเฟอร์เพื่อส่งให้กับโปรแกรมต่อไป รูปที่ 4.16 แสดงผังกระแสข้อมูลการทำงานนี้



รูปที่ 4.14 แสดงผังกระแสข้อมูลการทำงานของฟังก์ชัน 02H ในโปรแกรม SV_DISK.SYS

COMMAND CODE	DRIVE NO.	SECTOR NO.
1 BYTE	*1 BYTE*	*2 BYTES*
R	XX	XXXX

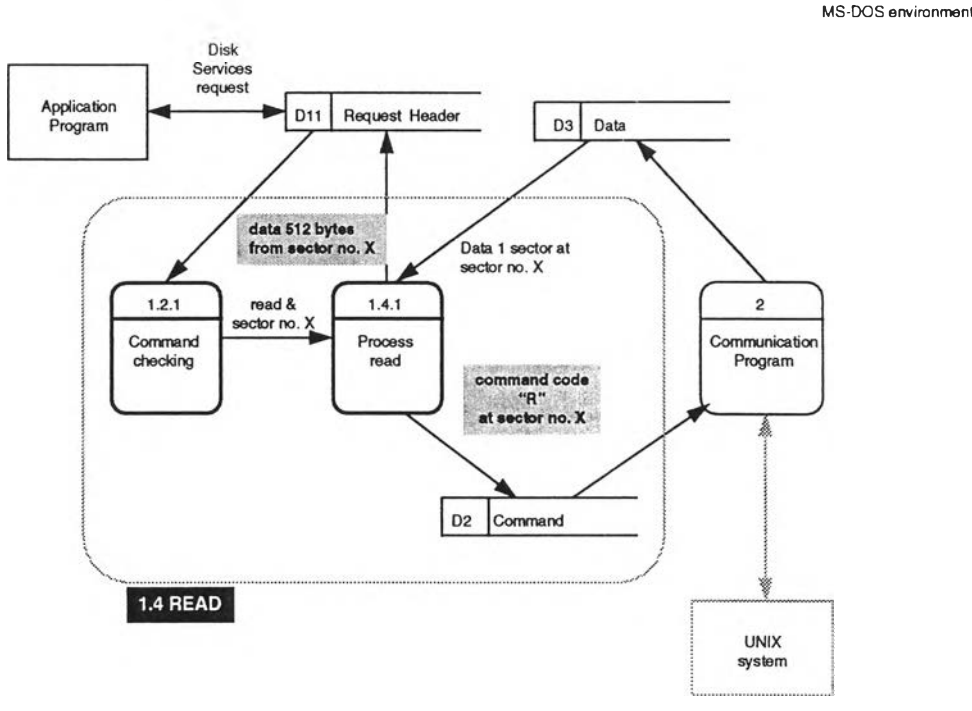
(1) รูปแบบคำสั่งการอ่านข้อมูลจำนวน 1 เซคเตอร์

COMMAND CODE	DATA1	DATA2	...	DATA n
1 BYTE	1 BYTE	1 BYTE		1 BYTE
D				

(2) รูปแบบของข้อมูลที่ได้รับ (n = 512 ไบต์ ในกรณีที่ Byte/Sector = 512)

รูปที่ 4.15 แสดงรูปแบบคำสั่งในฟังก์ชันการทำงาน 04H (Read)

4.6.1.5 *ฟังก์ชันการทำงานที่ 07H* การทำงานจะเป็นการบันทึกข้อมูลลงบนระบบจานบันทึกข้อมูลเสมือน ที่ตำแหน่งเซกเตอร์ที่กำหนดจำนวน 1 เซกเตอร์ หรือมากกว่าโดยที่รูปแบบของคำสั่งดังแสดงได้ในรูปที่ 4.17 การบันทึกจะทำไปอย่างต่อเนื่องจนกว่าจะมีการส่งคำสั่งบอกสิ้นสุดการบันทึก ซึ่งสามารถแสดงผังกระแสข้อมูลได้ดังรูปที่ 4.18



รูปที่ 4.16 แสดงผังกระแสข้อมูลการทำงานของฟังก์ชัน 04H ในโปรแกรม SV_DISK.SYS

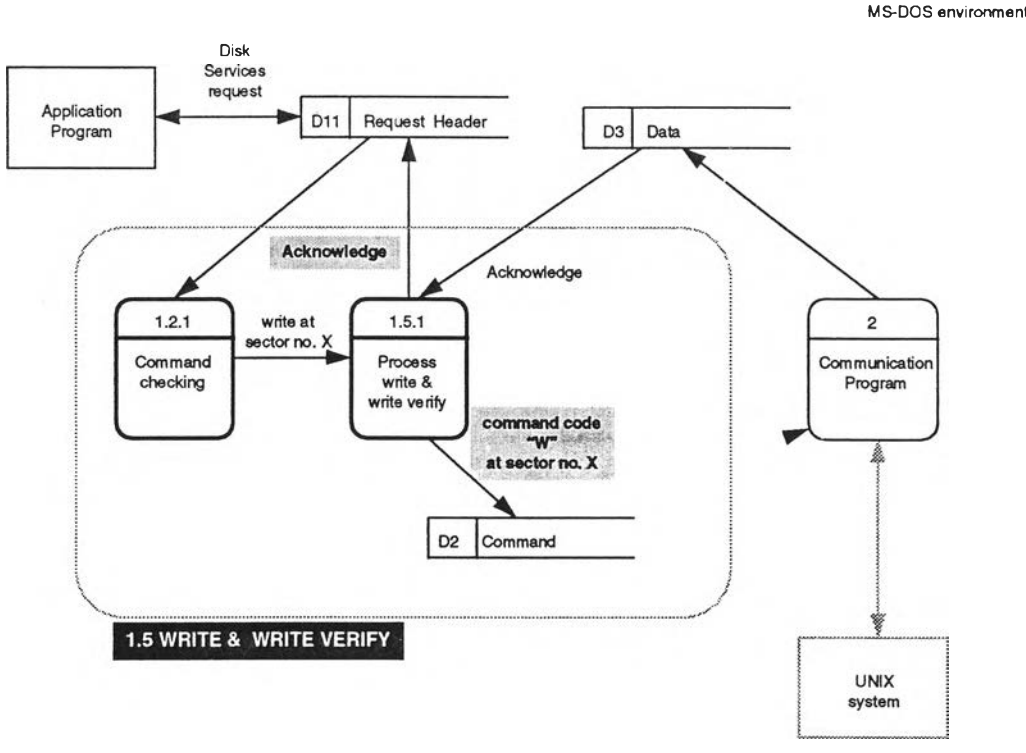
COMMAND CODE	DRIVE NO.	SECTOR NO.	SECTOR DATA
1 BYTE	1 BYTE	2 BYTES	1 .. 1024 BYTES
W	XX	XXXX	XXXXXX

(1) รูปแบบคำสั่งการเขียนข้อมูลจำนวน 1 เซคเตอร์

COMMAND CODE 1 BYTE
T

(2) รูปแบบคำสั่งสิ้นสุดการเขียนข้อมูล

รูปที่ 4.17 แสดงรูปแบบคำสั่งในฟังก์ชันการทำงาน Write



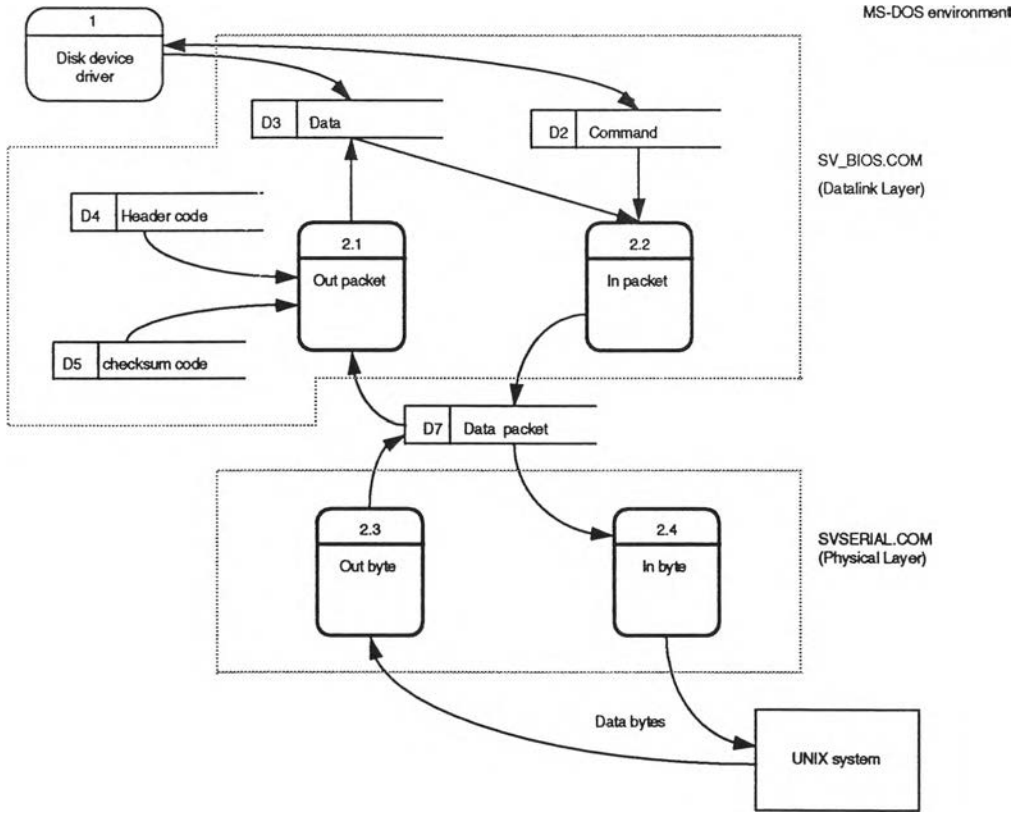
รูปที่ 4.18 แสดงผังกระแสข้อมูลการทำงานของฟังก์ชัน 07H ในโปรแกรม SV_DISK.SYS

4.6.1.6 *ฟังก์ชันการทำงานที่ 08H* การทำงานจะเหมือนกับฟังก์ชันการทำงาน 07H แต่จะมีการตรวจสอบว่ามีการป้องกันการเขียนหรือไม่เพิ่มเติมเข้ามา

4.6.1.7 *ฟังก์ชันการทำงานที่ 03H, 05H และ 06H* ไม่มีการทำงานใดๆ ในฟังก์ชันเหล่านี้

4.6.2 โปรแกรมสื่อสารในส่วนของระบบปฏิบัติการเอ็มเอสดอส

ประกอบไปด้วยโปรแกรม 2 โปรแกรม คือ SV_*BIOS.COM* เป็นโปรแกรมที่ทำงานรับส่งข้อมูลในระดับข่ายเชื่อมโยงข้อมูล และ SV*SERIAL.COM* เป็นโปรแกรมในระดับกายภาพ การทำงานของโปรแกรมทั้งสองโปรแกรมสามารถแสดงได้ด้วยผังกระแสข้อมูลดังรูปที่ 4.19 ซึ่งมีการทำงานดังนี้



รูปที่ 4.19 แสดงผังกระแสข้อมูลของโปรแกรมสื่อสารในส่วนของระบบปฏิบัติการเอ็มเอสดอส

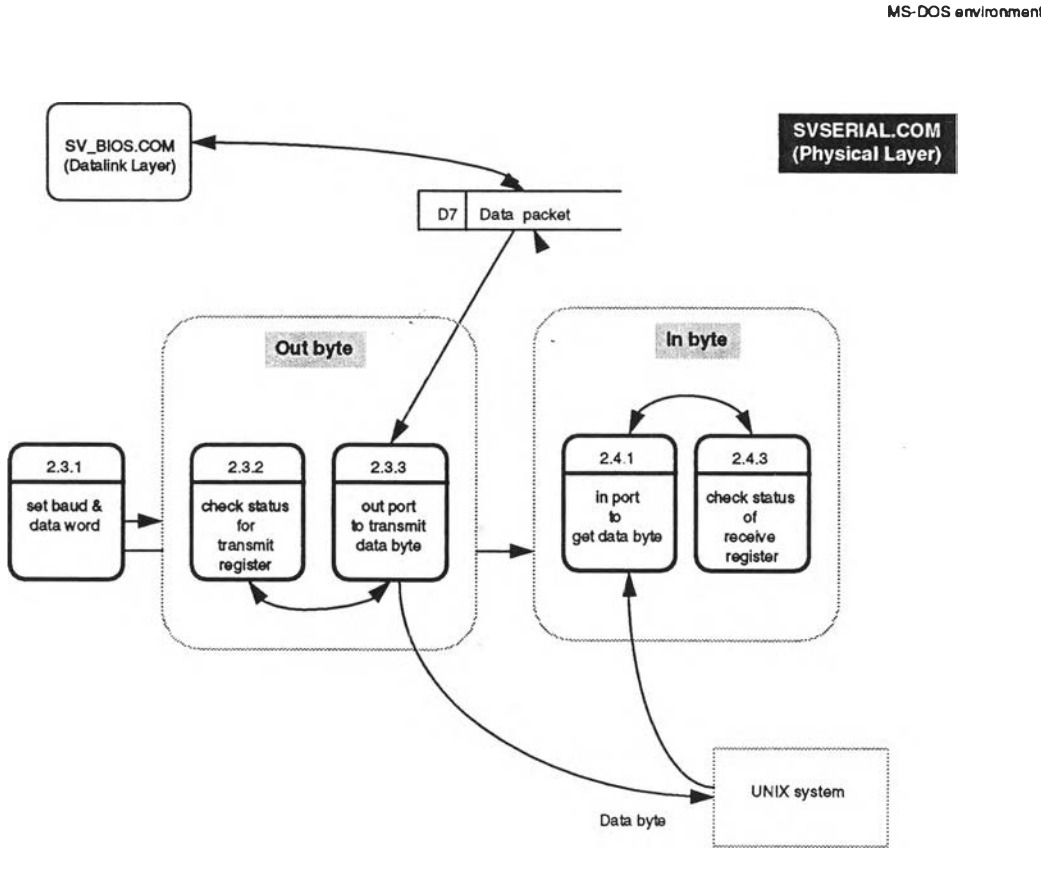
4.6.2.1 *โปรแกรม SVSERIAL.COM* โปรแกรมนี้จะทำหน้าที่ในการรับส่งข้อมูล ซึ่งประกอบไปด้วยฟังก์ชันการทำงานหลัก คือ รับข้อมูล 1ไบต์ (in byte) และส่งข้อมูล 1 ไบต์ (out byte) โปรแกรมนี้จะทำการติดตั้งค่าเริ่มต้นให้กับส่วนของฮาร์ดแวร์ เช่น ความเร็วในการรับส่ง หรือ รูปแบบของข้อมูลที่รับส่ง เป็นต้น โปรแกรมนี้จะถูกเรียกใช้จากโปรแกรมในระดับข่ายเชื่อมโยงข้อมูลอีกทีโดยอาศัยวิธีการขัดจังหวะ รูปที่ 4.20 แสดงให้เห็นผังกระแสข้อมูลการทำงานของโปรแกรมนี้

4.6.2.2 *โปรแกรม SV_BIOS.COM* เป็นโปรแกรมในระดับข่ายเชื่อมโยงข้อมูล ทำหน้าที่บรรจุคำสั่งที่ส่งมาจากโปรแกรมจานบันทึกข้อมูลโปรแกรมย่อยอุปกรณ์ลงในกลุ่มข้อมูลสื่อสารเพื่อส่งไปยังปลายทางและรับกลุ่มข้อมูลสื่อสารมาเปลี่ยนให้เป็นข้อมูลเดิม เพื่อส่งคืนให้กับโปรแกรมจานบันทึกข้อมูลโปรแกรมย่อยอุปกรณ์ต่อไป การทำงานหลักได้แก่ การส่งกลุ่มข้อมูลสื่อสาร (Out packet)

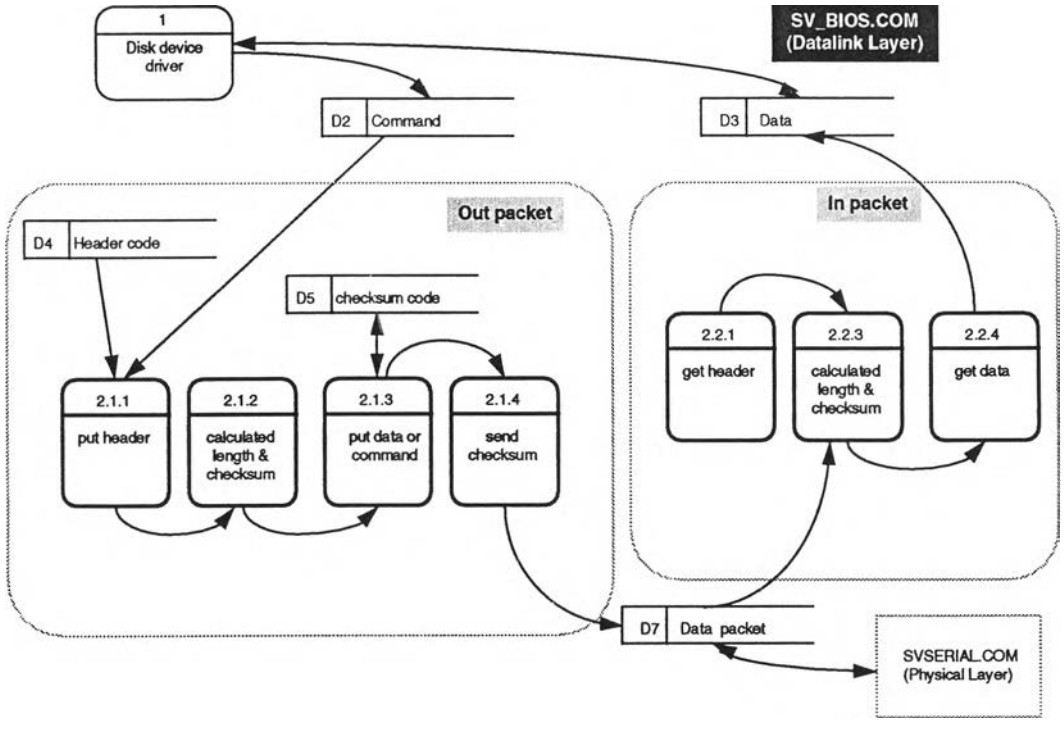
และรับกลุ่มข้อมูลสื่อสาร (In packet) โดยการประกอบส่วนหัว ความยาว และ ผลรวมการตรวจสอบเข้ากับคำสั่ง หรือ ข้อมูลที่ต้องการส่งเป็นกลุ่มข้อมูลสื่อสารแล้วจึงส่งออกไป ในทำนองกลับกันจะทำการถอดส่วนต่างๆ ของกลุ่มข้อมูลสื่อสารที่รับมาให้เหลือเฉพาะข้อมูลที่จะนำไปใช้ต่อไป รูปที่ 4.21 แสดงให้เห็นผังกระแสข้อมูลโปรแกรมนี้

4.6.3 โปรแกรมสื่อสารในส่วนของระบบปฏิบัติการยูนิกซ์บนเครื่องแม่ข่าย

ในทำนองเดียวกันกับการพัฒนาโปรแกรมการสื่อสารบนระบบปฏิบัติการเอ็มเอสดอส โปรแกรมนี้จะแบ่งการทำงานออกเป็น 2 ส่วน คือ *ระดับกายภาพ* และ *ระดับข่ายเชื่อมโยงข้อมูล* ซึ่งจะทำให้การพัฒนาสามารถและการทำความเข้าใจทำได้ง่ายขึ้น รูปที่ 4.22 แสดงให้เห็นผังกระแสข้อมูลของโปรแกรมสื่อสารนี้ โปรแกรมทั้งหมดมีรายละเอียดดังนี้ คือ



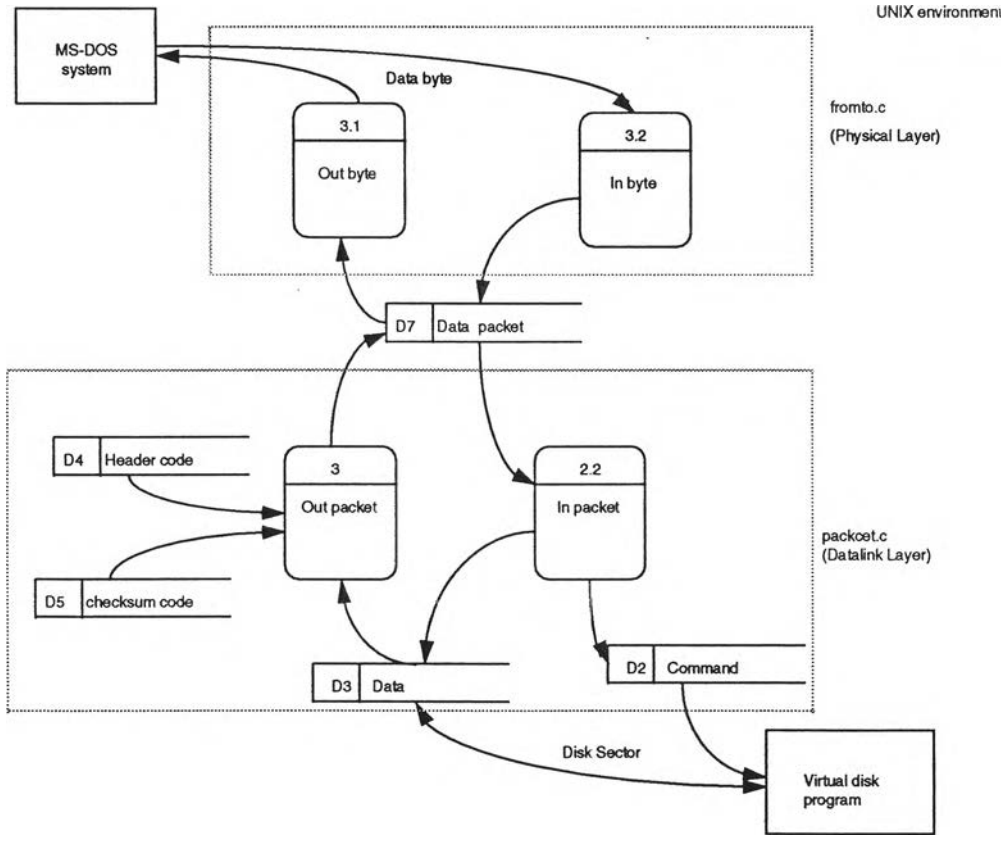
รูปที่ 4.20 แสดงผังกระแสข้อมูลของการทำงานของโปรแกรม SVSERIAL.COM



รูปที่ 4.21 แสดงผังกระแสข้อมูลการทำงานของโปรแกรม SV_BIOS.COM

4.6.3.1 *โปรแกรมอ่านบันทึกข้อมูลผ่านจอเทอร์มินอล*

เป็นโปรแกรมในระดับกายภาพ จากการที่ได้ศึกษามาแล้วพบว่าในระบบยูนิกซ์นั้นไม่นิยมให้ผู้ใช้ทำการติดต่อกับเทอร์มินอลโดยตรง แต่จะให้ผู้ใช้ติดต่อกับระบบเทอร์มินอลในลักษณะของแฟ้มแบบหนึ่ง ดังนั้น การส่งข้อมูลหรือรับข้อมูล 1 ไบต์จากทางเข้าออกสื่อสารของระบบปฏิบัติการยูนิกซ์จึงสามารถทำได้ โดยการใช้ฟังก์ชันการอ่านบันทึกแฟ้มที่อ้างอิงด้วยชื่อของจอเทอร์มินอลที่เราต้องการติดต่อด้วย การทำงานที่มี คือ การตั้งค่าเริ่มต้นให้กับโปรแกรมเทอร์มินอลไดรฟเวอร์ เช่น ความเร็วในการรับส่งข้อมูล โหมดการทำงานในแบบรอว์ เป็นต้น โปรแกรมนี้บรรจุอยู่ในโปรแกรมชื่อ *tty_open.c* และ *tty_raw.c* โปรแกรมการอ่านและบันทึกข้อมูลผ่านทางเข้าออกสื่อสารบรรจุไว้ในโปรแกรมชื่อ *fromto.c* และโปรแกรมติดต่อและยกเลิกการใช้จอเทอร์มินอล เพื่อให้ทางเข้าออกสื่อสารกลับคืนสภาพการทำงานตามปกติโปรแกรมนี้บรรจุอยู่ในโปรแกรมชื่อ *connect.c* ซึ่งทำหน้าที่ในการสร้างกระบวนการเพื่อทำงานนี้ด้วย สามารถแสดงผังกระแสข้อมูลได้ดังรูปที่ 4.23



รูปที่ 4.22 แสดงผังกระแสข้อมูลของโปรแกรมสื่อสารบนระบบปฏิบัติการยูนิกซ์

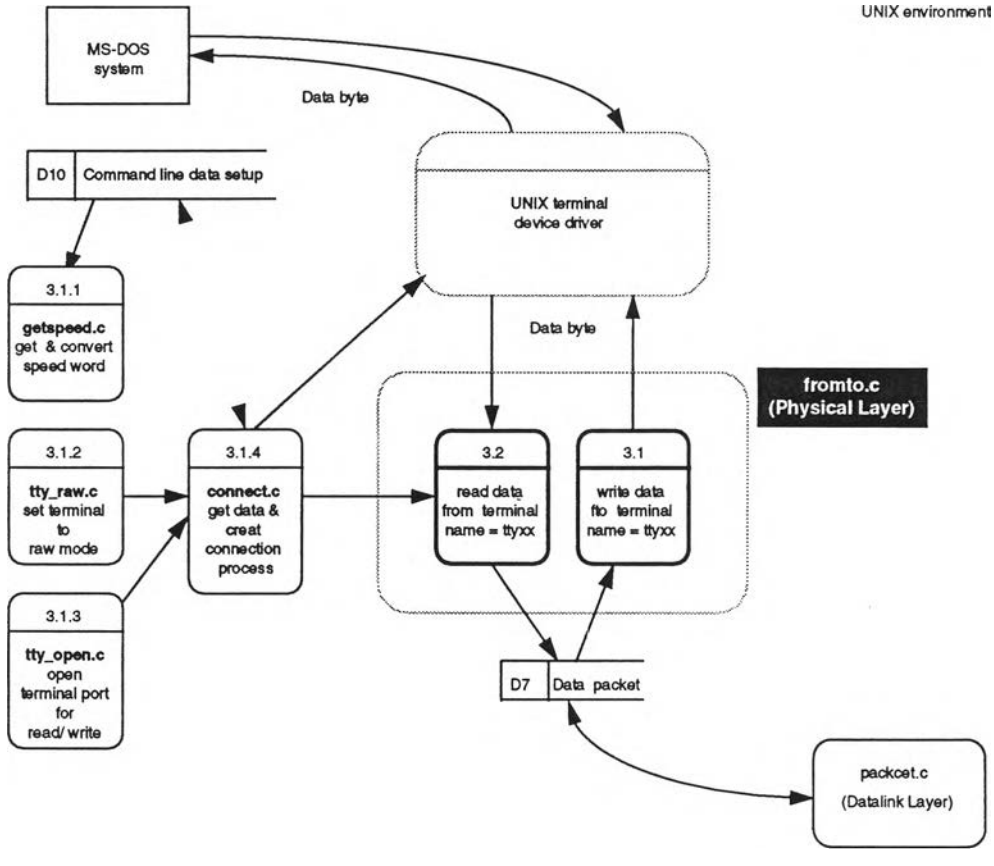
4.6.3.2 *โปรแกรมข่ายเชื่อมโยงข้อมูล*

โปรแกรมนี้จะทำหน้าที่ในการรับส่งข้อมูลแบบกลุ่มข้อมูลสื่อสาร เช่นเดียวกับในระบบปฏิบัติการเอ็มเอสดอส ซึ่งจะประกอบไปด้วยฟังก์ชันการทำงานที่เหมือนกัน คือ ส่งกลุ่มข้อมูลสื่อสารและรับกลุ่มข้อมูลสื่อสาร โปรแกรมนี้ชื่อว่า *packet.c* โดยแสดงผังกระแสข้อมูลได้ดังรูปที่ 4.24

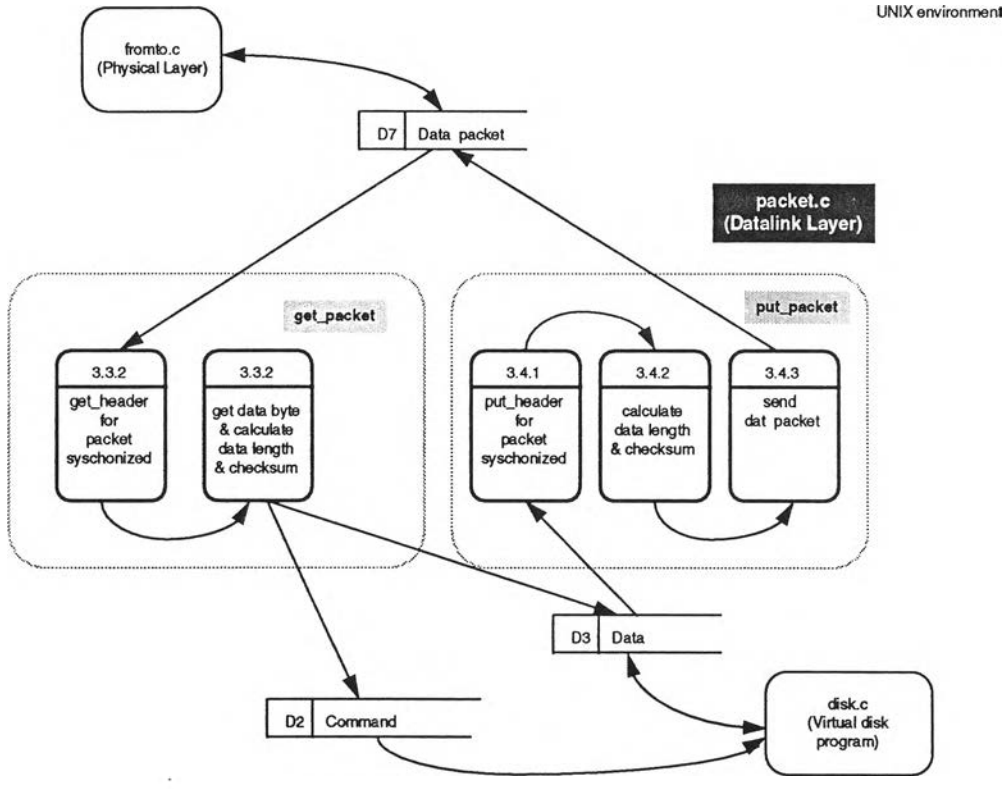
4.6.4 โปรแกรมเลียนการทำงานของจานบันทึกข้อมูลบนระบบปฏิบัติการยูนิกซ์

โปรแกรมเลียนการทำงานของจานบันทึกข้อมูลแบบระบบปฏิบัติการเอ็มเอสดอสในระบบดอสเชลล์นั้น พัฒนาโดยการสร้างแฟ้ม 1 แฟ้มบนระบบยูนิกซ์ให้เป็นเสมือนจานบันทึกข้อมูล 1 เครื่อง สำหรับใช้งานโดยเครื่องลูกข่าย อาศัยหลักการที่ว่าหน่วยสำรองข้อมูลในระบบปฏิบัติการเอ็มเอสดอสจะถูกมองเป็นเสมือนลูกโซ่ของเซคเตอร์ขนาดเท่า ๆ กันต่อเนื่องกันไป ข้อมูลที่กำหนดคุณลักษณะของ

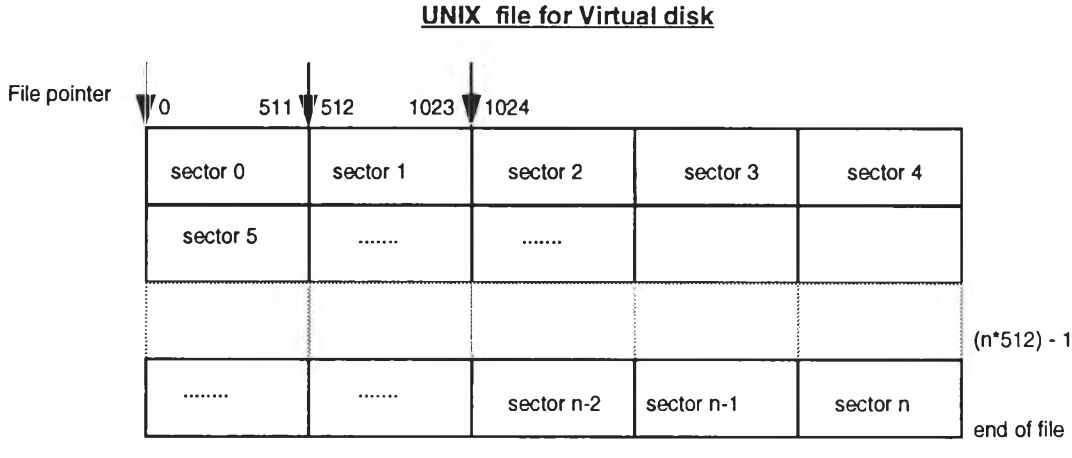
จานบันทึกข้อมูลจะกำหนดไว้ในบูตเซกเตอร์หรือเซกเตอร์ที่ 0 รูปที่ 4.25 แสดงให้เห็นภาพจำลองของจานบันทึกข้อมูลเสมือนในแฟ้มของระบบยูนิกซ์ และรูปที่ 4.26 แสดงให้เห็นผังกระแสข้อมูลการทำงานของโปรแกรมนี้ ซึ่งจะมีการทำงานอยู่ 3 ฟังก์ชัน คือ ฟังก์ชันที่ 1 ได้แก่การตรวจอุปกรณ์ว่ามีการเปลี่ยนหรือเปล่า ฟังก์ชันที่ 2 คือ ทำการอ่านข้อมูลจากจานบันทึกข้อมูลจำลองจำนวน 1 เซกเตอร์ และ ฟังก์ชันที่สุดท้ายเป็นการบันทึกข้อมูลลงในจานบันทึกข้อมูลเสมือนจำนวน 1 เซกเตอร์ การทำงานอ่านหรือบันทึกข้อมูลจะเป็นลักษณะการอ่านหรือบันทึกข้อมูลจากแฟ้มเป็น 1 ระเบียนแทน ซึ่งขนาดของระเบียนจะเท่ากับ 512 ไบต์ หรือเท่ากับขนาดของเซกเตอร์ของจานบันทึกข้อมูลในระบบปฏิบัติการเอ็มเอสดอสนั่นเอง โปรแกรมการทำงานนี้ชื่อว่า *disk.c* ซึ่งแสดงผังกระแสข้อมูลได้ดังรูปที่ 4.27



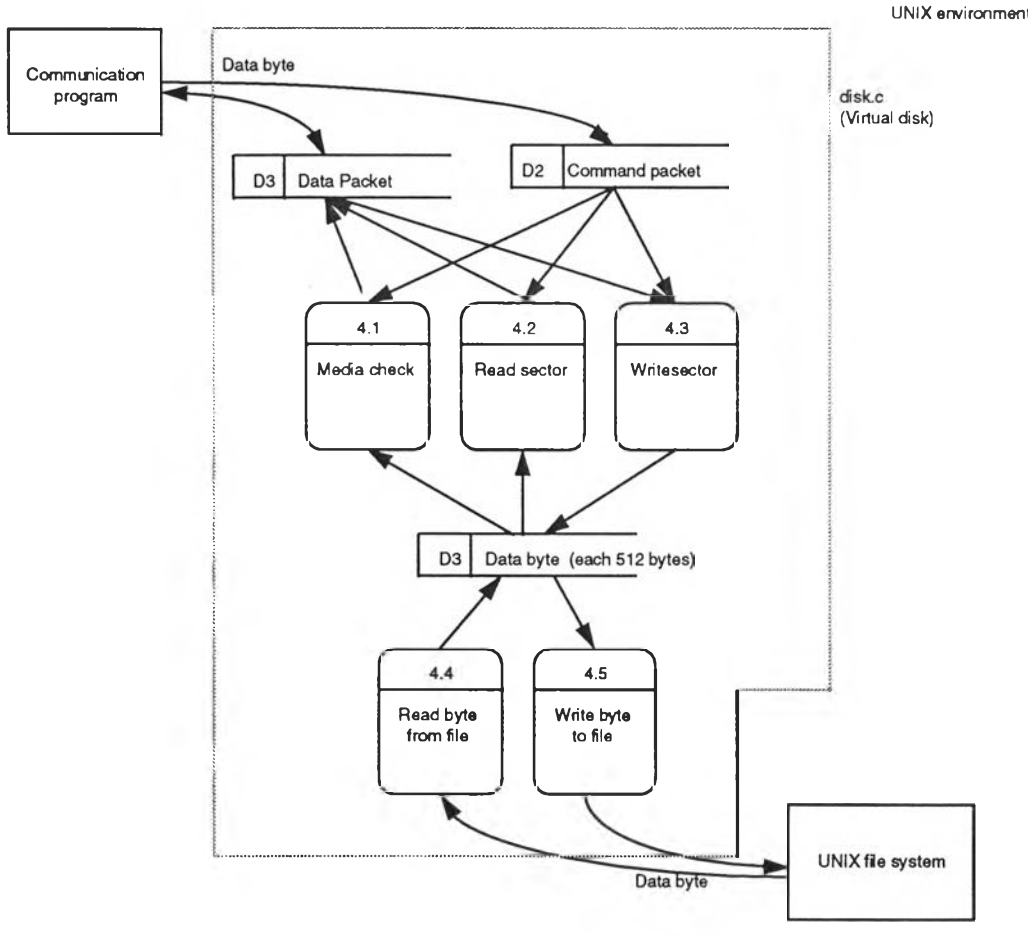
รูปที่ 4.23 แสดงผังกระแสข้อมูลของโปรแกรมการอ่านบันทึกข้อมูลผ่านจอเทอร์มินอล



รูปที่ 4.24 แสดงผังกระแสข้อมูลของโปรแกรมข่ายเชื่อมโยงข้อมูล



รูปที่ 4.25 แสดงภาพจำลองของโครงสร้างแฟ้มจานบันทึกข้อมูลเสมือน



รูปที่ 4.26 แสดงผังกระแสข้อมูลของโปรแกรมเลียนจานบันทึกข้อมูล

4.7 การติดตั้งระบบคอสเซลล์

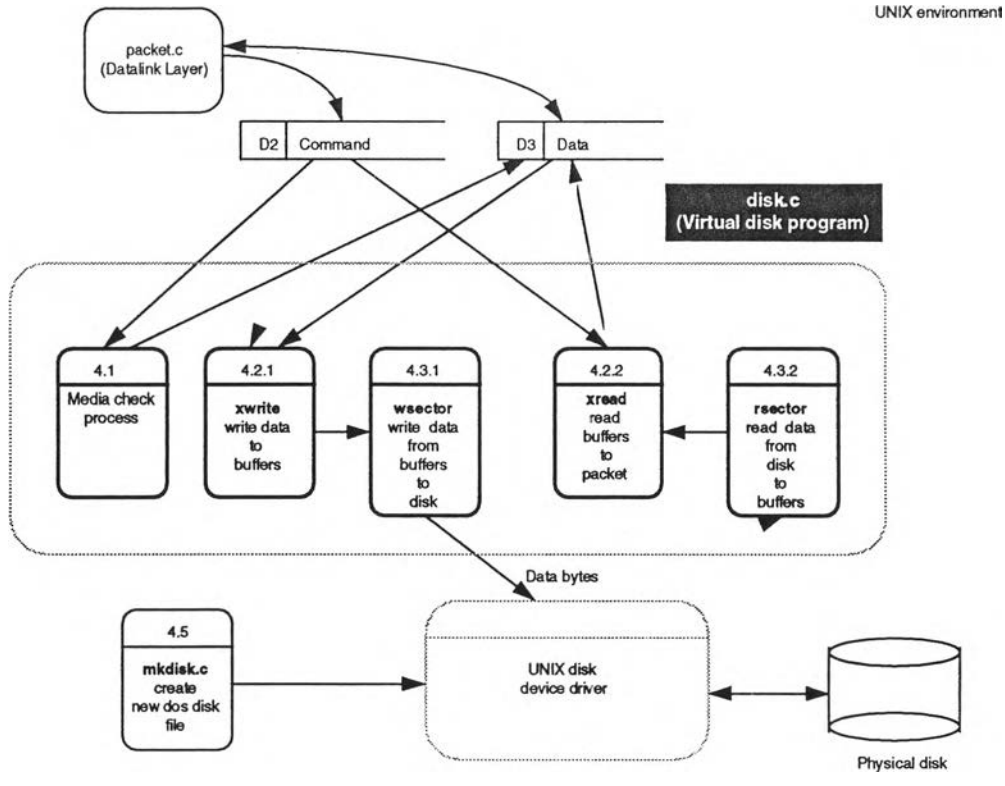
4.7.1 การติดตั้งระบบฮาร์ดแวร์

สามารถติดตั้งตามแบบระบบยูนิกซ์ปกติโดยผ่านทางเข้าออก RS-232C ที่จะเชื่อมโยงลูกข่ายเข้ากับเครื่องแม่ข่ายทำให้สามารถติดต่อสื่อสารกันได้ ซึ่งรายละเอียดได้กล่าวมาแล้วในบทต้น ๆ

4.7.2 การติดตั้งโปรแกรมบนเครื่องคอมพิวเตอร์ลูกข่าย

ในการติดตั้งบนเครื่องลูกข่ายซึ่งใช้ระบบปฏิบัติการเอ็มเอสดอส สามารถทำได้โดยทำการสำเนาแฟ้ม SV_BIOS.COM , SVSERIAL.COM และ SV_DISK.SYS ไว้ในแผ่นจานบันทึกข้อมูลชนิด

อ่อนที่ใช้ในการบูตเครื่องแล้วทำการแก้ไขแฟ้ม AUTOEXEC.BAT และแฟ้ม CONFIG.SYS ที่ใช้ในการเริ่มต้นการทำงานของระบบดังต่อไปนี้



รูปที่ 4.27 แสดงผังกระแสข้อมูลของโปรแกรมเลียนจานบันทึกข้อมูลโดยละเอียด

4.7.2.1 *การแก้ไขแฟ้ม AUTOEXEC.BAT* ทำการเพิ่มคำสั่งต่อไปนี้ในแฟ้ม คือ

A:\SV_BIOS และ A:\SVSERIAL /$3F8 /3 /12

4.7.2.2 *การแก้ไขแฟ้ม CONFIG.SYS* ทำการเพิ่มคำสั่งเข้าไปในแฟ้มดังต่อไปนี้ คือ

DEVICE = A:\SV_DISK.SYS /C /0

เมื่อทำการบันทึกข้อมูลแล้วให้ทำการบูตเครื่องลูกข่ายใหม่ จากนั้นสามารถใช้งานได้ตามปกติ โดยจานบันทึกข้อมูลเสมือนจะเป็นจานบันทึกข้อมูลที่มีชื่อลำดับถัดไปจากที่มีอยู่เดิมบนเครื่อง

ลูกข่าย แต่ก่อนอื่นต้องตรวจสอบดูด้วยว่าเครื่องแม่ข่ายได้วิ่งโปรแกรมจำลองจานบันทึกข้อมูลไว้แล้วการทำงานจึงจะถูกต้อง รูปที่ 4.28แสดงลำดับของโปรแกรมในหน่วยความจำของระบบปฏิบัติการเอ็มเอสดอส

รูปที่ 4.28 แสดงการลำดับและตำแหน่งการติดตั้งในหน่วยความจำ
ของโปรแกรม dosshell ต่าง ๆ ในระบบเอ็มเอสดอส

4.7.3 การติดตั้งโปรแกรมบนเครื่องคอมพิวเตอร์แม่ข่าย สามารถทำได้ดังต่อไปนี้

4.7.3.1 *การสำเนาโปรแกรม sv_shell* ซึ่งเป็นโปรแกรมคอสเชลล์ที่ทำการแปลภาษาแล้วลงในจานบันทึกข้อมูลในไดเรกทอรีที่ต้องการใช้งาน

4.7.3.2 *ทำการสร้างจานบันทึกข้อมูลเสมือน* โดยโปรแกรมสร้างแฟ้มที่สร้างไว้ในระบบปฏิบัติการยูนิกซ์ชื่อ *mkdisk.c*

4.7.3.3 *ทำการวิ่งโปรแกรม sv_shell* โดยใช้คำสั่ง ./sv_shell หรือ ./sv_shell & ในกรณีที่ต้องการให้โปรแกรมฝังตัวไว้

4.7.3.4 *ทำการกำหนดรูปแบบจานบันทึกข้อมูลเสมือน* โดยโปรแกรมกำหนดรูปแบบจากเครื่องลูกข่ายชื่อ *xformat.com* ก็สามารถนำไปใช้งานได้ตามปกติ

ในกรณีที่เครื่องแม่ข่ายเป็นระบบยูนิกซ์อื่น ๆ ที่ไม่ใช่ของบริษัท ซานตาครูซ โอเปอเรชัน ให้ทำการสำเนาโปรแกรมลงเครื่องโดยใช้คำสั่ง *doscp* ในไดเรกทอรีที่ต้องการใช้งาน ทำการแก้ไขแฟ้มโปรแกรมต้นฉบับในส่วนของรหัสสิ้นสุดบรรทัดและสิ้นสุดแฟ้มให้ตรงกับระบบปฏิบัติการยูนิกซ์ที่ใช้อยู่บนเครื่องแม่ข่ายนั้น ๆ และอีกประการหนึ่งก็คือ ให้แปลี่ยนค่าแฟลกในการเปิดใช้เทอร์มินอลในส่วนของการอินพุตดังตัวอย่างในรูปที่ 4.29 จากนั้นจึงทำการแปลภาษาโดยใช้ตัวแปลภาษาซีมาตรฐานของระบบก็สามารถทำงานได้เช่นเดียวกัน

```
/* change required characteristics */

tdescrip.c_iflag  =     DOSMODE;      /*(IXON|IXOFF); */
tdescrip.c_oflag  =     0;
tdescrip.c_lflag  =     0;
```

* *ค่าแฟลก* **DOSMODE** *ใช้สำหรับ ระบบที่ใช้ระบบปฏิบัติการเอสซีโอยูนิกซ์ และ* **0** *สำหรับระบบปฏิบัติการยูนิกซ์อื่น ๆ*

รูปที่ 4.29 แสดงแฟลกที่ใช้ในการเปิดใช้งานเทอร์มินอลในแฟ้ม tty_raw.c