หอสมุดกลาง จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 2

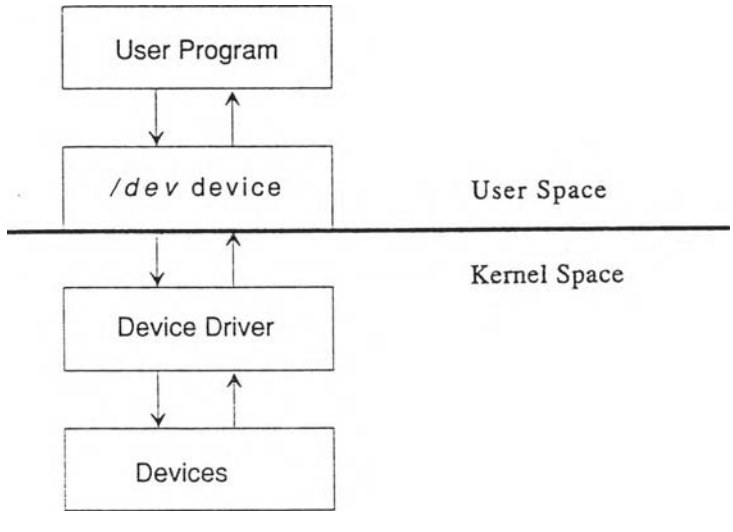
ทฤษฎีและแนวความคิดที่นำมาใช้ในงานวิจัย

2.1 โปรแกรมย่อยควบคุมอุปกรณ์บนระบบปฏิบัติการยูนิกซ์ (The device driver on UNIX system)

2.1.1 วิธีการควบคุมอุปกรณ์ของระบบปฏิบัติการยูนิกซ์ (EGAN AND TEIXEIKA, 1988)

ระบบปฏิบัติการยูนิกซ์ ได้ถูกออกแบบมาเพื่อให้เกิดความสะดวกในการใช้งานและการพัฒนาโปรแกรม โดยลดความซับซ้อนและสร้างระบบที่มีรูปแบบการติดต่อที่เป็นอันหนึ่งอันเดียวกัน (uniform interface) สำหรับองค์ประกอบต่าง ๆ ของระบบปฏิบัติการ

การควบคุมอุปกรณ์ต่าง ๆ ของระบบปฏิบัติการยูนิกซ์ก็เช่นเดียวกัน โปรแกรมของผู้ใช้จะติดต่อกับอุปกรณ์ในระบบในลักษณะเดียวกันกับการติดต่อกับ แฟ้มข้อมูลทั่วไป โดยจะมีชื่อแฟ้มข้อมูลอย่างน้อยหนึ่งชื่อที่เป็นช่องทางติดต่อกับอุปกรณ์แต่ละหน่วย โปรแกรมต่างๆจะติดต่อกับอุปกรณ์ผ่านชื่อแฟ้มข้อมูลประจำอุปกรณ์



ภาพที่ 2.1 แสดงการติดต่อระหว่างโปรแกรมประยุกต์และอุปกรณ์ผ่านโปรแกรมควบคุมอุปกรณ์

แฟ้มข้อมูลในระบบปฏิบัติการยูนิกซ์แบ่งออกได้เป็น 3 ประเภท คือ

1. แฟ้มข้อมูลปกติ (regular file)
2. แฟ้มข้อมูลสารบบ (directory file)
3. แฟ้มข้อมูลพิเศษ (special file) ซึ่งแบ่งออกเป็น
   - 3.1 ไปป์ (Pipes)
   - 3.2 เนมไปป์ (Named pipes) หรือไฟโฟ (First-In,First-Out)
   - 3.3 แฟ้มอุปกรณ์นำเข้า/ส่งออกข้อมูล (I/O device files) ซึ่งต่อไปจะขอเรียกสั้นๆ ว่า "แฟ้มอุปกรณ์"

เคอร์เนลจะทราบประเภทของแฟ้มข้อมูล โดยการตรวจที่ไอโนด (i-node) ของแฟ้มข้อมูลกล่าวคือ ที่ไอโนดจะมีข้อมูลบอกประเภทของแฟ้มไว้ นอกจากนี้ถ้าเป็นไอโนดของแฟ้มอุปกรณ์จะมีหมายเลขหลัก (major number) และหมายเลขรอง (minor number) ของอุปกรณ์อยู่

2.1.2 การจัดแบ่งประเภทอุปกรณ์ของระบบปฏิบัติการยูนิกซ์

ยูนิกซ์จะแบ่งอุปกรณ์ต่างๆ ที่ใช้งานในระบบออกเป็น 2 ประเภท คือ

1. อุปกรณ์แบบบล็อค (Block device)

หมายถึง อุปกรณ์ที่มีการแบ่งเก็บข้อมูลเป็นบล็อค ๆ ในขนาดบล็อคที่คงที่ โดย สามารถเข้าถึงข้อมูลบล็อคใดๆ ได้ในแบบสุ่ม (randomly addressable) การส่งผ่านข้อมูล ระหว่างอุปกรณ์และโปรเซสของผู้ใช้ (user process) จะผ่านระบบบัฟเฟอร์กลางของเคอร์เนล โดยเคอร์เนลมีระบบการดูแลบัฟเฟอร์ในตัวเอง รวมทั้งจะคอยรองรับความแตกต่างของขนาดข้อมูล ที่โปรเซสขอเขียนหรืออ่านกับอุปกรณ์ โดยจะแปลงให้อยู่ในรูปแบบของข้อมูลขนาดคงที่ และติดต่อเขียนหรืออ่านข้อมูล กับอุปกรณ์ในลักษณะของข้อมูลขนาดคงที่นั้น อุปกรณ์แบบบล็อคสามารถจะเก็บระบบแฟ้มข้อมูลของยูนิกซ์ (UNIX file system) ได้ และสามารถจะถูกใส่ (mount) เข้ามาในระบบแฟ้มข้อมูลยูนิกซ์ที่ทำงานอยู่ได้เช่นกัน

2. อุปกรณ์แบบคาแรคเตอร์ (Character device)

หมายถึง อุปกรณ์อื่นๆนอกเหนือจากอุปกรณ์ที่มีลักษณะแบบบล็อค โดยอุปกรณ์แบบคาแรคเตอร์บางประเภทไม่สามารถเก็บข้อมูลไว้ได้ เช่น จอภาพ และ เครื่องพิมพ์ เป็นต้น อุปกรณ์บางประเภทสามารถเก็บข้อมูลในขนาดใดๆ ที่แตกต่างกัน เช่น อุปกรณ์เทป การส่งผ่านข้อมูลระหว่างอุปกรณ์และโปรเซสของผู้ใช้ (user process) จะทำได้โดยตรง ไม่ผ่านระบบบัฟเฟอร์ของเคอร์เนล

อุปกรณ์แต่ละชนิดจะมีโปรแกรมควบคุมอุปกรณ์ (Device driver) ที่จะคอยควบคุมสั่งงานอุปกรณ์ตามการร้องขอของโปรแกรมผู้ใช้ โดยโปรแกรมควบคุมอุปกรณ์หนึ่งชุดจะควบคุมอุปกรณ์ชนิดนั้นได้หลายอุปกรณ์ โปรแกรมควบคุมอุปกรณ์เมื่อได้รับการพัฒนาขึ้นแล้ว จะถูกเชื่อมโยง(link) เข้าไปเป็นส่วนหนึ่งของเคอร์เนล การเรียกใช้โปรแกรมควบคุมอุปกรณ์จะกระทำโดยผ่าน ตารางข้อมูลของระบบ 2 ตาราง คือ ตารางสวิทช์สำหรับอุปกรณ์แบบบล็อค และตารางสวิทช์สำหรับอุปกรณ์แบบคาแรคเตอร์ ตารางสวิทช์แต่ละตารางจะแบ่งข้อมูลออกเป็นรายการ แต่ละรายการจะเก็บข้อมูลตำแหน่งที่ในหน่วยความจำ (address) ของโปรแกรมย่อยที่ทำหน้าที่ต่างๆ ของโปรแกรมควบคุมอุปกรณ์หนึ่งชนิด โดยลำดับที่ของรายการ ข้อมูลเป็นหมายเลขหลักของอุปกรณ์ (major device number)

block device switch table			
entry	open	close	strategy
0	gdopen	gdclose	gdstrategy
1	gtopen	gtclose	gtstrategy

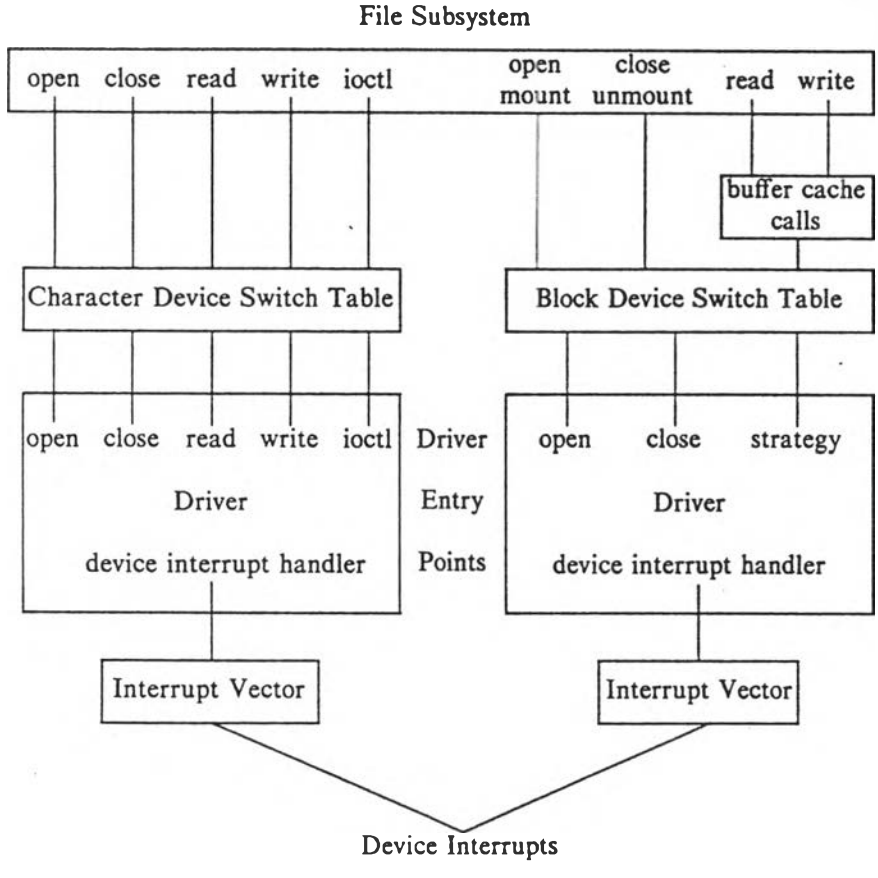
character device switch table					
entry	open	close	read	write	ioctl
0	conopen	conclose	conread	conwrite	conioctl
1	dzbopen	dzbclose	dzbread	dzbwrite	dzbioctl
2	syopen	nulldev	syread	sywrite	syioctl
3	nulldev	nulldev	mmread	mmwrite	nodev
4	gdopen	gdclose	gdread	gdwrite	nodev
5	gtopen	gtclose	gtread	gtwrite	nodev

ภาพที่ 2.2 แสดงตัวอย่างตารางสวิทช์สำหรับอุปกรณ์แบบบล็อค และแบบคาแรคเตอร์

เคอร์เนลจะเรียกใช้โปรแกรมควบคุมอุปกรณ์สำหรับอุปกรณ์แต่ละชนิดตามขั้นตอนดังนี้ เมื่อโปรแกรมผู้ใช้เรียกใช้อุปกรณ์ด้วยการเรียกใช้ฟังก์ชั่นของระบบ (function call) ต่างๆ ตัวอย่างเช่น ฟังก์ชั่น open พร้อมทั้งระบุชื่อแฟ้มอุปกรณ์ที่จะติดต่อด้วย เคอร์เนลจะเริ่มต้นค้นหา ไอโนดของแฟ้มดังกล่าวไปตามสารบบ เมื่อพบไอโนดของแฟ้มแล้วจะตรวจสอบว่า เป็นแฟ้มประเภทใด ในกรณีที่เป็นแฟ้มอุปกรณ์ จะตรวจสอบว่าเป็นแฟ้มอุปกรณ์แบบบล็อค หรือ แบบคาแรคเตอร์ และมีหมายเลขหลัก หมายเลขรองของอุปกรณ์เท่าไร? เคอร์เนลจะไปยังตารางสวิทช์ของอุปกรณ์ประเภทนั้น พร้อมทั้งค้นหารายการข้อมูลที่มีลำดับที่เป็นหมายเลขเดียวกันกับหมายเลขหลักของอุปกรณ์และเรียกใช้โปรแกรมย่อย ที่สัมพันธ์กับฟังก์ชั่นที่โปรแกรมผู้ใช้ร้องขอการทำงานมา ซึ่งในกรณีนี้คือโปรแกรมย่อย open โดยส่งข้อมูลหมายเลขหลักและรองให้กับโปรแกรมย่อยนั้นไปทำงานในขั้นตอนการทำงานของโปรแกรมย่อยอาจจะมีการเรียกใช้ โปรแกรมย่อยอื่นของโปรแกรม

ควบคุมอุปกรณ์ชุดเดียวกันทำงาน เช่น ในกรณีที่อุปกรณ์ทำงานตามคำสั่งที่ได้รับจากโปรแกรมย่อยเสร็จเรียบร้อย จะส่งสัญญาณขัดจังหวะ (Interrupt) เข้ามาที่หน่วยประมวลผลกลาง ยูนิกซ์จะตรวจสอบสัญญาณขัดจังหวะ ว่ามาจากอุปกรณ์ใด พร้อมทั้งเรียกโปรแกรมย่อยที่ทำหน้าที่ตอบรับสัญญาณขัดจังหวะ (Interrupt routine) มาทำงาน เป็นต้น

2.1.3 โปรแกรมย่อยที่จำเป็นต้องมีของโปรแกรมควบคุมอุปกรณ์แบบบล็อค



ภาพที่ 2.3 แสดงจุดเข้าของโปรแกรมควบคุมอุปกรณ์แบบบล็อคและแบบคาแรคเตอร์

โปรแกรมควบคุมอุปกรณ์แบบบล็อค จำเป็นต้องมีโปรแกรมย่อยดังต่อไปนี้

1. โปรแกรมเริ่มต้นการทำงานอุปกรณ์ (Init routine)

โปรแกรมย่อยนี้มีลักษณะพิเศษคือ จะถูกเรียกโดยอัตโนมัติในขบวนการเริ่มต้นการทำงานของเครื่อง (bootstrap) โดยเมื่อมีการอ่านเคอร์เนลเข้าสู่หน่วยความจำ แล้วชุดคำสั่งบรรจุ (loader) จะส่งต่อการทำงานให้กับเคอร์เนล ซึ่งจะทำการจัดหน่วยความจำต่างๆ และเรียกโปรแกรมเริ่มต้นการทำงานของอุปกรณ์ทุก ๆ ชนิดขึ้นมาทำงาน

โปรแกรมย่อยนี้จะทำหน้าที่แตกต่างกันตามชนิดของอุปกรณ์ เช่น เตรียมเนื้อที่หน่วยความจำสำหรับข้อมูลสถิต (static data) การล้างข้อมูลตัวแปรสถิต การขอจัดสรรทรัพยากรของระบบสำหรับการใช้งานของโปรแกรมควบคุมอุปกรณ์ การตรวจสอบว่าอุปกรณ์ที่ติดต่อด้วย เชื่อมตรง (on-line) หรือไม่ และส่งคำสั่งไปยังอุปกรณ์เพื่อเตรียมอุปกรณ์ให้พร้อมสำหรับการทำงาน เป็นต้น

โปรแกรมย่อยนี้อาจจะมีหรือไม่ก็ได้

2. โปรแกรมเปิดและปิดการติดต่อกับอุปกรณ์ (open and close routine)

โปรแกรมย่อยนี้จะถูกเรียกใช้งานเมื่อโปรแกรมผู้ใช้เรียกใช้ฟังก์ชั่น open หรือใช้คำสั่งใส่ (mount) อุปกรณ์เข้ากับระบบแฟ้มข้อมูลเดิม

โปรแกรมเปิดการติดต่อกับอุปกรณ์จะทำหน้าที่เตรียมอุปกรณ์ให้พร้อมสำหรับการทำงานตามคำสั่งติดต่อกับอุปกรณ์อื่นๆต่อไป เช่น ในกรณีของเครื่องพิมพ์จะส่งคำสั่งไปยังเครื่องพิมพ์ให้เลื่อนกระดาษขึ้นไปจนถึงต้นหน้าถัดไป เป็นต้น

โปรแกรมปิดการติดต่อกับอุปกรณ์จะถูกเรียกใช้งานเมื่อโปรเซสสุดท้ายที่เรียกใช้งานอุปกรณ์ได้เรียกใช้ฟังก์ชั่น close กับอุปกรณ์นั้น

โปรแกรมปิดการติดต่อกับอุปกรณ์จะทำหน้าที่ล้างข้อมูลในตัวแปรที่เกี่ยวข้องกับการควบคุมการทำงานของอุปกรณ์

โปรแกรมเปิดและปิดการติดต่อกับอุปกรณ์อาจทำหน้าที่ในการควบคุมให้มีการใช้งานอุปกรณ์เฉพาะโปรเซส (exclusive access) ตามที่ได้รับการสั่งมาจากฟังก์ชั่น open โดยโปรแกรมเปิดการติดต่อกับอุปกรณ์จะกำหนดตัวบ่งชี้ (flag) เพื่อระบุให้มีการใช้งานอุปกรณ์เฉพาะโปรเซส และโปรแกรมปิดการติดต่อกับอุปกรณ์จะทำหน้าที่ล้าง ค่าตัวบ่งชี้ออกไป

โปรแกรมย่อยนี้อาจจะไม่มีในโปรแกรมควบคุมอุปกรณ์ที่ทำงานไม่ซับซ้อน

3. โปรแกรมกลยุทธ (Strategy routines)

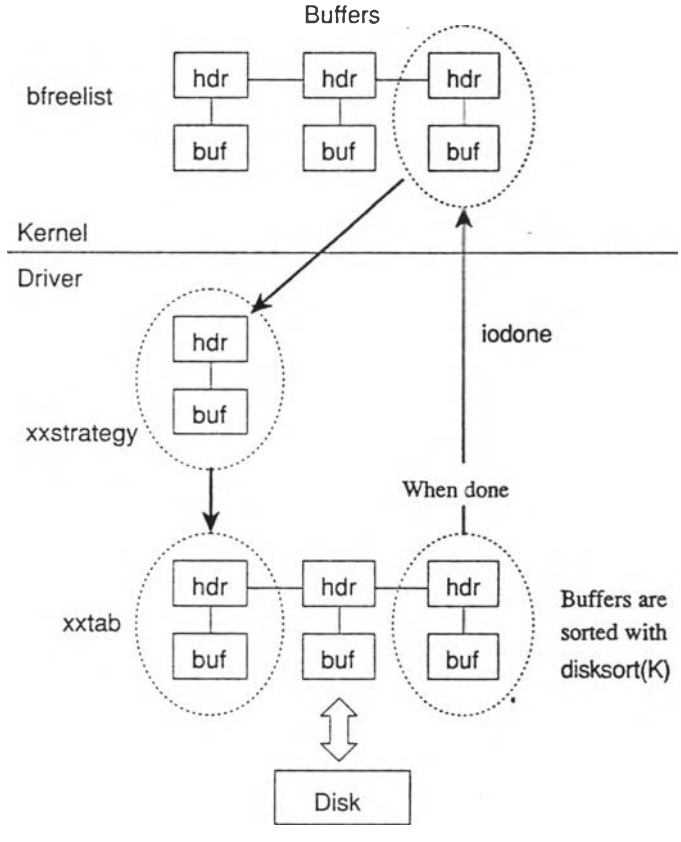
โปรแกรมย่อยนี้จะถูกเรียกใช้งานเมื่อเคอร์เนลได้รับคำสั่งให้ อ่าน หรือเขียนข้อมูล โดยเคอร์เนลจะเรียกให้โปรแกรมกลยุทธทำงานพร้อมกับส่งตัวชี้ตำแหน่งหน่วยความจำของบัฟเฟอร์ ที่จะใช้ในการเขียนหรืออ่านข้อมูล โดยบัฟเฟอร์จะมีข้อมูลเกี่ยวกับ ลักษณะการร้องขอ (อ่านหรือเขียน) ตำแหน่งของข้อมูลในอุปกรณ์ที่จะมีการเขียนหรืออ่าน ตัวชี้เพื่อเชื่อมต่อกับบัฟเฟอร์อื่นๆ เพื่อให้เคอร์เนลสามารถจะดูแลและเชื่อมโยงบัฟเฟอร์ต่างๆ ของระบบเข้าเป็นสายโซ่บัฟเฟอร์ได้(buffer chain)

โปรแกรมกลยุทธจะใช้ข้อมูลในบัฟเฟอร์ในการทำงานกับอุปกรณ์โดยไม่ใช้

ข้อมูลจากยูเอเรียโดยตรง (U area) เนื่องจากขณะที่โปรแกรมกลยุทธทำงาน ยูเอเรียอาจไม่ใช่ยูเอเรียของโปรเซสที่สั่งให้อุปกรณ์ทำงานในครั้งนี้

โปรแกรมกลยุทธจะทำหน้าที่เรียงลำดับคำสั่งที่ติดต่อกับอุปกรณ์ ให้เป็นไปตามตำแหน่งของการเข้าถึงข้อมูล เพื่อให้สะดวกและเพิ่มประสิทธิภาพในการทำงานของอุปกรณ์ และเรียกโปรแกรมเริ่มการทำงานเพื่อเริ่มการทำงานของอุปกรณ์ ตามลักษณะการร้องขอที่ปรากฏในบัฟเฟอร์

โปรแกรมกลยุทธจะต้องมีในโปรแกรมควบคุมอุปกรณ์แบบบล็อคทุกโปรแกรม



ภาพที่ 2.4 แสดงภาพการทำงานของโปรแกรมกลยุทธและฟังก์ชั่นของระบบต่างๆ กับ บัฟเฟอร์

4. โปรแกรมเริ่มต้นการทำงาน (start routine)

โปรแกรมย่อยนี้จะถูกเรียกโดยโปรแกรมกลยุทธ หรือโปรแกรมตอบรับการขัดจังหวะ โปรแกรมย่อยนี้มีหน้าที่ตรวจสอบว่าอุปกรณ์พร้อมทำงานคำสั่งต่อไปหรือไม่ ถ้าพร้อมจะตรวจสอบว่ามีการร้องขอให้อุปกรณ์นั้นทำงานอยู่หรือไม่ (ตรวจสอบว่ามีบัฟเฟอร์ที่รอการนำเข้า/ส่งออกข้อมูลของอุปกรณ์นั้นอยู่หรือไม่) ถ้ามีการร้องขอก็จะเริ่มการทำงานของอุปกรณ์ตามลำดับของการร้องขอ (ที่ได้ถูกเรียงลำดับแล้วโดยโปรแกรมกลยุทธ) ถ้าไม่มีการร้องขอก็จะส่งการทำงานกลับคืนโปรแกรมที่เรียกมา

โปรแกรมเริ่มต้นการทำงานนี้จะต้องมีในโปรแกรมควบคุมอุปกรณ์แบบบล็อคทุกโปรแกรม

5. โปรแกรมตอบรับการขัดจังหวะ (Interrupt service routine)

โปรแกรมย่อยนี้จะถูกเรียกให้ทำงาน เพื่อตอบรับการส่งสัญญาณขัดจังหวะของอุปกรณ์ เพื่อตรวจสอบผลของการทำงานของอุปกรณ์ ส่งข้อมูลที่ได้รับกลับไปยังโปรเซสที่ร้องขอการทำงานมา ทำการปลดปล่อยบัฟเฟอร์ และเรียกโปรแกรมเริ่มต้นการทำงาน เพื่อทำงานตามคำร้องขออื่นๆ ที่มีต่อไป

2.1.4 การส่งผ่านข้อมูลระหว่างโปรแกรมผู้ใช้และโปรแกรมควบคุมอุปกรณ์แบบบล็อค

โปรแกรมควบคุมอุปกรณ์แบบบล็อคและโปรแกรมควบคุมอุปกรณ์แบบคาแรคเตอร์ที่พัฒนาขึ้นตามมาตรฐานของโปรแกรมควบคุมอุปกรณ์ในระบบปฏิบัติการยูนิกซ์แล้ว เคอร์เนลจะทำหน้าที่ในการส่งผ่านข้อมูลพารามิเตอร์ของฟังก์ชั่นที่ผู้ใช้เรียกไปยังโปรแกรมควบคุมอุปกรณ์ และทำหน้าที่ส่งผ่านข้อมูลและสถานะการทำงานจากโปรแกรมควบคุมอุปกรณ์กลับมายังโปรแกรมผู้ใช้เอง โดย อาศัยขั้นตอนและองค์ประกอบต่างๆ จะขออธิบายรายละเอียดเฉพาะสำหรับโปรแกรมควบคุมอุปกรณ์แบบบล็อคที่ใช้ควบคุมอุปกรณ์ซีดีรอมเท่านั้น สำหรับโปรแกรมควบคุมอุปกรณ์แบบบล็อคของอุปกรณ์อื่นจะมีการทำงานที่ใกล้เคียงกัน

1. องค์ประกอบของการส่งผ่านข้อมูล คือ

1.1 ยูเอเรีย (U area)

เมื่อเคอร์เนลสร้างโปรเซสขึ้นมาจะสร้างเนื้อที่ข้อมูลส่วนหนึ่ง เรียกว่ายูเอเรียและรายการข้อมูลในตารางโปรเซส (process table) ที่สัมพันธ์กับโปรเซสขึ้นด้วย โดยทั้งยูเอเรียและรายการข้อมูลในตารางโปรเซสจะเก็บข้อมูลที่ใช้ควบคุมโปรเซส และข้อมูลสถานะของโปรเซสไว้

รายการข้อมูลในตารางโปรเซส จะเก็บข้อมูลเกี่ยวกับสถานะของโปรเซส เช่น สภาวะของโปรเซส (process state) ข้อมูลตำแหน่งของโปรเซสในหน่วยความจำหลักหรือตำแหน่งของโปรเซสและยูเอเรียในหน่วยความจำสำรอง (ขณะที่โปรเซสถูกเคอร์เนลเขียนลงในอุปกรณ์เก็บความจำสำรอง เนื่องจากต้องการใช้หน่วยความจำหลักในการทำงานอย่างอื่น) ข้อมูลเกี่ยวกับเจ้าของโปรเซส เหตุการณ์ (event) ที่โปรเซสรอให้เกิดขึ้น (ขณะที่โปรเซสกำลังพักการทำงาน (sleep)) สัญญาณ (signal) ที่โปรเซสได้รับแต่ยังไม่ได้ดำเนินการกับสัญญาณนั้น เป็นต้น รายการข้อมูลในตารางโปรเซสจะคงอยู่ในหน่วยความจำตลอดเวลา เพื่อให้เคอร์เนลสามารถมาค้นหา เปลี่ยนแปลงข้อมูลของโปรเซสได้ตลอดเวลาไม่ว่าโปรเซสจะกำลังทำงานอยู่หรือไม่

ยูเอเรียจะเก็บข้อมูลที่โปรเซสใช้ในขณะที่โปรเซสถูกเรียกขึ้นมาทำงาน เช่น ตารางแฟ้มข้อมูลของผู้ใช้ (user file descriptor) กองซ้อน (stack) สำหรับการเรียกฟังก์ชั่นของระบบ เป็นต้น ยูเอเรียจะถูกอ่านขึ้นมาในหน่วยความจำหรือเก็บลงสู่หน่วยความจำสำรองพร้อมกับโปรเซสเสมอ เมื่อโปรเซสทำงานจะมีการเปลี่ยนแปลงค่าข้อมูลต่าง ๆ ใน ยูเอเรียที่สัมพันธ์กับโปรเซสตลอดเวลา ขณะที่โปรเซสเรียกฟังก์ชั่นของระบบ (system call) ภาวะการทำงานของโปรเซสจะเปลี่ยนจากภาวะของผู้ใช้ (user mode) ไปเป็นภาวะของเคอร์เนล (kernel mode) ในขณะนั้นเคอร์เนลจะทำการแปลงตำแหน่งหน่วยความจำยูเอเรียของโปรเซสเข้ามาอยู่ในบริเวณหน่วยความจำที่เคอร์เนลอ้างถึงได้ เพื่อที่เคอร์เนลจะสามารถแก้ไขเปลี่ยนแปลงค่าข้อมูลในยูเอเรียของโปรเซสที่เรียกฟังก์ชั่นระบบได้ รวมทั้งสามารถอ่านค่าพารามิเตอร์ในยูเอเรียเพื่อส่งต่อให้กับโปรแกรมควบคุมอุปกรณ์ได้

ยูเอเรียเป็นโครงสร้างข้อมูลที่สำคัญในการทำงานของโปรเซสและในการส่งผ่านข้อมูลระหว่างโปรแกรมผู้ใช้และโปรแกรมควบคุมอุปกรณ์แบบบล็อค

1.2 ตารางแฟ้มข้อมูล (file table)

ตารางแฟ้มข้อมูลจะเก็บข้อมูลเกี่ยวกับ ตำแหน่งข้อมูลสัมพัทธ์ในแฟ้มข้อมูล หรือแฟ้มอุปกรณ์ที่เปิดใช้งาน รวมทั้งตัวชี้ไปยังไอโนดของแฟ้มข้อมูลหรือแฟ้มอุปกรณ์ที่เปิดใช้ เคอร์เนลจะอ่านค่าตำแหน่งข้อมูลสัมพัทธ์ไปเก็บไว้ที่ยูเอเรีย เพื่อส่งต่อให้กับโปรแกรมย่อยควบคุมอุปกรณ์นำไปใช้ในการหาตำแหน่งเริ่มต้นข้อมูลในอุปกรณ์ที่จะใช้ในการติดต่อส่งผ่านข้อมูลกับอุปกรณ์ในแต่ละคำสั่งที่มีการเรียกใช้

1.3 โครงสร้างข้อมูลแบบสถิต (static data structure)

โปรแกรมควบคุมอุปกรณ์ จะมีการสร้างโครงสร้างข้อมูลแบบสถิตซึ่งจะคงอยู่ในหน่วยความจำของเครื่องตลอดเวลา เพื่อใช้ในการส่งผ่านข้อมูลหรือสถานะการทำงานระหว่างโปรแกรมควบคุมอุปกรณ์และโปรแกรมผู้ใช้ โดยผ่านการทำงานของเคอร์เนล โครงสร้างข้อมูลแบบสถิตจะมีความสำคัญมากในการส่งผ่านสถานะการทำงานจาก โปรแกรมตอบรับการขัดจังหวะไปยังโปรแกรมผู้ใช้ เนื่องจากขณะที่มีสัญญาณขัดจังหวะจากอุปกรณ์ โปรเซสที่กำลังทำงานอยู่ อาจจะไม่ใช่โปรเซสที่ร้องขอการทำงานไปยังอุปกรณ์ ดังนั้นโปรแกรมตอบรับการขัดจังหวะจึงไม่สามารถใส่ข้อมูลหรือสถานะการทำงานของอุปกรณ์ในยูเอเรียขณะนั้นได้ จึงต้องเลี่ยงมาใส่ค่าข้อมูล และสถานะการทำงานที่โครงสร้างข้อมูลแบบสถิตชุดที่สัมพันธ์กับโปรแกรมควบคุมอุปกรณ์นั้น เมื่อโปรแกรมตอบรับสัญญาณขัดจังหวะทำงานเสร็จ จะทำการถ่ายทอดข้อมูลและสถานะการทำงานไปยังบัฟเฟอร์คำสั่ง ปลุกโปรเซส ที่เรียกใช้ฟังก์ชั่นติดต่ออุปกรณ์ครั้งนั้นขึ้นมาทำงานอีกครั้งหนึ่ง และทำการส่งบัฟเฟอร์คืนแก่เคอร์เนล เคอร์เนลจะถ่ายทอดข้อมูลในบัฟเฟอร์ไปยังยูเอเรียของโปรเซส ซึ่งจะถูกส่งต่อไปยังโปรเซสต่อไป

ทุกๆโปรแกรมควบคุมอุปกรณ์จะมีโครงสร้างข้อมูลแบบสถิตของแต่ละโปรแกรมเอง

2. ขั้นตอนการส่งผ่านข้อมูลของแต่ละฟังก์ชั่น

2.1 ฟังก์ชั่นเปิดการติดต่อกับอุปกรณ์ (open function)

รูปแบบการเรียกใช้ฟังก์ชั่น

fd = open(pathname,flags,mode);

fd คือ ตัวแปรที่มารอรับค่าส่งกลับจากฟังก์ชั่นซึ่งจะเป็นค่าตำแหน่งของรายการในตารางแฟ้มข้อมูลของโปรเซสผู้ใช้

pathname คือ ชื่อของแฟ้มอุปกรณ์

flags คือ ข้อมูลระบุลักษณะการเปิดแฟ้มข้อมูลเช่น เปิดเพื่ออ่าน เปิดเพื่อเขียน เป็นต้น

mode คือ ข้อมูลลักษณะการอนุญาติให้เข้าใช้แฟ้ม (เฉพาะในตอนสร้างแฟ้มข้อมูล)

รูปแบบของโปรแกรมย่อยควบคุมอุปกรณ์ที่ทำหน้าที่เปิดการติดต่อกับอุปกรณ์

xxopen(dev,flags,mode)

dev คือ เลขหมายอุปกรณ์ (ทั้งหลักและรอง)

flags คือ ข้อมูลระบุประเภทของการเปิดแฟ้ม

mode คือ ข้อมูลระบุการเปิดแฟ้มข้อมูล เช่น เปิดเพื่ออ่าน เปิดเพื่อเขียน เป็นต้น

การทำงานของเคอร์เนล

เคอร์เนลจะนำชื่อแฟ้มอุปกรณ์ไปค้นหาในระบบแฟ้มข้อมูล(file system) โดยค้นไปตามสารบบ (directory) เพื่อหาไอโนดของแฟ้มอุปกรณ์นี้ หลังจากพบไอโนด เคอร์เนลจะสร้างรายการในตารางแฟ้มข้อมูล เพื่อชี้ไปหาไอโนด พร้อมทั้งกำหนดค่า ตำแหน่งข้อมูลสัมพัทธ์เริ่มต้นเป็น 0 เคอร์เนลจะตรวจสอบว่าไอโนดเป็นไอโนดของอุปกรณ์แบบ บล็อค หรือแบบคาแรคเตอร์ เพื่อที่จะได้เลือกตารางสวิทช์สำหรับอุปกรณ์ที่ถูกต้อง หลังจากนั้นจะนำค่าเลขหมายหลัก (major number) และเลขหมายรอง (minor number)ของอุปกรณ์จากไอโนดไปค้นหาตำแหน่งของโปรแกรมเปิดการติดต่อกับอุปกรณ์ที่ต้องการ ในตารางสวิทช์สำหรับอุปกรณ์ หลังจากนั้นเรียกให้โปรแกรมทำงานโดยส่งพารามิเตอร์เลขหมายอุปกรณ์ (ทั้งหลักและรอง) และข้อมูลระบุประเภทของการเปิดแฟ้ม (open type) เช่น เปิดแฟ้ม เพื่อเป็นอุปกรณ์แบบ บล็อค เปิดแฟ้มเพื่อเป็นอุปกรณ์แบบคาแรคเตอร์ เปิดแฟ้มเพื่อเป็นอุปกรณ์แบบใส่ (mount) เป็นต้น

(ข้อมูลนี้ได้จากไอโนดและคำสั่งที่ใช้) ข้อมูลระบุการเข้าใช้แฟ้มอุปกรณ์ เช่น เปิดเพื่อเขียนข้อมูล เปิดเพื่ออ่านข้อมูล เป็นต้น ให้แก่โปรแกรมเปิดการติดต่อกับอุปกรณ์ เมื่อโปรแกรมทำงานสมบูรณ์โดยไม่มีความผิดพลาด เคอร์เนลจะส่งคืนค่าการทำงานของฟังก์ชั่นเป็นค่าตำแหน่งของรายการในตารางแฟ้มข้อมูลของโปรเซสผู้ใช้

2.2 ฟังก์ชั่นในการอ่านข้อมูล (read function)

รูปแบบการใช้งานฟังก์ชั่น

number = read(fd,buffer,count);

fd คือ ตำแหน่งที่ในตารางแฟ้มข้อมูลของโปรเซสผู้ใช้ที่บรรจุข้อมูลเกี่ยวกับอุปกรณ์นี้

buffer คือ ตำแหน่งหน่วยความจำที่รอรับข้อมูลจากการอ่าน

count คือ จำนวนข้อมูล(หน่วยเป็นไบต์)ที่ต้องการจะได้รับจากการอ่านครั้งนี้

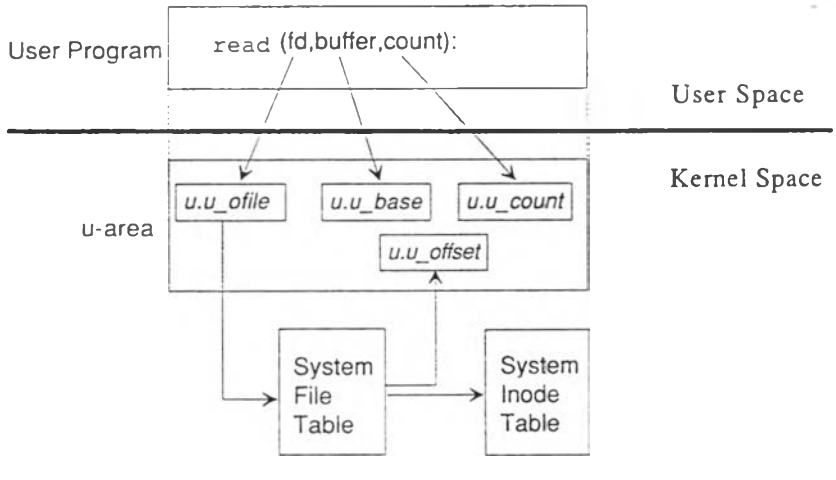
รูปแบบของโปรแกรมย่อยควบคุมอุปกรณ์ที่ทำหน้าที่อ่านข้อมูล

xxstrategy(bp)

bp คือ ตัวชี้ไปยังโครงสร้างข้อมูลบัฟเฟอร์ที่บรรจุคำสั่งและตำแหน่งของข้อมูลในอุปกรณ์ที่ต้องการอ่าน

การทำงานของเคอร์เนล

เคอร์เนลจะนำค่า fd สืบค้นเข้าไปหาตารางแฟ้มข้อมูลของโปรเซสผู้ใช้เพื่อหาตำแหน่งที่ใช้ในตารางแฟ้มข้อมูล เพื่อนำข้อมูลมาใส่ยูเอเรียดังนี้



ภาพที่ 2.5 แสดงการส่งผ่านข้อมูลระหว่างโปรเซสผู้ใช้กับโปรแกรมควบคุมอุปกรณ์ผ่านยูเอเรีย

1. นำค่า fd ใส่ในฟิลด์ u.u_ofile
2. นำค่าตำแหน่งหน่วยความจำของ buffer ใส่ใน u.u_base
3. นำค่าจำนวนข้อมูลที่ต้องการอ่านใส่ใน u.u_count
4. นำค่าตำแหน่งข้อมูลสัมพัทธ์ของอุปกรณ์(จากตารางแฟ้มข้อมูล) ใส่ใน u.u_offset

หลังจากกำหนดค่าต่าง ๆ เรียบร้อย เคอร์เนลจะเริ่มการทำงานเป็นวง (loop) จนกว่าจะสามารถจะส่งผ่านข้อมูลตามที่ระบุไว้ในฟังก์ชั่น read จนครบ หรือ จนกว่าจะสิ้นสุดข้อมูลที่มีอยู่ในอุปกรณ์ โดยเริ่มต้นจากการแปลงตำแหน่งข้อมูลสัมพัทธ์ ของแฟ้มอุปกรณ์ที่ปรากฏใน ฟิลด์ u.u_offset ไปเป็น หมายเลขของบล็อคข้อมูลในอุปกรณ์ และตำแหน่งข้อมูลในบล็อคข้อมูลนั้น เคอร์เนลจะตรวจสอบบัฟเฟอร์ในระบบ ว่ามีบัฟเฟอร์ใดที่บรรจุข้อมูลของบล็อคที่ต้องการดังกล่าว ถ้ามีจะนำบัฟเฟอร์นั้นมาใช้ ถ้าไม่มีบัฟเฟอร์ที่บรรจุข้อมูลของบล็อคที่ต้องการ เคอร์เนลจะขอบัฟเฟอร์ว่าง และ กำหนดคำสั่ง และค่าต่างๆ ในบัฟเฟอร์พร้อมกับเรียกโปรแกรมย่อยสำหรับอ่านข้อมูลของโปรแกรมควบคุมอุปกรณ์ในการทำงาน พร้อมกับส่งค่าพารามิเตอร์ เป็นตัวชี้บัฟเฟอร์ที่ต้องการข้อมูลไป หลังจากนั้นเมื่ออุปกรณ์อ่านข้อมูลที่ต้องการได้แล้ว เคอร์เนลจะทำการคัดลอกข้อมูลส่งไปเก็บยังตำแหน่งหน่วยความจำของบัฟเฟอร์ของโปรเซสผู้ใช้ และทำการปรับค่าข้อมูลต่างๆ ดังนี้

1. ทำการปรับค่าตำแหน่งข้อมูลสัมพัทธ์ ที่ปรากฏในยูเอเรีย คือ (u.u_offset)
2. เพิ่มตำแหน่งหน่วยความจำของบัฟเฟอร์ของโปรเซสผู้ใช้ตาม จำนวนข้อมูลที่ได้อ่านเข้ามา (u.u_base)
3. ลดค่าจำนวนนับของข้อมูลที่จะต้องอ่าน (u.u_count)

เป็นการจบการทำงานในรอบแรก เคอร์เนลจะตรวจสอบดูว่า จะต้องอ่านข้อมูลต่อไปหรือไม่ โดยพิจารณาที่จำนวนข้อมูลที่จะต้องอ่านในฟิลด์ (u.u_count) และทำการอ่านข้อมูลตามขั้นตอนเดิมต่อไปจนกว่าค่าข้อมูลในฟิลด์ u.u_count เป็น 0 หรือ จนกว่าจะไม่มีข้อมูลเหลือในอุปกรณ์ให้อ่านหรือจนกว่าเกิดความผิดพลาดในการอ่านข้อมูล หรือในการโอนถ่ายข้อมูลจากบัฟเฟอร์ ไปยังเนื้อที่หน่วยความจำของผู้ใช้ (user address space)

ในกรณีที่ข้อมูลที่ต้องการอ่านน้อยกว่าขนาดของบล็อคข้อมูลที่เคอร์เนลอ่านขึ้นมาในแต่ละครั้ง (ขนาดของบัฟเฟอร์) เคอร์เนลจะคัดลอกข้อมูลส่งต่อไปให้แก่โปรแกรมผู้ใช้เพียงเท่าที่มีการขอมา

หลังจากเคอร์เนลทำงานตามคำสั่ง อ่านข้อมูลที่ขอมาเรียบร้อย เคอร์เนลจะปรับค่าตำแหน่งข้อมูลสัมพัทธ์ของอุปกรณ์นี้ ที่ปรากฏในตารางแฟ้มข้อมูลให้เป็นค่าใหม่ที่ถูกต้องตามการ อ่านข้อมูลที่เกิดขึ้นจริง และส่งค่าจำนวนข้อมูลที่อ่านได้ กลับคืนไปให้แก่โปรแกรมผู้ใช้

เป็นค่าส่งกลับของฟังก์ชั่นอ่านข้อมูล โดยใส่ค่าความผิดพลาดที่ฟิลด์ u.u_error เป็น 0.

2.3 ฟังก์ชั่นในการปิดการติดต่ออุปกรณ์

รูปแบบการเรียกใช้ฟังก์ชั่น

close(fd);

fd คือ ตำแหน่งที่ในตารางแฟ้มข้อมูลของโปรเซสผู้ใช้

รูปแบบของโปรแกรมย่อยควบคุมอุปกรณ์ที่ทำหน้าที่ปิดการติดต่อกับอุปกรณ์

xxclose(dev,flag)

dev คือ เลขหมายอุปกรณ์ (ทั้งหลักและรอง)

flag คือ ข้อมูลระบุลักษณะการเปิดแฟ้มข้อมูล

การทำงานของเคอร์เนล

เคอร์เนลจะไปลดค่าตัวนับของรายการแฟ้มอุปกรณ์ ในตารางแฟ้มข้อมูล และ ที่ไอโนดหน่วยความจำของแฟ้มอุปกรณ์นี้ ถ้าหลังจากการลดค่าแล้วค่าตัวนับที่ ไอโนดหน่วยความจำมีค่าเป็น 0 แสดงว่าไม่มีโปรเซสใดเปิดใช้งานแฟ้มอุปกรณ์นี้อยู่ เคอร์เนลจะนำค่าเลขหมายหลัก ไปค้นหาตำแหน่งของโปรแกรมปิดการติดต่อกับอุปกรณ์ในตารางสวิทช์ของอุปกรณ์แบบบล็อค และเรียกโปรแกรมให้ทำงานโดยส่งค่าพารามิเตอร์ เป็นเลขหมายอุปกรณ์ และข้อมูลลักษณะการเปิดแฟ้มไปให้ ถ้าหลังการลดค่าแล้วค่าตัวนับที่ไอโนดหน่วยความจำไม่เป็น 0 จะไม่มีการเรียกใช้โปรแกรมย่อยควบคุมอุปกรณ์ในการปิดแฟ้มอุปกรณ์

เคอร์เนลจะทำการปิดแฟ้มข้อมูล และแฟ้มอุปกรณ์ ทุกแฟ้มที่โปรเซสเปิดใช้งานอยู่ทันทีที่โปรเซสจบการทำงาน

2.4 การส่งสถานะการทำงานของอุปกรณ์กลับสู่โปรแกรมผู้ใช้

รูปแบบของโปรแกรมย่อยควบคุมอุปกรณ์ที่ทำหน้าที่ตอบรับสัญญาณขัดจังหวะ

xxintr(vec_num)

vec_num คือ ตัวเลขที่แสดงหมายเลขของเวคเตอร์การขัดจังหวะ (interrupt vector number) เมื่ออุปกรณ์ทำงานเสร็จหรือเมื่ออุปกรณ์มีปัญหาในการทำงาน จะส่งสัญญาณขัดจังหวะเข้ามายังหน่วยประมวลผล ซึ่งระบบปฏิบัติการยูนิกซ์จะรับทราบสัญญาณขัดจังหวะ และค้นหาในตารางเวคเตอร์ของการขัดจังหวะ (interrupt vector table) เพื่อหาโปรแกรมที่จะต้องส่งการทำงานไปให้ หลังจากนั้นเคอร์เนล จะเรียกโปรแกรมดังกล่าวมาทำงาน พร้อมกับส่งพารามิเตอร์เป็นเลขหมายอุปกรณ์ไปให้

โปรแกรมตอบรับสัญญาณขัดจังหวะ จะตรวจสอบการทำงานของอุปกรณ์ว่าสมบูรณ์ หรือ มีข้อผิดพลาดใด ถ้ามีข้อผิดพลาดใดจะใส่ข้อผิดพลาดนั้นไว้ในโครงสร้างข้อมูลแบบสถิต (static data structure) ของอุปกรณ์นั้น

017195

2.2 ข้อกำหนดระบบการเชื่อมต่ออุปกรณ์แบบ SCSI (Small Computer System Interface)

ระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ (SCSI - Small Computer System Interface) ได้รับการพัฒนาขึ้นในปี ค.ศ. 1979 โดยบริษัท Shugart Associates เพื่อใช้ควบคุม Winchester disk drives ขนาด 8 นิ้ว ในชื่อว่า SASI (Shugart Associates System Interface) (Aitken, 1989) ต่อมาทางบริษัทได้พัฒนาให้สามารถใช้กับอุปกรณ์ต่างต่อกับเครื่องไมโครคอมพิวเตอร์อื่นๆด้วย เช่น Tape drives , printers , coprocessors เป็นต้น จากคุณลักษณะที่ดีของระบบเชื่อมต่อแบบนี้ จึงทำให้มีการนำไปใช้กันอย่างกว้างขวาง และคณะกรรมการกำหนดมาตรฐาน ANSI(American National Standard for Information Systems) ได้นำต้นแบบของระบบการเชื่อมต่ออุปกรณ์นี้ไปพิจารณา และเรียบเรียงเป็นข้อกำหนดระบบการเชื่อมต่ออุปกรณ์มาตรฐาน หมายเลข ANSI X3.131-1986 ในชื่อว่า Small Computer System Interface (SCSI) ซึ่งในปัจจุบันได้มีการนำไปใช้กันอย่างแพร่หลายในบริษัทผู้ผลิตเครื่องคอมพิวเตอร์ และอุปกรณ์ชั้นนำต่างๆเช่น Sun, Apple, NeXT เป็นต้น

ข้อกำหนดมาตรฐานของระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ ประกอบไปด้วยข้อกำหนดลักษณะทางกายภาพ ลักษณะทางสัญญาณไฟฟ้า และชุดคำสั่งที่ใช้ในสายสัญญาณที่อุปกรณ์ทุกชนิดในระบบจะใช้ในการติดต่อสื่อสารกัน ข้อกำหนดนี้กำหนดในลักษณะพื้นฐาน ขณะเดียวกันก็เปิดกว้างให้บริษัทผู้ผลิตแต่ละรายสามารถกำหนดรายละเอียดเพิ่มเติมในลักษณะต่าง ๆ เช่น

- รูปแบบและวิธีการติดต่อระหว่างโปรแกรมของเครื่องคอมพิวเตอร์ และสกาสซี่โฮสอแดปเตอร์ (SCSI host adapter)
- การทำงานของแผงควบคุมอุปกรณ์ (controller) เมื่อได้รับคำสั่งสกาสซี่
- ระบบการเชื่อมต่อของปลายทั้งสองข้างของสายสัญญาณสกาสซี่
- คำสั่งและข้อความพิเศษเพิ่มเติมที่บริษัทสามารถกำหนดขึ้นได้เอง (Vendor added commands and messages)

2.2.1 ลักษณะสำคัญของการเชื่อมต่ออุปกรณ์แบบสกาสซี่

1. ในสายสัญญาณสกาสซี่ 1 ระบบ สามารถเชื่อมต่ออุปกรณ์เข้าด้วยกันได้ทั้งหมด 8 อุปกรณ์ในลักษณะเชื่อมเป็นวงกลม (daisy chain) โดยอุปกรณ์อาจเป็นเครื่องคอมพิวเตอร์ อุปกรณ์ต่างต่อกับเครื่องคอมพิวเตอร์ หรือ แผงควบคุมอุปกรณ์ที่จะต่างต่อกับอุปกรณ์ชนิดเดียวกันต่อไปได้อีกไม่เกิน 8 อุปกรณ์ แต่ละอุปกรณ์ที่ต่างต่อกันในสายสัญญาณสกาสซี่จะมี

หมายเลขอุปกรณ์(ID) ประจำ ซึ่งหมายเลขอุปกรณ์มีตั้งแต่ 0 - 7 สำหรับในกรณีแผงควบคุมอุปกรณ์ที่ต่ออุปกรณ์ย่อยออกไปอีก 8 อุปกรณ์ อุปกรณ์ย่อยแต่ละอุปกรณ์จะมีหมายเลขอุปกรณ์เดียวกันเป็นหมายเลขอุปกรณ์ของแผงควบคุมอุปกรณ์ แต่มีหมายเลขโลจิคัลยูนิต (LUN) ต่างกันตั้งแต่ 0 - 7

2. อุปกรณ์ที่ต่อในสายสัญญาณจะจัดแบ่งออกเป็น 3 ประเภท คือ

2.1 อุปกรณ์ที่เป็นตัวเริ่มต้นการสื่อสารข้อมูล (Initiator) หมายถึง อุปกรณ์ที่เริ่มต้นส่งคำสั่งเพื่อร้องขอให้อุปกรณ์สกาสซีอื่นในสายสัญญาณทำงานตามต้องการ เช่น เครื่องคอมพิวเตอร์ เป็นต้น

2.2 อุปกรณ์ที่เป็นอุปกรณ์เป้าหมาย (Target) หมายถึง อุปกรณ์ที่ทำงานตามที่ได้รับการร้องขอของอุปกรณ์สกาสซีอื่นในสายสัญญาณ เช่น เครื่องพิมพ์ เครื่องบันทึกข้อมูลจานแม่เหล็กชนิดแข็ง (Hard Disk) เป็นต้น

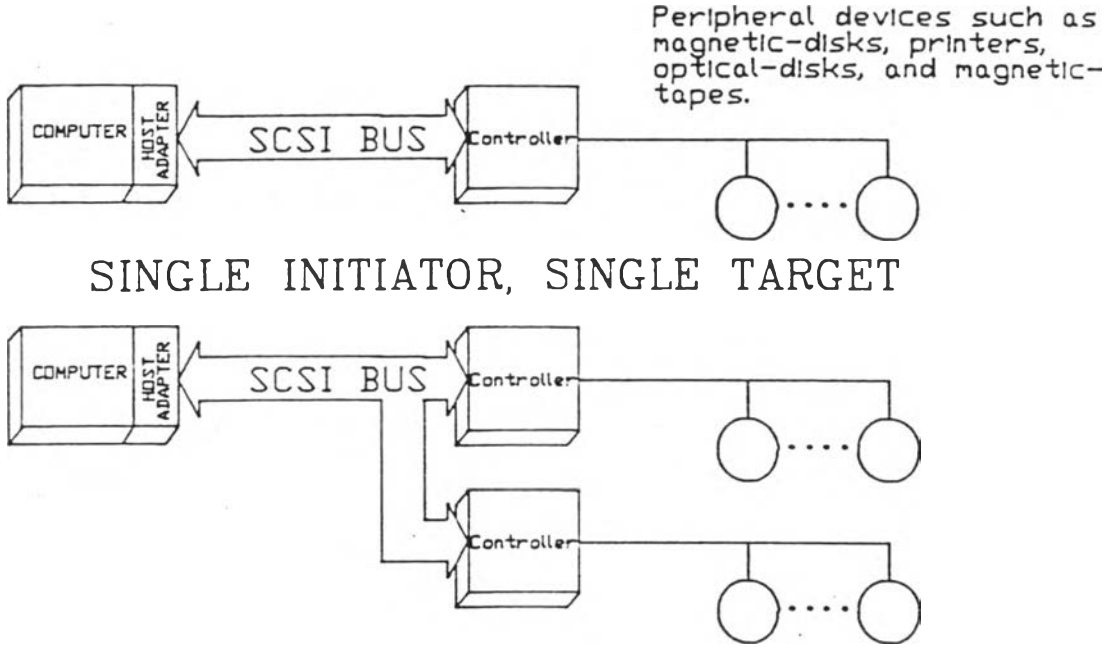
2.3 อุปกรณ์ที่เป็นได้ทั้งอุปกรณ์เริ่มต้นในการสื่อสารข้อมูล และอุปกรณ์เป้าหมายเช่น หน่วยประมวลผลร่วม (Coprocessor) ซึ่งได้รับการร้องขอให้ทำงานจากหน่วยประมวลผลกลาง และร้องขอการทำงานไปยังอุปกรณ์ต่างๆ เป็นต้น

ในระบบการเชื่อมต่ออุปกรณ์แบบสกาสซีหนึ่งระบบจะประกอบด้วย อุปกรณ์ประเภทใดจำนวนเท่าไรก็ได้ แต่รวมแล้วไม่เกิน 8 อุปกรณ์ดังนั้นถ้าในระบบสายสัญญาณประกอบด้วยคอมพิวเตอร์ 1 เครื่อง และแผงควบคุมอุปกรณ์ที่ต่อเชื่อมกับอุปกรณ์ 8 อุปกรณ์ อยู่ 7 แผงควบคุม ก็จะทำให้สามารถเชื่อมต่ออุปกรณ์ได้ถึง 56 อุปกรณ์พร้อมกัน (ภาพที่ 2.6)

3. การเชื่อมต่อของอุปกรณ์กับสายสัญญาณสกาสซี่

การเชื่อมต่อของเครื่องคอมพิวเตอร์กับสายสัญญาณสกาสซี่จะทำโดย ผ่านแผงวงจรสกาสซี่โฮสอแดปเตอร์ (SCSI host adapter) โดยสกาสซี่โฮสอแดปเตอร์จะแปลคำสั่งที่เฉพาะเจาะจงของโปรแกรมคอมพิวเตอร์ที่สั่งไปยังอุปกรณ์ต่าง ๆ ให้กลายเป็นคำสั่งของสกาสซี่ส่งไปตามสายสัญญาณถึงอุปกรณ์เป้าหมาย

การเชื่อมต่อของอุปกรณ์อื่นๆกับสายสัญญาณสกาสซี่นั้น อุปกรณ์สกาสซี่จะต้องมีแผงควบคุมอุปกรณ์ (controller) ที่สามารถรับคำสั่งสกาสซี่ เมื่อแผงควบคุมอุปกรณ์ได้รับคำสั่งสกาสซี่จะแปลคำสั่งสกาสซี่ไปเป็นคำสั่งที่เฉพาะเจาะจงกับอุปกรณ์และสั่งงานให้อุปกรณ์ทำงานต่อไป



ภาพที่ 2.6 แสดงการเชื่อมต่ออุปกรณ์เริ่มต้นการสื่อสารข้อมูลและอุปกรณ์เป้าหมายในระบบสายสัญญาณสกาสซี่

4. การขอเข้าใช้สายสัญญาณสกาสซี่

เนื่องจากในสายสัญญาณ 1 ระบบ สามารถจะต่อพ่วงกับอุปกรณ์ต่าง ๆ ได้ถึง 8 อุปกรณ์ ดังนั้นจึงต้องมีวิธีการควบคุมการขอใช้สายสัญญาณของอุปกรณ์ต่าง ๆ โดยจะมีข้อกำหนดดังนี้

4.1 ในเวลาใดเวลาหนึ่งสายสัญญาณสกาสซี่จะถูกใช้โดยอุปกรณ์เพียงคู่เดียวเสมอ

4.2 ถ้าอุปกรณ์ตั้งแต่ 2 อุปกรณ์ขึ้นไปขอเข้าใช้สายสัญญาณพร้อมกัน อุปกรณ์ที่มีระดับความสำคัญ (Priority) สูงกว่า จะเป็นผู้ได้ใช้สายสัญญาณก่อน โดยระดับความสำคัญจะพิจารณาจากหมายเลขประจำอุปกรณ์ (ID) หมายเลขอุปกรณ์มากจะมีระดับความสำคัญสูงกว่าหมายเลขอุปกรณ์น้อย

โดยเฉลี่ยแล้วทุกๆอุปกรณ์สามารถขอเข้าใช้สายสัญญาณได้ในเวลาน้อยกว่า 10^{-6} วินาที

5. การส่งผ่านข่าวสารในสายสัญญาณสกาสซี่

ข่าวสารที่ส่งถึงกันในสายสัญญาณสกาสซี่ แบ่งออกเป็น 4 ประเภท คือ

5.1 COMMAND หมายถึง คำสั่งที่อุปกรณ์ที่เป็นผู้เริ่มต้นการสื่อสารข้อมูลสั่งให้อุปกรณ์เป้าหมายทำงาน

5.2 DATA หมายถึง ข้อมูลที่มีการเคลื่อนย้ายระหว่างอุปกรณ์เป้าหมายและ อุปกรณ์ที่เป็นผู้เริ่มต้นการสื่อสารข้อมูล ซึ่งเป็นผลมาจากคำสั่งสกาสซี่

5.3 STATUS หมายถึง สถานะการทำงานของคำสั่งที่อุปกรณ์เป้าหมายได้ทำไป โดยสถานะนี้จะถูกส่งกลับไปยังอุปกรณ์ที่เป็นผู้เริ่มต้นการสื่อสารข้อมูล ทุกครั้งที่อุปกรณ์เป้าหมายทำงานแต่ละคำสั่งเสร็จสิ้น

5.4 MESSAGE หมายถึง ข้อความที่อุปกรณ์ที่สื่อสารกันส่งถึงกันเพื่อวัตถุประสงค์ ต่าง ๆ เช่น เพื่อบอกขอพักการสื่อสารข้อมูลชั่วคราว เพื่อระบุหมายเลขประจำอุปกรณ์ที่ใช้สื่อสารข้อมูลกัน เพื่อบอกรูปแบบการติดต่อสื่อสารกันว่าเป็นแบบสมวาร (synchronous) หรือ อสมวาร (asynchronous) เป็นต้น

การสื่อสารข้อมูลในสายสัญญาณสกาสซี่สามารถสื่อสารได้ทั้ง 2 พิธีการ(protocol)คือ

1. พิธีการสื่อสารข้อมูลแบบอสมวาร(Asynchronous transfer protocol) เป็นพิธีการพื้นฐานในการสื่อสารข้อมูลในสายสัญญาณสกาสซี่ อุปกรณ์สกาสซี่ทุกอุปกรณ์จำเป็นต้องสามารถสื่อสารด้วยพิธีการนี้ ความเร็วในการสื่อสารข้อมูลโดยเฉลี่ย 1-2 เมกะไบต์ต่อวินาที

2. พิธีการสื่อสารข้อมูลแบบสมวาร (Synchronous transfer protocol)

พิธีการนี้ใช้ในการสื่อสารเฉพาะข้อมูล (DATA) เท่านั้น และอุปกรณ์สกาสซี่จะสามารถสื่อสารด้วยพิธีการนี้หรือไม่ก็ได้ ก่อนการสื่อสารข้อมูลอุปกรณ์ทั้งคู่ต้องตรวจสอบความสามารถ ของการสื่อสารโดยพิธีการนี้กันก่อน โดยใช้ข้อความ (message) ความเร็วในการสื่อสารข้อมูลโดยเฉลี่ย 4 เมกะไบต์ต่อวินาที (MB/s)

การสื่อสารข้อมูลทั้ง 2 พิธีการ ควบคุมข้อมูลโดยใช้ลักษณะการโต้ตอบ (REQ/ACK handshake) จะมีการโต้ตอบทุก ๆ ครั้งที่มีการสื่อสารข้อมูล 1 ไบต์

6. คำสั่งที่ใช้ในการติดต่อสื่อสารระหว่างอุปกรณ์ในสายสัญญาณสกาสซี่

คำสั่งที่ใช้ในสายสัญญาณสกาสซี่ได้แบ่งออกเป็น 4 กลุ่ม ตามความจำเป็นในการใช้งาน ดังนี้

6.1 คำสั่งกลุ่มที่จำเป็นต้องมี (Mandatory command) ทุกๆ อุปกรณ์ที่มีระบบเชื่อมต่อแบบสกาสซี่ต้องสามารถรับและทำงานตามคำสั่งชุดนี้ครบถ้วน คำสั่งในชุดนี้สามารถอ้างอิงถึงข้อมูลได้น้อยกว่า คือ อ้างถึงข้อมูลได้ 2^{21} บล็อค กำหนดขนาดข้อมูลที่เคลื่อนย้ายได้ 256 บล็อคต่อ 1 คำสั่ง

6.2 คำสั่งกลุ่มเพิ่มเติม (Extended command) เป็นกลุ่มคำสั่งที่มีเพิ่มเติมขึ้น ถ้าอุปกรณ์ใดสามารถรับและทำงานตามชุดคำสั่งนี้จะทำให้ผู้เขียนโปรแกรมควบคุมอุปกรณ์มีความสะดวกในการเขียนโปรแกรมมากขึ้น เช่น มีคำสั่งในการสอบถามลักษณะจำเพาะของอุปกรณ์ (Device-necessary attributes) ซึ่งทำให้สามารถเขียนโปรแกรมควบคุมอุปกรณ์ที่เรียนรู้ลักษณะของอุปกรณ์ได้เลย (Self-configuring software drivers) คำสั่งในชุดนี้ สามารถอ้างอิงข้อมูลได้ 2^{32} บล็อค และกำหนดขนาดการเคลื่อนย้ายข้อมูลได้ 65,535 บล็อค

6.3 คำสั่งกลุ่มเพิ่มเติมที่เป็นทางเลือก (Optional command) อุปกรณ์สกาสซี่จะมีหรือไม่มีคำสั่งใดในกลุ่มคำสั่งนี้ก็ได้

6.4 คำสั่งเพิ่มเติมของบริษัทผู้ผลิต (Vendor unique command)ผู้ผลิตอุปกรณ์สกาสซี่สามารถจะเพิ่มเติมคำสั่งที่เฉพาะเจาะจงของตนเอง เข้าไปในอุปกรณ์ได้ตามขอบเขตที่มาตรฐานเปิดไว้

จากคำสั่งทั้ง 4 กลุ่ม ข้อกำหนดมาตรฐานได้นำคำสั่งมาแยกตามประเภทของอุปกรณ์สกาสซี่ออกได้เป็นคำสั่งทั่วไปสำหรับทุกอุปกรณ์ และคำสั่งสำหรับอุปกรณ์ประเภทต่าง ๆ 6 ประเภทคือ

1. คำสั่งสำหรับอุปกรณ์ที่มีการเข้าถึงโดยตรง (Direct Access Devices) เช่น เครื่องบันทึกข้อมูลแบบจานแม่เหล็กชนิดแข็ง (Hard Disk) เป็นต้น

2. คำสั่งสำหรับอุปกรณ์ที่มีการเข้าถึงข้อมูลแบบตามลำดับ (Sequential

Access Device) เช่น เครื่องบันทึกข้อมูลเทป เป็นต้น

3. คำสั่งสำหรับอุปกรณ์ที่เป็นเครื่องพิมพ์ (Printer Device)

4. คำสั่งสำหรับอุปกรณ์ที่ทำหน้าที่ประมวลผล (Processor Devices)

5. คำสั่งสำหรับอุปกรณ์เก็บข้อมูลที่เขียนข้อมูลเข้าได้ครั้งเดียว แต่อ่านได้มากครั้ง (Write Once Read Multiple Devices) เช่น WORM (Write Once Read Many) เป็นต้น

6. คำสั่งสำหรับอุปกรณ์ที่อ่านได้อย่างเดียว เช่น CD-ROM (Compact Disk Read Only Memory) เป็นต้น

จะกล่าวถึงรายละเอียดของการทำงานแต่ละคำสั่งในขั้นตอนต่อไป

7. ขั้นตอนของสายสัญญาณสกาสซี่

การส่งสัญญาณเพื่อสื่อสารข้อมูลระหว่างอุปกรณ์ที่ต่อเชื่อมกันด้วยสายสัญญาณสกาสซี่นั้น ประกอบด้วยขั้นตอนของสัญญาณต่าง ๆ 8 ขั้นตอนด้วยกัน คือ

7.1 ขั้นตอนปลอดการสื่อสาร (BUS FREE phase)

เป็นขั้นตอนที่ไม่มีอุปกรณ์ใด ๆ ใช้สายสัญญาณสกาสซี่เลย

7.2 ขั้นตอนการแข่งขันเพื่อขอใช้สายสัญญาณ (ARBITRATION phase)

เป็นขั้นตอนที่มีอุปกรณ์สกาสซี่มากกว่า 1 อุปกรณ์ต้องการจะขอใช้สายสัญญาณสกาสซี่ ถ้าอุปกรณ์สกาสซี่มากกว่า 1 อุปกรณ์ขอใช้สายสัญญาณในเวลาเดียวกัน อุปกรณ์ที่มีระดับความสำคัญสูงกว่าจะได้รับสิทธิในการใช้สายสัญญาณก่อน

ในระบบที่มีอุปกรณ์ที่เป็นตัวเริ่มต้นในการสื่อสารข้อมูล (initiator) เพียงอุปกรณ์เดียวไม่ต้องมีขั้นตอนนี้ เนื่องจากไม่มีอุปกรณ์ใดมาแข่งขันเพื่อขอใช้สายสัญญาณ

7.3 ขั้นตอนการระบุอุปกรณ์ที่จะติดต่อสื่อสารเพื่อทำการสื่อสาร (SELECTION phase)

เมื่ออุปกรณ์ที่เป็นตัวเริ่มต้นในการสื่อสารข้อมูลได้เข้าใช้สายสัญญาณสกาสซี่ จะส่งสัญญาณระบุหมายเลขอุปกรณ์ที่จะติดต่อสื่อสารด้วย เพื่อเป็นขั้นตอนการเริ่มต้นการติดต่อสื่อสารข้อมูล

7.4 ขั้นตอนการระบุอุปกรณ์ที่จะติดต่อสื่อสาร เพื่อทำการสื่อสารข้อมูลต่อจากที่พักไว้ (SELECTION phase) ขั้นตอนนี้จะมีในระบบที่มีการแข่งขันเพื่อขอใช้สายสัญญาณเพื่อเพิ่มโอกาสในการใช้สายสัญญาณของอุปกรณ์ต่าง ๆ กล่าวคือ เมื่ออุปกรณ์ที่ได้ใช้สายสัญญาณจะยังไม่ใช้สายสัญญาณในการสื่อสารช่วงเวลาหนึ่ง อุปกรณ์นั้นจะปลดปล่อยสายสัญญาณให้ว่าง เพื่อเปิดโอกาสให้อุปกรณ์อื่นเข้าใช้สายสัญญาณก่อน โดยการส่งข้อความ (messages) ไปบอกอุปกรณ์ที่ติดต่อสื่อสารด้วยว่าจะขอพักการติดต่อสื่อสาร หลังจากอุปกรณ์อื่นทำงานเสร็จและสายสัญญาณว่าง

ลงอีกครั้งหนึ่ง อุปกรณ์ที่ขอพักการสื่อสารนี้จะเข้าแข่งขันเพื่อขอใช้สายสัญญาณ เมื่อได้ใช้สายสัญญาณจะส่งข้อความเพื่อระบุอุปกรณ์ที่จะติดต่อสื่อสาร เพื่อทำการติดต่อสื่อสารข้อมูลต่อจากที่พักไว้

7.5 ขั้นตอนการส่งคำสั่ง (COMMAND phase)

เป็นขั้นตอนที่ส่งคำสั่งจากอุปกรณ์เริ่มต้นการสื่อสารข้อมูลไปยังอุปกรณ์เป้าหมาย

7.6 ขั้นตอนการส่งข้อมูล (DATA phase)

เป็นขั้นตอนที่มีการเคลื่อนย้ายข้อมูลระหว่างอุปกรณ์ที่ติดต่อสื่อสารกัน พอจะแยกออกได้เป็น 2 แบบ คือ ขั้นตอนการส่งข้อมูลจากอุปกรณ์เริ่มต้นสื่อสารข้อมูลไปยังอุปกรณ์เป้าหมาย (DATA OUT phase) ขั้นตอนการส่งข้อมูลจากอุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นสื่อสารข้อมูล (DATA IN phase)

7.7 ขั้นตอนการส่งสถานะการทำงาน (STATUS phase)

เมื่ออุปกรณ์เป้าหมายทำงานแต่ละคำสั่งเสร็จสิ้นลงจะส่งสถานะการทำงาน (STATUS) กลับไปยังอุปกรณ์ที่เป็นผู้สั่งคำสั่งนั้นในขั้นตอนนี้

7.8 ขั้นตอนการส่งข้อความระหว่างอุปกรณ์ที่ติดต่อสื่อสารข้อมูล (MESSAGE phase)

เป็นขั้นตอนที่อุปกรณ์ที่ติดต่อสื่อสารกันส่งข้อความถึงกันเพื่อวัตถุประสงค์ต่างๆ เช่น ข้อความที่รายงานสภาวะการทำงาน (COMMAND COMPLETE , ABORT) ข้อความที่ขอหยุดพักการสื่อสารข้อมูล (DISCONNECT) ข้อความที่ระบุหมายเลขอุปกรณ์ที่จะสื่อสารด้วย (IDENTIFY) เป็นต้น แบ่งออกเป็น 2 แบบ คือ การส่งข้อความจากอุปกรณ์เริ่มต้นสื่อสารข้อมูลไปยังอุปกรณ์เป้าหมาย (MESSAGE OUT phase) และ การส่งข้อความจากอุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นสื่อสารข้อมูล (MESSAGE IN phase)

ขั้นตอนที่ 7.5 7.6 7.7 7.8 รวมกันเรียกว่า ขั้นตอนการเคลื่อนย้ายข่าวสาร (Information transfer phase)

การเปลี่ยนแปลงของขั้นตอนในสายสัญญาณ แยกออกได้เป็น 3 แบบ ดังนี้

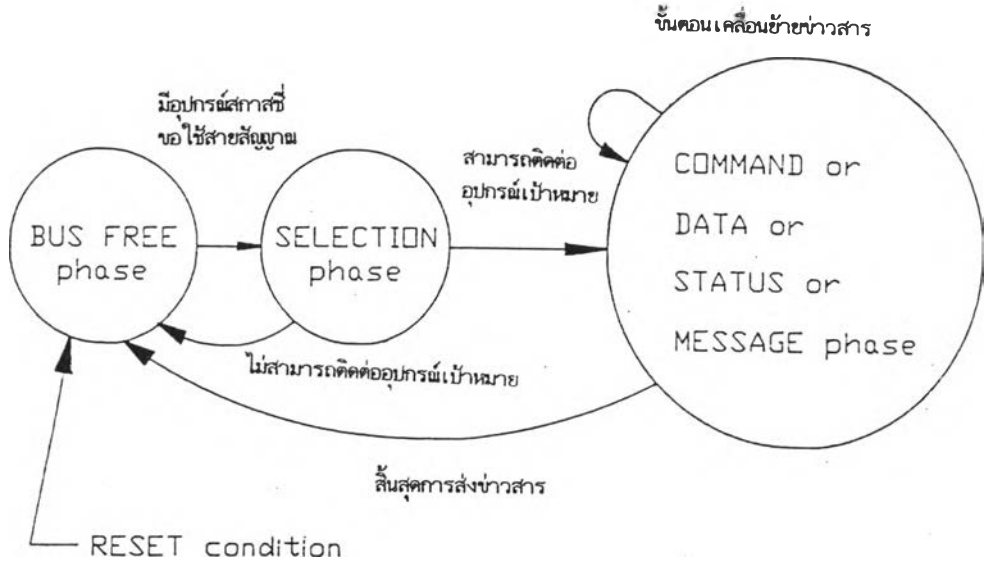
1. ระบบที่ไม่มีการแข่งขันเข้าใช้สายสัญญาณ (Nonarbitrating Systems)

ขั้นตอนของระบบจะเริ่มจากขั้นตอนปลอดการสื่อสาร ไปสู่การขอระบุอุปกรณ์เป้าหมายที่จะติดต่อด้วย และเข้าสู่ขั้นตอนการเคลื่อนย้ายข่าวสารต่าง ๆ

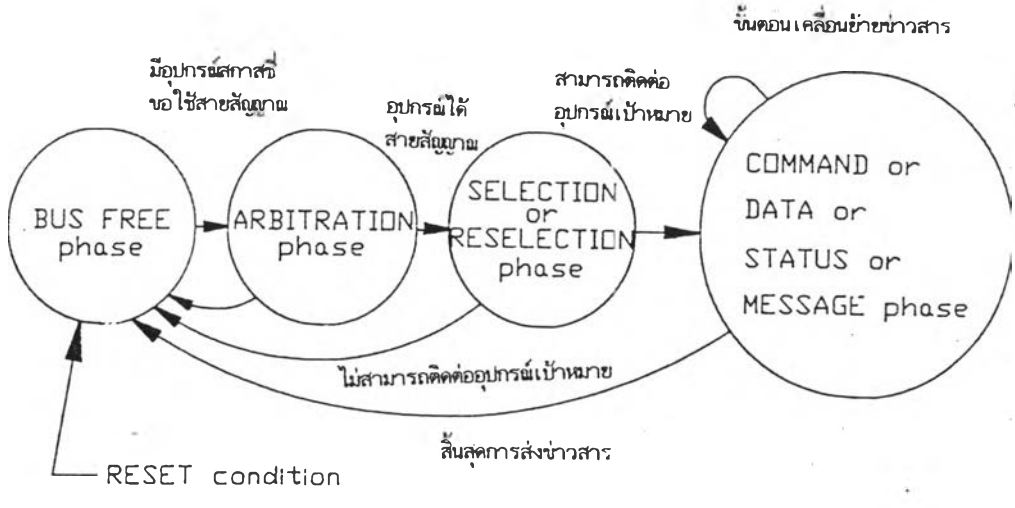
2. ระบบที่มีการแข่งขันเข้าใช้สายสัญญาณ (Arbitrating Systems)

ขั้นตอนของระบบจะเริ่มต้นจากขั้นตอนปลอดการสื่อสารไปสู่ ขั้นตอนการแข่งขันเพื่อขอใช้สายสัญญาณ อุปกรณ์ที่ได้ใช้สายสัญญาณจะระบุอุปกรณ์เป้าหมาย และเข้าสู่ขั้นตอนการเคลื่อนย้ายข่าวสารต่าง ๆ

สำหรับทั้ง 2 ระบบ เมื่อเข้าสู่ขั้นตอนการเคลื่อนย้ายข่าวสาร จะมีการเปลี่ยนแปลงระหว่างขั้นตอนเหล่านั้นในลำดับก็ได้



ภาพที่ 2.6 การเปลี่ยนแปลงของขั้นตอนในสายสัญญาณในระบบที่ไม่มีการแข่งขันเข้าใช้สายสัญญาณ



ภาพที่ 2.7 การเปลี่ยนแปลงของขั้นตอนในสายสัญญาณในระบบที่มีการแข่งขันเข้าใช้สายสัญญาณ

8. การอ้างถึงข้อมูลในอุปกรณ์ (Addressing)

โปรแกรมคอมพิวเตอร์ จะมองเห็นข้อมูลบนอุปกรณ์เก็บข้อมูลแบบเข้าถึงโดยตรง (Direct access storage device) และเข้าถึงแบบตามลำดับ (Sequential access storage device) ที่เป็นสกาสซี่ใด ๆ ในลักษณะของบล็อคข้อมูลขนาดเท่าๆ กันที่เรียงต่อเนื่องกันไป และอ้างถึงข้อมูลได้โดยใช้หมายเลขบล็อค โดยไม่จำเป็นต้องทราบรายละเอียดการจัดเก็บข้อมูลทางกายภาพที่แท้จริงของอุปกรณ์ ซึ่งทำให้เกิดความสะดวกอย่างมากในการพัฒนาโปรแกรมเพื่อใช้งานอุปกรณ์

9. สถานะเงื่อนไขของสายสัญญาณสกาสซี่ (SCSI Bus Conditions)

ในสายสัญญาณสกาสซี่มีสถานะเงื่อนไข 2 สถานะ ซึ่งเมื่อเกิดสถานะเงื่อนไขขึ้นในสายสัญญาณ จะมีผลต่อการทำงานของอุปกรณ์สกาสซี่ในระบบ และอาจมีผลให้เกิดการเปลี่ยนลำดับขั้นตอนในสายสัญญาณ (phase sequence) ในขณะนั้นได้ สถานะเงื่อนไขทั้ง 2 คือ

9.1 สถานะเงื่อนไขแอคเทนชั่น (ATTENTION Condition) เป็นสถานะที่ อุปกรณ์ที่ทำหน้าที่เป็นผู้เริ่มต้นการสื่อสารข้อมูลสร้างขึ้นเพื่อให้ อุปกรณ์เป้าหมายทราบว่ามีข้อความ (message) ต้องการจะส่งถึงอุปกรณ์เป้าหมาย ซึ่งจะมีผลทำให้อุปกรณ์เป้าหมายส่งสัญญาณมาขอข้อความไป

9.2 สถานะเงื่อนไขรีเซต (RESET Condition) เป็นสถานะที่ทุกๆ อุปกรณ์ ในระบบสามารถสร้างขึ้น เพื่อบอกให้อุปกรณ์สกาสซี่อื่นๆ ในระบบปลดปล่อยสายสัญญาณ และ ดำเนินการตามขั้นตอนที่กำหนดไว้ในแต่ละอุปกรณ์เมื่อได้รับสัญญาณรีเซต สถานะเงื่อนไขรีเซตนี้ จะมีความสำคัญและมีผลทันทีไม่ว่าขณะนั้น สายสัญญาณกำลังอยู่ในขั้นตอนใด (phase) หรือสถานะไหน (condition) หลังสถานะเงื่อนไขรีเซต สายสัญญาณจะเข้าสู่ขั้นตอนปลอดการสื่อสารเสมอ

อุปกรณ์สกาสซี่แต่ละชนิดจะมีขั้นตอนการดำเนินงานเมื่อได้รับสัญญาณรีเซตแตกต่างกัน 2 แบบ ขึ้นกับว่าอุปกรณ์นั้นถูกกำหนดเป็นแบบใด คือ

9.2.1 อุปกรณ์ที่ถูกกำหนดขั้นตอนแบบฮาร์ดรีเซต (HARD RESET OPTION) เมื่อได้รับสัญญาณรีเซต อุปกรณ์จะดำเนินการเป็นขั้นตอนดังนี้

9.2.1.1 ยกเลิกคำสั่งทั้งหมดที่กำลังทำอยู่

9.2.1.2 ปลดปล่อยอุปกรณ์สกาสซี่ที่ขอสำรองไว้ใช้งาน

9.2.1.3 ปรับภาวะการดำเนินงาน (mode) ของอุปกรณ์ไปสู่ภาวะดั้งเดิมของอุปกรณ์ (default mode)

9.2.2 อุปกรณ์ที่ถูกกำหนดขั้นตอนแบบซอฟท์รีเซต (SOFT RESET OPTION) เมื่อได้รับสัญญาณรีเซต อุปกรณ์จะดำเนินการเป็นขั้นตอน ดังนี้

9.2.2.1 พยายามทำงานคำสั่งที่มีการระบุชัดเจนถึงอุปกรณ์เริ่มต้นการสื่อสารและอุปกรณ์เป้าหมาย (fully identified) ให้เสร็จสิ้นลง

9.2.2.2 คงการสำรองอุปกรณ์สกาสซี่ไว้

9.2.2.3 คงสภาพภาวะการดำเนินงานของอุปกรณ์ในขณะนั้นไว้

การกำหนดซอฟท์รีเซต เพื่อให้อุปกรณ์เริ่มต้นการสื่อสารหนึ่งสามารถสั่งรีเซตสายสัญญาณโดยไม่กระทบกระเทือนการทำงานของ อุปกรณ์เริ่มต้นการสื่อสารอื่น ๆ ในระบบที่มีหลายอุปกรณ์เริ่มต้นการสื่อสาร (multiple initiator)

มาตรฐานระบบเชื่อมต่ออุปกรณ์สกาสซี่ได้กำหนดเงื่อนไขการทำงานต่างๆ ดังนี้

1. อุปกรณ์เริ่มต้นการสื่อสารจะถือว่าคำสั่งที่ส่งไปยังอุปกรณ์เป้าหมาย มีการระบุชัดเจน (fully identified) ก็ต่อเมื่อมีการส่งข้อความ IDENTIFY ไปยังอุปกรณ์เป้าหมาย และอุปกรณ์เป้าหมายตอบกลับโดยการเปลี่ยนขั้นตอนในสายสัญญาณสกาสซี่ ไปเป็นขั้นตอนต่าง ๆ ในขั้นตอนการเคลื่อนย้ายข่าวสาร และเริ่มต้นการส่งข้อมูลอย่างน้อย 1 ไบต์

2. อุปกรณ์เป้าหมายจะถือว่าคำสั่งที่ได้รับมีการระบุชัดเจนเมื่ออุปกรณ์เป้าหมายได้รับข้อความ IDENTIFY

3. ถ้าอุปกรณ์เริ่มต้นการสื่อสารส่งคำสั่งเลือกอุปกรณ์เป้าหมายไปยังอุปกรณ์เป้าหมาย ที่กำลังทำงานตามคำสั่งที่ได้รับจากอุปกรณ์เริ่มต้นการสื่อสารข้อมูลอยู่ อุปกรณ์เป้าหมายจะยกเลิกคำสั่งที่ได้รับเดิม และทำงานตามคำสั่งใหม่

4. ถ้าอุปกรณ์เป้าหมายระบุเลือกอุปกรณ์เริ่มต้นการสื่อสาร โดยที่อุปกรณ์เริ่มต้นการสื่อสารไม่มีข้อมูลจะส่งให้ อุปกรณ์เริ่มต้นการสื่อสารจะยกเลิกคำสั่งโดยการส่งข้อความ ABORT

5. อุปกรณ์เริ่มต้นการสื่อสารจะถือว่าคำสั่งที่ส่งไปสมบูรณ์ เมื่อได้ถอนสัญญาณตอบ (ACK) หลังจากที่ได้รับข้อความ COMMAND COMPLETE

6. อุปกรณ์เป้าหมายจะถือว่าคำสั่งที่ทำเสร็จสมบูรณ์ เมื่ออุปกรณ์นั้นได้พบว่าไม่มีสัญญาณ ACK และ ATN หลังจากส่งข้อความ COMMAND COMPLETE

7. เมื่อได้รับข้อความ SAVE DATA POINTER อุปกรณ์เริ่มต้นการสื่อสารข้อมูลจะทำการบันทึก DATA POINTER ให้เสร็จสิ้นก่อน แล้วจึงถอนสัญญาณ ACK

8. อุปกรณ์เป้าหมายจะถือว่าอุปกรณ์เริ่มต้นการสื่อสารข้อมูลได้บันทึก DATA POINTER เรียบร้อยเมื่ออุปกรณ์นั้นไม่มีสัญญาณ ACK และ ATN หลังจากอุปกรณ์เป้าหมายได้ส่งข้อความ SAVE DATA POINTER

9. ถ้าในสายสัญญาณมีเงื่อนไขรีเซตเกิดขึ้น หลังจากเวลาที่อุปกรณ์เป้าหมายใส่สัญญาณ REQ เพื่อส่งข้อความ SAVE DATA POINTER และก่อนเวลาที่อุปกรณ์เป้าหมายพบว่าไม่มี

สัญญาณ ACK อุปกรณ์เป้าหมายจะยุติการทำงานของคำสั่ง และส่งสถานะการทำงาน CHECK CONDITION ไปยังอุปกรณ์เริ่มต้นการสื่อสาร ซึ่งถ้าในระบบ EXTENDED SENSE KEY อุปกรณ์เป้าหมายจะส่ง SENSE KEY ที่แสดงถึงการยกเลิกคำสั่ง (ABORTED COMMAND) เนื่องจากในขณะนั้นอุปกรณ์เป้าหมายไม่ทราบว่าอุปกรณ์เริ่มต้นการสื่อสารข้อมูลได้เก็บ DATA POINTER ไว้เรียบร้อยแล้วหรือไม่

หมายเหตุ ในเงื่อนไขการทำงานข้อที่ 6,8 ถ้ามีสัญญาณ ATN อยู่อุปกรณ์เป้าหมายจะเปลี่ยนขั้นตอนของสายสัญญาณไปสู่ขั้นตอนข้อความ MESSAGE OUT) และพยายามส่งข้อความ

ถ้ามีเงื่อนไขรีเซตเกิดขึ้น อุปกรณ์เป้าหมายจะถือว่าอุปกรณ์เริ่มต้นการสื่อสารไม่ได้รับข้อความ COMMAND COMPLETE หรือ SAVE DATA POINTER MESSAGE

ในกรณีของข้อความ COMMAND COMPLETE อุปกรณ์เป้าหมายจะขอติดต่อกับอุปกรณ์เริ่มต้นการสื่อสารข้อมูลใหม่ และขอส่งข้อความ COMMAND COMPLETE อีกครั้งหนึ่ง

ในกรณีของข้อความ SAVE DATA POINTER อุปกรณ์เป้าหมายจะขอติดต่อกับอุปกรณ์เริ่มต้นการสื่อสาร และดำเนินการตามในข้อ 9

10. ตัวชี้ข่าวสาร (SCSI Pointer)

เมื่ออุปกรณ์เริ่มต้นการสื่อสารต้องการจะส่งคำสั่งใดไปยังอุปกรณ์เป้าหมาย อุปกรณ์เริ่มต้นการสื่อสารจะเตรียมหน่วยความจำไว้ 3 บริเวณ สำหรับการเก็บข้อมูล คำสั่ง และสถานะไว้สำหรับการติดต่อสื่อสารกับอุปกรณ์เป้าหมายโดยได้เตรียมตัวชี้ (pointer) กำกับแต่ละบริเวณ อุปกรณ์เป้าหมายจะสามารถเข้าถึงข้อมูลโดยใช้ตัวชี้

ตัวชี้จะถูกเตรียมเป็น 2 ชุด คือ

ตัวชี้ชุดปัจจุบัน (Current pointer) จะชี้ไปยังตำแหน่งทำงานปัจจุบันของข้อมูล (DATA POINTER) คำสั่ง (COMMAND POINTER) และสถานะการทำงาน (STATUS POINTER) ในอุปกรณ์เริ่มต้นการสื่อสาร 1 อุปกรณ์จะมีตัวชี้ชุดปัจจุบันเพียงชุดเดียวเท่านั้นซึ่งจะสัมพันธ์กับคำสั่งที่ทำงานปัจจุบัน

ตัวชี้ชุดเก็บ (Saved pointer) จะมีจำนวนชุดเท่ากับจำนวนคำสั่งที่กำลังทำงานอยู่ รวมทั้งคำสั่งที่พักการทำงานไว้เนื่องจากอุปกรณ์เป้าหมายของคำสั่งนั้นกำลังทำงาน

เมื่อเริ่มต้นการทำงานแต่ละคำสั่งตัวชี้ชุดปัจจุบัน และตัวชี้ชุดเก็บของคำสั่งนั้นจะชี้ไปยังตำแหน่งเริ่มต้นของคำสั่ง ข้อมูล และสถานะ เมื่ออุปกรณ์เป้าหมายเริ่มการทำงานจะเปลี่ยนตัวชี้ชุดปัจจุบันไปตามการทำงาน โดยตัวชี้ชุดเก็บยังมีค่าเหมือนเดิมเมื่ออุปกรณ์เป้าหมายส่งข้อความ SAVE DATA POINTER ไปยังอุปกรณ์เริ่มต้นการสื่อสาร อุปกรณ์เริ่มต้นการสื่อสารจะทำสำเนาของตัวชี้ชุดปัจจุบันไปเก็บไว้ในตัวชี้ชุดเก็บคำสั่งนั้น และเมื่ออุปกรณ์เป้าหมายส่งข้อความ

DISCONNECT อุปกรณ์เริ่มต้นการสื่อสารจะพักการทำงานคำสั่งนั้น หลังจากนั้นอุปกรณ์เริ่มต้นการสื่อสารอาจสั่งงานคำสั่งอื่น ไปยังอุปกรณ์เป้าหมายอื่นต่อไป โดยใช้ตัวชี้ชุดปัจจุบันเป็นชุด ทำงาน

เมื่ออุปกรณ์เป้าหมายของคำสั่งแรกทำงานเสร็จ และสายสัญญาณว่างลง อุปกรณ์เป้าหมายจะติดต่อกลับมายังอุปกรณ์เริ่มต้นสื่อสารข้อมูลเพื่อขอทำงานต่อ และส่งข้อความ RESTORE POINTER เพื่อให้อุปกรณ์เริ่มต้นการสื่อสารคัดลอกข้อมูลตัวชี้ จากตัวชี้ตัวชี้ชุดเก็บของคำสั่งนั้น มายังตัวชี้ชุดปัจจุบัน เพื่อให้อุปกรณ์เป้าหมายทำงานต่อจากที่พักไว้ได้

ตัวชี้ข่าวสารจะมีความสำคัญในระบบเชื่อมต่ออุปกรณ์แบบสกาสซี่ที่มีอุปกรณ์เริ่มต้นการสื่อสารหลายอุปกรณ์ และมีระบบการแข่งขันเพื่อเข้าใช้สายสัญญาณ ซึ่งในระบบดังกล่าวจะมีการพักและดำเนินงานต่อจากที่พัก

11. ข้อกำหนดของข้อความสื่อสาร (Message System Specification)

ข้อความสื่อสารที่ส่งถึงกันระหว่างอุปกรณ์เริ่มต้นการสื่อสารและอุปกรณ์เป้าหมาย มีจุดประสงค์หลักเพื่อจัดการดูแล และเรื่องการเชื่อมต่อเส้นทางสื่อสารระหว่างอุปกรณ์ทางกายภาพ (physical path management) รวมทั้งเพื่อรายงานการสิ้นสุดการทำงานตามคำสั่งที่อุปกรณ์เป้าหมายได้รับ

11.1 พิธีการของข้อความ (Message Protocol)

อุปกรณ์สกาสซี่ทุกอุปกรณ์จะสามารถส่งข้อความ COMMAND COMPLETE เสมอ และโดยอาจจะสามารถรับ/ส่งข้อความอื่นหรือไม่ก็ได้

สำหรับอุปกรณ์ที่สามารถรับ/ส่งข้อความได้มากกว่า COMMAND COMPLETE จะใช้วิธีการระบุอุปกรณ์ที่เริ่มต้นการสื่อสารข้อมูล และอุปกรณ์เป้าหมายใช้ข้อความ IDENTIFY ไม่ใช้เลขหมายโลจิคัลของอุปกรณ์ที่ปรากฏในส่วนคำสั่ง กล่าวคือ

เมื่ออุปกรณ์ที่เริ่มต้นการสื่อสารข้อมูลระบุอุปกรณ์เป้าหมายได้แล้ว ข้อความแรกที่ส่งไปยังอุปกรณ์เป้าหมายคือ ข้อความ IDENTIFY เพื่อสร้างการเชื่อมต่อทางกายภาพ (physical path connection) ระหว่างอุปกรณ์นั้น และในทำนองเดียวกัน เมื่ออุปกรณ์เป้าหมายระบุอุปกรณ์เริ่มต้นการสื่อสารข้อมูลเพื่อขอสื่อสารต่อจากที่พักไว้ ข้อความแรกที่อุปกรณ์เป้าหมายส่งไปยังอุปกรณ์เริ่มต้นการสื่อสารข้อมูล คือ ข้อความ IDENTIFY โดยจะมีเพียงเลขหมายโลจิคัลเดียวของอุปกรณ์ที่ถูก IDENTIFY การส่งข้อความ IDENTIFY ไปยังเลขหมายโลจิคัลอื่น เป็นครั้งที่สอง หลังจากการระบุอุปกรณ์เป้าหมายได้แล้วนั้น จะไม่มีผลแต่อย่างไร

Code	Type	Description	Direction	
01h	M	COMMAND COMPLETE	In	
01h	O	EXTENDED MESSAGE	In	Out
02h	O	SAVE DATA POINTER	In	
03h	O	RESTORE POINTERS	In	
04h	O	DISCONNECT	In	
05h	O	INITIATOR DETECTED ERROR		Out
06h	O	ABORT		Out
07h	O	MESSAGE REJECT	In	Out
08h	O	NO OPERATION		Out
09h	O	MESSAGE PARITY ERROR		Out
0Ah	O	LINKED COMMAND COMPLETE		Out
0Bh	O	LINKED COMMAND COMPLETE (WITH FLAG)	In	
0Ch	O	BUS DEVICE RESET		Out
0Dh-7 Fh	R	Reserved Codes		Out
80-ffh	O	IDENTIFY	In	Out

ตารางที่ 2.1 แสดงรหัส ประเภท และทิศทางการเคลื่อนย้ายของข้อความ (messages)

11.2 ข้อความ (Messages)

ข้อความที่ใช้สื่อสารระหว่างอุปกรณ์ในสายสัญญาณสกาสซี่ แบ่งออกเป็น 2 ประเภท คือ

11.2.1 ข้อความขนาดไบต์เดียว (Single byte messages) ประกอบด้วยข้อความ

11.2.1.1 COMMAND COMPLETE รหัส 00h(ข้อความบังคับ) เป็นข้อความที่อุปกรณ์สกาสซี่ทุกอุปกรณ์จะต้องมีและสามารถส่งข้อความนี้ได้ ข้อความนี้จะถูกส่งจาก อุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นการสื่อสาร เพื่อแสดงว่า อุปกรณ์เป้าหมายได้ทำงานตามคำสั่งแล้ว และได้ส่งสถานะการทำงานไปยังอุปกรณ์เริ่มต้นการสื่อสารแล้ว (สถานะการทำงานจะเป็นตัวบ่งบอกว่าการทำงานคำสั่งดังกล่าวสมบูรณ์หรือไม่ อย่างไร?)

11.2.1.2 SAVE DATA POINTER รหัส 01h (ข้อความไม่บังคับ) เป็นข้อความที่ถูกส่งจากอุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นการสื่อสารข้อมูล เพื่อให้อุปกรณ์เริ่มต้นการสื่อสารคัดลอกข้อมูลตัวชี้ชุดปัจจุบันไปยังตัวชี้ชุดเก็บของคำสั่งเดิมของอุปกรณ์ เลขหมายโลจิคัล ที่กำลังติดต่อสื่อสารด้วยในขณะนั้น

11.2.1.3 RESTORE POINTERS รหัส 03h(ข้อความไม่บังคับ) เป็นข้อความที่ถูกส่งจากอุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นการสื่อสารข้อมูลเพื่อให้อุปกรณ์เริ่มต้นการสื่อสารข้อมูล คัดลอกข้อมูลตัวชี้ชุดเก็บของอุปกรณ์เลขหมายโลจิคัล ที่กำลังติดต่อสื่อสารด้วย ในขณะนั้นมายังตัวชี้ชุดปัจจุบัน เพื่อทำงานต่อ

11.2.1.4 DISCONNECT รหัส 04h(ข้อความไม่บังคับ) เป็นข้อความที่ถูกส่งจากอุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นการสื่อสารข้อมูล เพื่อขอพักการสื่อสารไว้ โดยจะมีการติดต่อเพื่อขอสื่อสารข้อมูลต่อจากที่พักไว้ในครั้งนี้ภายหลัง

ถ้าอุปกรณ์เริ่มต้นการสื่อข้อมูลตรวจสอบสายสัญญาณพบว่าเป็นช่วงปลอดการสื่อสาร ในขณะที่ยังสื่อสารกับอุปกรณ์เป้าหมายยังไม่เสร็จสิ้น และไม่ได้รับข้อความ DISCONNECT หรือ COMMAND COMPLETE อุปกรณ์เริ่มต้นการสื่อสารจะถือว่ามีความผิดพลาดเกิดขึ้นในการสื่อสาร การทำงานตามคำสั่งดังกล่าวไม่สมบูรณ์

ข้อความ DISCONNECT ไม่มีผลต่อการคัดลอกข้อมูลตัวชี้แต่อย่างใด ดังนั้นเพื่อให้อุปกรณ์เป้าหมายสามารถทำงานต่อจากที่ค้างไว้ได้ในภายหลังจะต้องมีการส่งข้อความ SAVE DATA POINTER ก่อนแล้วตามด้วยข้อความ DISCONNECT

11.2.1.5 INITIATOR DETECTED ERROR รหัส 05h (ข้อความไม่บังคับ) เป็นข้อความที่ถูกส่งจากอุปกรณ์เริ่มต้นการสื่อสารข้อมูลไปยังอุปกรณ์เป้าหมายเพื่อแจ้งว่าได้มีความผิดพลาดเกิดขึ้น (เช่น การเกิดตรวจสอบพบว่าค่าพาริตี้ผิดพลาด (parity error)) อุปกรณ์เป้าหมายจะสั่งงานตามคำสั่งนั้นอีกครั้ง

ก่อนที่จะทำงานต่ออุปกรณ์เป้าหมายอาจจะส่งข้อความ RESTORE POINTERS หรือ ข้อความ DISCONNECT แล้วตามด้วยการ CONNECT เพื่อให้ตัวชี้ชุดปัจจุบัน คัดลอกข้อมูลจากตัวชี้ชุดเก็บมาใช้งาน เนื่องจากขณะนั้น ตัวชี้ชุดปัจจุบันอาจมีค่าข้อมูลที่ไม่สัมพันธ์กับการทำงานเลยก็ได้

11.2.1.6 ABORT รหัส 06h (ข้อความไม่บังคับ) เป็นข้อความที่ถูกส่งจากอุปกรณ์เริ่มต้นการสื่อสารข้อมูลเพื่อยกเลิกการทำงานในขณะนั้น

ในกรณีที่อุปกรณ์เริ่มต้นการสื่อสารมีการระบุอุปกรณ์เป้าหมายโดยใช้ IDENTIFY อุปกรณ์เป้าหมายจะลบทั้งข้อมูลและสถานะการทำงานของอุปกรณ์เริ่มต้นการสื่อสารนั้น โดยยังคงข้อมูลและสถานะการทำงานของอุปกรณ์เริ่มต้นการสื่อสาร

อื่นเอาไว้ หลังจากนั้นอุปกรณ์เป้าหมายจะเปลี่ยนขั้นตอนของสายสัญญาณเป็นขั้นตอนปลอดการสื่อสาร ในกรณีที่อุปกรณ์เริ่มต้นการสื่อสารไม่มีการระบุอุปกรณ์เป้าหมายโดยใช้ IDENTIFY อุปกรณ์เป้าหมายจะเปลี่ยนขั้นตอนของสายสัญญาณ เป็นขั้นตอนปลอดสัญญาณ โดยไม่มีการส่งทั้งสถานะหรือข้อความใด ๆ กลับไปยังอุปกรณ์เริ่มต้นการสื่อสารข้อมูลเลย

11.2.1.6 MESSAGE REJECT รหัส 07h (ข้อความไม่บังคับ) เป็นข้อความที่อุปกรณ์เริ่มต้นการสื่อสารหรืออุปกรณ์เป้าหมายส่งถึงกัน เพื่อระบุว่าข้อความที่ ได้รับไม่เหมาะสม หรือไม่รู้จักข้อความนั้น

11.2.1.8 NO OPERATION รหัส 08h (ข้อความไม่บังคับ) เป็นข้อความที่อุปกรณ์เริ่มต้นการสื่อสารส่งไปยังอุปกรณ์เป้าหมาย เพื่อตอบสนองต่อการขอข้อความของอุปกรณ์เป้าหมาย ว่าในขณะนั้น อุปกรณ์เริ่มต้นการสื่อสารไม่มีข้อความที่จะส่ง

11.2.1.9 MESSAGE PARITY ERROR รหัส 09h (ข้อความไม่บังคับ) เป็นข้อความที่ส่งจากอุปกรณ์เริ่มต้นการสื่อสารข้อมูลไปยังอุปกรณ์เป้าหมาย เพื่อระบุว่าข้อความที่อุปกรณ์เริ่มต้นการสื่อสารได้รับในครั้งล่าสุด มีค่าพาริตี้ผิดพลาด จำนวน 1 หรือ หลายไบต์

11.2.1.10 LINKED COMMAND COMPLETE รหัส 0Ah (ข้อความไม่บังคับ) เป็นข้อความที่ถูกส่ง จากอุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นการสื่อสารเพื่อระบุว่า การทำงานตามชุดคำสั่ง(คำสั่งตั้งแต่ 2 คำสั่งขึ้นไป ที่ส่งไปยังอุปกรณ์เป้าหมายเดียวกัน ต่อเนื่องกันไป (LINKED COMMAND) เสร็จแล้ว และได้จัดส่งสถานะการทำงานของคำสั่งกลับคืน ไปยังอุปกรณ์เริ่มต้นการสื่อสารข้อมูลแล้ว

เมื่ออุปกรณ์เริ่มต้นการสื่อสารได้รับ ข้อความนี้จะกำหนดตัวชี้ไปยังจุดเริ่มต้นของชุดคำสั่งใหม่

11.2.1.11 LINKED COMMAND COMPLETE (WITH FLAG) 0Bh (ข้อความไม่บังคับเป็นข้อความในลักษณะเดียวกับ 11.2.1.10) โดยใช้กับชุดคำสั่ง (LINKED COMMAND) ที่มีการกำหนดค่าแฟลกบิต (flag bit) เป็น 1 โดยทั่วไปแล้วข้อความนี้จะใช้เพื่อก่อให้เกิดการขัดจังหวะ (interrupt) ในอุปกรณ์เริ่มต้นการสื่อสารในระหว่างการทำงานตามชุดคำสั่ง 2 ชุด

11.2.1.12 BUS DEVICE RESET รหัส 0Ch (ข้อความไม่บังคับ) เป็นข้อความที่อุปกรณ์เริ่มต้นการสื่อสารข้อมูล ส่งไปยังอุปกรณ์เป้าหมายเพื่อขอยกเลิกคำสั่งปัจจุบันทุกคำสั่ง เมื่ออุปกรณ์เป้าหมายได้รับข้อความนี้จะปรับเข้าสู่สถานะเริ่มต้นของอุปกรณ์ และยกเลิกการทำงานตามคำสั่ง ของอุปกรณ์เริ่มต้นการสื่อสารทุกอุปกรณ์

11.2.1.13 RESERVED รหัส 0Dh ถึง 7Fh ข้อความรหัสเหล่านี้สำรองไว้ใช้ในอนาคต

11.2.1.14 IDENTIFY รหัส 80h ถึง FFh (ข้อความไม่บังคับ) เป็นข้อความที่ทั้งอุปกรณ์เริ่มต้นการสื่อสารและอุปกรณ์เป้าหมายส่งถึงกัน เพื่อตกลงเส้นทางเชื่อมต่อการสื่อสารกันทางกายภาพ มีรายละเอียดของการกำหนดบิตของข้อมูล ดังนี้

บิต 7 จะเป็น 1 เสมอเพื่อแสดงว่าเป็น IDENTIFY MESSAGE

บิต 6 จะกำหนดเป็น 1 โดยอุปกรณ์เริ่มต้นการสื่อสาร เพื่อแสดงว่าอุปกรณ์เริ่มต้นการสื่อสารสามารถที่จะรองรับการหยุดพักและการเชื่อมต่อการสื่อสารใหม่ได้

บิต 5-3 สำรอง

บิต 2-0 แสดงเลขหมายโลจิคัลของอุปกรณ์เป้าหมาย (ถ้าอุปกรณ์เริ่มต้นการสื่อสารเป็นผู้ส่งข้อความ) หรือเลขหมายโลจิคัลของทุกอุปกรณ์เริ่มต้นการสื่อสาร (ถ้าอุปกรณ์เป้าหมายเป็นผู้ส่งข้อความ)

ถ้าอุปกรณ์เป้าหมายส่งข้อความ IDENTIFY ไปยังอุปกรณ์เริ่มต้นการสื่อสาร เพื่อขอสื่อสารข้อมูลต่อจากที่พักไว้ (Reconnection) อุปกรณ์เริ่มต้นการสื่อสารจะทำการคัดลอกตัวชี้ชุดเก็บมาสู่ตัวชี้ชุดปัจจุบันโดยอัตโนมัติ

11.2.2 ข้อความขนาดหลายไบต์ (Multi bytes messages)

ข้อความขนาดหลายไบต์ใช้ในกรณีที่ข้อความที่จะสื่อสารกันไม่ สามารถแสดงได้โดยใช้ขนาดเพียงไบต์เดียว การส่งข้อความขนาดหลายไบต์ จะเริ่มต้นด้วยการส่งข้อความไบต์แรก เป็น EXTENDED MESSAGE (รหัส 01h) ตามด้วยไบต์ที่สองเป็น EXTENDED MESSAGE LENGTH และไบต์ที่สามเป็น EXTENDED MESSAGE CODE หลังจากนั้นเป็นข้อมูลประกอบ EXTENDED MESSAGE (EXTENDED MESSAGE ARGUMENTS) ซึ่งจะมีกี่ไบต์ก็ได้ (ตารางที่ 2.2)

Byte	Value	Description
0	01h	Extended message
1	n	Extended message length
2	y	Extended message code
3-n+1	x	Extended message arguments

ตารางที่ 2.2 แสดงโครงสร้างข้อมูลของ EXTENDED MESSAGE

Code (y)	Description
00h	MODIFY DATA POINTER (Optional)
01h	SYNCHRONOUS DATA TRANSFER REQUEST (Optional)
02h	EXTENDED IDENTIFY (Optional)
03h - 7Fh	Reserved
80h - FFh	Vendor Unique

ตารางที่ 2.3 แสดงรหัสของ EXTENDED MESSAGE

EXTENDED MESSAGE LENGTH จะเป็นผลรวมของขนาดของ EXTENDED MESSAGE CODE กับ EXTENDED MESSAGE ARGUMENTS ถ้า EXTENDED MESSAGE LENGTH เป็น 0 จะหมายถึงว่าจะมี EXTENDED MESSAGE CODE 1 ไบต์ และ EXTENDED MESSAGE ARGUMENTS 255 ไบต์

EXTENDED MESSAGE CODE (ตารางที่ 2.3) ประกอบด้วย

11.2.2.1 MODIFY DATA POINTER รหัส 00h (ข้อความไม่บังคับ) เป็นข้อความที่ส่งจากอุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นการสื่อสาร เพื่อขอเปลี่ยนแปลงตัวชี้ข้อมูลปัจจุบัน (CURRENT DATA POINTER) โดยการให้บวกข้อมูล (เป็นตัวเลขที่มีเครื่องหมาย) ที่ส่งมากับข้อความนี้เข้ากับตัวชี้ข้อมูล

(ดูรายละเอียดโครงสร้างข้อมูลในข้อกำหนดมาตรฐานสกาสชี่)

11.2.2.2 SYNCHRONOUS DATA TRANSFER REQUEST รหัส 01h (ข้อความไม่บังคับ) เป็นข้อความที่ทั้งอุปกรณ์เริ่มต้นการสื่อสารข้อมูล และอุปกรณ์เป้าหมายส่งถึงกัน เพื่อยืนยันที่จะสื่อสารกันด้วยพิธีการสมวาร หรือส่งถึงกันในกรณีที่การสื่อสารแบบสมวารของอุปกรณ์ทั้งสองชะงักลง เนื่องจากสายสัญญาณถูกทำให้อยู่ในสภาพรีเซท ("HARD" RESET) หรือได้รับ BUS DEVICE RESET

(ดูรายละเอียดโครงสร้างข้อมูลในข้อกำหนดมาตรฐานสกาสชี่)

11.2.2.3 EXTENDED IDENTIFY MESSAGE รหัส 02h (ข้อความไม่บังคับ) เป็นข้อความในลักษณะเดียวกับ IDENTIFY คือส่งโดยทั้งอุปกรณ์เริ่มต้นการสื่อสารข้อมูล และอุปกรณ์เป้าหมายเพื่อประกอบกับ IDENTIFY MESSAGE ให้มีความสามารถในการอ้างถึงหมายเลขโลจิคัลย่อยของหมายเลขโลจิคัลใหญ่ เนื่องจากในข้อความนี้สามารถระบุหมายเลขโลจิคัลย่อยได้ถึง 256 หมายเลข (1 ไบต์) เมื่อประกอบกับการระบุหมายเลขโลจิคัลได้ 8 หมายเลข จะทำให้อ้างถึง 2048 หมายเลขโลจิคัลได้

12. สถานะการทำงานของคำสั่ง (Status byte)

สถานะการทำงานของคำสั่ง จะถูกส่งจากอุปกรณ์เป้าหมายไปยังอุปกรณ์เริ่มต้นการสื่อสารหลังจากเสร็จสิ้นการทำงานแต่ละคำสั่ง

สถานะการทำงานจะหายไปถ้าอุปกรณ์เป้าหมายได้รับข้อความ ABORT หรือ ข้อความ BUS DEVICE RESET หรือ สายสัญญาณอยู่ในสภาวะฮาร์ดรีเซต

โครงสร้างของสถานะคำสั่ง และ ค่าของสถานะ แสดงในตารางที่ 2.4

Bit Byte	7	6	5	4	3	2	1	0
0	Reserve	Vendor Unique		Status byte code				V

ตารางที่ 2.4 แสดงโครงสร้างสถานะการทำงานของคำสั่ง (Status byte)

Bits of Status Byte								Status(es) Represented
7	6	5	4	3	2	1	0	
R	V	V	0	0	0	0	V	GOOD
R	V	V	0	0	0	1	V	CHECK CONDITION
R	V	V	0	0	1	0	V	CONDITION MET/GOOD
R	V	V	0	0	1	1	V	Reserved
R	V	V	0	1	0	0	V	BUSY
R	V	V	0	1	0	1	V	Reserved
R	V	V	0	1	1	0	V	Reserved
R	V	V	0	1	1	1	V	Reserved
R	V	V	1	0	0	0	V	INTERMEDIATED/GOOD
R	V	V	1	0	0	1	V	Reserved
R	V	V	1	0	1	0	V	INTERMEDIATED/CONDITION MET/GOOD
R	V	V	1	0	1	1	V	Reserved
R	V	V	1	1	0	0	V	RESERVATION CONFLICT
R	V	V	1	1	0	1	V	Reserved
R	V	V	1	1	1	0	V	Reserved
R	V	V	1	1	1	1	V	Reserved

ตารางที่ 2.5 แสดงรหัสสถานะการทำงาน (Status byte code)

ซึ่งมีคำอธิบายสถานะดังนี้ (ตารางที่ 2.5)

GOOD หมายถึง ว่าอุปกรณ์เป้าหมายได้ทำงานตามคำสั่งที่ได้รับมาเสร็จสิ้นสมบูรณ์

CHECK CONDITION หมายถึง มีความผิดพลาดต่าง ๆ เกิดขึ้นในการทำงานตามคำสั่งและมีเซนซ์คาต้า ที่จะแสดงสาเหตุความผิดพลาดรอให้อุปกรณ์เริ่มต้นการสื่อสารมาอ่านไป

CONDITION MET หมายถึง การพบข้อมูลตามเงื่อนไข ที่ระบุในคำสั่ง SEARCH DATA สถานะนี้จะไม่ทำลายการเชื่อมกันของชุดคำสั่ง และจะส่งเลขหมายโลจิคัลบล็อคที่พบข้อมูลกลับคืน ไปยังอุปกรณ์เริ่มต้นการสื่อสารเมื่อได้รับคำสั่ง REQUEST SENSE

BUSY หมายถึง อุปกรณ์เป้าหมายกำลังทำงานอยู่ไม่สามารถทำงานตามคำสั่งใหม่ที่เพิ่งได้รับมาได้ การแก้ไขปัญหาทำได้โดยการสั่งให้ทำงานคำสั่งนั้นใหม่อีกครั้ง

INTERMEDIATE หมายถึง การเสร็จสิ้นการทำงานแต่ละคำสั่งในชุดคำสั่ง (ยกเว้นคำสั่งสุดท้าย) ถ้ามีความผิดพลาดเกิดขึ้นในการทำงานแต่ละคำสั่งจะให้ค่าสถานะอื่น เช่น CHECK CONDITION หรือ RESERVATION CONFLICT เป็นต้น

ถ้ามีการสั่งงานชุดคำสั่ง แต่ไม่มีสถานะ INTERMEDIATE ส่งคืนมายังอุปกรณ์เริ่มต้นการสื่อสารแสดงว่า ชุดคำสั่งแตกออกจากกัน อุปกรณ์เป้าหมายจะไม่ทำงานตามคำสั่งในชุดนั้นต่อไปอีก

RESERVATION CONFLICT หมายถึง คำสั่งที่ทำให้เกิดสถานะนี้เกี่ยวข้องกับอุปกรณ์เป้าหมาย หรือบางส่วนของอุปกรณ์เป้าหมายที่ถูกสำรองไว้ใช้เป็นการส่วนตัวโดยอุปกรณ์สถาสชี่อื่นแล้ว

<u>ข้อดีของระบบการเชื่อมต่ออุปกรณ์แบบสกาสชี่</u>

1. ระบบการเชื่อมต่ออุปกรณ์แบบสกาสชี่ เป็นระบบการสื่อสารกลาง ซึ่งมีคำสั่งครอบคลุมถึงอุปกรณ์หลายประเภท ทำให้สามารถพ่วงต่ออุปกรณ์ต่างประเภทกัน เข้ามาสู่ระบบการสื่อสารระบบเดียวกันได้ นอกจากเป็นการลดความซับซ้อนระบบการเชื่อมต่ออุปกรณ์ของเครื่องคอมพิวเตอร์แล้ว ยังเป็นการลดค่าใช้จ่าย และทำให้เกิดความเป็นอันหนึ่งอันเดียวกันของระบบเชื่อมต่ออุปกรณ์คอมพิวเตอร์

2. ระบบการเชื่อมต่ออุปกรณ์แบบสกาสชี่ ไม่ได้เข้าควบคุมอุปกรณ์ต่าง ๆ โดยตรง เป็นเพียงทางผ่านของคำสั่ง ข้อมูล ข้อความ สถานะ และสัญญาณต่าง ๆ ระหว่างอุปกรณ์ที่ต่อ

i17731002y

เชื่อมกันในระบบ หน้าที่การควบคุมอุปกรณ์ ยังคงเป็นของแผงควบคุมอุปกรณ์ที่ต่อกับอุปกรณ์ต่าง ๆ โดยตรง ซึ่งแผงควบคุมอุปกรณ์จะต้องสามารถรับคำสั่งสกาสซี่ มาแปลเป็นคำสั่งเฉพาะเจาะจงเพื่อสั่งต่ออุปกรณ์ให้ทำงานตามต้องการจากรูปแบบการติดต่ออุปกรณ์แบบนี้ ทำให้เกิดข้อดี คือ

2.1 ผู้ผลิตอุปกรณ์แต่ละชนิดสามารถผลิต หรือเลือกใช้แผงควบคุมที่เหมาะสมและมีประสิทธภาพ โดยไม่ต้องคำนึงถึงว่าอุปกรณ์จะถูกนำไปพ่วงต่อกับคอมพิวเตอร์ระบบใด

2.2 แผงควบคุมอุปกรณ์แบบสกาสซี่ มักจะผนวกเข้าเป็นส่วนหนึ่งของอุปกรณ์ (embed controller) ทำให้สายสัญญาณที่ต่อระหว่างแผงควบคุมอุปกรณ์ และอุปกรณ์สั้น ลดโอกาสของการถูกรบกวนจากสัญญาณรอบข้าง และเพิ่มความเร็วในการติดต่อกัน

2.3 เครื่องคอมพิวเตอร์ที่มีระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ เป็นอิสระจากอุปกรณ์พ่วงต่อมากขึ้น (device independent)

3. ระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ สามารถส่งข้อมูลได้ด้วยความเร็วสูง คือสามารถส่งข้อมูลได้ถึง 1-2 เมกะไบต์ต่อวินาที (MB/s) ในแบบอสมวาร และ 4 เมกะไบต์ต่อวินาที (MB/s) ในแบบสมวาร

4. มีระบบควบคุมการเข้าใช้สายสัญญาณที่มีประสิทธิภาพ

5. ระบบการเชื่อมต่อแบบสกาสซี่ 1 ระบบ สามารถเชื่อมต่ออุปกรณ์จำนวนมากเข้าด้วยกัน คือได้สูงสุดถึง 56 อุปกรณ์ โดยคอมพิวเตอร์ 1 เครื่องสามารถต่อสกาสซี่โฮสอแดปเตอร์ได้ถึง 4 ชุด (เกิดระบบเชื่อมต่อได้ 4 ระบบ)

6. มีบริษัทผู้ผลิตอุปกรณ์ พ่วงต่อกับเครื่องคอมพิวเตอร์ และเครื่องคอมพิวเตอร์จำนวนมากได้นำระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ไปใช้ ซึ่งเป็นประโยชน์ทั้งผู้ผลิต และผู้ใช้ กล่าวคือผู้ผลิตจะสามารถขายอุปกรณ์ได้มากขึ้น เนื่องจากใช้งานได้กับ คอมพิวเตอร์หลายระบบขึ้น ทางด้านผู้ใช้นอกจากสามารถจะเลือกซื้ออุปกรณ์ได้มากชนิด แล้วยังได้อุปกรณ์ในราคาที่ถูกลง เนื่องจากกลไกของตลาด

<u>ข้อเสียของระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่</u>

1. ถ้าใช้ระบบการเชื่อมต่อนี้กับอุปกรณ์จำนวนน้อยชนิด ต้นทุนของระบบจะสูงกว่าระบบการเชื่อมต่อแบบอื่น เนื่องจากระบบการเชื่อมต่ออุปกรณ์แบบนี้ โดยพื้นฐานเป็นการเพิ่มเติมความซับซ้อนขึ้นในอุปกรณ์และเครื่องคอมพิวเตอร์ กล่าวคือจะต้องมีสกาสซี่โฮสอแดปเตอร์ (SCSI HOST ADAPTER)ขึ้นในเครื่องคอมพิวเตอร์ ในส่วนของอุปกรณ์สกาสซี่ ก็จะต้องมีแผงควบคุมอุปกรณ์ที่สามารถรับคำสั่งสกาสซี่จากสายสัญญาณ

2. ระบบการเชื่อมต่อนี้ ต้องมีโปรแกรมควบคุม (driver) เพิ่มขึ้นควบคุมและติดต่อกับสกาสซี่โฮสอแดปเตอร์ ซึ่งโปรแกรมนี้ต้องจัดหาเพิ่มเติม

3. อุปกรณ์แบบสกาสีมักจะถูกออกแบบและผลิตมาในลักษณะที่เป็นอุปกรณ์ต่างต่อแบบภายนอก (external) เพื่อเพิ่มประโยชน์ในลักษณะของการเคลื่อนย้ายสับเปลี่ยนระหว่างระบบ (Interchangeability) ทำให้แต่ละอุปกรณ์ต้องมีภาชนะบรรจุ (case) และแหล่งจ่ายไฟ (power supply) แยกต่างหากจากเครื่องคอมพิวเตอร์ ซึ่งเป็นการเพิ่มค่าใช้จ่าย (American National Standard Institute , Inc. ,1986)

2.2.2 ลักษณะคำสั่งที่ใช้สำหรับอุปกรณ์สกาสี่

ลักษณะทั่วไปของคำสั่งที่ใช้ในสายสัญญาณสกาสี่ คือ

1. คำสั่งทุกคำสั่งถ้าเกี่ยวข้องกับข้อมูลในอุปกรณ์สกาสี่แล้ว จะมองเห็นข้อมูลในรูปแบบของข้อมูลที่เรียงต่อเนื่องกันไปเป็นบล็อคขนาดคงที่ โดยขนาดและโครงสร้างของข้อมูลจะเป็นไปตามชนิดของอุปกรณ์ที่กำหนดไว้ในระบบการเชื่อมต่ออุปกรณ์แบบสกาสี่ แผงควบคุมอุปกรณ์สกาสี่จะทำหน้าที่แปลงตำแหน่งทางกายภาพของข้อมูลบนอุปกรณ์สกาสี่จริงไปเป็นตำแหน่งทางโลจิคัลในลักษณะบล็อคข้อมูล

2. คำสั่งที่สั่งไปยังอุปกรณ์สกาสี่ อาจเป็นคำสั่งเดี่ยวหรือ ชุดของคำสั่งไปยังอุปกรณ์เป้าหมายเดียวกัน ในกรณีที่เป็นชุดคำสั่ง แต่ละคำสั่งจะเชื่อม (link) ต่อกันโดยการกำหนดให้ link flag ที่ control byte เป็น 1 ในกรณีเป็นชุดคำสั่ง อุปกรณ์เป้าหมายจะทำงานแต่ละคำสั่งต่อเนื่องไปจนจบชุดคำสั่ง

3. เมื่ออุปกรณ์เป้าหมายทำงานแต่ละคำสั่งเสร็จสิ้น อุปกรณ์เป้าหมายจะส่งสถานะการทำงาน (status byte) กลับไปยังอุปกรณ์เริ่มต้นการสื่อสาร เพื่อแสดงผลของการทำงานตามคำสั่งว่า สมบูรณ์หรือผิดพลาดอย่างไร

ในกรณีที่ผลของคำสั่งไม่สามารถแสดงได้ด้วยสถานะเพียงไบต์เดียว อุปกรณ์เป้าหมายจะส่งสถานะ CHECK CONDITION เพื่อให้อุปกรณ์เริ่มต้นการสื่อสารข้อมูลทราบว่ามีสถานะการทำงานเพิ่มเติม ซึ่งอุปกรณ์เริ่มต้นการสื่อสารข้อมูลจะส่งคำสั่ง REQUEST SENSE เพื่อขอสถานะการทำงานเพิ่มเติมอีกที

4. การระบุหมายเลขโลจิคัลยูนิตของอุปกรณ์เป้าหมาย ระบุได้ 2 วิธีการ คือ

4.1 ระบุไปเป็นส่วนหนึ่งของคำสั่งในโครงสร้างคำสั่ง (COMMAND DESCRIPTOR BLOCK) นั้น 3 บิตสุดท้าย (บิต 5-7) ของไบต์ที่สอง เป็นส่วนของการระบุหมายเลขโลจิคัลยูนิตของอุปกรณ์เป้าหมาย ซึ่งอุปกรณ์สกาสี่ที่ไม่สามารถใช้ข้อความ IDENTIFY ได้จะใช้วิธีนี้ในการระบุ

4.2 ระบุในข้อความ IDENTIFY ซึ่งอุปกรณ์เริ่มต้นการสื่อสารหรืออุปกรณ์เป้าหมายส่งตามหลังขั้นตอนการเลือกอุปกรณ์เป้าหมาย (SELECTION PHASE) หรือการเลือกอุปกรณ์เริ่มต้นการสื่อสารข้อมูล (RESELECTION PHASE)

5. ในโครงสร้างของคำสั่งหลาย ๆ คำสั่ง ในขณะนี้จะมีการสำรองเนื้อที่บางไบต์บางบิต หรือแม้กระทั่งรหัสคำสั่งบางรหัส เพื่อไว้ใช้ในอนาคต โดยจะกำหนดเป็นมาตรฐานส่วนเพิ่มเติมอีกครั้งหนึ่ง ดังนั้นการสั่งงานคำสั่งต่าง ๆ ในขณะนี้ควรกำหนดให้ข้อมูลในเนื้อที่สำรองเหล่านี้เป็น 0 มิฉะนั้นเมื่ออุปกรณ์สกาสซี่ได้รับคำสั่งที่มีเนื้อที่สำรองเหล่านี้ไม่เป็น 0 จะไม่ทำงานตามคำสั่งพร้อมกับส่งสถานะการทำงาน CHECK CONDITION กลับไปยังอุปกรณ์เริ่มต้นการสื่อสาร และเมื่อได้รับคำสั่ง REQUEST SENSE จะส่งคืนค่า EXTENDED SENSE เป็นค่า ILLEGAL REQUEST

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Logical Block Address(if required)(MSB)				
2	Logical Block Address (if required)							
3	Logical Block Address(if required)(LSB)							
4	Transfer Length (if required)							
5	Control Byte							

ตารางที่ 2.6 แสดงโครงสร้างคำสั่งขนาด 6 ไบต์

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				RelAdr
2	Logical Block Address(if required)(MSB)							
3	Logical Block Address (if required)							
4	Logical Block Address (if required)							
5	Logical Block Address(if required)(LSB)							
6	Reserved							
7	Transfer Length (if required) (MSB)							
8	Transfer Length (if required) (LSB)							
9	Control Byte							

ตารางที่ 2.7 แสดงโครงสร้างคำสั่งขนาด 10 ไบต์

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				RelAdr
2	Logical Block Address (if required) (MSB)							
3	Logical Block Address (if required)							
4	Logical Block Address (if required)							
5	Logical Block Address (if required) (LSB)							
6	Reserved							
7	Reserved							
8	Reserved							
9	Transfer Length (if required) (MSB)							
10	Transfer Length (if required) (LSB)							
11	Control Byte							

ตารางที่ 2.8 แสดงโครงสร้างคำสั่งขนาด 12 ไบต์

6. โครงสร้างของคำสั่งสกาสซี่

คำสั่งสกาสซี่ที่กำหนดโดย ANSI มี 3 ขนาด คือ

- คำสั่งขนาด 6 ไบต์ (six-byte commands) ตารางที่ 2.6
- คำสั่งขนาด 10 ไบต์ (ten-byte commands) ตารางที่ 2.7
- คำสั่งขนาด 12 ไบต์ (twelve-byte commands) ตารางที่ 2.8

โดยคำสั่งทั้ง 3 ขนาดมีส่วนประกอบในลักษณะเดียวกัน คือ

6.1 รหัสทำงาน (Operation code) ส่วนนี้จะเป็นไบต์แรกของคำสั่ง โดยแบ่งออกเป็น 2 ส่วน คือ

6.1.1 รหัสกลุ่มคำสั่ง (Group code) ใช้เนื้อที่ 3 บิต คือบิตที่ 5-7 โดยกำหนดไว้ 8 กลุ่มคำสั่ง (คำสั่งที่ 00h - 07h)

6.1.1 รหัสคำสั่ง (command code) ใช้เนื้อที่ 5 บิต คือบิตที่ 0-4 โดยกำหนดไว้ 32 คำสั่ง ต่อ 1 กลุ่มคำสั่ง (คำสั่งที่ 00h - 1Fh)

โดยกำหนดชัดเจนดังนี้ว่า

คำสั่งกลุ่มที่ 0 เป็นคำสั่งขนาด 6 ไบต์
คำสั่งกลุ่มที่ 1 เป็นคำสั่งขนาด 10 ไบต์
คำสั่งกลุ่มที่ 2 สำรองไว้ใช้ในอนาคต
คำสั่งกลุ่มที่ 3 สำรองไว้ใช้ในอนาคต
คำสั่งกลุ่มที่ 4 สำรองไว้ใช้ในอนาคต
คำสั่งกลุ่มที่ 5 เป็นคำสั่งขนาด 12 ไบต์
คำสั่งกลุ่มที่ 6 สำรองไว้สำหรับผู้ผลิตใช้เองตามต้องการ
คำสั่งกลุ่มที่ 7 สำรองไว้สำหรับผู้ผลิตใช้เองตามต้องการ

Bit Byte	7	6	5	4	3	2	1	0
0	Group Code			Command Code				

ตารางที่ 2.9 แสดงโครงสร้างรหัสทำงาน

6.2 ส่วนระบุหมายเลขโลจิคัลยูนิต (logical unit number)

ใช้เนื้อที่ 3 บิต ของไบต์ที่สอง คือ บิตที่ 5-7 เพื่อระบุหมายเลขโลจิคัลยูนิตของอุปกรณ์เป้าหมาย ซึ่งอาจเป็นอุปกรณ์จริง (physical device) หรืออุปกรณ์เสมือน (virtual device) ส่วนนี้จะใช้ในกรณีที่อุปกรณ์สกาสซี่ไม่สามารถรับข้อความ IDENTIFY ในกรณีอุปกรณ์สกาสซี่สามารถรับข้อความ IDENTIFY อุปกรณ์นั้นจะใช้หมายเลขโลจิคัลยูนิตที่ข้อความ IDENTIFY แทนโดยละเลยต่อหมายเลขโลจิคัลยูนิตของคำสั่ง

ข้อควรปฏิบัติที่ดี คือ ไม่ว่าอุปกรณ์สกาสซี่จะรับหรือไม่รับข้อความ IDENTIFY ควรจะใส่หมายเลขโลจิคัลยูนิตของส่วนคำสั่งทุกคำสั่งให้ตรงกับหมายเลขโลจิคัลยูนิตของข้อความ IDENTIFY

6.3 ตำแหน่งโลจิคัลบล็อคที่คำสั่งทำงานด้วย (Logical Block Address)

โลจิคัลบล็อคของอุปกรณ์สกาสซี่ทุกอุปกรณ์จะเริ่มด้วยหมายเลข 0 ต่อเนื่องกันไปจนถึงหมายเลขสูงสุดที่มีของอุปกรณ์ การอ้างอิงถึงตำแหน่งของโลจิคัลบล็อค ในอุปกรณ์สกาสซี่ของคำสั่งแต่ละกลุ่ม จะมีความสามารถในการอ้างไม่เท่ากัน คือ

คำสั่งกลุ่มที่ 0 (คำสั่งขนาด 6 ไบต์) ได้เตรียมเนื้อที่ในคำสั่งสำหรับการอ้างตำแหน่งโลจิคัลบล็อคไว้ 21 บิต (อ้างได้ 2^{21} บล็อค)

คำสั่งกลุ่มที่ 1,5 (คำสั่งขนาด 10 และ 12 ไบต์) ได้เตรียมเนื้อที่ในคำสั่งสำหรับการอ้างตำแหน่งโลจิคัลบล็อคไว้ 32 บิต (อ้างได้ 2^{32} บล็อค)

อุปกรณ์ที่สื่อสารกัน จะต้องทราบขนาดของบล็อคข้อมูลที่จะสื่อสารกัน (ทราบได้ด้วยการใช้คำสั่ง READ CAPACITY หรือ MODE SENSE) เพื่อความสะดวกในการเตรียมเนื้อที่หน่วยความจำ

6.4 บิตแสดงตำแหน่งข้อมูลสัมพันธ์ (Relative Address Bit)

บิตนี้จะมีในคำสั่งกลุ่มที่ 1,5 (คำสั่งขนาด 10,12 ไบต์) โดยจะใช้เนื้อที่บิตที่ 0 ของไบต์ที่สอง การกำหนดให้บิตนี้เป็น 1 จะหมายความว่าข้อมูลตำแหน่งโลจิคัลบล็อคในส่วนของคำสั่งนี้เป็นส่วนเติมต่อที่ 2 (two's complement displacement) การคำนวณหาตำแหน่ง โลจิคัลบล็อคที่คำสั่งปัจจุบันจะทำงานด้วย ทำได้โดยการนำเอาส่วนเติมต่อที่ 2 ไปรวมเข้ากับตำแหน่งโลจิคัลบล็อคที่อุปกรณ์เป้าหมายนั้นทำงานด้วยล่าสุด

ลักษณะการคำนวณตำแหน่งโลจิคัลบล็อค แบบนี้ใช้ในการสร้างงานเป็นชุดคำสั่ง โดยคำสั่งที่ทำงานก่อนหน้าคำสั่งที่มีการกำหนดบิตนี้ให้เป็น 1 ได้ทำงานกับตำแหน่งโลจิคัลตำแหน่งใดตำแหน่งหนึ่งในอุปกรณ์เป้าหมายแล้ว

6.5 ขนาดของข้อมูลที่คำสั่งทำงานด้วย (Transfer Length)

ขนาดของข้อมูลที่คำสั่งทำงานด้วย หมายถึงขนาดของข้อมูลที่มีการโอนย้าย

กัน ระหว่างอุปกรณ์ปกติแล้วจะใช้หน่วยเป็นจำนวนบล็อคของข้อมูล ซึ่งแตกต่างกันไปตามแต่ละคำสั่ง คือ

คำสั่งที่มีเนื้อที่สำหรับจำนวนบล็อคข้อมูล 1 ไบต์ จะยอมให้มีการโอนย้ายข้อมูลไม่เกิน 256 บล็อค โดยถ้ากำหนดค่าจำนวนบล็อคเป็น 0 หมายถึงจะโอนย้าย 256 บล็อคข้อมูล

คำสั่งที่มีเนื้อที่สำหรับจำนวนบล็อคข้อมูล 2 ไบต์ จะยอมให้มีการโอนย้ายข้อมูลไม่เกิน 65,535 บล็อค ในกรณีที่ค่าจำนวนบล็อคเป็น 0 หมายถึง จะไม่มีการโอนย้ายข้อมูล

มีบางคำสั่งที่กำหนดเนื้อที่สำหรับขนาดข้อมูลที่จะโอนย้ายมากกว่า 2 ไบต์ เช่น คำสั่ง MODE SENSE เป็นต้น

ชื่อที่ใช้เรียกส่วนนี้ของคำสั่งแตกต่างกันไปในแต่ละคำสั่ง เช่น

ในคำสั่งที่ส่งพารามิเตอร์ (parameters) ไปยังอุปกรณ์เป้าหมายจะใช้ชื่อสำหรับเรียกส่วนนี้ว่า parameter list length

ในคำสั่งที่สั่งให้อุปกรณ์เป้าหมายส่งค่าพารามิเตอร์ ของอุปกรณ์กลับมายังอุปกรณ์เริ่มต้นการสื่อสารจะใช้ชื่อว่า ALLOCATION LENGTH ซึ่งหมายถึงเนื้อที่ที่อุปกรณ์เริ่มต้นการสื่อสารเตรียมไว้ให้สำหรับข้อมูลที่อุปกรณ์เป้าหมายจะส่งคืนมา ในกรณีนี้อุปกรณ์เป้าหมายจะส่งข้อมูลจำนวนเท่ากับ ALLOCATION LENGTH หรือ เท่ากับจำนวนข้อมูลที่มีอยู่

6.6 ไบต์ควบคุม (Control Byte) ตารางที่ 2.10

จะเป็นไบต์สุดท้ายที่กำกับติดไปในทุกคำสั่ง ซึ่งมีส่วนประกอบดังนี้

Bit Byte	7	6	5	4	3	2	1	0
Last	Vendor unique		Reserved				Flag	Link

ตารางที่ 2.10 แสดงโครงสร้างไบต์ควบคุม

Bit	คำอธิบาย
7 - 6	ขึ้นกับการกำหนดของบริษัทผู้ผลิต
5 - 2	สำรองไว้ใช้งานในอนาคต
1	Flag bit - การใช้งานจะใช้คู่กับ Link bit โดย Flag bit จะมีความหมายถ้า Link bit มีค่าเป็น 1 กล่าวคือ ถ้า Link bit เป็น 1 และ Flag bit เป็น 0 เมื่ออุปกรณ์เป้าหมายทำงานจบแต่ละคำสั่งจะส่งข้อความ LINKED COMMAND COMPLETE ถ้า Link bit เป็น 1 และ Flag bit เป็น 1 เมื่ออุปกรณ์เป้าหมายทำงานจบแต่ละคำสั่งจะส่งข้อความ LINKED COMMAND COMPLETE WITH FLAG
0	Link bit - การใช้งานบิตนี้ เพื่อแสดงว่าคำสั่งนี้เป็นชุดคำสั่ง (Linked command) ถ้า Linked bit เป็น 1 เมื่อจบการทำงานแต่ละคำสั่งในชุดคำสั่ง อุปกรณ์เป้าหมายจะส่งสถานะการทำงาน INTERMEDIATED กลับคืนไปยังอุปกรณ์เริ่มต้นการสื่อสารข้อมูล พร้อมทั้งข้อความอย่างใดอย่างหนึ่งดังที่ระบุใน Flag bit

บิตที่ 0 และ 1 จะใช้ในคำสั่งที่ส่งถึงอุปกรณ์เป้าหมายที่สามารถรับชุดคำสั่ง (linked command) ได้ ถ้าอุปกรณ์เป้าหมายไม่สามารถรับชุดคำสั่งและได้รับคำสั่งที่มีการกำหนดค่าให้บิตใดบิตหนึ่งหรือทั้งสองบิต จะส่งคืนสถานะการทำงาน CHECK CONDITION และถ้าอุปกรณ์สามารถส่งเอกเทนเซนซ์คาด้า จะส่งค่าเอกเทนเซนซ์คาด้าเป็น ILLEGAL REQUEST

ตารางที่ 2.11 แสดงคำอธิบายบิตต่างๆ ในไบต์ควบคุม

6.7 คำสั่งสกาสซี่ได้ถูกจัดแบ่งตามประเภทอุปกรณ์ออกเป็น 6 ประเภทคำสั่ง ซึ่งมีรายละเอียดโครงสร้างคำสั่งและการทำงานแตกต่างกันไป ในส่วนการทำวิทยานิพนธ์นี้ได้ใช้คำสั่งในชุดที่ 6 คือคำสั่งสำหรับอุปกรณ์ในลักษณะอ่านได้อย่างเดียวและสามารถเข้าได้โดยตรง (Read Only Direct Access Devices) ซึ่งจะแสดงรายละเอียดทุกคำสั่งต่อไป สำหรับ

คำสั่งในชุดอื่น สามารถศึกษาเพิ่มเติมได้ใน เอกสารกำหนดมาตรฐาน ANSI X3.131-1986 "Small Computer System Interface (SCSI)"

คำสั่งสำหรับอุปกรณ์แบบเข้าถึงโดยตรงที่อ่านข้อมูลได้อย่างเดียว คือ

6.7.1 คำสั่งกลุ่มที่ 0

ในส่วนของวิทยานิพนธ์ฉบับนี้จะขอกล่าวถึงเฉพาะคำสั่งที่ทำงานได้กับระบบปฏิบัติงาน SCO UNIX SYSTEM V/386 ฉบับที่ 3.2.0 รุ่น 1.0 เท่านั้น

6.7.1.1 คำสั่ง TEST UNIT READY รหัส 00h ชนิด Optional ตารางที่ 2.12

หน้าที่ ใช้เพื่อทดสอบว่าอุปกรณ์เป้าหมายตามหมายเลขโลจิคัลที่ระบุพร้อมสำหรับการทำงานหรือไม่ คำสั่งนี้ไม่ได้สั่งให้อุปกรณ์เป้าหมายทดสอบตัวเอง (self test) เพียงแต่ส่งคำสั่งบางคำสั่งที่เข้าใช้ข้อมูลในอุปกรณ์ ถ้าอุปกรณ์เป้าหมายรับและทำงานโดยไม่ส่งสถานะ CHECK CONDITION (ส่งสถานะ GOOD) จะแสดงว่าอุปกรณ์เป้าหมายนั้นพร้อมที่จะทำงานต่อได้

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				
2	Reserved							
3	Reserved							
4	Reserved							
5	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.12 แสดงโครงสร้างคำสั่ง TEST UNIT READY

6.7.1.2 คำสั่ง REZERO UNIT รหัส 01h ชนิด Optional ตารางที่ 2.13

หน้าที่ เพื่อกำหนดอุปกรณ์เป้าหมายให้อยู่ในสภาวะที่เฉพาะเจาะจงสภาวะหนึ่ง ตามที่กำหนดโดยผู้ผลิตอุปกรณ์สภาวะนั้น

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				
2	Reserved							
3	Reserved							
4	Reserved							
5	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.13 แสดงโครงสร้างคำสั่ง REZERO UNIT

6.7.1.3 คำสั่ง REQUEST SENSE รหัส 03h ชนิด Mandatory ตารางที่ 2.14

หน้าที่ เพื่อขอสถานะการทำงานเพิ่มเติมจากอุปกรณ์เป้าหมาย โดยสถานะการทำงานเพิ่มเติม จะใช้ชื่อว่า เซนซ์ดาต้า (SENSE DATA)

เซนซ์ดาต้า จะถูกเก็บไว้ในอุปกรณ์เป้าหมาย เพื่อรอคำสั่ง REQUEST SENSE จากอุปกรณ์เริ่มต้นสื่อสาร คำสั่งอื่นที่ไม่ใช่ REQUEST SENSE จะทำให้ เซนซ์ดาต้าถูกลบไป

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				
2	Reserved							
3	Reserved							
4	Allocation Length							
5	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.14 แสดงโครงสร้างคำสั่ง REQUEST SENSE

ALLOCATION LENGTH คือจำนวนเนื้อที่หน่วยความจำที่อุปกรณ์เริ่มต้นการสื่อสารเตรียมไว้สำหรับ เซนซ์ดาต้า ถ้ามีค่าเป็น 0 หมายถึงอุปกรณ์เริ่มต้นการสื่อสารเตรียมเนื้อที่ไว้ให้ 4 ไบต์ อุปกรณ์เป้าหมายจะส่งเซนซ์ดาต้า เท่าจำนวนเนื้อที่ที่อุปกรณ์เริ่มต้นการสื่อสารเตรียมไว้ หรือจนกว่าจะหมดเซนซ์ดาต้า

ถ้าเกิดความผิดพลาดขึ้นเนื่องจากการทำงานตามคำสั่ง REQUEST SENSE อุปกรณ์เป้าหมายจะดำเนินการ ดังนี้

1. ถ้าเป็นความผิดพลาดรุนแรง (fatal error) อุปกรณ์เป้าหมายจะไม่สามารถส่งเซนซ์ดาต้าที่ถูกต้องแก่อุปกรณ์เริ่มต้นการสื่อสาร ความผิดพลาดรุนแรง ได้แก่
   - ส่วนของคำสั่งที่เป็นเนื้อที่สำรอง (reserved) ไม่เป็น 0
   - มีพาริตี้ผิดพลาดในสายสัญญาณข้อมูล (DATA BUS)
   - อุปกรณ์เป้าหมายทำงานผิดพลาด ไม่สามารถส่งคืน เซนซ์ดาต้าได้
2. ถ้าเป็นความผิดพลาดไม่รุนแรง (nonfatal error) อุปกรณ์เป้าหมายจะ

ยังคงสามารถส่งเซนซ์คาด้า และสถานะการทำงาน GOOD กลับไปยังอุปกรณ์เริ่มต้นการสื่อสาร

อุปกรณ์เป้าหมายอาจจะสามารถส่งนอนเอกเทนเซนซ์คาด้า (nonextended sense data) หรือ เอกเทนเซนซ์คาด้า (extended sense data) หรือ สามารถส่งได้ทั้ง 2 ประเภท

(หมายเหตุ ในกรณีที่อุปกรณ์เป้าหมายสามารถส่งเซนซ์คาด้าได้ทั้ง 2 แบบ อุปกรณ์เป้าหมายควรจะส่ง นอนเอกเทนเซนซ์คาด้า ในกรณีที่คำสั่ง REQUEST SENSE มีค่า ALLOCATION LENGTH เป็น 0 โดยปกติแล้วการเลือกใช้รูปแบบของเซนซ์คาด้า ขึ้นกับระดับชั้นของความผิดพลาด (error class) กล่าวคือ ระดับชั้นของความผิดพลาด 0-6 (ระดับชั้นของความผิดพลาด 0-6 กำหนดโดยบริษัทผู้ผลิตอุปกรณ์) จะใช้นอนเอกเทนเซนซ์คาด้า ในกรณีระดับความผิดพลาดเป็น 7 จะใช้เอกเทนเซนซ์คาด้า)

นอนเอกเทนเซนซ์คาด้า (Nonextended Sense) แสดงในตารางที่ 2.15

Bit Byte	7	6	5	4	3	2	1	0
0	AdValid	Error Class			Error Code			
1	Vendor Unique				Logical Block Address (MSB)			
2					Logical Block Address			
3					Logical Block Address (LSB)			

ตารางที่ 2.15 แสดงโครงสร้างนอนเอกเทนเซนซ์คาด้า

บิตแสดงความถูกต้องของตำแหน่งข้อมูล (address valid bit-Advalid) ถ้าเป็น 1 แสดงว่าข้อมูลที่ปรากฏในเลขหมายโลจิคัลบล็อค มีความถูกต้องและสัมพันธ์กับรหัสแสดงความผิดพลาด

ระดับชั้นความผิดพลาด มีค่า 0-6 กำหนดโดยบริษัทผู้ผลิตอุปกรณ์

รหัสของความผิดพลาด กำหนดโดยบริษัทผู้ผลิต

Bit Byte	7	6	5	4	3	2	1	0
0	Valid	Error Class (7)			Error Code (0)			
1	Segment Number							
2	Filemark	EOM	ILI	Reserved	Sense Key			
3	Information Byte (MSB)							
4	Information Byte							
5	Information Byte							
6	Information Byte (LSB)							
7	Additional Sense Length (n)							
8- n+7	Additional Sense Bytes							

ตารางที่ 2.16 แสดงโครงสร้างเอกเทนเซนซ์คาต้า

เอกเทนเซนซ์คาต้า (Extended Sense) ตารางที่ 2.16

- บิตแสดงความถูกต้อง (Valid bit) ถ้ามีค่าเป็น 0 แสดงว่า ไบต์ข้อมูล (Information bytes)จะไม่ถูกต้อง ถ้ามีค่าเป็น 1 แสดงว่าไบต์ข้อมูลจะบรรจุข้อมูลดังต่อไปนี้
- ถ้าเป็นเซนซ์คาต้าสำหรับคำสั่งของอุปกรณ์ประเภทเข้าถึงได้โดยตรง (direct

access device) อุปกรณ์ที่เขียนได้ครั้งเดียวแต่อ่านได้มากครั้ง (write-once read-multiple devices) อ่านได้อย่างเดียว (read-only direct-access devices) ไบต์ข้อมูลจะเป็นเลขหมายโลจิคัลบล็อคของข้อมูลที่มีความสัมพันธ์กับเซนซ์คาต้า

- ถ้าเป็นเซนซ์คาต้าสำหรับอุปกรณ์ประเภทเข้าถึงแบบตามลำดับ (sequential access devices) เครื่องพิมพ์ (printer devices) อุปกรณ์ประมวลผล (processor devices) ไบต์แสดงข้อมูลจะเป็นความแตกต่างของขนาดข้อมูลที่ขอกับขนาด ข้อมูลจริง ทั้งในหน่วยของไบต์และบล็อคข้อมูล (ค่าลบแสดงในรูปของส่วนต่อเติมที่ 2 two's complement)

- ถ้าเป็นเซนซ์คาต้าสำหรับคำสั่ง COPY, COMPARE, COPY AND VERIFY ไบต์แสดงข้อมูลจะเป็นความแตกต่างของจำนวนบล็อคข้อมูลที่ขอ และจำนวนบล็อคข้อมูลจริงที่ได้มีการคัดลอก (ด้วยคำสั่ง COPY, COPY AND VERIFY) หรือเปรียบเทียบ (COMPARE)

- ระดับชั้นความผิดพลาด (Error class) จะเป็น 7 เสมอ สำหรับเอกเทนเซนซ์คาต้า

- รหัสความผิดพลาด (Error code)

ถ้าเป็น 0 แสดงว่า เซนซ์คาต้าอยู่ในรูปแบบนอนเอกเทนเซนซ์คาต้า

ถ้าเป็น Fh แสดงว่า เซนซ์คาต้าอยู่ในรูปแบบนอนเอกเทนเซนซ์คาต้า ที่เฉพาะของบริษัทผู้ผลิตอุปกรณ์

ถ้าเป็น 1h-Eh แสดงว่า เป็นรหัสสำรองไว้ใช้ในอนาคต

- เลขหมายเซกเมนต์ (Segment number)

แสดงเลขหมายเซกเมนต์เดสคริปเตอร์ปัจจุบัน (current segment descriptor) ถ้าเป็นเอกเทนเซนซ์ สำหรับคำสั่ง COPY, COMPARE, COPY AND VERIFY ค่าหมายเลขเซกเมนต์มีได้ตั้งแต่ 0-255 ทั้งหมด 256 เซกเมนต์

- บิตไฟล์มาร์ค (Filemark bit)

บิตนี้ให้ระบุว่าคำสั่งปัจจุบันได้อ่านไฟล์มาร์ค (file mark) บิตนี้ใช้สำหรับอุปกรณ์ที่มีการเข้าถึงแบบตามลำดับ (segmential access device)

- บิตเอนออฟมีเดียม (End of medium bit)

บิตนี้ใช้ระบุสภาวะของการหมดเนื้อที่ของการทำงาน (end of medium condition) ใช้สำหรับอุปกรณ์ประเภทเข้าถึงได้แบบตามลำดับ หรือประเภทเครื่องพิมพ์

- บิตแสดงความผิดพลาดของขนาดข้อมูล (incorrect length indicator ILI)

บิตนี้จะใช้แสดงว่า ขนาดข้อมูลของโลจิคัลบล็อคที่ระบุมาในคำสั่ง ไม่สัมพันธ์กับขนาดข้อมูลของโลจิคัลบล็อคจริงในอุปกรณ์

- เซนซ์คีย์ (SENSE KEY)

  เป็นรหัสแสดงความผิดพลาดต่าง ๆ ตามตารางที่ 2.17, 2.18, 2.19

- ขนาดข้อมูลเซนซ์เพิ่มเติม (Addition Sense Length)

- ขนาดข้อมูลเซนซ์เพิ่มเติมระบุจำนวนของเซนซ์ไบต์ที่ต่อท้ายเอกเทนเซนซ์คาต้า ขนาดของข้อมูลนี้จะเป็นไปตามขนาดข้อมูลเซนซ์เพิ่มเติมโดยไม่ขึ้นกับ ALLOCATION LENGTH ที่ระบุไว้ในคำสั่ง REQUEST SENSE

- ข้อมูลเซนซ์เพิ่มเติม (Addition Sense bytes)

  เป็นไปตามการกำหนดของผู้ผลิตอุปกรณ์แต่ละราย

Sense Key	Description
0h	NO SENSE. เซนซ์คีย์นี้เป็นการระบุว่าโลจิคัลยูนิตนี้ไม่มีเซนซ์คีย์ ซึ่งอาจเป็นผลมาจากการทำงานตามคำสั่งได้สมบูรณ์ หรือ ความผิดพลาดเนื่องจากมีการกำหนดให้บิตข้อมูล filemark , EOM , ILI เป็น 1
1h	RECOVERED ERROR. เซนซ์คีย์นี้เป็นการระบุว่าโลจิคัลยูนิตนี้ได้ทำงานตามคำสั่งสมบูรณ์โดยในขณะที่ทำงานตามคำสั่งพบความผิดพลาด บางอย่างที่แก้ไขได้แล้ว ข้อมูลความผิดพลาดตรวจสอบได้จากข้อมูลเซนซ์เพิ่มเติม
2h	NOT READY. เซนซ์คีย์นี้เป็นการระบุว่าการอ้างถึงตำแหน่งข้อมูลในโลจิคัลยูนิตไม่ถูกต้อง
3h	MEDIUM ERROR. เซนซ์คีย์นี้เป็นการระบุว่าการทำงานตามคำสั่งสิ้นสุดลงเนื่องจากมีความผิดพลาดที่ไม่สามารถแก้ไขได้ ซึ่งอาจเกิดขึ้นจากมีปัญหาที่สื่อ เช่น สื่อมีสภาพเปลี่ยน หรือ การบันทึกข้อมูลผิดพลาด
4h	HARDWARE ERROR. เซนซ์คีย์นี้เป็นการระบุว่ามีความผิดพลาดที่ไม่สามารถแก้ไขได้ เกิดขึ้นขณะทำงานตามคำสั่งหรือขณะอุปกรณ์ทดสอบตัวเอง ซึ่งมีสาเหตุมาจากอุปกรณ์

ตารางที่ 2.17 แสดงรหัสของเซนซ์คีย์รหัส 0h - 4h (sense key)

Sense Key	Description
5h	ILLEGAL REQUEST. เซนซ์คีย์นี้เป็นการระบุว่ามีข้อมูลที่ไม่ถูกต้องในโครงสร้างของคำสั่ง หรือ ในพารามิเตอร์ของคำสั่ง การทำงานนั้นถ้าข้อมูลที่ไม่ถูกต้องปรากฏในโครงสร้างของคำสั่ง อุปกรณ์เป้าหมายจะไม่เปลี่ยนแปลงสภาพสื่อ แต่ถ้าข้อมูลที่ไม่ถูกต้องปรากฏในพารามิเตอร์ของคำสั่ง อุปกรณ์เป้าหมายอาจเปลี่ยนแปลงสื่อไปแล้ว
6h	UNIT ATTENTION. เซนซ์คีย์นี้เป็นการระบุว่ามีการเปลี่ยนแปลงสื่อเกิดขึ้น หรือ อุปกรณ์เป้าหมายได้รับสัญญาณรีเซต
7h	DATA ATTENTION. เซนซ์คีย์นี้เป็นการระบุว่าคำสั่งที่เกี่ยวข้องกับการเขียน หรือ อ่านข้อมูล ได้ทำกับบล็อคข้อมูลที่ถูกระบุป้องกันการเขียนอ่าน ดังนั้นจะไม่มีการเขียนหรืออ่านข้อมูลเกิดขึ้น
8h	BLANK CHECK. เซนซ์คีย์นี้เป็นการระบุว่าอุปกรณ์ประเภทเขียนได้ครั้งเดียวแต่อ่านได้มาก หรือ อุปกรณ์ประเภทเข้าถึงข้อมูลตามลำดับ พบบล็อคข้อมูลว่างขณะที่มีการอ่านข้อมูล หรือ อุปกรณ์ประเภทเขียนได้ครั้งเดียวแต่อ่านได้มากครั้งพบบล็อคข้อมูลที่ไม่ว่างขณะกำลังเขียนข้อมูล
9h	Vendor unique. เซนซ์คีย์นี้เป็นการระบุเซนซ์คีย์ของแต่ละบริษัทผู้ผลิต
Ah	COPY ABORTED. เซนซ์คีย์เป็นการระบุว่าคำสั่ง COPY , COMPARE หรือ COPY AND VERIFY สิ้นสุดการทำงานลงเนื่องจากเกิดความผิดพลาดที่อุปกรณ์ต้นทาง หรือ อุปกรณ์ปลายทาง หรือ ทั้งสองอุปกรณ์
Bh	ABORTED COMMAND. เซนซ์คีย์เป็นการระบุว่าอุปกรณ์เป้าหมายยุติการทำงานการสั่งให้ทำงานคำสั่งนี้ใหม่โดยอุปกรณ์เริ่มต้นการสื่อสารอาจจะช่วยแก้ไขได้

ตารางที่ 2.18 แสดงรหัสของเซนซ์คีย์รหัส 5h - Bh (sense key)

Sense Key	Description
Ch	EQUAL. เซนซ์คีย์เป็นการระบุว่าคำสั่งในการค้นหาข้อมูลได้พบข้อมูลที่ตรงตามเงื่อนไข
Dh	VOLUMNE OVERFLOW. เซนซ์คีย์เป็นการระบุว่ามีข้อมูลเหลือค้างในบัฟเฟอร์ โดยไม่สามารถเขียนลงสู่อุปกรณ์เป้าหมายได้อีก เนื่องจากสื่อบนอุปกรณ์เป้าหมายเต็ม อาจจะใช้คำสั่ง RECOVER BUFFERED DATA เพื่ออ่านข้อมูลคืนจากบัฟเฟอร์
Eh	MISCOMPARE. เซนซ์คีย์เป็นการระบุว่าข้อมูลดั้งต้นไม่เท่ากับข้อมูลที่อ่านจากสื่อ
Fh	สำรองไว้ใช้ในอนาคต

ตารางที่ 2.19 แสดงรหัสของเซนซ์คีย์รหัส Ch - Fh (sense key)

6.7.1.4 คำสั่ง READ รหัส 08h ชนิด Optional

(ตารางที่ 2.20)

หน้าที่ เพื่อให้อุปกรณ์เป้าหมายโอนถ่ายข้อมูลให้แก่อุปกรณ์เริ่มต้นการสื่อสาร

- เลขหมายโลจิคัลบล็อคหมายถึง โลจิคัลบล็อคที่คำสั่ง READ จะเริ่มต้นโอนถ่ายข้อมูล

- ขนาดข้อมูลที่จะโอนถ่าย (transfer length) จะระบุจำนวนโลจิคัลบล็อคที่ต่อเนื่องกันที่จะถูกโอนถ่ายโดยคำสั่ง READ ถ้าระบุขนาดข้อมูลเป็น 0 หมายถึงขอโอนถ่ายข้อมูล 256 โลจิคัลบล็อค ถ้าขนาดข้อมูลเป็นอย่างอื่นก็หมายถึงจำนวนบล็อคเท่านั้นที่จะถูกโอนถ่าย

ถ้ามีการขอใช้อุปกรณ์เป้าหมายหรือส่วนของอุปกรณ์เป้าหมาย (Extent of logical unit) ที่ถูกสำรองไว้ใช้เป็นการส่วนตัวโดยอุปกรณ์สกาสซี่อื่น อุปกรณ์

เป้าหมายจะส่งสถานะ RESERVATION CONFLICT ให้แก่อุปกรณ์เริ่มต้นการสื่อสาร

เซนซ์คีย์ที่มีความสัมพันธ์กับคำสั่ง READ

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number		Logical Block Address (MSB)					
2	Logical Block Address							
3	Logical Block Address (LSB)							
4	Transfer Length							
5	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.20 แสดงโครงสร้างคำสั่ง READ

เซนซ์คีย์	เงื่อนไข
ILLEGAL REQUEST	เลขหมายโลจิคัลบล็อคไม่ถูกต้อง[1]
UNIT ATTENTION	อุปกรณ์เป้าหมายถูกรีเซตหรือมีการเปลี่ยนมัชฌิมกลาง (medium) ของอุปกรณ์เป้าหมายหลังจากการทำงานตามคำสั่งล่าสุด
MEDIUM ERROR	มีความผิดพลาดจากการอ่านข้อมูลที่แก้ไขไม่ได้
ABORTED COMMAND	ความผิดพลาดที่สามารถจะแก้ไขโดยการสั่งให้ทำงานคำสั่งเดิมซ้ำ
BLANK CHECK	มีการพยายามอ่านบล็อคว่างหรือบล็อคที่ยังไม่ได้มีการเขียนข้อมูล[2]

หมายเหตุ (1) ไบต์แสดงข้อมูล ของเอกเทนเซนซ์จะถูกกำหนดเป็นเลขหมายโลจิคัลบล็อคแรกที่ไม่ถูกต้อง

(2) ไบต์แสดงข้อมูลของเอกเทนเซนซ์จะถูกกำหนดเป็นเลขหมายโลจิคัลบล็อคแรกที่เป็นที่ว่าง ข้อมูลเฉพาะที่อยู่ก่อนโลจิคัลบล็อคที่เป็นที่ว่างเท่านั้นจะถูกโอนถ่ายไป

6.7.1.5 คำสั่ง SEEK รหัส 0Bh ประเภท Optional

(ตารางที่ 2.21)

หน้าที่ เพื่อให้อุปกรณ์เป้าหมายแสวงหา (seek) เลขหมายโลจิคัลบล็อคที่ระบุไว้ในคำสั่ง

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Logical Block Address (MSB)				
2	Logical Block Address							
3	Logical Block Address (LSB)							
4	Reserved							
5	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.21 แสดงโครงสร้างคำสั่ง SEEK

6.7.1.6 คำสั่ง INQUIRY รหัส 11h ประเภท Extended

(ตารางที่ 2.22)

<u>หน้าที่</u> เพื่อให้อุปกรณ์เป้าหมายส่งข้อมูลพารามิเตอร์ต่าง ๆ ทั้งของอุปกรณ์เป้าหมายและของอุปกรณ์รอบข้างที่ต่อกับอุปกรณ์เป้าหมาย ไปยังอุปกรณ์เริ่มต้นการสื่อสารข้อมูล

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				
2	Reserved							
3	Reserved							
4	Allocation Length							
5	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.22 แสดงโครงสร้างคำสั่ง INQUIRY

- ALLOCATION LENGTH นั้น เป็นการระบุเนื้อที่หน่วยความจำ(ไบต์สูงสุด) ที่ อุปกรณ์เริ่มต้นการสื่อสารเตรียมไว้สำหรับรองรับข้อมูลพารามิเตอร์ การกำหนด ALLOCATION LENGTH เป็น 0 แสดงว่าคำสั่งนี้ไม่ต้องการข้อมูลส่งกลับจากอุปกรณ์เป้าหมาย(ไม่ถือว่าคำสั่งผิด)

มาตรฐาน ANSI ระบุว่าอุปกรณ์เป้าหมายควรจะส่งข้อมูลพารามิเตอร์ได้ตลอด ถึงแม้ว่าอุปกรณ์จะไม่สามารถทำงานตามคำสั่งสถานะชนิดอื่น ๆ ก็ตาม

ในกรณีที่อุปกรณ์เป้าหมายอยู่ในสภาวะเอดเทนชั่น (UNIT ATTENTION) ซึ่งเกิดจากอุปกรณ์ได้รับสัญญาณรีเซต หรือมีชนิดของอุปกรณ์ถูกเปลี่ยนไป อุปกรณ์เป้าหมายก็ควรจะทำงานคำสั่ง INQUIRY โดยการส่งข้อมูลพารามิเตอร์ และไม่ลบล้างสภาวะเอดเทนชั่นออก

ข้อมูลพารามิเตอร์ของอุปกรณ์จะถูกส่งกลับมาในโครงสร้างตาม ตารางที่ 2.23 ซึ่งมีข้อมูลส่วนต้น 5 ไบต์ และตามด้วยข้อมูลจำเพาะของแต่ละบริษัทผู้ผลิต

1. ประเภทของอุปกรณ์ (Peripheral Device Type) (ตารางที่ 2.24)
2. บิตแสดงสภาพการเปลี่ยนมัชฌิม (Removable medium bit RMB)

บิตนี้ถ้าเป็น 0 แสดงว่าไม่สามารถเปลี่ยนแปลงมัชฌิมได้

บิตนี้ถ้าเป็น 1 แสดงว่าสามารถเปลี่ยนแปลงมัชฌิมได้

3. ข้อมูลแสดงรหัสจำเพาะของชนิดอุปกรณ์ (Device-type qualifier)

มีอุปกรณ์สกาสซี่บางอุปกรณ์ที่มีลักษณะเช่นนี้ เพื่อให้เกิดความสะดวกแก่โปรแกรมขึ้น (อาจกำหนดโดยใช้สวิทช์ที่อุปกรณ์ หรือโดยวิธีการอื่น) เพื่อให้เกิดความสะดวกแก่โปรแกรมคุมอุปกรณ์ ที่มีความสามารถในการเรียนรู้ลักษณะอุปกรณ์ได้เอง (self configuring software) ตรวจสอบว่าอุปกรณ์ที่ต่อที่เลขหมายโลจิคัลดังกล่าวเป็นอุปกรณ์ชนิดใด ซึ่งคุณลักษณะนี้มีประโยชน์อย่างมากสำหรับระบบที่มีมัชฌิมเปลี่ยนแปลงได้หลายชนิด (multiple types of removable medium)

สำหรับอุปกรณ์สกาสซี่ที่ไม่มีลักษณะเช่นนี้จะให้ค่า 0 ในข้อมูล

4. ข้อมูลแสดงความสอดคล้องกับมาตรฐานต่าง ๆ

สามารถระบุได้ว่าอุปกรณ์สกาสซี่นี้สอดคล้องตามมาตรฐานใด ซึ่งอาจจะสอดคล้องมากกว่าหนึ่งมาตรฐานก็ได้ มาตรฐานที่มีให้ระบุได้ คือ

มาตรฐานของ ISO (โดยองค์กร International Organization for Standardization)

มาตรฐานของ ECMA (โดยองค์กร The European Computer Manufacturers Association)

มาตรฐานของ ANSI (โดยองค์กร The American National Standard for Information System)

การสอดคล้องกับมาตรฐานใดแสดงโดย ค่าข้อมูลที่ไม่เป็น 0 ในบริเวณต่าง ๆ โดยค่าที่แสดงจะต้องนำไปตรวจสอบกับมาตรฐานต่าง ๆ เหล่านี้ เพื่อหารุ่นของมาตรฐานต่อไป สำหรับ ANSI แล้ว เลขหมายต่าง ๆ จะเป็นไปตาม ตารางที่ 2.25

5. ขนาดข้อมูลเพิ่มเติม (Addition Length)

แสดงขนาดข้อมูลเพิ่มเติมที่บริษัทผู้ผลิตกำหนดขึ้น และส่ง

ตามหลังข้อมูลของ INQUIRY ถ้าขนาด ALLOCATION LENGTH ในคำสั่ง INQUIRY กำหนดไว้น้อยกว่าข้อมูลทั้งหมดที่จะส่ง ข้อมูลจะถูกตัดทอนลง แต่ขนาดข้อมูลเพิ่มเติมจะยังคงมีค่าเท่าเดิม (ไม่ได้ถูกลดลงให้เท่ากับขนาดข้อมูลที่ส่งได้จริง)

Bit Byte	7	6	5	4	3	2	1	0
0	Peripheral Device Type							
1	RMB	Device-Type Qualifier						
2	ISO Version		ECMA Version			ANSI-Approved Version		
3	Reserved							
4	Additional Length (n)							
	Vendor Unique Parameters							
5- n+4	Vendor Unique Parameter Bytes							

ตารางที่ 2.23 แสดงโครงสร้างข้อมูลของคำสั่ง INQUIRY

Code	Description
00h	Direct-access device (e.g., magnetic disk)
01h	Sequential-access device (e.g., magnetic tape)
02h	Printer device
03h	Processor device
04h	Write-once read-multiple device (e.g., some optical disk)
05h	Read-only direct-access device (e.g., some optical disks)
06h-7Eh	Reserved
7Eh	Logical unit not present
80h-FFh	Vendor unique

ตารางที่ 2.24 แสดงรหัสประเภทอุปกรณ์ที่ปรากฏในโครงสร้างข้อมูลคำสั่ง INQUIRY

Code	Description
0h	Version is unspecified.
1h	This version. This code shall be used by SCSI devices that claim to comply with this standard.
2h-7h	Reserved

ตารางที่ 2.25 แสดงรหัสรุ่นของมาตรฐาน ANSI

6.7.1.7 คำสั่ง MODE SELECT รหัส 15h ประเภท Optional ตารางที่ 2.26

หน้าที่ เพื่อให้อุปกรณ์เริ่มต้นการสื่อสารระบุมัชฌิม เลขหมายโลจิคัล และค่าพารามิเตอร์ของอุปกรณ์ต่าง ๆ ไปยังอุปกรณ์เป้าหมายโดย

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				
2	Reserved							
3	Reserved							
4	Parameter List Length							
5	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.26 แสดงโครงสร้างคำสั่ง MODE SELECT

- ขนาดข้อมูลพารามิเตอร์ (parameter list length)

เป็นขนาดข้อมูล (หน่วยไบต์) ของพารามิเตอร์คำสั่ง MODE SELECT ที่จะส่งจากอุปกรณ์เริ่มต้นการสื่อสารข้อมูลไปยังอุปกรณ์เป้าหมาย อาจมีค่าเป็น 0 ได้ ซึ่งหมายความว่าไม่มีการส่งข้อมูลพารามิเตอร์ไปยังอุปกรณ์เป้าหมาย (ไม่ถือว่าผิด)

ข้อมูลพารามิเตอร์ประกอบด้วยข้อมูลส่วนต้น 4 ไบต์ และชุดข้อมูล แสดงลักษณะของบล็อคชุดละ 8 ไบต์ ซึ่งอาจไม่มี หรือมีกี่ชุดก็ได้ ตามด้วยข้อมูลจำเพาะของผู้ผลิต ตาราง 2.27

Bit	7	6	5	4	3	2	1	0
0	Reserved							
1	Reserved							
2	Reserved							EBC
3	Block Descriptor Length							
	Block Descriptor (s)							
0	Reserved							
1	Number of Blocks (MSB)							
2	Number of Blocks							
3	Number of Blocks (LBS)							
4	Reserved							
5	Block Length (MSB)							
6	Block Length							
7	Block Length (LSB)							
0-n	Vendor Unique(MSB) Parameter Byte(s)							

ตารางที่ 2.27 แสดงโครงสร้างพารามิเตอร์ของ MODE SELECT

ข้อมูลที่บรรจุในพารามิเตอร์ประกอบด้วย

1. บิตแสดงการตรวจสอบว่าข้อมูลว่างหรือไม่

ถ้าเป็น 0 จะไม่มีการตรวจสอบว่ามัชฌิมเป็นที่ว่างระหว่างการเขียนข้อมูลหรือไม่

ถ้าเป็น 1 จะมีการตรวจสอบถ้าพบว่ามัชฌิมไม่เป็นที่ว่างระหว่างการเขียนข้อมูล อุปกรณ์เป้าหมายจะหยุดการทำงานคำสั่ง และส่งสถานะการทำงาน CHECK CONDITION พร้อมกับค่าเซนซ์คีย์เป็น BLANK CHECK

สำหรับอุปกรณ์ที่อ่านได้อย่างเดียว บิตนี้ไม่ถูกใช้(สำรองไว้)

2. ขนาดของข้อมูลแสดงลักษณะบล็อค (The block descriptor length)

เป็นขนาดข้อมูลแสดงลักษณะบล็อคทั้งหมดที่ส่งมาพร้อมกับรายการพารามิเตอร์นี้ ซึ่งจะมีค่าเท่ากับจำนวนชุดข้อมูลแสดงลักษณะบล็อค คูณกับ 8 ไบต์ โดยค่านี้จะไม่รวมค่าพารามิเตอร์ เฉพาะของบริษัทผู้ผลิต ค่าขนาดข้อมูลอาจเป็น 0 ได้

แต่ละชุดของข้อมูลแสดงลักษณะบล็อค จะบ่งบอกถึงคุณลักษณะของมัชฌิมทั้งอุปกรณ์ซึ่งอาจจะมีลักษณะบล็อคข้อมูลได้หลายแบบ โดยแต่ละชุดจะแสดง

- จำนวนโลจิคัลบล็อคของข้อมูลที่มีขนาดเท่ากับขนาดบล็อคที่ระบุไว้ในชุดข้อมูลแสดงลักษณะบล็อคเดียวกัน
- ขนาดโลจิคัลบล็อค (เป็นไบต์)

6.7.1.8 คำสั่ง MODE SENSE รหัส 1Ah ประเภท Optional (ตารางที่ 2.28)

<u>หน้าที่</u> เป็นคำสั่งที่ส่งไปยังอุปกรณ์เป้าหมายเพื่อให้อุปกรณ์เป้าหมายรายงานข้อมูลเกี่ยวกับสภาพมัชฌิม เลขหมายโลจิคัล (logical unit) และพารามิเตอร์ของอุปกรณ์รอบข้างที่ต่อกับอุปกรณ์เป้าหมาย

คำสั่งนี้เป็นคำสั่งที่ทำงานตรงกันข้ามกับคำสั่ง MODE SELECT

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				
2	Reserved							
3	Reserved							
4	Allocation Length							
5	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.28 แสดงโครงสร้างคำสั่ง MODE SENSE

Bit	7	6	5	4	3	2	1	0
0	Sense Data Length							
1	Medium Type							
2	WP	Reserved						EBC
3	Block Descriptor Length							
	Block Descriptor (s)							
0	Reserved							
1	Number of Blocks (MSB)							
2	Number of Blocks							
3	Number of Blocks (LBS)							
4	Reserved							
5	Block Length (MSB)							
6	Block Length							
7	Block Length (LSB)							
	Vendor Unique Parameter(s)							
0-n	Vendor Unique Parameter(s) Parameter Byte(s)							

ตารางที่ 2.29 แสดงโครงสร้างพารามิเตอร์ของ MODE SENSE

ข้อมูลที่ระบุในคำสั่งคือ

- ขนาดเนื้อที่ที่เตรียมไว้ให้สำหรับข้อมูล (ALLOCATION LENGTH) ขนาดเนื้อที่อาจเป็น 0 ได้ ซึ่งแสดงว่าไม่ต้องการข้อมูลจากอุปกรณ์เป้าหมาย ในกรณีเป็นตัวเลขอื่นจะหมายถึงเนื้อที่สูงสุดที่เตรียมไว้สำหรับข้อมูลที่ส่งมาจากอุปกรณ์เป้าหมาย

ข้อมูลที่ส่งคืนมาจากคำสั่งนี้ประกอบด้วยข้อมูลส่วนต้น 4 ไบต์ ตามด้วยชุดของข้อมูลที่แสดงลักษณะบล็อค และข้อมูลพารามิเตอร์ที่จำเพาะของผู้ผลิตแต่ละราย ตามตารางที่ 2.29

ข้อมูลที่ส่งคืนจากคำสั่ง MODE SENSE ประกอบด้วย

1. ขนาดข้อมูลทั้งหมด (The sense data length) เป็นขนาดข้อมูลที่ส่งคืนจากคำสั่ง MODE SENSE ทั้งหมด โดยไม่รวมไบต์ของขนาดข้อมูล (เพราะฉะนั้นขนาดข้อมูลทั้งหมดจริงต้องเท่ากับ ขนาดข้อมูลที่ระบุบวกหนึ่ง)
2. รหัสแสดงประเภทของมัชฌิม (Code values for medium type) มีค่าดังนี้

   00h หมายถึง เป็นประเภทมัชฌิมตามค่าปริยาย (Default) และเพียงมัชฌิมชนิดเดียว

   01h - 7Fh สำรองไว้

   80h - FFh กำหนดโดยผู้ผลิตแต่ละราย
3. บิตแสดงการตรวจสอบว่าข้อมูลว่างหรือไม่ เหมือนกับใน MODE SELECT
4. บิตแสดงสถานะป้องกันการเขียนข้อมูล (Write protected bit -WP)

ถ้าบิตนี้เป็น 0 แสดงว่า สามารถเขียนข้อมูลเข้ามัชฌิมได้

ถ้าบิตนี้เป็น 1 แสดงว่า ไม่สามารถเขียนข้อมูลเข้ามัชฌิมได้บิตนี้ไม่ได้ใช้ในอุปกรณ์สกาสซี่ที่เป็นแบบอ่านข้อมูลได้อย่างเดียว

5. ขนาดของข้อมูลแสดงลักษณะบล็อค (The block descriptor length) เหมือนกับใน MODE SELECT

6.7.1.9 คำสั่ง READ CAPACITY รหัส 25h ประเภท Extended ตารางที่ 2.30

หน้าที่ เพื่อให้อุปกรณ์เริ่มต้นการสื่อสารข้อมูลร้องขอข้อมูลเกี่ยวกับความจุของอุปกรณ์เลขหมายโลจิคัลที่ระบุ โครงสร้างคำสั่งปรากฏในตาราง 2.30 ซึ่งมีส่วนประกอบดังนี้

Bit Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Reserved				RelAdr
2	Logical Block Address (MSB)							
3	Logical Block Address							
4	Logical Block Address							
5	Logical Block Address (LSB)							
6	Reserved							
7	Reserved							
8	Vendor Unique		Reserved					PMI
9	Vendor Unique		Reserved				Flag	Link

ตารางที่ 2.30 แสดงโครงสร้างคำสั่ง READ CAPACITY

- บิตพาร์เชี่ยลมีเดียมอินดิเคเรเตอร์ (Partial medium indicator - PMI) บิตนี้จะเป็นตัวกำหนดว่าข้อมูลที่ได้รับจากอุปกรณ์เป้าหมายนั้น หมายถึงอะไร

ถ้าบิตนี้ เป็น 0 ข้อมูลเลขหมายโลจิคัลบล็อคและขนาดบล็อค (ไบต์) ที่ได้รับจากคำสั่งนี้เป็นเลขหมายโลจิคัลบล็อค และขนาดของบล็อคสุดท้ายของอุปกรณ์เลขหมายโลจิคัลนั้น ซึ่งในกรณีนี้ข้อมูลตำแหน่งโลจิคัลบล็อคในส่วนของคำสั่ง READ CAPACITY ไม่มีความสำคัญ อาจกำหนดให้เป็น 0

ถ้าบิตนี้ เป็น 1 ข้อมูลเลขหมายโลจิคัลบล็อคและขนาดบล็อค (ไบต์) ที่ได้รับจากคำสั่งนี้เป็นเลขหมายโลจิคัลบล็อค และขนาดบล็อคที่มีการโอนถ่ายล่าสุด เลขหมายโลจิคัลบล็อคนี้ อาจจะเท่ากับหรือมากกว่าเลขหมายโลจิคัลบล็อคที่ระบุในคำสั่ง READ CAPACITY

2.3 วิธีการควบคุมอุปกรณ์ที่มีระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ในระบบปฏิบัติการ SCO UNIX SYSTEM V/386

ระบบปฏิบัติการ SCO UNIX SYSTEM V/386 เป็นระบบปฏิบัติการยูนิกซ์ในตระกูล SYSTEM V บนเครื่องไมโครคอมพิวเตอร์ ที่ใช้หน่วยประมวลผลกลาง ของ Intel หมายเลข 80386 ระบบปฏิบัติการนี้ ฉบับที่ 3.2.0 รุ่น 1 ได้พัฒนาระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ ไว้ในส่วนของ kernel ด้วย โดยพัฒนาตามมาตรฐานของ ANSI หมายเลข X3.131-1986 ซึ่งมีข้อกำหนดในการใช้งานดังนี้

1. ข้อกำหนดทางฮาร์ดแวร์

1.1 สกาสซี่โฮสอแดปเตอร์ที่จะใช้กับระบบปฏิบัติการ SCO UNIX SYSTEM V/386 รุ่นนี้จะต้องเป็นของบริษัทอแดปเทค (Adaptec) รุ่น AHA-154X ที่สามารถกำหนดเลขที่อยู่ฐาน (base address) เป็น 330h และหมายเลขการขัดจังหวะ (IRQ) เป็น 11 ได้

1.2 คอมพิวเตอร์แม่งาน (host computer) 1 ระบบ สามารถบรรจุสกาสซี่โฮสอแดปเตอร์ได้ 2 แผงวงจร โดยแต่ละแผงวงจรจะต่ออุปกรณ์หรือแผงควบคุมอุปกรณ์ (controller) ได้ 7 หน่วยซึ่งแผงควบคุมอุปกรณ์แต่ละหน่วยจะต่อกับอุปกรณ์ประเภทเดียวกันได้ 8 อุปกรณ์ ดังนั้นในระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ 1 ระบบจะพ่วงต่ออุปกรณ์ได้สูงสุด 56 อุปกรณ์ และคอมพิวเตอร์แม่งาน 1 ระบบ จะพ่วงต่ออุปกรณ์ได้สูงสุด 112 อุปกรณ์

2. ข้อกำหนดทางซอฟท์แวร์

2.1 ระบบปฏิบัติการ SCO UNIX SYSTEM V/386 รุ่นนี้ ได้แบ่งส่วนชุดคำสั่ง

เพื่อติดต่อและควบคุมอุปกรณ์ผ่านระบบการเชื่อมต่ออุปกรณ์แบบสกาสซีออกเป็น 3 ส่วน ดังนี้

2.1.1 โปรแกรมควบคุมโฮสอแดปเตอร์ (host adapter driver)

โปรแกรมนี้ทำหน้าที่ในการติดต่อและควบคุมโฮสอแดปเตอร์ให้ส่งสัญญาณ หรือตอบรับสัญญาณกับสายสัญญาณสกาสซี ซึ่งในระบบปฏิบัติการรุ่นนี้ได้มีโปรแกรมควบคุมสกาสซีโฮสอแดปเตอร์ของบริษัทอแดปเทคอยู่ในระบบแล้ว

โปรแกรมย่อยของโปรแกรมควบคุมโฮสอแดปเตอร์ ประกอบด้วย

- โปรแกรมเริ่มต้นการทำงานของโฮสอแดปเตอร์ (init routine)

โปรแกรมย่อยนี้ทำหน้าที่ตรวจสอบโครงแบบ (configuration) ของอุปกรณ์ต่างๆ และทำการกำหนดค่าเริ่มต้นของแผงควบคุมอุปกรณ์ต่าง ๆ โปรแกรมย่อยนี้จะถูกเรียกให้ทำงานโดยเคอร์เนลตอนเริ่มต้นการทำงานของระบบปฏิบัติการยูนิกซ์

- โปรแกรมที่เป็นจุดเข้า (entry routine)

โปรแกรมย่อยนี้จะถูกเรียกโดยโปรแกรมควบคุมอุปกรณ์ต่างๆ เพื่อส่งคำสั่งไปยังอุปกรณ์สกาสซีต่าง ๆ โปรแกรมนี้จะทำหน้าที่แปลงคำสั่งสกาสซีที่ได้รับจากโปรแกรมควบคุมอุปกรณ์ เป็นคำสั่งที่ส่งต่อให้กับโฮสอแดปเตอร์ได้

โปรแกรมย่อยนี้จะสามารถเก็บรักษาโครงสร้างข้อมูลร้องขอที่ถูกส่งมาแต่ยังไม่ได้ทำงานได้ถึง 16 ชุด โดยจัดเรียงการทำงานตามลำดับที่ได้รับ

- โปรแกรมสำหรับปรับควบคุมการนำเข้า/ส่งออก (i/o control command routine)

โปรแกรมย่อยนี้สำหรับให้โปรแกรมควบคุมอุปกรณ์สามารถปรับค่าพารามิเตอร์ของอุปกรณ์บางอย่าง เช่น เวลาของบัสเปิด/ปิด เป็นต้น

- โปรแกรมตอบรับสัญญาณขัดจังหวะ (Interrupt routine)

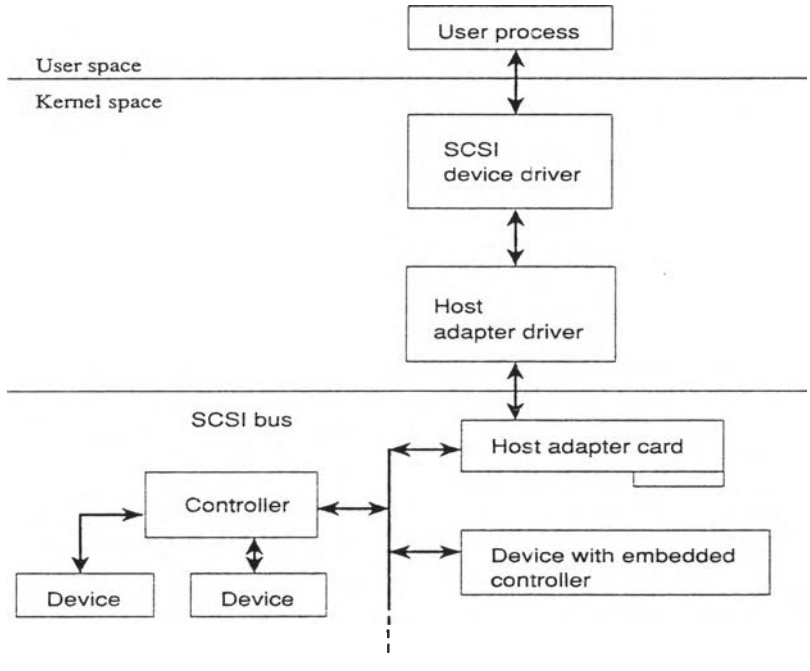
โปรแกรมนี้จะทำหน้าที่ตอบรับสัญญาณขัดจังหวะของอุปกรณ์ที่ส่งมายังโฮสอแดปเตอร์ เมื่อได้รับสัญญาณขัดจังหวะโปรแกรมจะตรวจสอบสถานะต่าง ๆ เช่น สถานะสกาสซีโฮสอแดปเตอร์ สถานะอุปกรณ์ สถานะของคำสั่ง และตำแหน่งของข้อมูล บรรจุสถานเหล่านี้ในโครงสร้างข้อมูลร้องขอ ตรวจสอบว่าการขัดจังหวะมาจากอุปกรณ์ใดแล้ว ส่งการทำงานกลับคืนไปให้โปรแกรมย่อยควบคุมอุปกรณ์ส่วนที่เกี่ยวข้องกับการตอบรับสัญญาณขัดจังหวะ ไปทำงานต่อโดยส่งโครงสร้างข้อมูลร้องขอกลับคืนไปให้

2.1.2 โปรแกรมควบคุมอุปกรณ์สกาสซี (SCSI device driver)

โปรแกรมนี้ทำหน้าที่ในการควบคุมการทำงานของอุปกรณ์สกาสซีแต่ละชนิด โดยในระบบปฏิบัติการนี้ โปรแกรมควบคุมอุปกรณ์สกาสซีจะถูกพัฒนาขึ้นมาสำหรับ

อุปกรณ์สกาสซี่ แต่ละชนิดในรูปแบบของโปรแกรมควบคุมอุปกรณ์ทั่วไป เพียงแต่การติดต่อกับอุปกรณ์นั้นจะกระทำผ่านโปรแกรมควบคุมโฮสอแดปเตอร์ และขณะที่อุปกรณ์ส่งสัญญาณขัดจังหวะเข้ามา โปรแกรมควบคุมโฮสอแดปเตอร์จะรับสัญญาณขัดจังหวะก่อน แล้วจึงส่งต่อการทำงานให้กับโปรแกรมควบคุมอุปกรณ์ส่วนที่เกี่ยวข้องกับการตอบรับการขัดจังหวะจะได้กล่าวถึงรายละเอียดของโปรแกรมย่อยในขั้นตอนต่อไป

2.1.3 ชุดคำสั่งฟังก์ชั่นที่เกี่ยวข้องกับการเชื่อมต่ออุปกรณ์แบบสกาสซี่ ระบบปฏิบัติการนี้ได้เตรียมชุดคำสั่งฟังก์ชั่นที่เกี่ยวข้องกับการเชื่อมต่ออุปกรณ์แบบสกาสซี่ไว้ซึ่งโปรแกรมควบคุมอุปกรณ์ สามารถเรียกใช้งานได้ เพื่อความเป็นอันหนึ่งอันเดียวกัน เพิ่มประสิทธิภาพ และลดความซ้ำซ้อนของชุดคำสั่งในโปรแกรมควบคุมอุปกรณ์แบบสกาสซี่ชนิดต่าง ๆ



ภาพที่ 2.8 แสดงระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ของระบบปฏิบัติการ SCO UNIX SYSTEM V/386 และการทำงานของระบบโปรแกรมควบคุมอุปกรณ์

2.2 ระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ที่ ระบบปฏิบัติการรุ่นนี้รองรับได้ในปัจจุบัน เป็นในรูปแบบของอุปกรณ์เริ่มต้นการสื่อสารเพียงอุปกรณ์เดียว และอุปกรณ์เป้าหมายหลายอุปกรณ์ (single initiator multiple targets)

2.3 การพัฒนาโปรแกรมควบคุมอุปกรณ์สกาสชี่บนระบบปฏิบัติการ SCO UNIX SYSTEM V/386

2.3.1 ระบบปฏิบัติการนี้มีชุดคำสั่งที่จะสร้างและติดตั้งตารางโครงแบบ (configuration table) และโครงสร้างข้อมูลแบบสถิต สำหรับอุปกรณ์ประเภทต่างๆไว้ เพื่อความสะดวกในการติดตั้งโปรแกรมควบคุมอุปกรณ์ชนิดใหม่เข้ามาในระบบ โปรแกรมควบคุมอุปกรณ์ที่ต้องการใช้ชุดคำสั่งในการสร้างและติดตั้งตารางโครงแบบ และโครงสร้างข้อมูลแบบสถิตจะต้องตั้งชื่อนำหน้าโปรแกรมย่อยควบคุมอุปกรณ์ (driver prefix) ตามชื่อย่อที่ระบุไว้ข้างท้ายอุปกรณ์แต่ละประเภทและใส่ข้อมูลเข้าไปในแฟ้มข้อมูลของระบบต่างๆ ตามที่ระบบปฏิบัติการนี้ระบุไว้ (ดูรายละเอียดในภาคผนวก ก.)

2.3.2 โปรแกรมควบคุมอุปกรณ์สกาสชี่จะติดต่อกับอุปกรณ์โดยผ่าน โปรแกรมควบคุมโฮสอแดปเตอร์ ทั้งในขั้นตอนการส่งคำสั่งและข้อมูลไปยังอุปกรณ์ รวมทั้งในขั้นตอนที่อุปกกรณ์ส่งสัญญาณขัดจังหวะเข้ามาในระบบ ในกรณีการส่งคำสั่งไปยังอุปกรณ์ โปรแกรมควบคุมอุปกรณ์จะเตรียมโครงสร้างข้อมูลร้องขอชุดหนึ่ง (request block) ซึ่งประกอบด้วย ข้อมูล คำสั่งตำแหน่งของข้อมูลที่ต้องการ เป็นต้น และส่งโครงสร้างข้อมูลชุดนี้ไปยังโปรแกรมควบคุมโฮสอแดปเตอร์ โปรแกรมควบคุมโฮสอแดปเตอร์จะรับโครงสร้างข้อมูลมาแปลงและส่ง คำสั่งไปยังโฮสอแดปเตอร์ ให้ส่งสัญญาณไปตามสายสัญญาณสกาสชี่ถึงอุปกรณ์เป้าหมายตามคำสั่งที่ได้รับมา

เมื่ออุปกรณ์เป้าหมายทำงานเสร็จจะส่งสัญญาณขัดจังหวะเข้ามาตามสายสัญญาณสกาสชี่ มายังโฮสอแดปเตอร์ ซึ่งจะส่งต่อสัญญาณขัดจังหวะให้กับโปรแกรมย่อยควบคุมโฮสอแดปเตอร์ ส่วนที่เกี่ยวข้องกับการตอบรับสัญญาณขัดจังหวะ โปรแกรมย่อยจะตรวจสอบสถานะของการทำงานคำสั่ง บรรจุสถานะต่าง ๆ เข้าในโครงสร้างข้อมูลร้องขอชุดที่เกี่ยวข้องกับการเกิดสัญญาณขัดจังหวะของอุปกรณ์ และเรียกโปรแกรมย่อยควบคุมอุปกรณ์ในส่วนที่เกี่ยวข้องกับการตอบรับสัญญาณขัดจังหวะ พร้อมกับส่งโครงสร้างข้อมูลร้องขอกลับคืน เพื่อให้โปรแกรมย่อยควบคุมอุปกรณ์นำข้อมูลไปใช้งาน หรือตรวจสอบสถานะการทำงานได้

2.3.3 ในระบบปฏิบัติการรุ่นปัจจุบันจะรองรับการใช้งานคำสั่งสกาสชี่แบบเดี่ยว ๆ เท่านั้นยังไม่สามารถใช้คำสั่งชุด (linked command) ได้

2.3.4 โครงสร้างข้อมูลร้องขอที่สำคัญ ประกอบด้วย

ฟิลด์	ประเภทข้อมูล	ข้อมูลที่บรรจุ
1. ฟิลด์ที่ต้องกำหนดโดยโปรแกรมควบคุมอุปกรณ์		
req_type	unsigned short	ประเภทของการร้องขอ แบ่งออกเป็น - SCSI_INIT เป็นคำสั่งเพื่อเตรียมโฮสอแดปเตอร์ให้พร้อมสำหรับการทำงานคำสั่งต่าง ๆ - SCSI_INFO เป็นคำสั่งเพื่อขอข้อมูลเกี่ยวกับความสามารถของโฮสอแดปเตอร์ - SCSI_SEND ใช้สำหรับส่งคำสั่งสกาสซี่ต่างๆ
opcode	char	รหัสคำสั่งสกาสซี่
id	char	เลขหมายอุปกรณ์ของอุปกรณ์เป้าหมาย
lun	char	เลขหมายโลจิคัลของอุปกรณ์
ha_num	char	โฮสอแดปเตอร์ที่อุปกรณ์เป้าหมายต่อพ่วงอยู่ (ระบบปฏิบัติงานรุ่นปัจจุบันยอมให้มีโฮสอแดปเตอร์ได้ 2 หน่วย)
ext_p	struct exten*	ถ้า req_type เป็น SCSI_INFO โฮสอแดปเตอร์จะส่งข้อมูลกลับมาโดยบรรจุตำแหน่งหน่วยความจำแรกของข้อมูลในฟิลด์นี้
dir	char	ทิศทางของการเคลื่อนย้ายข้อมูลของคำสั่งนี้แบ่งออกเป็น - SCSI_IN ย้ายข้อมูลจากอุปกรณ์มายังคอมพิวเตอร์แม่งาน - SCSI_OUT ย้ายข้อมูลจากคอมพิวเตอร์แม่งานไปยังอุปกรณ์
cmdlen	char	ขนาดของคำสั่ง (เป็นไบต์)
data_len	paddr_t	ขนาดของข้อมูลที่จะเคลื่อนย้ายโดยคำสั่งนี้
data_ptr	paddr_t	ตัวชี้ไปยังตำแหน่งหน่วยความจำจริงซึ่งจะมีการเคลื่อนย้ายข้อมูล
data_blk	paddr_t	ตำแหน่งโลจิคัลบล็อคของข้อมูลในอุปกรณ์ที่เกี่ยวข้องกับคำสั่งนี้

ฟิลด์	ประเภทข้อมูล	ข้อมูลที่บรรจุ
scsi_cmd	union scsi_cdb	โครงสร้างข้อมูลของคำสั่งสกาสชี่
r_count	char	จำนวนครั้งที่ขอให้โฮสอแดปเตอร์ลองทำคำสั่งเดิมซ้ำถ้าเกิดความผิดพลาดที่ไม่ได้ทำให้อุปกรณ์ส่งสถานะ CHECK CONDITION กลับมาที่อุปกรณ์เริ่มต้นการสื่อสารข้อมูล (โฮสอแดปเตอร์บางบริษัทอาจไม่ทำคำสั่งเดิมซ้ำ)
hacmd	char	ถ้าประเภทการร้องขอเป็น SCSI_INIT การกำหนด hacmd เป็น SCSI_DISK_INFO จะเป็นการขอข้อมูลเกี่ยวกับ BIOS-compatibility geometry ของอุปกรณ์ ถ้าไม่ต้องการข้อมูลให้กำหนดฟิลด์นี้เป็น SESI_NO_INFO
*(io_intr)()	int	ตัวชี้ตำแหน่งของหน่วยความจำของโปรแกรมตอบรับสัญญาณขัดจังหวะของโปรแกรมควบคุมอุปกรณ์

2. ฟิลด์ที่โฮสอแดปเตอร์เป็นผู้ที่ใส่ค่า

ฟิลด์	ประเภทข้อมูล	ข้อมูลที่บรรจุ
host_sts	char	สถานะของโฮสอแดปเตอร์หลังการทำคำสั่งสกาสชี่ ถ้าเป็น 0 แสดงว่าการทำงานสมบูรณ์
target_sts	char	สถานะของอุปกรณ์เป้าหมายหลังการทำคำสั่งสกาสชี่ ถ้าเป็น 0 แสดงว่าการทำงานสมบูรณ์
req_status	char	สถานะการทำงานคำสั่งนั้น

3. ฟิลด์ที่ใช้โดยโปรแกรมควบคุมอุปกรณ์ (เลือกใช้ตามเหมาะสม)

ฟิลด์	ประเภทข้อมูล	ข้อมูลที่บรรจุ
req_id	padd_t	เลขหมายของโครงสร้างข้อมูลแบบร้องขอ
use_flag	char	ข้อมูลแสดงสภาพการใช้งาน ถ้าเป็น 0 แสดงว่าโครงสร้างนี้ว่างสำหรับการใช้งาน
internal	char	ใช้ตามความเหมาะสม
jq	char	ใช้ตามความเหมาะสม
s1	char *	ใช้ตามความเหมาะสม
s2	char *	ใช้ตามความเหมาะสม
ctlr	char	ตำแหน่งหน่วยความจำของแผงควบคุมอุปกรณ์
adapter	char	เลขที่อยู่ฐานของโฮสอแดปเตอร์
rbuf	struct buf *	บัฟเฟอร์สำหรับใช้งานกับคำสั่งสกาสซี่นี้

4. ฟิลด์ที่สำรองไว้ใช้ในอนาคต การใช้งานปัจจุบันให้กำหนดเป็น 0 คือ

ฟิลด์	ประเภทข้อมูล
sense_len	char
scsi_sense	paddr_t
link_ptr	paddr_t
link_id	char

หมายเหตุ ประเภทข้อมูล

paddr_t หมายถึง ตำแหน่งหน่วยความจำจริง (physical memory address) มีขนาดเป็น unsigned long

สำหรับโครงสร้างข้อมูล exten , sesi_cdb และ buf ดูรายละเอียดได้ในแฟ้มข้อมูล header ของ ภาคผนวก ข.

พิธีการของระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่ ประกอบด้วย การสื่อสารกันระหว่างอุปกรณ์เริ่มต้นการสื่อสารและอุปกรณ์เป้าหมาย สิ่งที่สื่อสารกันประกอบด้วย คำสั่ง ข้อมูล ข้อความ สถานะการทำงานของคำสั่งและเซนซ์ดาต้า สำหรับระบบการเชื่อมต่ออุปกรณ์แบบสกาสซี่บนระบบปฏิบัติการยูนิกซ์นี้ องค์ประกอบของการสื่อสารจะอยู่ในฟิลด์ต่าง ๆ ของโครงสร้างข้อมูลร้องขอ ดังนี้

คำสั่งสกาสซี่ จะอยู่ในฟิลด์ opcode และในโครงสร้างคำสั่ง sesi_cdb

ข้อมูล ตำแหน่งที่ที่จะบรรจุข้อมูลจะอยู่ในฟิลด์ data_ptr
ขนาดข้อมูลที่ต้องการจะอยู่ในฟิลด์ data_len

ข้อความ ไม่ปรากฏในโครงสร้างข้อมูลการร้องขอ แต่โปรแกรมควบคุมโฮสอแดปเตอร์จะตรวจสอบข้อความเอง ถ้ามีปัญหาจะแสดงออกในฟิลด์ host_sts target_sts

สถานะการทำงานคำสั่ง จะอยู่ในฟิลด์ target_sts
ถ้าสถานะการทำงานของคำสั่งเป็น CHECK CONDITION โปรแกรมควบคุมโฮสอแดปเตอร์จะส่งคำสั่ง REGUEST SENSE ไปยังอุปกรณ์เป้าหมายอุปกรณ์นั้นเพื่อขอเซนซ์ดาต้าให้โดยอัตโนมัติ ซึ่งข้อมูลของเซนซ์ดาต้าจะปรากฏในฟิลด์ scsi_cdb.raw

2.3.5 โปรแกรมควบคุมอุปกรณ์สามารถจะตรวจสอบสถานะการส่งคำสั่ง (โครงสร้างข้อมูลแบบร้องขอ) ไปยังโปรแกรมควบคุมโฮสอแดปเตอร์ ได้ดังนี้

2.3.5.1 ถ้าค่าตอบกลับ (return value) เป็น 0 แสดงว่า การส่งคำสั่งไปยังโปรแกรมควบคุมอุปกรณ์เกิดขึ้นโดยสมบูรณ์

2.3.5.2 ถ้าค่าตอบกลับ เป็น -1 แสดงว่ามีความผิดพลาดเกิดขึ้นซึ่งอาจเกิดจาก

- โปรแกรมควบคุมโฮสอแดปเตอร์ไม่สามารถรับคำสั่งอื่นต่อไปได้อีก (คำสั่งที่ได้รับและยังไม่ได้ทำงานมีมากกว่า 16 คำสั่ง)

- มีความผิดพลาดในโปรแกรมควบคุมโฮสอแดปเตอร์

ในกรณีที่โปรแกรมควบคุมอุปกรณ์เตรียมโครงสร้างข้อมูลร้องขอผิดค่าตอบกลับจะยังคงเป็น 0 โดยโปรแกรมควบคุมโฮสอแดปเตอร์จะใส่ค่าความผิดพลาดในฟิลด์ host_sts และส่งคืนโครงสร้างข้อมูลร้องขอไปยังโปรแกรมย่อยควบคุมอุปกรณ์ส่วนตอบรับสัญญาณบัดจังหวะ