การเขียนโปรแกรมบนวินโดวส์โดยติดต่อกับผู้ใช้ด้วยจอสัมผัส

Application on Windows user interface with Touch Screen

โดย

นาย พีระ เหลืองรัตนะแสง

นาย วีระศักดิ์ เจริญรัตน์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2536

ปริญญานิพนธ์ปีการศึกษา 2536

ภาควิชาคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สภาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหราลาดกระบัง

เรื่อง การเขียน Application บน Windows โดยใช้ user-interface แบบ Touch screen

ผู้จัดทำ

1. นาย พีระ เหลืองรัตนะแสง
2. นาย วีระศักดิ์ เจริญรัตน์

----------อาจารย์ที่ปรึกษา

(อาจารย์ วัชระ ฉัตรวิริยะ)

การเขียนโปรแกรมบนวินโดวส์โดยติดต่อกับผู้ใช้ด้วยจอสัมผัส

Application on Windows user interface with Touch Screen

โดย นาย พีระ เหลืองรัตนะแสง

นาย วีระศักดิ์ เจริญรัตน์

อาจารย์ที่ปรึกษา อาจารย์วัชระ ฉัตรวิริยะ

บทคัดย่อ

ในการติดต่อโดยใช้จอสัมผัสและระบบหลายสื่อนั้น เป็นวิธีที่ทำให้ผู้ใช้เกิดความรู้สึกคุ้นเคยและง่ายในการติดต่อกับเครื่อง จอสัมผัสที่เราใช้นี้เป็นจอสัมผัสระบบที่ใช้ความต้านทานในการหาตำแหน่ง ของจุดสัมผัสบนผิวจอ เราได้นำระบบการติดต่อแบบหลายสื่อ และจอสัมผัสมาประยุกต์ในการติดต่อกับโปรแกรม ซึ่งเป็นโปรแกรมที่สร้างเพื่อใช้ช่วยในการสอนวิชาไมโครโปรเซสเซอร์เบื้องต้น

ABSTRACT

The method to interface with computer by using touch screen and multimedia is the way that can make user friendly. Touch screen that we use is the switching-touch screen with resistors array to detect the signal on the screen. And we apply the touch screen and multimedia to interface with program that we make to instruction in Introduction to Microprocessors.

สารบัญ

บทที่2
แนวคิดในการออกแบบระบบ

ในการออกแบบโปรแกรมของเรานั้น จะมีลักษณะที่คล้ายกับการเล่นเทป หรือการดูทีวี ซึ่งผู้ใช้สามรถดูส่วนของบทเรียนซ้ำไปมาได้ และมีบทเรียนที่ให้ผู้ใช้เลือกสิ่งที่ต้องการเรียนรู้เองโดยการกดในรูปของสิ่งนั้น ซึ่งก็จะเป็นลักษณะคล้ายการถามในห้องเรียนเมื่อผู้เรียน สงสัยในบางอย่างที่ตนเรียน ในการใช้จอสัมผัสจะทำให้ผู้เรียนไม่ต้องมากังวลกับการติดต่อ ที่ยุ่งยากอย่างเดิม เช่นการป้อนคำสั่งด้วยแป้นพิมพ์ รวมทั้งยังมีภาพเคลื่อนไหว เป็นการกระตุ้นความสนใจของผู้เรียนไม่ให้เกิดความเบื่อต่อบทเรียน พร้อมทั้งมีเสียงบรรยายที่เป็นภาษาไทย ทำให้เกิดความเข้าใจได้ดีขึ้น

สำหรับการสร้างโปรแกรมในการช่วยสอนวิชาไมโครโปรเซสเซอร์ขั้นพื้นฐานนี้ เป็นโปรแกรมเพื่อเป็นตัวอย่างในการนำรูปแบบการใช้สื่อหลายสื่อ ในการติดต่อกับผู้ใช้ เพื่อให้เกิดความสนใจต่อบทเรียน และเพื่อเป็นแนวทางในการสร้าง โปรแกรมเพื่อช่วยในการสอนในวิชาอื่น ๆ ต่อไป

สำหรับการทำงานของโปรแกรมนี้นั้น เป็นการทำงานในลักษณะที่ต่อเนื่องกันไป ผู้เรียนอาจเรียนบทใดก่อนก็ได้ แต่ในแต่ละบท ก็จะเป็นการเรียนแบบต่อเนื่องไป ซึ่งอาจข้ามในช่วงของการเรียนโดยการ กด เพื่อให้ข้ามในส่วนที่กำลังอธิบายอยู่ แต่ก็จะเป็นลำดับต่อเนื่องไปจนจบบท และเมื่อจบบทเรียนแล้ว ผู้เรียนสามารถเริ่มเรียนในบทใหม่ได้ หรืออาจเรียนในบทเรียนเดิมซ้ำอีกก็ได้

ในการสร้างบทเรียนนั้น เราจะเริ่มจาก การกล่าวในเรื่องทั่วไปของไมโครโปรเซสเซอร์ และประโยชน์ของไมโครโปรเซสเซอร์ที่เราเห็นในชีวิต ประจำวัน รวมทั้งลักษณะคร่าว ๆ ของไมโครโปรเซสเซอร์

ในบทที่ 2 จะเป็นเรื่องที่เกี่ยวกับโครงสร้างทั่วไป และลักษณะของไมโครโปรเซสเซอร์ที่พบเห็นอยู่โดยทั่วไป รวมทั้งมีการอธิบายเกี่ยวกับคำศัพท์ที่เกี่ยวข้องกับไมโครโปรเซสเซอร์บางคำ

ในบทที่ 3 จะเป็นเรื่องของโครงสร้างภายในของตัวไมโครโปรเซสเซอร์ ซึ่งประกอบด้วยส่วนทำงานต่าง ๆ อย่างคร่าว ๆ

เนื่องจากว่าเป็นบทเรียนที่ช่วยในการสอน ดังนั้นเนื้อหาวิชานั้นจึงยังไม่เน้นความละเอียดมากนัก เป็นเพียงแต่การกล่าวโดยรวม ๆ ซึ่งก็จะช่วยให้ผู้เรียนเข้าใจภาพรวม ๆ ของวิชานี้ได้

2.1 การคำนวณและการสร้าง

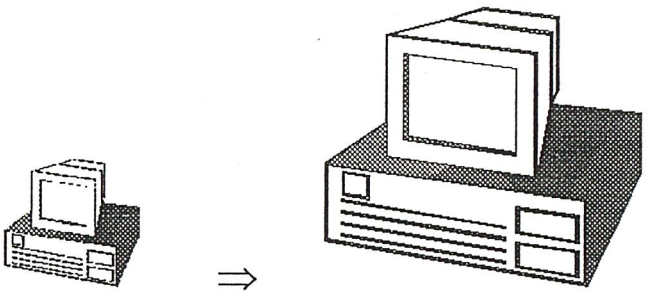
ขั้นตอนการออกแบบโปรแกรมแบบออกเป็น 3 ขั้นตอนคือ

2.1.1. การเขียน script(บทความ) ของตัวโปรแกรม

การเขียน script เราจะทำการเขียนเป็น รูปภาพและ เสียงที่ให้ในสำหรับแต่ละรูปที่จะทำงาน

โดยที่เราจะก็หนดเป็น Frame กับ Action ตัวอย่างเช่น

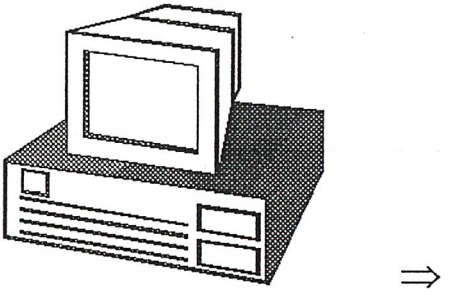
Frame ที่ 1



เราจะทำาการพูด " ปัจจุบันนี้คอมพิเตอร์ได้เข้ามามีบทบาทในชีวิตประจำวันเรามากขึ้น

Action คอมพิวเตอร์ขยายขึ้นจากขนาดเล็กไปยังขนาดใหญ่

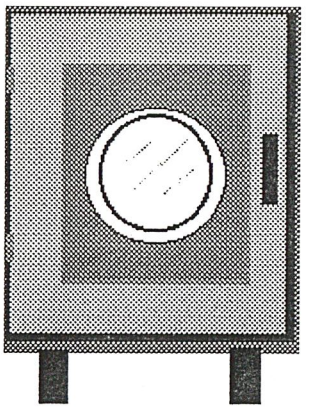
Frame ที่ 2



ซึ่งจะเห็นได้จากเครื่องใช้ในชีวิตประจำวันที่มีระบบคอมพิวเตอร์ควบคุมอยู่

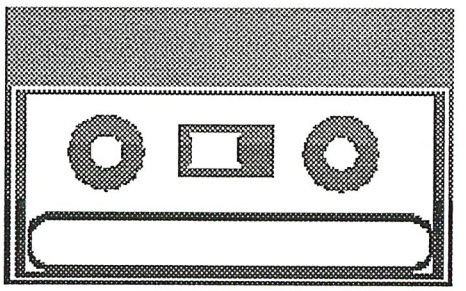
Action เป็นการที่เครื่องคอมพิวเตอร์นี้ทำการเคลื่อนที่

Frame ที่ 3



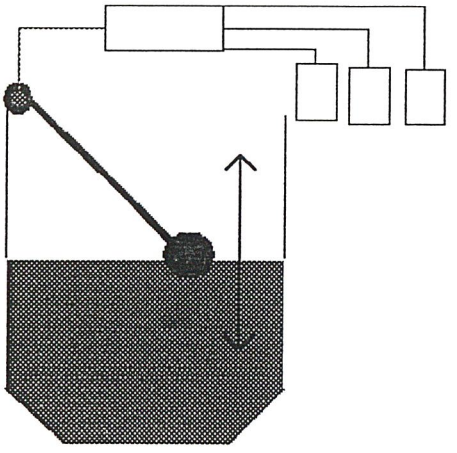
เช่นเครื่องซักผ้า

Frame ที่ 4



เครื่องเล่นเทป

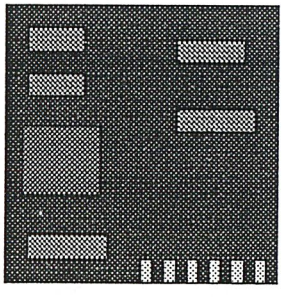
Frame ที่ 4



เครื่องวัดระดับของเหลวเป็นต้น

Action ระดับน้ำไหลขึ้นลง

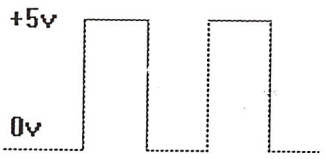
Frame ที่ 5



ในระบบคอมพิวเตอร์นั้นถ้าเราพิจารณาให้ดีจะพบว่าจะมีอุปกรณ์ที่เป็นตัวสำคัญในการทำงานนั้นคือตัว Microprocessor ซึ่งไมโครโปรเซสเซอร์นี้จะมีส่วนที่รับข้อมูลเข้าและส่งข้อมูลออกที่เราเห็นเป็นขาของตัวไมโครโปรเซสเซอร์

Action ส่วนที่เป็นไมโครโปรเซสเซอร์(ที่เห็นเป็นสีแดง)ทำการกระพิบ

Frame ที่ 6

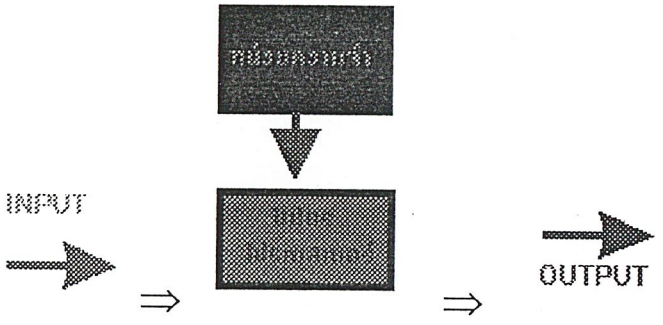


⇒

ซึ่งข้อมูลดังกล่าวนี้เป็นสัญญาณดิจิตอลคือเป็น 0 หรือ 5 Volt

Action รูปสัญญาณเคลื่อนที่

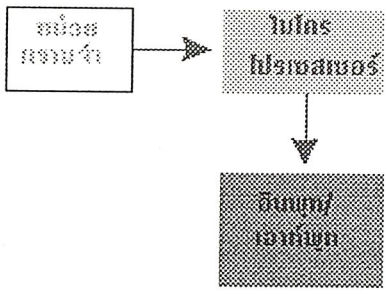
Frame ที่ 7



ตัวไมโครโปรเซสเซอร์จะรับสัญญาณเข้ามาแล้วทำการประมวณผลตามสัญญาณนัคำสั่งที่อยู่ในหน่วยความจำจะได้ผลลัพธ์เป็นสัญญาณออกจากตัวไมโครโปรเซสเซอร์

Acito เป็นสัญญาณรูปลูกศรเข้าเคลื่อนที่เข้ามาดังในรูปแล้วเป็นรูปไมโครโปรเซสเซอร์กระพิบ แล้วตามด้วยรูปหน่วยความจำกระพิบและตามด้วยรูปไมโครโปรเซสเซอร์กระพิบอีกทีหนึ่งและจะเป็นรูปลูกศรออกดังรูปเคลื่อนที่ออกไป

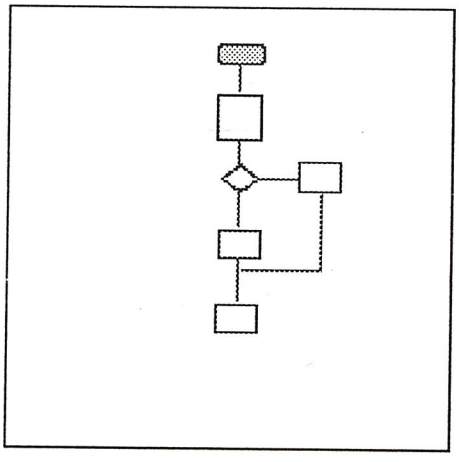
Frame ที่ 8



อาจกล่าวได้ว่าระบบคอมพิวเตอร์โดยทั่วไปจะประกอบด้วยหน่วยความจำหน่วยประมวณผลกลางหรือตัวไมโครโปรเซสเซอร์และหน่วยรับส่งข้อมูล

Action รูปภาพทำการกระพริบตามลำดับคำพูดคือ หน่ายประมวณผลกลางต่อมาคือ ไมโครโปรเซสเซอร์และสุดท้ายมากคือหน่วยรับส่งข้อมูล

Frame ที่ 9



เนื่องจากระบบคอมพิวเตอร์จะทำงานตามคำสั่งที่ได้เขียนไว้ดังนั้นเราจึงต้องมีการวางแผนการเรียนคำสั่งเหล่านี้ให้ดี

Action รูปเคลื่อนที่จากบนลงล่าง

Frame ที่ 10

ซึ่งต้องใช้ความพยายามในการศึกษามาก

Action เป็นรูปหนังสือตกลงมาแล้วเคลื่อนที่ออกไป

ต่อจากที่เราได้ทำการเขียน Script ดังตัวอย่างข้างบนแล้วเราได้ทำการนำมาทำรูปและเสียงให้ได้ตามรูปแบบที่ได้เขียนดังข้างบนนี้

2.1.2 การกำหนดระยะเวลาในการทำงานของ รูป เสียง และ เวลา

เราสามารถรู้จักเวลาการทำงานของเสียงในแต่ละช่วงเราสามารถรู้ file เสียงของเราว่ามีระยะเวลาในการทำงานนานขนาดไหนได้โดย

- ดูจากโปรแกรม Sound Recorder ซึ่งเราจะต้องทำการเปิด file เสียง file นั้นแล้วเราจะเห็นเวลาที่บอกหน่วยเป็น วินาที

- ดูจาก file เสียง file นั้นโดยตรงโดยดูจาก wave header file ซึ่งเรานำมาจาก struct waveformat_tag ใน record nSamplesPerSec และ nBlockAlign nSamplesPerSec ซึ่งเป็นส่วนที่บอกความถี่ในการนำข้อมูลเข้า ซึ่งความถี่ที่นำข้อมูลเข้านี้แบ่งออกเป็น 11.025 kilohertz , 22.05 kilohertz และ 44.1 kilohertz และ nBlockAlign เป็นตัวบอกว่า เป็นการบอกขนาดของข้อมูลคือ 8 bit , 16 bit ซึ่งเราจะต้องนำข้อมูลทั้งสองนี้มาหารด้วยขนาดของข้อมูลที่เป็นข้อมูลเสียง แล้วนำค่าที่หารได้ซึ่งเป็นค่าของเวลาทั้งหมดนำมาหารด้วย ขนาดของข้อมูลซึ่งเราจะได้เป็นเวลาทั้งหมดแล้วเรามานำ 60 ไปหารค่าที่ได้ออกมาเป็นหน่วยวินาที และ นำ 100 ไปหารอีกซึ่งจะได้เป็นหน่วยมิลิวินาทีแล้วเราก็จะได้ค่าเวลาใน file เสียงนั้นตามที่ต้องการ

หลักการคิดในการใช้เสียงและภาพในการแสดงผลงาน เนื่องจากการทำงานกับ file เสียงนี้ต้องทำการติดต่อกับ Sound Card ซึ่งเป็นการติดต่อ input/output ทำให้เราไม่สามารถติดต่อ input/output ตัวอื่นไปด้วยกันได้เพราะจะทำให้การส่งส่วน file เสียงไม่ต่อเนื่องได้เราจึงต้องมีกฎที่ว่าในขณะที่เล่น file เสียงรูปภาพทั้งหมดที่ใช้กับ file เสียงนั้นต้องถูก load ขึ้นมาอยู่ในหน่วยความจำเสียก่อนเราถึงจะทำการเล่น file เสียงได้ แต่ถ้าเราไม่ต้องการเล่นเสียงเราจะอ่านรูปมาก่อนล่วงหน้าหรือไม่ก็ได้

เมื่อเรารู้ระยะเวลาของ file เสียงแล้วตามที่ได้ระบุดังข้างบนนี้เราก็จะต้องมาดูที่ Timer ซึ่งมีตัวแปรที่ทำการเก็บจำนวน delay time ได้ซึ่งถ้าเท่ากับ 1 จะเท่ากับ 1 millisecond ซึ่งถ้าเราเซ็ตเป็นค่าอื่นก็จะเป็นค่านั้นคูณด้วย 1 millisecond ไปแล้วนำมาคำนวณกับเวลาที่ได้จาก file เสียงทำให้เรารู้เวลาที่ file เสียงนั้นใช้ในรูปของ message WM_TIMER ได้อีกด้วย

2.1.3. นำการคำนวณทั้งหมดมาใส่ในโปรแกรม

เริ่มแรกเราทำการก็หนดค่าตัวแปรต่างๆ Frame ดังตัวอย่างข้างบนนี้เป็น Script ซึ่งจะเราให้ตัวแปร Frame เป็นตัวแปรที่กำหนดค่า Frame ตัวแปร Frame นี้จะเพิ่มค่าได้ก็ต่อเมื่อ file เสียงนั้นจบลงแล้วเวลาในการทำงานของ file เสียงนี้เราจะมีเวลาที่ได้จากการ interrupt timer เป็นตัวกำหนดเวลาที่จะเข้ามา แล้วเราจะมีตัวแปร i เป็นตัวแปรซึ่งจะทำการเพิ่มค่าในลักษณะของ subframe ซึ่งจะถูกตั้งค่ากลับเป็น 1 ในขณะที่เราเพิ่มค่าของ Frame ไป Frame ถัดไปนอกจากนี้ในโปรแกรมจะมีตัวแปร waits กับ waitt ซึ่งเป็นตัวแปรที่รอการทำงานของ file เสียงเสร็จกับรอคอยด้วย timer ซึ่งจะเป็นตัวแปรสำหรับรอค่าในระหว่างที่โปรแกรมไม่ได้ทำงานอะไร หรือรอจังหวะในการทำงาน เป็นขั้นที่แต่ละขั้นใช้เวลานานกว่า 1 message WM_TIMER

การสร้าง Windows ในโปรแกรมเราได้ทำการสร้าง Windows ชนิดที่ไม่มีของกับ system bar ซึ่งชนิดของ Windows ชนิดนี้คือ WS_POPUP ซึ่งจะเหมือนรูปภาพคือจะไม่มีอะไรทั้งนั้นไม่ว่าจะเป็น คอนโทรลเมนู คอนโทรลเมนูบ็อกซ์ ไตเติ้ลบาร์ เส้นเมนู ปุ่ม minimize ปุ่ม maximize แล้วเราทำการสร้าง Windows ให้มีขนาดเต็มจอโดยคำสั่ง GetSystemMetrics และมี parameter SM_CXSCREEN ซึ่งเป็นของเขตของแกน X และ SM_CYSCREEN ซึ่งเป็นของแขตของแกน Y ในการสร้าง Windows เพื่อที่จะให้มีขนาดเต็มจอภาพก่อนที่เราจะทำการแสดง Windows เราจะทำการสร้าง Timer โดยคำสั่ง SetTimer ซึ่งในโปรแกรมเรากำหนด delay = 100 ซึ่งเป็นการส่ง message เข้ามา 10 ครั้ง ใน 1 วินาที

บทที่ 3
ลักษณะของอุปกรณ์ที่ใช้

3.1 ลักษณะของวินโดวส์ (Windows)

ระบบปฏิบัติการวินโดวส์ มีการติดต่อกับผู้ใช้แบบกราฟิก (Graphic) ทำให้ผู้ใช้เกิดความเข้าใจได้ง่าย และทำให้เรียนรู้รูปแบบการติดต่อได้เร็ว รวมทั้งสามารถ รัน(run) โปรแกรม(program) ต่าง ๆ พร้อมกันได้ และยังสามารถ ใช้ข้อมูลร่วมกันกับโปรแกรมอื่นได้ ทำให้เป็นที่นิยมกันมากในปัจจุบัน เราอาจกล่าวถึงลักษณะโปรแกรมที่รันบนวินโดวส์ ได้ดังนี้

3.1.1 ลักษณะโปรแกรมบนวินโดวส์

บนระบบวินโดวส์มีการแบ่งแยกแต่ละโปรแกรมออกจากกัน เนื่องจากวินโดวส์ เป็นระบบที่สามารถรันโปรแกรมหลาย ๆ โปรแกรมพร้อม ๆ กันได้ จึงต้องมีการแบ่งแยกแต่ละโปรแกรม ออกจากกันให้ชัดเจน โดย วินโดวส์จะใช้กรอบสี่เหลี่ยมเพื่อแบ่งแยกโปรแกรมต่าง ๆ ออกจากกัน โดยกรอบสี่เหลี่ยมนี้เราจะเรียกว่าวินโดว์ (window) แต่ละโปรแกรมจะมีวินโดว์ เป็นของตัวเอง เราสามารถมองเห็นการทำงานของโปรแกรมต่าง ๆ ได้ตาม ความต้องการของผู้ใช้ โดยที่แต่ละโปรแกรมนั้นจะแบ่งพื้นที่ของหน้าจอตามที่ผู้ใช้ต้องการ และสามารถที่จะติดต่อ กับโปรแกรมต่าง ๆ ได้ในรูปแบบที่คล้ายกัน เช่น มีปุ่มบังคับสครอลล์บาร์(scroll bar) เมนู (menu) เป็นต้น ทำให้ง่ายต่อการเรียนรู้การใช้โปรแกรมใหม่ ๆ

โปรแกรม บนวินโดวส์ โดยทั่วไปจะต้องประกอบไปด้วย ส่วนประกอบหลักดังนี้

3.1.1.1 เมนฟังก์ชัน

เมนฟังก์ชัน ซึ่งจะต้องชื่อ Winmain เสมอ ทำหน้าที่คล้าย ฟังก์ชันเมน (main function) ในภาษาซีมาตรฐาน โดยมีหน้าที่ดังนี้

1 การเตรียมงาน เช่นการรีจิสเตอร์วินโดว์ การสร้างวินโดว์

การสร้างวินโดว์ ทุกครั้งต้องมีการสร้าง วินโดว์คลาส ของวินโดว์นั้นเสียก่อน ซึ่งจะเป็นส่วนที่เก็บลักษณะ ของวินโดว์เอาไว้ เช่น ขนาด ตำแหน่ง และลักษณะอื่น ๆ ซึ่งวินโดว์คลาสนี้จะเป็นโครงสร้างข้อมูลที่มีอยู่แล้วใน วินโดวส์ ซึ่งสามารถอ้างใช้ได้เลย จากนั้นก็ใส่ค่าต่าง ๆ ในตัวแปรเหล่านั้น แล้วทำการรีจิสเตอร์คลาสที่เราสร้าง เมื่อ รีจิสเตอร์เสร็จเรียบร้อยแล้ว จะสามารถสร้างวินโดว์นั้นได้โดยใช้ฟังก์ชั้นในการสร้างวินโดว์ เมื่อสร้างสำเร็จ ก็ให้ แสดงวินโดว์นั้นโดยใช้ฟังก์ชั้นในการแสดงวินโดว์ และมีการเปลี่ยนแปลงวินโดว์ด้วย เช่นเมื่อมีการสร้างใหม่ หรือมี การเคลื่อนย้าย

2 การสร้างเมสเสจลูป(message Loop) เพื่อส่ง เมสเสจ ไปยังฟังก์ชันต่าง ๆ โดยการอ่าน เมสเสจ

จากคิวของระบบแล้วส่งต่อไปให้ยังฟังก์ชันต่าง ๆ เมสเสจเหล่านี้จะถูกส่งมาจากอินพุทต่าง ๆ ของวินโดวส์ เช่น คีย์บอร์ด (keyboard) เมาส์ (mouse) หรือไทเมอร์ (timer)

3 การจบโปรแกรมเมื่อได้รับ เมสเสจ WM_QUIT เมื่อจะจบโปรแกรมจะมีการส่ง เมสเสจ WM_QUIT ลงใน คิวของระบบ เมื่ออ่านเมสเสจเข้ามาจะให้ค่าเป็น null ทำให้ออกจาก ลูป ของเมสเสจ และสามารถทำการจบ โปรแกรมได้

3.1.1.2 ฟังก์ชันประจำวินโดว์

ฟังก์ชันอื่น ๆ ซึ่งมีลักษณะเหมือนฟังก์ชันในภาษาซี เป็นส่วนที่ทำงานจริง ๆ ในโปรแกรม เราจะเรียกว่า ฟังก์ชันประจำวินโดว์ ซึ่งจะทำหน้าที่ตอบสนอง เมสเสจ ต่าง ๆ ที่ได้รับ ซึ่งจะต้องมีลักษณะดังนี้

ต้องเป็นการเรียกแบบ FAR และผ่านค่าตัวแปรแบบ PASCAL และต้องเป็นชื่อเดียวกันกับฟังชันที่ระบุ ตอนที่รีจิสเตอร์คลาสของวินโดว์ หน้าที่ของฟังชันประจำวินโดว์คือ รับ เมสเสจ จากตัว Winmain และโดยตรงจาก วินโดวส์ แล้วทำการตอบสนอง เมสเสจ ต่าง ๆ ซึ่งบาง เมสเสจ อาจมีการส่งค่าพารามิเตอร์มาด้วย โดยจะส่งมา กับตัวแปร lParam หรือ wParam

เมสเสจ ที่ได้รับจากฟังก์ชันเมนนั้นจะเป็นเมสเสจ ของอินพุทต่าง ๆ เช่น เมาส์ คีย์บอร์ด ส่วน เมสเสจ ที่รับจาก วินโดวส์ โดยตรงนั้น ได้แก่ เมสเสจพิเศษอื่น ๆ เช่น เมื่อมีการเปลี่ยนขนาดของวินโดว์ จะมีการส่ง เมสเสจ WM_PAINT เพื่อให้มีการวาดพื้นที่ใช้งานใหม่ หรือเมื่อจบโปรแกรมก็จะมีการส่งเมสเสจเพื่อที่จะทำลายวินโดว์นั้น

3.1.1.3 คอนโทรล (Control)

คอนโทรล เป็นปุ่มบังคับชนิดต่าง ๆ สำหรับให้โปรแกรมติดต่อกับผู้ใช้งานได้ง่ายเข้า คอนโทรลเป็นวินโดว์ Child ชนิดหนึ่ง ที่มีการกำหนดคลาสไว้ใช้งานล่วงหน้าอยู่แล้ว การใช้งานคอนโทรลมักเป็นการใช้สำหรับงานง่าย ๆ เช่น รับข้อมูลบางอย่าง หรือการแสดงผลทางจอภาพแบบง่าย ๆ ซึ่งเราสามารถสร้างได้ 2 ลักษณะ คือ

ใช้งานร่วมกับกรอบข้อความ

ใช้งานร่วมกับวินโดว์ทั่วไป

การใช้งานร่วมกับวินโดว์ทั่วไปนั้น เราสร้างโดยใช้ฟังก์ชัน CreateWindow เหมือนกับการสร้างวินโดว์อื่น เพียงแต่เมื่อสร้างแล้ว การผ่านค่าจะแตกต่างกันเท่านั้น โดยที่ ค่าคลาสของวินโดว์เป็นคลาสของคอนโทรล

รูปแบบของคอนโทรล

วินโดว์ที่เป็น Parrent ของคอนโทรลนั้นต้องกำหนดตอนสร้างคลาสของคอนโทรล ค่า ID เพื่ออ้างอิงเป็นค่าของคอนโทรล

เมื่อสร้างวินโดว์แล้ว เราจะได้แฮนเดิลของคอนโทรลนั้นกลับมา เพื่อใช้งานของวินโดว์ของคอนโทรล เพื่อ จะวาดใหม่ หรือการเคลื่อนย้าย หรือทำลาย หรือใช้งานอื่น ๆ ที่เกี่ยวกับคอนโทรล

การเลือกคลาสของคอนโทรล คลาสของคอนโทรลจะเป็นตัวกำหนดการทำงานของคอนโทรล และลักษณะ การแสดงตัวของคอนโทรล และสิ่งที่ผู้ใช้จะใช้งานโดยกระทำต่อคอนโทรลนั้น ๆ คลาสของคอนโทรลนี้จะเป็นตัว กำหนดว่าคอนโทรลนั้นจะเป็นอย่างไร ซึ่งเมสเสจ มีคลาสของคอนโทรลเอาไว้ให้เราใช้อยู่แล้วดังนี้

คลาส BUTTON เป็นคอนโทรลสำหรับสร้างวินโดว์ชนิดปุ่มบังคับ ที่มีข้อความติดไว้บนปุ่มนั้น สำหรับให้ผู้ใช้ ได้เลือกสั่งงานตามข้อความบนปุ่มนั้น

EDIT เป็นวินโดว์ที่ผู้ใช้สามารถใส่ข้อความหรือแก้ไขข้อความในส่วนนั้นได้

LISTBOX เป็นวินโดว์สำหรับใส่รายชื่อสิ่งต่าง ๆ ผู้ใช้สามารถเลือกชื่อเหล่านั้นออกมาใช้งานได้

COMBOBOX เป็นวินโดว์ที่ประกอบด้วยคอนโทรลหลายชนิด โดยที่ใช้คอนโทรล STATIC หรือ EDIT มาทำงานร่วมกันกับกรอบชื่อ

SCROLL BAR เป็นคอนโทรลที่มีรูปร่างและหน้าที่การใช้งานที่คล้ายกับ สครอลล์บาร์ ในวินโดว์ทั่วไป จะอยู่บริเวณริม ๆของวินโดว์ แต่ สครอลล์บาร์ ที่สร้างขึ้นใหม่นี้จะวางไว้ในตำแหน่งใดๆ บนพื้นที่ ใช้งานก็ได้

STATIC เป็นวินโดว์ที่มีข้อความสั้น ๆ หรือเป็นรูปภาพเพื่อใช้ในการอธิบาย หรือติดชื่อให้กับคอนโทรลอื่น หรือแยกคอนโทรลออกเป็นกลุ่ม

การเลือกใช้คอนโทรลแบบต่าง ๆ

รูปแบบของคอนโทรล จะขึ้นอยู่ว่าเราเลือกคอนโทรลเป็นคลาสอะไร คลาสของคอนโทรลนั้นจะหมายถึงการใช้งานของคอนโทรล ซึ่งเราสามารถใช้คลาสร่วมกันได้ รูปแบบของคอนโทรลที่เราใช้กันอยู่มีดังนี้

BS_PUSHBUTTON เป็นคอนโทรลปุ่มบังคับแบบตอบสนองทันที ซึ่งจะเป็นช่องสี่เหลี่ยมเพื่อให้ผู้ใช้สามารถใช้เคอร์เซอร์ไปคลิกภายในได้ ซึ่งภายในจะมีข้อความสั้น ๆ เพื่อบอกว่าเป็นปุ่มที่ทำงานอะไร เมื่อผู้ใช้กดปุ่ม โปรแกรมก็จะตอบสนองการสั่งงานนี้ทันที

BS_DEFPUSHBUTTON หมายถึงปุ่มบังคับที่จะรอรับการกดเอนเทอร์(enter) อยู่แล้ว ถ้าไม่มีการเลือกปุ่มอื่น เป็นการกำหนดปุ่มที่มีการใช้งานบ่อย ๆ เมื่อไม่มีการเลือกปุ่มอื่น ก็จะเป็นการเลือกปุ่มนี้ไปโดยปริยาย

BS_CHECKBOX เป็นปุ่มบังคับเพื่อให้เลือกว่า ใช่-ไม่ใช่ ซึ่งจะมีให้เลือกอยู่สองอย่าง เมื่อเรา

เลือกว่าใช่ จะมีเครื่องหมายกากบาท ถ้าเราเลือกไม่ใช่ ก็จะไม่มีเครื่องหมายกากบาทนั้น

BS_RADIOBUTTON เป็นปุ่มให้เลือกอย่างใดอย่างหนึ่ง ซึ่งจะมีลักษณะเป็นวงกลม ซึ่งจะมีคำบรรยายอยู่ทางขวามือของปุ่ม

ES_LEFT เป็นคอนโทรลที่สามารถเขียนข้อความเข้าไปได้บรรทัดเดียว โดยอักษรจะชิดซ้ายเสมอ

ES_MULTILINE เป็นคอนโทรลที่สามารถที่จะเขียนข้อความเข้าไปได้หลายบรรทัด

SS_LEFT เป็นคอนโทรลที่บรรจุอักษรชิดขอบขวาไว้

SS_RIGHT เป็นคอนโทรลที่บรรจุอักษรชิดขอบซ้ายไว้

LBS_STANDARD เป็นคอนโทรลชนิดกรอบรายชื่อมารตฐาน ซึ่งจะมี สครอลล์บาร์ และส่งสัญญาณไปยัง parrent วินโดว์ เมื่อผู้ใช้เลือกไอเท็มใดใน listbox

CBS_DROPDOWN เป็นคอนโทรล COMBOBOX ที่ประกอบไปด้วย คอนโทรลชนิด EDIT กับกรอบรายชื่อที่ซ่อนไว้ เมื่อผู้ใช้คลิกปุ่มกดที่ตำแหน่งถัดจาก EDIT ไปทางขวา กรอบรายชื่อก็จะปรากฏออกมา และเมื่อผู้ใช้สามารถเลือกไอเท็มที่ปรากฏในส่วนของ EDIT

การหา Parent ให้กับคอนโทรล

เนื่องจากว่าคอนโทรลเป็นวินโดว์ Child แบบหนึ่ง จึงต้องมีวินโดว์ parent เราจะให้ค่าแฮนเดิลของวินโดว์parent กับฟังก์ชันสร้างวินโดว์ทุกครั้ง ซึ่งในการสร้างคอนโทรลเราจะใส่ค่าแฮนเดิลนี้ไว้ในตัวแปร hWndParent เมื่อมีการเปลี่ยนแปลงกับวินโดว์ที่เป็น parent ก็จะมีผลกระทบกับวินโดว์ Child เสมอ เช่นการย้ายที่ การวาดใหม่ การทำลาย และขนาดของคอนโทรลนั้นจะถูกจำกัดโดยขนาดของวินโดว์ที่เป็น parentด้วย นั่นคือ คอนโทรลจะปรากฏตัวอยู่ในพื้นที่ของวินโดว์ parent และการเคลื่อนย้ายคอนโทรลก็ทำได้เฉพาะในพื้นที่ของ วินโดว์ parent เท่านั้น

การกำหนดค่าอ้างอิงของคอนโทรล

ค่าอ้างอิงของคอนโทรลจะเป็นค่าที่ใช้เพื่อการแบ่งแยกคอนโทรลแต่ละตัวออกจากกัน โดยที่ค่านี้จะกำหนดเป็นตัวเลขซึ่งมีค่าเป็นอะไรก็ได้ โดยที่ค่าอ้างอิงแต่ละตัวจะต้องไม่ซ้ำกัน ซึ่งค่านี้เราจะกำหนดในตัวแปร hMenu เมื่อมีการสร้างคอนโทรลขึ้นมา

การใช้งานของคอนโทรล เมื่อผู้ใช้กระทำสิ่งใด ๆ ต่อคอนโทรล คอนโทรลจะส่งรหัสการกระทำเหล่านั้นให้แก่วินโดว์parent โดยผ่านทาง เมสเสจ WM_COMMAND โดยที่ตัวแปร wParam จะเก็บค่าอ้างอิงของคอนโทรลที่ส่ง เมสเสจ นั้นมา และตัวแปร lParam จะเก็บรหัสที่เป็นคำสั่งที่ผู้ใช้เลือก

3.1.2 การรับข้อมูลโดยผ่านทางเมสเสจ(message)

เมื่อ วินโดวส์ รับข้อจากอุปกรณ์ต่าง ๆ เข้า มาจะเก็บไว้ในคิวเมสเสจ เมื่อโปรแกรมต้องการอ่านข้อมูลก็เพียงแต่อ่านจากคิวเมสเสจ(message Que) เท่านั้น โดยที่ข้อมูลจะอยู่ในรูปของ เมสเสจ ซึ่งมีลักษณะ

การใช้งานที่คล้ายกัน ทำให้การเขียนโปรแกรมง่ายและสะดวกขึ้น

3.1.3 ระบบกราฟิก

วินโดวส์ มีคำสั่งเกี่ยวกับระบบกราฟิกให้ใช้มากมาย และเราสามารถใช้คำสั่งเหล่านี้ได้กับอุปกรณ์แสดงผลทุกชนิด ไม่ว่าจะเป็นจอภาพชนิดต่าง ๆ เครื่องพิมพ์ต่างชนิดกัน โดยที่ใช้คำสั่งและรูปแบบเดิม โดยที่วินโดวส์จะอาศัย device driver ที่เหมาะสมกับอุปกรณ์แสดงผลชนิดต่าง ๆในการแสดงผล

3.1.4 ระบบหลายงาน

เนื่องจากวินโดวส์ สามารถที่จะรันโปรแกรมต่าง ๆ พร้อมกันได้จึงต้องมีการ แบ่งปันรีซอร์ส resource ต่าง ๆ ให้ใช้ร่วมกันได้อย่างเหมาะสม ไม่ว่าจะเป็น จอภาพ หน่วยความจำ จะต้องให้วินโดวส์เป็นผู้จัดการให้เสมอ

ลักษณะของโปรแกรมที่เขียนบน วินโดวส์ โดยทั่วไปจะมีการติดต่อกับผู้ใช้ในลักษณะที่คล้ายกันดังนี้

1.วินโดว์(window)

วินโดว์เป็นส่วนที่ติดต่อกับผู้ใช้อย่างพื้นฐานโดยที่ทุกโปรแกรมจะต้องมีวินโดว์เสมอ วินโดว์จะประกอบไปด้วย เมนูบาร์(menu bar) สโครลล์บาร์(scroll bar) ไตเติลบาร์ (title bar) และอื่นๆ โดยที่ส่วนต่าง ๆ เหล่านี้จะถูกกำหนดตอนที่เราสร้างวินโดว์ ซึ่งเราสามารถให้มีส่วนต่าง ๆ ได้หรืออาจไม่ให้มีบางอย่างที่ไม่จำเป็นสำหรับโปรแกรมเราได้

2.เมนู (menu)

เมนู เป็นส่วนที่รับข้อมูลหลักของโปรแกรม ประกอบไปด้วยรายการคำสั่งต่าง ๆ ซึ่งผู้ใช้สามารถเลือกดูและใช้งานได้ โดยที่เมื่อผู้ใช้เลือกเมนูแล้ว วินโดวส์ จะส่งคำสั่งที่ผู้ใช้เลือกผ่านทางเมสเสจคิวเพื่อให้โปรแกรมตอบสนองต่อคำสั่งนั้น ๆได้

การสร้างเมนูของโปรแกรมต่าง ๆ โดยการกำหนดใน resource ไฟล์ โดยกำหนดให้เป็นการนำresource เกี่ยวกับเมนูมาใช้โดยอาจทำให้เป็น popup เมนูก็ได้ และอาจทำให้เมนูเป็น popup ที่ซ้อนกันได้ โดยการกำหนดเกี่ยวกับเมนูจะอยู่ระหว่าง begin กับ end

3.กรอบข้อความ (Dialog box)

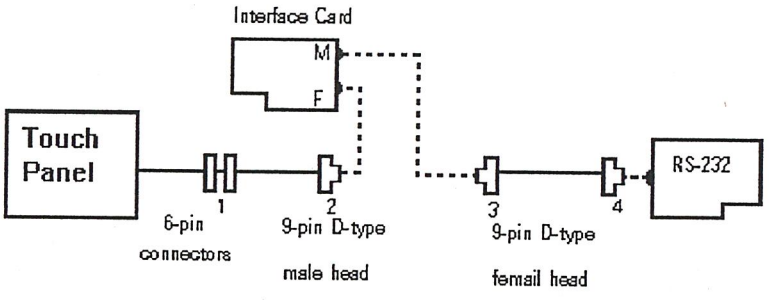
เป็นวินโดว์ชนิดหนึ่ง ที่สร้างเพื่อติดต่อกับผู้ใช้ เพื่อทำหน้าที่เฉพาะบางอย่าง เมื่อใช้เสร็จก็จะทำลายตัวเองไป และจะถูกสร้างอีกเมื่อมีการเรียกใช้ในครั้งต่อไป ซึ่งกรอบข้อความนี้มีอยู่สองลักษณะ คือ แบบที่จะต้องตอบสนองกรอบข้อความอย่างเดียวในขณะที่ใช้กรอบข้อความอยู่ นั่นคือจะเลือกไปยังวินโดว์อื่นขณะที่มีกรอบข้อความชนิดนี้อยู่ไม่ได้ และอีกแบบหนึ่งคือ กรอบข้อความที่เราสามารถเลือกไปวินโดว์อื่นได้ ถึงจะมีกรอบข้อความนั้นอยู่ก็ตาม ภายในกรอบข้อความอาจมีปุ่มต่าง ๆ เพื่อรับคำสั่งให้ทำงาน และใน

บางครั้งเราอาจใช้กรอบข้อความแทนวินโดว์หลักก็ได้ ส่วนในโปรแกรมที่เราพัฒนานั้น เราไม่มีการใช้กรอบข้อความ เนื่องจากการทำงานที่เป็นไปตามที่กำหนดไว้ในสคริปแล้ว เพียงแต่ให้ผู้ใช้เลือกคำสั่งที่ต้องการให้ทำเท่านั้น ไม่มีคำสั่งพิเศษอื่น ๆ มากนัก ดังนั้นเราจึงจะไม่กล่าวถึงการสร้างกรอบข้อความในที่นี้

4.เมสเสจ เป็นส่วนที่ วินโดวส์ ติดต่อกับโปรแกรมต่าง ๆ โดยจะส่ง เมสเสจ ที่เหมาะสมสำหรับแต่ละโปรแกรมเมื่อผู้ใช้มีการเลือกคำสั่งที่ใช้งานกับโปรแกรมนั้น ๆ หรือเมื่อผู้ใช้ต้องการติดต่อกับโปรแกรมนั้น วินโดวส์ จะเป็นตัวส่ง เมสเสจ เหล่านี้ไปยังโปรแกรมต่าง ๆ อย่างเหมาะสม ในโปรแกรมนั้นจะดักเอาบาง เมสเสจ ที่จำเป็นต้องใช้และตอบสนอง ซึ่งจะมีการเลือก เมสเสจ ที่ใช้งานเท่านั้น

3.2 จอสัมผัส (Touch Screen)

touch screen นี้มีหลายแบบ ซึ่งการเป็นการ scan line หรือ เป็น membran ต่างๆ เมื่อ user ทำการกด จะให้สัญญาณออกมา ซึ่งสัญญานี้ จะมีการเปลี่ยนแปลง เฟส หรือ แอมปิจูล และ นำไป แปลงเป็นสัญญาณดิจิตอล ทำการส่งเข้าไปยังคอมพิวเตอร์ ในลักษณะแทน mouse แต่การรับสัญญาณ นี้เราจะรับการกดเพียงอย่างเดียว ลักษณะการ Interface โดยการต่อ Hard ware ที่ใช้



ซึ่งเมื่อเราทำการต่อได้ดังรูปข้างบนแล้วเราจะต้องทำการเปลี่ยน Driver ของ Windows ซึ่งจะต้องเปลี่ยน Driver Mouse ของ Windows ให้เป็น INGMOUSE.DRV แล้วก็จะสามารถทำ Touch Screen ต่อกับ Windows ได้

บทที่ 4
การเลือกเครื่องมือ

เครื่องมือที่เราใช้ในการสร้างโปรแกรมนี้นั้นเริ่มตั้งแต่ ระบบปฏิบัติการ การใช้ภาษาในการโปรแกรม การใช้เครื่องมือในการสร้างภาพหรือสร้างเสียง และการใช้จอสัมผัสในการติดต่อกับผู้ใช้

การติดต่อในรูปแบบกราฟิกที่อาศัยภาพ ในการสื่อความหมายนั้น ทำให้ผู้ใช้เกิดความเข้าใจได้มากกว่าการใช้ข้อความอย่างเดียว ซึ่งระบบปฏิบัติการวินโดวส์นั้นมีรูปแบบการติดต่อแบบกราฟิกอยู่แล้ว ทำให้เราได้เลือกระบบปฏิบัติการบนวินโดวส์ และยังสามารถทำงานหลายอย่างพร้อมกันได้ เช่นขณะที่มีการสร้างภาพเคลื่อนไหวนั้น ก็สามารถรับข้อมูลเข้าพร้อมกันได้ และยังสามารถสร้างเสียงเพื่ออธิบายภาพเหล่านั้นไปพร้อมกันได้อีกด้วย ส่วนการสร้างภาพนั้นมีหลายวิธี ซึ่งเราได้ใช้ทั้งโปรแกรมในการวาดภาพ เช่น ฮาร์ดวาดกราฟิก คอลเรลดอลล์ และโปรแกรมในการสร้างภาพอื่น ๆ รวมทั้งการใช้สแกนเนอร์สแกนภาพเข้าไปด้วย ส่วนการสร้างเสียงนั้นได้จากการอัดเสียงโดยโปรแกรมสำเหร็จรูปที่ใช้ในการอัดเสียงและแต่งเสียงเพื่อให้ได้เสียงตามที่เราต้องการ และส่วนสุดท้ายคือการใช้จอสัมผัสในการติดต่อกับผู้ใช้นั้นต้องมีการติดตั้งจอสัมผัสตามขั้นตอนที่ได้ระบุไว้สำหรับจอสัมผัสนั้น ๆซึ่งสามารถอธิบายส่วนต่าง ๆ ได้ดังนี้

4.1 ใช้ระบบวินโดวส์เป็นระบบปฏิบัติการสำหรับโปรแกรม

ในการแสดงผลทางด้านกราฟิกนั้น ระบบวินโดวส์นั้นจะมีคำสั่งที่เกี่ยวกับกราฟิกให้ใช้อยู่อย่างมาก โดยสามารนำสิ่งที่มีอยู่แล้ว เช่น รีซอร์ส(resource) ต่าง ๆ มาใช้ได้ทันที ซึ่งก็ทำให้สะดวกในการสร้างอย่างมาก ดังนั้นเราจึงคิดว่าการสร้างโปรแกรมบนระบบวินโดวส์นั้นจะเป็นทางเลือกที่เหมาะสม รวมทั้งยังสนับสนุนการรันโปรแกรมต่าง ๆ พร้อมกันด้วยซึ่งก็เป็นสิ่งที่เป็นประโยชน์ต่อการสร้างโปรแกรมด้วย เนื่องจากว่า ในขณะที่ทำการสร้างภาพเคลื่อนไหวนั้น จะมีการติดต่อกับผู้ใช้ตลอดเวลาและมีการบรรยายเป็นเสียงประกอบกันไปด้วย ทำให้ต้องทำงานต่าง ๆ พร้อม ๆ กัน ซึ่งบนระบบวินโดวส์สามารถทำเรื่องเหล่านี้ได้ดีอยู่แล้ว

4.1.1 วินโดวส์ (Windows)

การติดต่อกับโปรแกรมในระบบวินโดวส์นั้น จะติดต่อโดยผ่านทาง เมสเสจ ซึ่งในโปรแกรมจะมีส่วนที่ทำหน้าที่คอยรับ เมสเสจ ที่วินโดวส์ ส่งไปให้ เราเรียกส่วนนี้ว่า เมสเสจลูป(message loop)เมื่อวินโดวส์ได้ส่งเมสเสจให้โปรแกรมแล้ว โปรแกรมก็จะนำเมสเสจนั้นมาตีความและทำตามคำสั่งของเมสเสจนั้น ซึ่งอาจมีการแสดงข้อมูลออกทางจอภาพ หรือมีการใช้เสียงประกอบ

4.1.1.1 การติดต่อกับ input

การที่วินโดวส์ จะรู้ว่ามีอินพุทเข้ามาจากผู้ใช้แล้วหรือยังก็คือ การมีเมสเสจปรากฏในคิวซึ่ง เมสเสจเหล่านี้ก็จะเกิดขึ้นเมื่อผู้ใช้ลากเมาส์ คลิกเมาส์ หรือดับเบลิลคลิกเมาส์ กดคีย์บอร์ด และอื่น ๆ สรุปทางที่จะทำให้เกิดเมสเสจของอินพุทได้เป็นดังนี้

ทางคีย์บอร์ด เกิดเมื่อผู้ใช้กดหรือปล่อยคีย์

ทางตัวอักษร เมื่อมีการกดคีย์ แล้ว วินโดวส์ แปรให้เป็นรหัสตัวอักษรเสียก่อนแล้วส่งเมสเสจของตัวอักษรนี้ไปยังคิว

ทางเมาส์ เกิดเมื่อผู้ใช้ลากเมาส์ไปมา กดปุ่ม ปล่อยปุ่ม ก็จะเกิดเมสเสจต่าง ๆ กันไป

ทางสัญญาณนาฬิกาของระบบ เป็นเมสเสจที่ได้มาจากตัวนาฬิกาของระบบ

ทาง สครอลล์บาร์ เกิดขึ้นเนื่องจากผู้ใช้มีการกระทำกับ สครอลล์บาร์

ทางเมนู เกิดจากผู้ใช้มีการเลือกเมนูของโปรแกรม

สำหรับอินพุทที่มาจากคีย์บอร์ด เมาส์ ไทเมอร์ จะเป็นอินพุทที่มาจากฮาร์ดแวร์โดยตรง ซึ่ง วินโดวส์ จะผ่านเข้ามาให้โปรแกรมทางคิวของระบบ ส่วนที่มาจากการแปรงตัวอักษร จากเมนู หรือสครอลล์บาร์ เป็นอินพุทที่เกิดขึ้นเพื่อสนองตอบการทำงานของเมาส์และคีย์บอร์ด นอกพื้นที่ทำงาน และโดยส่วนมาก วินโดวส์ จะผ่านเมสเสจเหล่านี้ให้กับฟังก์ชันของโปรแกรมโดยตรง

เมสเสจดังกล่าวนี้ยังแบ่งได้เป็นสองชนิดใหญ่ ๆ คือ แบบที่ วินโดวส์ ส่งผ่านเข้ามาทางคิวของระบบ (ในรูปของโครงสร้างแบบ MSG) ให้เมสเสจลูปของโปรแกรมเป็นผู้ดึงขึ้นมา แล้วแจกจ่ายไปยังฟังก์ชันที่เหมาะสม ส่วนอีกแบบหนึ่งคือส่งโดยตรงไปยังฟังก์ชันของโปรแกรมเลย ซึ่งทั้งสองวิธีนี้จะมีการผ่านค่าตัวแปรที่เป็นอากิวเมนต์ 4 ตัวคือ hWnd,message,wParam,lParam

ความแตกต่างของเมสเสจทั้งสองอย่างอยู่ตรงที่ โครงสร้าง MSG จะมีตำแหน่งของเคอร์เซอร์ของเมาส์และเวลาของระบบเพิ่มเข้ามา

รายละเอียดของอินพุทของวินโดวส์มีดังนี้

4.1.1.1.1 คีย์บอร์ด คีย์บอร์ดนับว่าเป็นอุปกรณ์การติดต่อกับวินโดวส์ ที่สำคัญตัวหนึ่ง ซึ่งเมื่อวินโดวส์ รับข้อมูลจากคีย์บอร์ดแล้วก็จะเก็บไว้ในคิวเมสเสจ ซึ่ง เมสเสจที่เกิดขึ้นเพราะคีย์บอร์ดมีดังนี้

WM_KEYDOWN	เกิดจากการกดคีย์
WM_KEYUP	เกิดจากการปล่อยคีย์
WM_SYSKEYDOWN	เกิดจากการกดคีย์ของระบบ
WM_SYSKEYUP	เกิดจากการปล่อยคีย์ของระบบ

ซึ่งเมสเสจทั้ง 4 นี้จะอยู่ในส่วนของ wParam จะเป็นรหัสของคีย์ที่ถูกกด ส่วนคีย์พิเศษ เช่น กด

คอนโทรลค้าง(Ctrl) หรือกดชิพค้าง(Shift) จะอยู่ใน lParam ซึ่งโปรแกรมที่จะได้รับ เมสเสจ ของคีย์บอร์ดนี้จะเป็นโปรแกรมที่ได้รับโฟกัสของอินพุต (Focus input) หรืออาจเรียกว่าเป็นโปรแกรมที่แอคทีฟอยู่นั่นเอง

เมื่อรับเมสเสจของคีย์บอร์ดเข้ามาแล้ว โปรแกรมจะมีการแปลงรหัสเหล่านั้น ให้เป็นรหัสแอนซิ (ANSI) โดยฟังก์ชัน TranslateMessage ซึ่งจะแปลงรหัสเป็นแอนซิ และสร้างเมสเสจ WM_CHAR หรือ WM_SYSCHAR โดยที่รหัสนั้นจะเก็บในตัวแปร wParam

การรับอินพุทจากการแปลงตัวอักษร

โปรแกรมที่ต้องการรับคีย์โดยรับเป็นเมสเสจของตัวอักษรเข้ามา จะต้องมีการใช้ฟังก์ชัน TranslateMessage ซึ่งฟังก์ชันนี้จะแปลงคีย์ที่กดให้เป็นรหัส แอนซิ และสร้างเมสเสจ WM_CHAR หรือ WM_SYSCHAR ขึ้นมา โดยเก็บรหัส ANSI ของตัวอักษรนั้นในตัวแปร wParam

4.1.1.1.2 เมาส์ สำหรับ เมสเสจ ที่เกิดจากเมาส์นั้นมีมากมาย แต่ที่เราใช้ในโปรแกรม จะใช้เฉพาะเมสเสจ WM_LBUTTONDOWN และ WM_LBUTTONUP เนื่องจากว่า มือคนเรานั้นไม่แน่นอนทำให้การรับ เมสเสจ ตัออื่น ๆ ทำได้ไม่ดีนัก การเก็บตำแหน่งของการกดเมาส์จะเก็บในตัวแปร lParam ส่วนการเก็บสถานะของการกดคีย์บอร์ขณะที่มีการกดเมาส์นั้นจะเก็บในตัวแปร wParam

วินโดวส์จะส่งเมสเสจ เหล่านี้ให้กับวินโดว์ที่เคอร์เซอร์อยู่ในบริเวณของวินโดว์นั้น หากเลื่อนเคอร์เซอร์ออกจากวินโดว์นั้นแล้ว ก็จะไม่มีการส่ง เมสเสจ ออกให้กับวินโดว์เดิมอีก แต่เราอาจให้มีการส่งเมสเสจ ให้กับวินโดว์ใดโดยเฉพาะโดยใช้ฟังก์ชัน SetCapture ซึ่งมักจะใช้ในกรณีที่กำลังทำงานสำคัญกับเมาส์อยู่ เมื่อทำงานนั้นเสร็จแล้ว ก็ควรปล่อยให้การส่งเมสเสจเป็นไปตามปกติโดยใช้ฟังก์ชัน ReleaseCapture

4.1.1.1.3 ไทเมอร์ สำหรับในโปรแกรมที่ต้องการให้มีการทำงานเป็นคาบ ๆทุกครั้งที่ถึงช่วงเวลาหนึ่งนั้น เราสามารถทำได้โดยกำหนดเวลา หรือตั้งเวลา เพื่อให้ วินโดวส์ ส่ง เมสเสจ มาเมื่อถึงเวลาที่กำหนด การตั้งเวลานั้นทำโดยใช้ฟังก์ชัน SetTimer ซึ่งจะเป็นการกำหนดเวลาในหน่วนมิลลิวินาทีให้วินโดวส์ ส่ง เมสเสจ WM_TIMER ทุกครั้งที่ถึงเวลาที่กำหนดไว้ ซึ่งการส่ง เมสเสจ นี้อาจส่งไปยังคิวของระบบ หรือส่งไปยังฟังก์ชันที่ต้องการก็ได้ เมื่อเราต้องการยกเลิกใช้ไทเมอร์นั้น ก็ควรทำลายไทเมอร์ตัวนั้นโดยใช้ฟังก์ชัน KillTimer

การรับอินพุทจาก สครอลล์บาร์ เมสเสจที่จะได้จาก สครอลล์บาร์ เมื่อผู้ใช้คลิกที่ลูกศร ของสครอลล์บาร์ คือ WM_HSCROLL หรือWM_VSCROLL ซึ่งส่วนมากจะใช้กับ วินโดว์ที่ไม่สามารถ แสดงข้อมูลได้ทั้งหมดในหน้าจอเดียว การที่วินโดว์จะมี สครอลล์บาร์ ได้นั้นต้องเป็นวินโดว์ที่มีการบอกสไตล์เป็น WS_HSCROLL หรือ WS_VSCROLL ตอนที่สร้างวินโดว์นั้น ซึ่งเมื่อสร้างวินโดว์โดยบอกสไตล์ เป็น WS_HSCROLL หรือ WS_VSCROLL แล้ว วินโดว์นั้นจะแสดง สครอลล์บาร์ และพร้อมที่จะรับเมสเสจจาก

สครอลล์บาร์ โดยอัตโนมัติ เมสเสจที่รับมานั้นจะมีชนิดของการใช้ สครอลล์บาร์ ตามมากับเมสเสจใน wParamด้วย ดังนั้นเมื่อตรวจค่าที่ส่งมาใน wParam ก็จะทราบได้ว่ามีการใช้งาน สครอลล์บาร์ นั้นอย่างไร

การรับอินพุทจากเมนู

เมื่อผู้ใช้เลือกคำสั่งจากเมนูต่าง ๆ วินโดว์ที่เป็นเจ้าของเมนูนั้นจะได้เมสเสจ 2 ชนิดจากเมนู คือ

WM_SYSCOMMAND เกิดขึ้นเมื่อผู้ใช้เลือกคำสั่งในเมนูของระบบ

WM_COMMAND เกิดขึ้นเมื่อผู้ใช้เลือกคำสั่งในเมนูธรรมดาของโปรแกรม

4.1.1.2 การทำเมสเสจลูป

เมื่อเรากดคีย์บอร์ดหนึ่งคีย์ วินโดวส์ จะรับทราบและสร้างเมสเสจสำหรับคีย์บอร์ดที่เหมาะสมวางลงบน เมสเสจคิวของระบบ แล้วส่งต่อไปยังคิวประจำโปรแกรมของโปรแกรมที่เหมาะสม จากนั้นเมสเสจจะถูกอ่านแล้วตีความ ซึ่งก็จะเป็นเมสเสจของตัวหนังสือในรูปของรหัสแอนซิ(ANSI) ผ่านมาทางเมสเสจชื่อ WM_CHAR แล้วแจกจ่ายเมสเสจของคีย์บอร์ด ไปยังฟังก์ชันประจำโปรแกรมที่เหมาะสม หากฟังก์ชันประจำโปรแกรมนั้นตอบสนองเมสเสจด้วยการแสดงตัวหนังสือออกทางจอภาพ ก็จะใช้ฟังก์ชัน TextOut ในการแสดงตัวอักษรบนพื้นที่ทำงานของวินโดว์นั้น

วินโดวส์ ไม่เพียงแต่สามารถจัดการเมสเสจให้แก่โปรแกรมเดียวเท่านั้น แต่ยังสามารถรวบรวมเมสเสจและแจกจ่ายไปยังโปรแกรมต่าง ๆ หลาย ๆโปรแกรมพร้อมกันได้ ซึ่งแต่ละโปรแกรมก็จะมี เมสเสจลูป ของตน เพื่อคอยรับและแจกจ่าย ไปให้กับฟังก์ชันประจำโปรแกรมของตน

การส่งเมสเสจ นอกจากจะส่งทางคิวประจำโปรแกรมแล้ว วินโดวส์ ยังสามารถส่งโดยตรงได้ด้วยเมสเสจบางอย่างเช่น เมื่อ วินโดวส์ ตอบรับการทำลายวินโดว์ ก็จะเริ่มส่งเมสเสจชื่อ WM_DESTROY ไปยังฟังก์ชันประจำโปรแกรมโดยตรง (โดยไม่ผ่านเมสเสจคิวเลย) หลังจากนั้น ฟังก์ชันประจำโปรแกรมก็จะตอบสนองโดยการส่งสัญญาณให้แก่ วินโดวส์ ว่าวินโดว์ได้ถูกทำลายแล้ว โปรแกรมควรจะเลิกทำงาน ก็จะใช้ฟังก์ชัน PostQuitMessage ส่งเมสเสจชื่อWM_QUIT ไปยังคิวของโปรแกรมนั้น และเมื่อเมสเสจลูปได้รับเมสเสจ WM_QUIT ก็จะเลิกทำงาน

บทที่ 5
แผนการทำงาน

ในการทำโครงงานครั้งนี้ได้มีการแบ่งขั้นตอนการดำเนินงานดังต่อไปนี้

5.1 การศึกษาการเขียนโปรแกรม

เนื่องจากเราได้เลือกใช้ระบบปฏิบัติการบนวินโดวส์ ดังนั้น จึงต้องมีการศึกษาการเขียนโปรแกรมบนระบบวินโดวส์ ซึ่งเราได้นำหลักการสร้างภาพบนวินโดวส์มาใช้ ซึ่งในการสร้างภาพบนวินโดวส์นั้นเราสามารถทำได้หลายวิธีด้วยกันคือ

5.1.1 สร้างภาพแล้วทำการดึง(load)ภาพนั้นเข้ามาใช้

การสร้างภาพจากภายนอกโปรแกรมแล้วทำการดึงรูปภาพเหล่านั้นเข้ามาโดย การอ่านจากไฟล์ภาพเข้ามา ซึ่งต้องมีการตรวจสอบว่าเป็นไฟล์ภาพชนิดไหน และต้องการหน่วยความจำขนาดไหน หลังจากนั้นก็อ่านส่วนต่าง ๆ ของไฟล์เข้าสู่หน่วยความจำ และนำค่าอ้างอิงของไฟล์นั้น มาใช้ในการอ้างอิงถึงภาพนั้น ๆ

5.1.2 นำภาพไปเป็นส่วนหนึ่งของรีซอร์ส

การสร้างภาพจากโปรแกรมภายนอก แล้วนำภาพเหล่านั้นเก็บเป็นรีซอร์สส่วนหนึ่ง เมื่อต้องการเรียกใช้ไฟล์นั้น ก็สามารถอ้างได้โดยการดึงออกมาจากรีซอร์สที่ได้กำหนดไว้ โดยค่าอ้างอิงจะได้จากการดึงรีซอร์สมาใช้งาน ซึ่งการสร้างภาพจากโปรแกรมภายนอกนี้ บางทีอาจเป็นการสแกนภาพซึ่งอาจวาดหรือถ่ายเข้ามาก็ได้ แล้วเปลี่ยนให้เป็นฟอร์แมท(format)บีเอ็มพี(bmp) ซึ่งสามารถนำมาเป็นส่วนหนึ่งของรีซอร์สได้

5.1.3 สร้างภาพโดยใช้ฟังก์ชันในการวาดรูปของวินโดวส์

การสร้างภาพโดยใช้ฟังก์ชันที่มีในวินโดวส์ ในการสร้างภาพลักษณะนี้ต้องมีการบอกรายละเอียดต่างๆ ของภาพให้กับฟังก์ชันเหล่านั้น เช่นการสร้างวงกลม หรือการสร้างสี่เหลี่ยม ซึ่งต้องรู้จุดพิกัดต่าง ๆ และขนาดที่ต้องการสร้าง ซึ่งในวินโดวส์มีฟังก์ชันให้สร้างภาพเหล่านี้มากมาย และยังมีรูปแบบในการสร้างมากมาย สำหรับรายละเอียดเรื่องการเขียนโปรแกรมต่าง ๆ นั้น ผู้สนใจสามารถอ่านได้จากหนังสืออ้างอิงท้ายเล่ม

5.2.การทำภาพเคลื่อนไหว

ในการทำภาพเคลื่อนไหวนั้นมีด้วยกันหลายวิธี ซึ่งอาจแบ่งได้ดังนี้คือ

5.2.1 การสร้างภาพเคลื่อนไหวโดยวาดภาพใหม่ขึ้นมา

การสร้างภาพเคลื่อนไหวชนิดนี้เป็นการสร้างขึ้นมาใหม่ซึ่งเป็นภาพที่มีการเคลื่อนไหวทั้งหมด เมื่อ

เรานำเอาภาพต่าง ๆ เหล่านี้มาแสดงบนจอภาพแล้วและแสดงภาพต่าง ๆ ตามลำดับโดยมีจังหวะเวลาที่ได้กำหนดไว้ ก็จะทำให้เห็นว่า ส่วนต่าง ๆ เกิดการเคลื่อนไหว

5.2.2 การสร้างภาพเคลื่อนไหวโดยการวาดเฉพาะส่วนที่เคลื่อนไหว

การสร้างภาพเคลื่อนไหวนี้ เป็นการเคลื่อนไหวเพียงบางส่วนของรูปทั้งหมด นั่นคือจะไม่วาดภาพทั้งหมดที่จะแสดงบนจอ จะมีฉากหลังที่ไม่เคลื่อนไหว และมีส่วนที่เคลื่อนไหวซึ่งก็เป็นส่วนย่อย ๆ ของภาพนั้น ซึ่งการสร้างภาพเคลื่อนไหวนี้ ขนาดของวัตถุที่เคลื่อนไหวนั้นจะมีขนาดเล็ก ทำให้ใช้หน่วยความจำน้อยลง ซึ่งจะใช้ในการเคลื่อนไหวเฉพาะรายละเอียดบางส่วนเท่านั้น

5.3.การสร้างเสียงในโปรแกรม

การสร้างเสียงในโปรแกรมเรานั้นโดยการ สั่งให้การ์ดเสียงอ่านไฟล์ที่เก็บไว้ ซึ่งก็จะถูกควบคุมโดยโปรแกรม นั่นก็คือเมื่อถึงจุดที่ต้องอ่านเสียงอะไร ก็ส่งคำสั่งเพื่อให้อ่านไฟล์เสียงนั้นจากไฟล์ที่เราเก็บไว้ ซึ่งในบางครั้งเราอาจให้มีการเล่นเครื่องเล่นเสียงไปพร้อมกันกับการอ่านจากไฟล์เสียง

5.3.1การเล่น file เสียงโดยติดต่อกับ Sound Card บน Windows

ใน Windows มีคำสั่งเกี่ยวกับการเล่น file เสียงโดยการแบ่งการเปิด file เสียงแบบ low-level(เป็นการติดต่อกับ Sound Card โดยตรง) และ high-level(เป็นการติดต่อกับ file เสียงโดยการส่ง message ไปให้กับ windows แล้ว windows จะตอบสนองการทำงานให้) สำหรับการทำงานแบบ low-level นั้นเราสามารถปรับ Parameter(คือ ค่าตัวแปร ต่างๆ ซึ่งใช้กับ Sound Card ชนิดนั้นๆ)ต่างๆ ของ Sound Card ได้ เช่น volumn(ความดังของ file เสียง) ความเร็วในการเล่น file เสียง ซึ่ง parameter พวกนี้จะสามารถทำงานได้กับ Sound Card แบบใดแบบหนึ่งเท่านั้น

เราแบ่งกลุ่มของคำสั่งในการติดต่อกับ Sound Card ตามลำดับความสามารถในการทำงานของคำสั่งเหล่านั้น

5.3.1.1. MessageBeep (Message) ซึ่ง Message นี้ได้แก่ MB_OK,MB_ICONASTERISK, MB_ICONEXCLATION, MB_ICONHAND, MB_ICONQUESTION ซึ่ง Message เหล่านี้เราสามารถระบุได้ว่าจะให้เป็น file เสียง file อะไร โดยจะตั้งค่าเหล่านี้ใน Windows โดย icon Sound ใน icon Control Panel

5.3.1.2. sndPlaySound เป็นคำส่งในการเล่น file เสียง โดยเราสามารถ หยุด file เสียงได้โดยการส่ง parameter NULL และสามารถเล่น file เสียงได้ทั้งเล่นแบบ Sync (โปรแกรมคอยการเล่น file เสียง) และแบบ Async (โปรแกรมส่งคำสั่งแล้วทำงานอย่างอื่นต่อไป) ได้ และยังสามารถ เล่น file เสียงซ้ำกับ เป็น Loop(คือเล่นแล้วแล่นอีกจนกว่าจะสั่งหยุด)ได้อีกด้วย

แต่ คำสั่งนี้จะเล่น file เสียงขนาดใหญ่มากไม่ได้ซึ่งหมายความว่าจะเล่นแบบ file เดียวเป็นระยะเวลานาน ๆ มากไม่ได้ และการหยุดนี้ยังไม่สามารถกลับมาเริ่มตำแหน่งเดิมได้ต้องกลับไปเริ่มต้นเล่น file

เสียงเริ่มต้นใหม่หมด

5.3.1.3. กลุ่มคำสั่ง MCI เป็นกลุ่มคำสั่งที่ใช้ในงาน Multi media โดยเป็นคำสั่งที่ใช้ในการติดต่อกับ device(อุปกรณ์)ต่างๆ ทั้งอุปกรณ์ภายนอก และ อุปกรณ์ภายในชนิดของ device ต่างๆ

การใช้คำสั่งในกลุ่ม MCI นั้นเป็นการใส่ค่าตัวแปรลงใน Structure(ตัวแปรชนิดโครงสร้าง) แล้วแต่ชนิดของ Parameter ใน MCI เช่น ในการเปิด file เราจะส่งตัวแปรโครงสร้าง MCI_OPEN_PARMS ไปถ้าเป็นการ play เราจะส่งตัวแปรโดยใส่ข้อมูลโครงสร้างแบบ MCI_PLAY_PARMS และ ถ้าเป็นการหยุดเรา จะส่งค่าตัวแปรโดยใส่ข้อมูลชนิด MCI_GENERIC_PARMS เป็นต้น เราจะทำการใส่ค่าตัวแปรลงในโครงสร้างดังกล่าวแล้วทำการเรียกคำสั่ง mciSendComand เพื่อให้ทำงานตามนั้น และการส่งคำสั่งนี้สามารถตามด้วย MCI_NOTIFY ซึ่งเมื่อทำงานในคำสั่งดังกล่าวเสร็จแล้ว Windows จะทำการส่ง Message MM_MCINOTIFY มายัง Windows และ ถ้ามี MCI_WAIT Windows จะทำการคอยให้คำสั่ง MCI นั้นทำงานเสร็จเสียก่อนจะ return การทำงานให้โปรแกรมอีกครั้งหนึ่งซึ่ง Message ทั้งสองนี้สามารถใช้คู่กันได้และไม่ใช้ก็ได้ด้วย

การทำงานกับ file เสียง ด้วยคำสั่งในกลุ่มของ MCI นี้สามารถทำงานได้ทั้ง file ขนาดใหญ่และ fileขนาดเล็ก สามารถทำการหยุด file เสียงนั้นชั่วคราว สามารถตั้งตำแหน่งให้เริ่มทำงานและหยุดได้อีกด้วย การทำงานโดยใช้คำสั่ง MCI นี้จะง่ายต่อการใช้งานคือสามารถทำงานได้โดยคำสั่งเดียวซึ่งจะเขียนโปรแกรมสะดวก

5.3.1.4. กลุ่มของคำสั่ง waveOut การใช้คำสั่ง waveOut นี้เราจะต้องเป็นการจัดการในการนำ file เสียงขึ้นมาใน buffer ก่อนแล้วทำการส่งไปยัง Sound Card การที่เราจะนำ file เสียงขึ้นมาใน buffer นั้นเราจะต้องสร้าง Buffer ซึ่ง Buffer ที่ใช้ในงานนี้เรียกว่า RIFF (Resource Interchange File Format)

RIFF file นี้ มีลักษณะคล้าย link list คือ จะมีข้อมูลแบบโครงสร้างคือมีชนิดของ buffer ซึ่งอยู่ใน record ของ FOURCC ซึ่งเป็นตัวเก็บ character 4 ตัวในการบอกถึงลักษณะของข้อมูลเหล่านั้นแบ่งออกเป็น

PAL = Palette File Format

RDIB = RIFF Device Independent Bitmap Format

RMID = RIFF MIDI Format

RMMD = RIFF Multimedia Movie File Format

WAVE = Waveform Audio Format

ซึ่งส่วนที่กำหนดชนิดของ buffer เหล่านี้เราจะต้องบอกเป็นอันดับแรก และทำการสร้าง buffer ที่ชี้ข้อมูลตัวถัดไปโดยการใช้คำสั่ง mmioDescend ซึ่งเป็นการชี้ข้อมูลตัวถัดไปเป็นชิ้นเป็น Header file ตามชนิดของ Resource ซึ่งในการเก็บข้อมูลนี้เราจะต้องทำเป็นตัวแปร pointer ซึ่งชี้ไปยังที่เก็บตัวแปรดังกล่าว โดย

ส่วน Header file นี้เราจะทำการให้ FOURCC = fmt และเราจะต้องทำการลงไปแล้วสร้าง ส่วนที่เก็บข้อมูลซึ่งเราให้ FOURCC = DATA ซึ่งเป็นการเก็บข้อมูลถัดไป

การใช้สั่งกลุ่ม waveOut นี้ประกอบด้วย

ก่อนที่เราจะเริ่มเล่น file เสียงเราต้องตรวจสอบความสามารถของ Hardward(Sound Card) เสียก่อนเริ่มการทำงาน เช่น ความสามารถของ Sound Card และเลขที่ของ Sound Card หลังจากนั้นเราจะเริ่มการทำงานโดยการ waveOutOpen เป็นการเริ่มการทำงานก่อนที่จะเล่น file เสียง เมื่อเราเปิด file เสียงสำเร็จเราจะต้องตรวจสอบ ขนาดของ buffer ของ Sound Card ก่อนที่เราจะเริ่มส่งข้อมูลไปยัง Sound Card เราใช้คำสั่ง waveOutprepareHeader ซึ่งเป็นคำสั่งในการ ปล่อยข้อมูลใน buffer และทำการจองหน่วยความจำในการเขียน file เสียงลงไป และเราจะใช้คำสั่ง waveOutWrite เพื่อที่จะเขียนข้อมูลของ file เสียงลงใน Sound Card เพื่อที่จะเล่น file เสียง การใช้คำสั่ง waveOut นี้ยังมีคำสั่งในการตั้งความดังของเสียงด้วยคือ waveOutSetVolumn ซึ่งสามารถตั้งค่าได้เฉพาะ Sound Card ที่มีการทำงานนี้เท่านั้น นอกจากนี้การทำงานของคำสั่งกลุ่ม waveOut นี้คล้ายกับ MCI คือเมื่อทำงานเสร็จคือMessage MM_MOM_CLOSE,MM_MOM_OPEN เมื่อWindowsจบการทำงานตามที่ระบุ คือ เปิด และ ปิด และ MM_MOM_DONE เป็น Message ที่ส่งมาให้ถ้าคำสั่งนั้นต้องการโดยระบุที่คำสั่งที่ทำการเรียกคำสั่งนั้น

การเปรียบเทียบการใช้กลุ่มคำสั่ง waveOut และกลุ่มคำสั่ง MCI

คำสั่งกลุ่ม waveOut สามารถใช้ความสามารถสูงสุดของ Sound Card ซึ่งดีกว่าการใช้คำสั่ง MCI แต่คำสั่งกลุ่ม MCI นี้ดีกว่าคือเราไม่ต้องทำการนำ file เสียงขึ้นมาในหน่วยความจำเองแต่ทำการให้ที่อยู่ของ file เสียงที่ต้องการเล่นแล้ว Windows ก็จะทำการนำ file เสียงนั้นขึ้นมาใส่ในหน่วยความจำเองและสามารถใช้คำสั่งเดียวในการเล่น fileเสียง ซึ่งจะทำให้การทำงานสะดวกกว่า และการที่ให้ Windows จะการเองนี้จะลดความผิดพลาดลงได้มาก

Windows Media Control Interface (MCI)

MCI เป็นกลุ่มของคำสั่งเป็นวิธีการควบคุมอุปกรณ์ multi media ต่างๆ ซึ่งที่อุปกรณ์ต่างๆ ใช้ งานเรียกว่า resource

MCI เป็นรูปแบบคำสั่งที่ไม่ขึ้นกับรูปแบบของโปรแกรมต่างๆ และ อุปกรณ์ multi media Hardware MCI นี้ใช้คำสั่งตัวอักษรให้โปรแกรมทำงานเช่น open play และ close ซึ่งใช้ควบคุมอุปกรณ์ multimedia โดยการส่งคำสั่ง mciSendString() ซึ่งฟังก์ชั่นต่างๆ เหล่านี้อยู่ใน file MMSYSTEM.DLL ซึ่งเป็นๆ Windows dynamic link library ซึ่งสำหรับควบคุมอุปกรณ์ต่างๆ ซึ่งคำสั่ง MCI นี้ สามารถใช้กับภาษาได้หลายภาษาเช่น Visual Basic, Turbo Pascal for Windows , C, และ C++ เป็นต้น

การใช้คำสั่ง MCI กับ file เสียง และ audio compact disc

การใช้คำสั่ง MCI กับ audio compact disc โดยเริ่มต้นจาก track 6 สิ้นสุด trace 7 เป็นดังนี้

```
open cdaudio
set cdaudio time format tmsf
play cdaudio from 6 to 7
close cdaudio
```

รูปแบบของการใช้คำสั่ง MCI ในการเล่น file เสียง 10,000 samples แรก

```
open c:\mmdata\redrobin.wav type waveaudio alias robin
set robin time format sammples
play robin from 1 to 10,000 wait
close robin
```

จากตัวอย่างข้างบนนี้เราใช้คำสั่ง MCI แบ่งได้ดังนี้

เป็นคำสั่งพื้นฐานเริ่มต้นด้วย open play และ close

เมื่อเราทำการเปิดเราต้องบอกชนิดของอุปกรณ์ที่เราทำการเปิดด้วยเช่นถ้าเราจะเปิดเพื่อทำงานใน file เสียง เราจะต้องเปิด waveaudio device และ ถ้าเราจะเปิดเพื่อทำงาน audio compact disc เราจะต้องทำการเปิดเป็น cdaudio

เราสามารถใช้คำสั่ง set เพื่อเป็นการระบุเวลาในการเล่น cdaudio และ waveaudio ได้

นอกจากนั้นเรายังสามารถใช้คำสั่ง from กับ to เพื่อเป็นการระบุอีกแบบหนึ่งเช่น waveaudio จะเป็นการระบุจำนวน samples ส่วน cdaudio จะเป็นการระบุช่วงของ trace เป็นต้น

คำสั่งพื้นฐานของ MCI เช่น

load	เป็นการในข้อมูลจาก disk
pause	เป็นการหยุดการทำงานชั่วคราว
play	เป็นการส่งข้อมูลออกไปยังอุปกรณ์
record	เป็นการรับข้อมูลจากอุปกรณ์
resume	เป็นการทำงานต่อเนื่องจากคำสั่ง pause
save	เป็นการเก็บข้อมูลลง disk
seek	เป็นการหาโดยสามารถหาไปข้างหรือ ถอยหลังได้
set	เป็นคำสั่งควบคุมการทำงานของอุปกรณ์
status	เป็นคำสั่งในการรับข้อมูลจากอุปกรณ์

stop เป็นคำสั่งที่หยุดการทำงาน

นอกจากคำสั่งพื้นฐานแล้ว MCI ยังมีคำสั่งเพิ่มเติมอีกเช่น

copy เป็นคัดลอกข้อมูลไปยัง Clipbord

cut เป็นการตัวข้อมูลบางส่วนไปยัง Clipbord

delete เป็นการนำข้อมูลออกจาก MCI

MCI นี้สามารถใช้ได้กับอุปกรณ์

Device Type	คำอธิบาย	MCI driver
AVIVideo	Microsoft Audio Video Interleaved	MCIAVI.DRV
cdaudio	Audio CD player	MCICDA.DRV
dat	Digital audiotype player	ชื่อของบริษัทที่ทำ
digitalvidio	Digital video in a windows	ชื่อของบริษัทที่ทำ
mixer	Media Vision MMMIXER.DLL	MVMIXER.DRV
MMMovie	Multimedia movie player	MCIMMP.DRV
scanner	Image scanner	ชื่อของบริษัทที่ทำ
sequencer	MIDI sequencer	MCISEQ.DRV
vcr	Videotype recorder or player	ชื่อของบริษัทที่ทำ
videodisc	Pioneer LD-V4200 videodisc player	MCIPIONR.DRV
waveaudio	Digitized wavefrom audio device	MCIWAVE.DRV

การใช้ MCI กับ file เสียงทำเริ่มโดยการ

```
MCI_OPEN_PARMS mciopen;
int id;
mciopen.lpstrDeviceType="waveaudio";
mciopen.lpstrElementName="c:\windows\chore.wav"
mciSendCommand(0,MCI_OPEN,MCI_OPEN_TYPE | MCI_OPEN_ELEMENT,
(DWOR)(LPVOID)&mciopen);
id=mciopen.wDeviceID;
```

เมื่อเราทำการเปิด ไฟล์เสียงเราจะต้องส่งค่าผ่าน object MCI_OPEN_PARMS ซึ่งค่าต่างๆ ที่เราทำการส่งเช่นชนิดของ อุปกรณ์ที่เราทำการเปิด และ ชื่อของ file เสียงที่เราทำการเปิดเป็นต้น

เมื่อเราเปิด ไฟล์เสียงสำเร็จแล้วเราต้องทำการเก็บค่า id ของ file นั้นไว้เพื่อทำการทำงานต่อไปเพื่อที่จะอ้างอิงถึง ไฟล์เสียงนั้นได้ ทั้งหมดตามข้างบนนี้เป็นการเปิด device สำหรับเล่น ไฟล์เสียง ซึ่งถ้าเราทำการเปิดไม่สำเร็จเราสามารถรับค่าที่ทำการส่งค่ากลับจากคำสั่ง mciSendcommand ได้แล้วเรานำค่าที่ได้มาทำการเขียนแล้วส่งไปยัง ฟังก์ชั่น mciGetErrorString ได้โดยที่เราจะต้องมีตัวแปรสำหรับรับ message ซึ่งจะให้คำตอบของค่าที่ผิดพลาดได้

ต่อมาเมื่อเราสามารถเปิด ไฟล์ เสียงสำเร็จแล้วเราทำการส่ง file เสียงที่เปิดได้ไปยังอุปกรณ์นั้นเริ่มต้นด้วย

```
MCI_PLAY_PARMS mciplay;
mciplay.dwCallback=(DWORD)hwnd;
mciSendCommand(id,MCI_PLAY,MCI_NOTIFY,
(DWORD)(LPVOID)&mciplay);
```

โดยที่ค่า MCI_NOTIFY เป็นการระบุว่าเมื่อทำงานตามคำสั่งเสร็จแล้วจะทำการส่ง message MM_MCINOTIFY ไปยังโปรแกรมซึ่งเมื่อเรารับ message MM_MCINOTIFY นี้แล้วเราก็สามารถทำงานอย่างอื่นเช่น

```
case MM_NOTIFY:
mciSendCommand(LOWORD(lParam),MCI_CLOSE,MCI_WAIT,NULL);
break;
```

ซึ่งจากข้างบนเมื่อเรารับ message MM_NOTIFY แล้วเราทำการปิด file เสียงนั้นทันทีซึ่งนอกจากระบุค่า MCI_NOTIFY แล้วเรายังสามารถระบุค่าอื่นๆ ได้อีกเช่น ค่า MCI_WAIT ซึ่งเป็นค่าที่ระบุว่าเมื่อส่งค่าคำสั่งนี้แล้วโปรแกรมจะทำการรอจนกว่าจะทำคำสั่งนี้เสร็จซึ่งทั้ง MCI_NOTIFY และ MCI_WAIT นี้สามารถใช้ร่วมกันได้ซึ่งนอกจากนี้ยังมีค่าอื่นๆ ที่สามารถส่งไปได้แต่ขึ้นอยู่กับ คำสั่งเช่น คำสั่ง MCI_PLAY นี้นอกจากจะมี MCI_WAIT กับ MCI_NOTIFY แล้วยังมี MCI_FROM กับ MCI_TO ด้วยเพื่อเป็นการกำหนดช่วงในการเล่น ไฟล์เสียงอีกเป็นต้น

เมื่อเราเริ่มเล่น ไฟล์เสียงแล้วเราจะทำการเลิกเล่นกลางคันได้โดยส่ง message MCI_STOP ไปยัง windows ซึ่งเขียนโปรแกรมได้ดังนี้

```
MCI_GENERIC_PARMS mcigen;
mcigen.dwCallback=hwnd;
mciSendCommand(id,MCI_STOP,MCI_NOTIFYIMCI_WAIT,
```

(DWORD)(LPVOID)&mcigen);

และเมื่อ Stop เสร็จจะทำการส่ง MM_NOTIFY ไปยัง windows แล้วเราจะสามารถทำการปิด file เสียงได้ดังการรับ message ตามข้างบนนี้

การใช้ MCI เหมือนกับการทำ ไฟล์เสียง แต่เราจะต้องเปลี่ยน Device(ชื่อของอุปกรณ์) ให้เป็น cdaudio เราจะทำการเริ่มต้นโปรแกรมด้วย

```
MCI_OPEN_PARMS mciOpen;
int id;
mciOpen.lpstrDeviceType="cdaudio";
mciSendCommand(NULL,MCI_OPEN,MCI_OPEN_TYPE,
(DWORD)(LPVOID)&mciOpen);
id=mciOpen.wDeviceID;
```

จะเห็นว่าการใช้คำสั่ง MCI เป็นคำสั่งที่ร้องขอทำการเปิดอุปกรณ์ต่างๆ นั้นใช้งาน โดยที่เราส่ง Object ไปยัง windows โดยในพารามิเตอร์ MCI_OPEN_PARMS

```
MCI_SET_PARMS mciSet;
mciSet.dwTimeFormat=MCI_FORMAT_TMSF;
mciSendCommand(id,MCI_SET,MCI_SET_TIME_FORMAT,
(DWORD)(LPVOID)&mciSet)
```

ซึ่งคำสั่ง MCI_SET นี้ส่งค่า MCI_SET_TIME_FORMAT เราส่งค่า MCI_FORMAT_TMSF เป็นการสั่งให้เก็บค่าของเวลาเป็น trace,นาที,วินาที,และ frames

ต่อมาเราสามารถรู้จำนวน traces ทั้งหมดโดยการใช้คำสั่ง MCI_STATUS ดังนี้

```
MCI_STATUS_PARMS mciStat;
unsigned int tracks;
mciStat.dwItem=MCI_STATUS_NUMBER_OF_TRACKS;
mciSendCommand(id,MCI_STATUS,MCI_STATUS_ITEM,
(DWORD)(LPVOID)&mciStat)
tracks=(unsigned int)mciStat.dwReturn;
```

เมื่อเรารู้ค่าจำนวน tracks ทั้งหมดแล้วเราสามารถหาค่าของเวลาที่ใช้ในการทำงานแต่ละ tracks ได้ โดยส่งดังตัวอย่าง

```
unsigned int i;
for(i=1;i<=tracks;++i){
mciStat.dwItem=MCI_STATUS_LENGTH;
mciStat.dwTrack=i;
mciSendCommand(id,MCI_STATUS,MCI_STATUS_ITEM | MCI_TRACK,
(DWORD)(LPVOID)&mciStat)
wsprintf(b," Track %02u - %02u:%02u:%02u",i,
MCI_MSF_MINUTE(mciStat.dwRetrun),
MCI_MSF_SECOND(mciStat.dwRetrun),
MCI_MSF_FRAME(mciStat.dwReturn));
}
```

จะเห็นได้ว่าจากตัวอย่างข้างบนนี้เราจะทราบจำนวน tracks และ เวลาในแต่ละ track ซึ่งก็คือเวลาในแต่ละเพลงนั้นเองสุดท้ายนี้เมื่อเราสามารถทำตามข้างบนได้หมดแล้วเราจะทำการสั่งให้มันทำงานโดย

```
mciPlay.dwFrom=MCI_MAKE_TMSF (track,
MCI_MSF_MINUTE(0),
MCI_MSF_SECOND(0),
MCI_MSF_FRAME(0));
mciPlay.dwCallback=(DWORD)hwnd;
mciSendCommand(id,MCI_PLAY,MCI_FROMIMCI_TOIMCI_NOTIFY,
(DWORD)(LPVOID)&mciPlay);
```

เมื่อเราทำการเล่น Compact disc เสร็จเรียบร้อยแล้วเราจะทำการปิดอุปกรณ์ตัวนี้ได้โดย

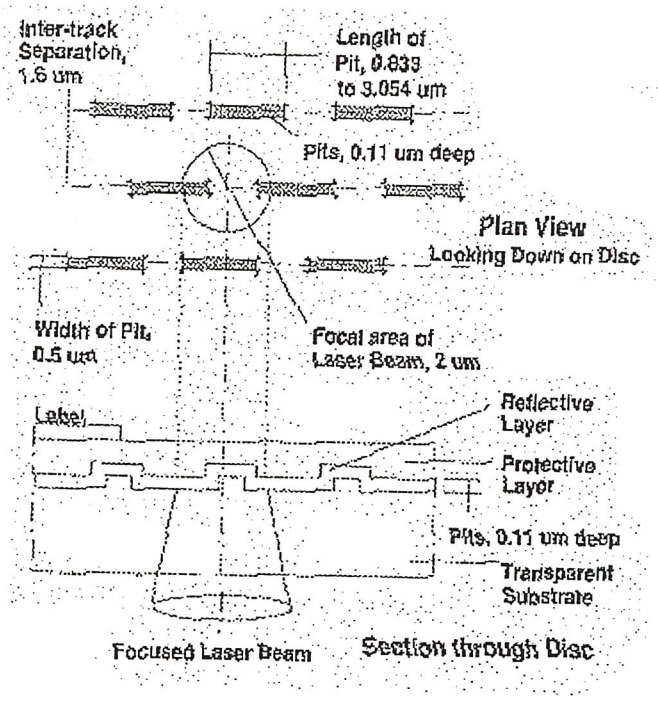
```
mciSendCommand(id,MCI_CLOSE,0,NULL);
```

เป็นการจบการทำงาน

5.3.2 เรื่องเกี่ยวกับ compact disc

compact disc เป็นตัวเก็บข้อมูลของเสียงในลักษณะเช่นเดียวกับไฟล์เสียงโดยที่สามารถเก็บข้อมูลขนาด 16 บิทที่ Sampling rate 44.1 kHz เป็นเวลา 74 นาทีซึ่งเมื่อเทียบกับการเก็บข้อมูลเป็นไบต์แล้วสามารถเก็บได้ถึง 600 M ซึ่งการเก็บข้อมูลใน compact discs นี้อยู่ในรูปขนาดของข้อมูล 16 บิทซึ่งเรียกว่า pit แผ่น compact disc นี้ ทำมาจาก อะลูมีเนียมโดยที่ 1 pit นี้มีขนาด 1 ไมโครเมตร โดยที่ compact disc นี้จะมีตัวอ่านที่เรียกว่า เซมิคอนดักเตอร์ เลเซอร์ (semiconductor laser)ซึ่งจะทำการอ่านค่า pit pit ที่ไม่สะท้อนแสงจะ

ถูกเก็บค่า 1 ในเลขฐาน 2 ส่วน pit ที่สามารถสะท้อนแสงได้จะถูกเก็บเป็น 0 ในเลขฐาน 2



5.4.การติดต่อกับจอสัมผัส

ในการใช้งานจอสัมผัสนั้น จะเป็นเสมือนการใช้งานเมาส์ตัวหนึ่ง ซึ่งต้องมีการเปลียน ไดร์ฟเวอร์ของเมาส์ก่อน ให้เป็นไดร์ฟเวอร์ของจอสัมผัส ซึ่งเมื่อจอสัมผัสได้รับการติดตั้ง แล้ว เราก็สามารถใช้จอสัมผัสแทนเมาส์ได้เลย โดยเมื่อมีการกดที่จอ จะมีสัญญาณ ส่งไปยัง การ์ดแปรสัญญาณของจอ เพื่อเปลี่ยนให้เป็นสัญญาณในลักษณะของเมาส์ จากนั้นสัญญาณนี้จะส่งเข้าไปยังพอร์ตอนุกรมของเครื่อง และตัวโปรแกรมก็จะทำการแปลงสัญญาณที่ได้รับมา ให้เป็นสัญญาณในลักษณะเช่นเดียวกับสัญญาณของเมาส์ แล้วส่งให้โปรแกรมที่ต้องการใช้งาน เพื่อนำสัญญาณนี้ไปใช้งานต่อไป

5.5 ปัญหาที่เกิดขึ้นในการดำเนินงาน

ในการสร้างภาพเคลื่อไหวนั้น เมื่อภาพเคลื่อไหวไป จะเกิดส่วนเดิมของภาพปรากฏบนฉากหลัง และเมื่อรูปเคลื่อไปอีกก็จะมีฉากหลังเพิ่มขึ้นมากอีก ทำให้เมื่อภาพเคลื่อที่ไปก็จะยังมีบางส่วนของภาพเดิมปรากฏบนจอ วิธีแก้คือ วาดภาพโดยให้มีฉากหลังที่ใหญ่เท่ากับหรือมากกว่าส่วนที่ต้องการให้เคลื่อนที่ ซึ่งฉากหลังนี้จะมีสีเดียวกับสีพื้น ทำให้เมื่อวัตถุเคลือนไปส่วนที่เป็นฉากหลังของภาพเดิมก็จะลบรูปเดิม และ

เนื่องจากฉากหลังมีสีเดียวกับสีพื้น ทำให้เราไม่เห็นว่ามีส่วนของภาพเหลืออยู่ในจอ

ในการสร้างภาพนั้น เมื่อเราอ่านภาพและมีการอ่านเสียงไปพร้อมกัน ทำให้การอ่านเสียงเกิดขาดจังหวะเป็นช่วง ๆ ทำให้ได้เสียงที่ไม่เรียบ การแก้ไขคือทำการอ่านภาพจำนวนเท่าที่เราต้องการก่อน จากนั้นจึงรันโปรแกรม เมื่อมีการอ่านเสียงก็จะไม่มีการแย่งกันอ่านภาพอีก ทำให้การอ่านเสียงเรียบขึ้น ซึ่งก็ทำให้ได้เสียงที่เรียบขึ้น

ในการใช้รีซอร์สนั้น เมื่อมีการใช้แล้วและเมื่อเลิกใช้แล้ว ต้องทำการคืนรีซอร์สนั้น เพราะรีซอร์สมีจำนวนจำกัด ถ้าเกิดใช้แล้วไม่มีการคืนจะทำให้รีซอร์สถูกใช้ไปจนหมด และโปรแกรมก็ไม่สามารถรันได้ต่อไป

บทที่ 6
การเก็บข้อมูล การนำลงคอมพิวเตอร์

6.1 การสร้างภาพและการเก็บภาพ

รูปภาพที่เราสร้างนั้นเป็นการนำเอาจุดต่าง ๆ มาประกอบกันเป็นภาพ โดยที่จุดเหล่านี้จะใช้หน่วยความจำ ในการกำหนด ซึ่งจะบอกว่าจุดนั้ ๆ มีสีอะไร เมื่อนำจุดต่าง ๆ มาเรียงกันประกอบเป็นภาพ ก็จะได้ภาพที่มีทั้งส่วนที่มีสีต่าง ๆ และส่วนที่มืด รูปภาพที่แสดงด้วยจุดต่าง ๆ นี้เราจะเรียกว่า
บิทแมบ (Bitmap) ซึ่งบิทแมทที่ใช้นี้มีลักษณะโครงสร้าง2 รูปแบบ คือ

6.1.1. DIB (Device Independent Bitmap) ซึ่งได้แบบออกเป็น 2 แบบคือ

ของ วินโดวส์ 3.0

ของ Presentation Manger 1.2

6.1.2. DDB (Device Dependent Bitmap)

DIB นี้เป็นการเก็บรูปภาพโดยทั่วไปส่วน DDB นี้จะเป็นรูปภาพที่แสดงออกมาในจอภาพในขณะใดขณะหนึ่งซึ่งก็คือ DIB เปลี่ยนเป็น DDB ก่อนนำแสดงออกมาทางจอภาพนั้นเอง ที่จำเป็นต้องเก็บเป็น DIB เพราะว่า driver ของจอภาพมีหลายลักษณะนั้นเอง

โครงสร้างของ DIB แบ่งออกเป็นส่วนใหญ่ 4 ส่วนคือ

1. A file header
2. A bitmap information header
3. ตารางสี
4. An array ของข้อมูลที่ถูกออกแบบเป็น bitmap

1. ส่วน Bitmap File Header แบ่งคือ

```
typedef struct tagBITMAPFILEHEADER {
WORD bfType;
DWORD bfSize;
WORD bfReserved1;
WORD bfReserved2;
DWORD bfOffBits;
} BITMAPFILEHEADER;
```

ซึ่ง bfType เป็นการระบุถึงชนิดของ file ซึ่งต้องเป็นค่า 0x4D42

bfSize เป็นการบอกขนาดของ file เป็นหน่วย byte

bfReserved1,bfReserved2 ต้องมีค่าเท่ากับ 0

bfOffBits เป็นการระบุถึงขนาดของข้อมูลโครงสร้าง BITMAPFILEHEADER

2. ส่วน Bitmap Information Header ซึ่งส่วนนี้แบ่งออกเป็น 2 ส่วนย่อยซึ่งในส่วนย่อยนี้ ส่วนของ bitmap information header ซึ่งจะเป็นส่วนที่แตกต่างกันสำหรับ file แบบ วินโดวส์ 3.0 กับแบบ Presentation Manger 1.2

```
typedef struct tagBITMAPINFO {
BITMAPINFOHEADER bmiHeader;
RGBQUAD          bmiColors[1]
}BITMAPINFO;
```

ซึ่งในส่วน BITMAPINFOHEADER นี้แบ่งออกเป็น

ของ วินโดวส์ 3.0

```
typedef struct tagBITMAPINFOHEADER {
DWORD biSize;
DWORD biWidth;
DWORD biHeight;
WORD  biPlanes;
WORD  biBitCount;
DWORD biCompression;
DWORD biSizeImage;
DWORD biXPelsPerMeter;
DWORD biYPelsPerMeter;
DWORD biClrUsed;
DWORD biClrImportant;
} BITMAPINFOHEADER;
```

ของ Presentation Manager 1.2

```
typedef struct tagBITMAPINFOHEADER {
```

```
    DWORD biSize;
    DWORD biWidth;
    DWORD biHeight;
    WORD  biPlanes;
    WORD  biBitCount;
} BITMAPCOREHEADER;
```

ซึ่ง biSize เป็นตัวบอกขนาดของ bitmap info header ซึ่งจะไม่เท่ากันระหว่าง วินโดวส์ 3.0 กับ Presentation Manager 1.2

biWidth ความกว้างของ bitmap ในหน่วยของ pixels

biHeight ความสูงของ bitmap ในหน่วยของ pixels

biPlanes เป็นการระบุรานาดของอุปกรณ์ปลายทาง ต้องเท่ากับ 1

biBitCount เป็นการระบุจำนวน bit ใน 1 pexel ซึ่งเป็นตัวแปรค่าในส่วนของ Color Table

ส่วนของ Bitmap Color Table ซึ่งเป็นข้อมูลแบบโครงสร้างซึ่งมีรูปแบบดังนี้

```
typedef struct tagRGBQUAD {
BYTE rgbBlue;
BYTE rgbGreen;
BYTE rgbRed;
BYTE rgbReserved;
} RGBQUAD;
```

ซึ่ง rgbBlue เป็นส่วนระบุความหนาแน่นของสีฟ้า ,สีน้ำเงิน

rgbGreen เป็นส่วนระบุความหนาแน่นของสีเขียว

rgbRed เป็นส่วนระบุความหนาแน่นของสีแดง

rgbReserved ส่วนนี้ต้องมีค่าเป็น 0

ซึ่งการแปรค่าในส่วนของตารางสีนี้จะแปรค่าจากตัวแปรโครงสร้าง biBitCount ซึ่งได้กล่าวดังข้างต้น คือ

biBitCount = 1 แสดงว่ารูปภาพ bitmap นี้เป็นภาพสำหรับจอ Monochrome

biBitCount = 4 แสดงว่ารูปภาพ bitmap นี้เป็นภาพสี 16 สี

biBitCount = 8 แสดงว่ารูปภาพ bitmap นี้เป็นภาพสี 256 สี

biBitCount = 24 แสดงว่ารูปภาพ bitmap นี้เป็นภาพสีซึ่งแสดงสีได้สูงสุด 2 กำลัง 24 สี

ซึ่งการแสดงรูปภาพ bitmap นี้ วินโดวส์ ได้มีคำสั่งที่สนับสนุนมากพอสมควรโดยที่ วินโดวส์ นี้มีตัวแปร HBITMAP ซึ่งทำการเก็บค่า DDB ซึ่งเป็น HANDLE ซึ่งทำการเก็บค่าของรูปภาพซึ่งแปรงมาจาก DIB ตามที่ได้กล่าวมาข้างต้น นอกจากนี้ วินโดวส์ ได้มีการเก็บรูปภาพ DIB ไว้ Resource ซึ่งจะทำการรวมกับโปรแกรมก่อนที่โปรแกรมเราจะทำงานได้อีกด้วยโดยใช้คำสั่ง LoadBitmap แล้วในค่าที่ได้เก็บได้ในตัวแปร HBITMAP อีกด้วย นอกจากนี้ วินโดวส์ ยังมีคำสั่งที่สำหรับนำเอารูปภาพ bitmap ออกหน้าจอซึ่งมี 2 คำสั่งใหญ่ ๆคือ

1. BitBlt ซึ่งเป็นคำสั่งที่นำเอารูป DDB ออกจอ
2. SetDIBitsToDevice ซึ่งเป็นคำสั่งที่นำเอารูป DIB ออกสู่จอภาพ

การส่งรูปภาพออกสู่จอภาพ โดยคำสั่ง BitBlt เป็นคำสั่งที่สามารถนำรูปภาพออกสู่จอภาพได้รวดเร็วที่สุดเพราะเป็นการนำข้อมูลจากหน่วยความจำออกสู่หน่วยความจำของจอภาพซึ่งการใช้คำสั่งนี้เราจะต้องมีการสร้างรูปภาพ bitmap ในหน่วยความจำเสียก่อนซึ่งคำสั่งโดยทั่วไปนี้เป็นคำสั่งเช่น ถ้าเราจะเอารูปภาพ bitmap จาก resource เราจะใช้คำสั่ง LoadBitmap ดังที่ได้กล่าวดังข้างบนนี้ แต่ ถ้าเราจะนำรูปภาพ DIB ออกสู่จอภาพโดยวิธีนี้เราต้องทำการเปลี่ยนให้เป็น DDB เสียก่อนซึ่ง DIB นี้จะเป็น file จุด bmp ข้างนอกซึ่งคำสั่งที่เราจะใช้ก็คือ CreateDIBitmap ซึ่งเราจะได้ DDB ออกมาแล้วถึงใช้คำสั่ง BitBlt ออกสู่จอภาพอีกที นอกจากเราจะเอารูปภาพจากสองแหล่งตามที่กล่าวมาแล้วเรายังสามรถสร้างตัวแปร HBITMAP ขึ้นเองแล้วทำการวาดรูปลงไปในโปรแกรมหรือทำเพื่อประมวณผลภาพซึ่งเราจะใช้เป็น temporary โดยการทำคำสั่งเช่นAND ,OR ,NOT ได้อีกคือคำสั่ง CreateCompatibleBitmap ได้อีกด้วย

การส่งรูปภาพออกสู่จอภาพโดยคำสั่ง SetDIBitsToDevice ซึ่งเป็นคำสั่งที่ทำงานได้ช้ากว่าคำสั่ง BitBlt แต่ข้อดีกว่าตรงที่เปลืองหน่วยความจำน้อยกว่าเพราเราไม่ต้องเสียหน่วยความจำในการสร้าง HANDLE HBITMAP ซึ่งเป็น DDB ซึ่งมีผลให้เปลืองเนื้อที่ในหน่วยความจำมากกว่า

การทำรูป animation เป็นการทำรูปภาพเคลื่อนไหวนี้เราจะต้องใช้ DDB ซึ่งจะสามารถทำงานได้อย่างรวดเร็วโดยที่คนมองไม่รู้สึกถึงการเปลี่ยนแปรงนอกจากนี้เรายังสามารถแสดงรูปออกมาโดยการสร้าง mark ซึ่งเป็นการเก็บรักษารูป Background ได้โดยไม่เปลี่ยนเป็นรูปเราทั้งหมดแต่จะเป็นการนำรูปออกมาเฉพาะส่วนที่อยู่ใน mark เท่านั้นการทำงานคำเรานำคำสั่ง BitBlt มาทำการประมวณผลรูปโดยใช้ logic ต่างๆ ที่ใช้กันคือ AND กับ OR ซึ่งในคำสั่ง BitBlt นี้เรียกว่า SRCAND กับ SRCPAINT ตามลำดับ

6.2 การสร้างเสียงและการเก็บเสียง

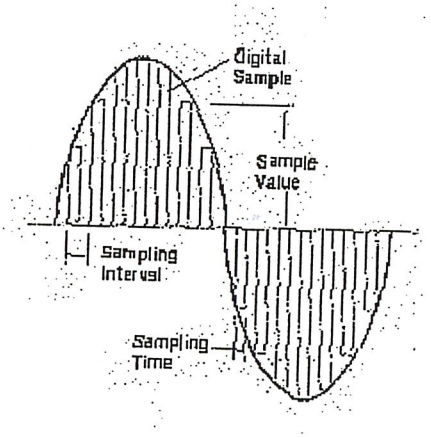
ในการสร้างเสียงนั้นเราใช้การ์ดเสียงเพื่อการอัดเสียง หลังจากนั้นก็นำเสียงที่อัดมาใช้ใน โปรแกรม ซึ่งการเก็บข้อมูลของการ์ดเสียงนั้นมีลักษณะดังนี้

การ์ดเสียง(Sound Card)เป็นอุปกรณ์ที่ทำการแปลงสัญญาณเสียงซึ่งเป็นสัญญาณ อานาลอกเป็นสัญญาณ ดิจิตอล ADC (Analog to digital converter) เพื่อที่จะสามารถเก็บสัญญาณเสียงนั้นบนคอมพิวเตอร์ และนำออกมาใช้โดยแปลงกลับจากสัญญาณ ดิจิตอลเป็นสัญญาณ อานาลอก ADC (Digital to analog converter) ADC นี้ทำการวัดค่าแอมปลิจูทของคลื่น sine ซึ่งเวลาที่ใช้ในการเก็บค่าของข้อมูลนี้เรียกว่า Sampling rate ซึ่งปกติค่าของ Sampling rate ของ Sound Card โดยทั่วไปจะมีค่าเท่ากับ 44.1 ,22.05 ,11.025 และ 5.0125 การที่จะเราเก็บข้อมูลเสียงโดยใช้ Sampling Rate สูงๆ นี้จะทำให้การเก็บเสียงมีความชัดเจนยิ่งขึ้น โดยถ้าเราเก็บเสียง Sampling Rate เพิ่มขึ้นอีก 1 เท่าจะทำให้เราจะต้องเปลืองที่เก็บเพิ่มขึ้นอีก 1 เท่าด้วย

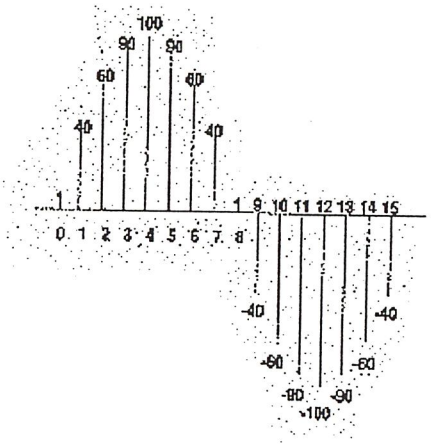
Sampled เสียงนี้เรียกว่า Pulse Code Modulation (PCM)

การวัดแอมปลิจูลคือการเก็บค่าความแตกต่างของระดับของเสียงหรือเรียกว่า resolution ถ้า 8 bit จะให้ความแตกต่างได้ 256 ค่าของ แอมปลิจูท และ 16 บิทจะให้ความแตกต่างได้ 65,536 ค่า การเพิ่มจำนวนบิทมากจะทำให้เสียงมีค่าความถูกต้องเพิ่มมากขึ้นเพราะสามารถบอกความแตกต่างได้มากกว่าซึ่งถ้าเราทำการเก็บค่าของเสียงเป็น 16 บิทจะทำให้เราเปลืองเนื้อที่ในการเก็บเพิ่มขึ้นอีกเท่าตัวเช่นกัน นอกจากนี้ซึ่งถ้าเราทำการเก็บเสียงเป็นแบบ stereo และ 44.1 kHz และ 16 bit จะต้องเปลืองเนื้อที่ Harddisk 10 M ต่อ 1 นาที

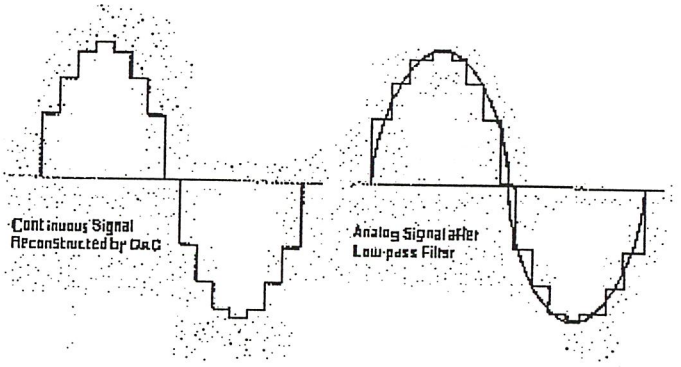
ซึ่งปกติ Audio Sound CD จะใช้ Sampling rate = 44.1 kHz และ 16 Bit และ DAT Digital Audio Type จะใช้ Sampling rate = 48 kHz และเก็บข้อมูล 16 bit เป็นต้น



การแซมปลิ่งสัญญาณอะนาลอกเพื่อหาค่าในรูปดิจิตอลของแต่ละจุดออกมา



ค่าที่ได้จากการแซมปลิ่งสัญญาณอนาลอก ซึ่งเป็นค่าดิจิตอล



การสร้างสัญญาณอะนาลอกขึ้นมาโดยใช้ข้อมูลจากดิจิตอล

บทที่ 7
บทสรุปและวิจารณ์

คอมพิวเตอร์ ได้เข้ามามีบทบาทอย่างมาก ในชีวิตประจำวันของเรา ไม่ว่าจะเป็นทางด้านธุรกิจ การแพทย์ การทำงานต่าง ๆ รวมทั้งทางด้านการศึกษา ทั้งนี้เพราะคอมพิวเตอร์ทำงานได้รวดเร็วและยังทำงานได้อย่างถูกต้องด้วย รวมทั้งในปัจจุบันนี้ราคาของเครื่องคอมพิวเตอร์ก็ถูกลงกว่าแต่เดิมมาก จึงทำให้ผู้คนหันมาใช้คอมพิวเตอร์กันมากขึ้น สำหรับในด้านการศึกษานั้น จะเห็นว่าคอมพิวเตอร์นั้น มีประโยชน์อย่าง มากในการสอน เพราะจะช่วยทั้งผู้เรียน ได้เรียนรู้ ได้โดยตรง สามารถติดต่อกับเครื่องได้ และยังเป็นประโยชน์ต่อผู้สอน ที่สามารถเตรียมเนื้อหา และบทเรียนได้อย่างเป็นระบบ รวมทั้งยังสอนได้ในปริมาณมากด้วย ในการสื่อสารกับเครื่องคอมพิวเตอร์นั้นอาจทำให้ผู้ใช้พอใจ ถ้ามีการสื่อสารกันกับผู้ใช้ ในรูปแบบ ที่ทำให้ผู้ใช้รู้สึกว่าเป็นกันเองมาก เช่น การใช้ภาพและเสียงในการอธิบาย หรือการติดต่ออย่างอื่น ๆ ที่สามารถเข้าใจได้ง่าย ซึ่งสิ่งเหล่านี้มีการใช้กันมากขึ้นเรื่อย ๆ ตามการพัฒนาของระบบคอมพิวเตอร์ และ วินโดวส์ ก็มีรูปแบบการติดต่อโดยใช้กราฟิก และสื่อต่าง ๆ เช่น ภาพหรือเสียง และมีฟังก์ชัน ในการติดต่อลักษณะนี้ให้ใช้มากมาย การใช้จอสัมผัส ช่วยในการติดต่อ ก็ทำให้ผู้ใช้ เกิดความเป็นกันเองมากขึ้น ซึ่งลักษณะ จะคล้ายกับการติดต่อแบบใช้เมาส์ แต่การออกแบบบท เรียนให้สามารถอธิบายเรื่องราวต่าง ๆ นั้น ต้องใช้เวลาและการเรียง ลำดับขั้นที่ดี เพื่อผู้เรียนจะได้ไม่เกิดการสับสน และต้องคำนึงถึง ความเป็นไปได้ในการใช้คอมพิวเตอร์ เพื่อสอนบทเรียนที่สร้างขึ้นนั้นด้วย ซึ่งในบทเรียนที่เราได้สร้างขึ้นมานี้ก็เพื่อให้เห็นภาพพจน์ ของการสอนโดยคอมพิวเตอร์ โดยใช้ภาพ เสียง ร่วมกับ touch screen ในการสื่อสารกับคอมพิวเตอร์ ซึ่งฟังก์ชันต่าง ๆ ในโปรแกรมยังสามารถนำไปประยุกต์ใช้ได้ต่อไป และทฤษฎีต่าง ๆ ก็เพื่ออธิบายขั้นตอนการทำงาน เพื่อเป็นแนวทางในการศึกษาสำหรับผู้ที่สนใจต่อ ไป

กิตติกรรมประกาศ

ขอขอบคุณท่านอาจารย์วัชระ ฉัตรวิริยะ ที่ได้ให้ความรู้และคำแนะนำ รวมทั้งจัดหาอุปกรณ์ที่ใช้ในการทำโปรเจคนี้

ขอขอบคุณพี่ปู ที่ช่วยดูแลระบบแลนให้ทำงานได้เป็นปกติ

หนังสืออ้างอิง

1 จิรพัฒน์ จันทร์เจิดศักดิ์ "การเขียนโปรแกรมบน Microsoft Windows", ซีเอ็ดยูเคชัน, 608 หน้า, 2536.

2 Ben Ezzel, "Windows 3.1 Graphics Programming", Zd Press, 480 p., 1992.

3 Gilmore, "Microprocessors Principles And Applications", McGraw-Hill Publishing Company, 508 p., 1989.

4 James L. Conger, "Windows API Bible", Waite Group Press, 1014 p., 1992.

5 Lee Adams, "C For Windows Animation Programming", Windcrest/McGraw-Hill, 634 p., 1993.

6 Microsoft Corporation, "Microsoft Windows Software Development Kit", Microsoft Corporation, 1990.

7 Nabajyoti Barkakati, "Imaging and Animation for Windows", Sams Publishing, 448 p., 1993.

8 Paul J. Perry, "Your Borland C++ Consultant", Sams Publishing, 449 p., 1993.

9 Que, "Windows 3.1 Multimedia", Book Disk, 912 p., 1992.

10 Steve Rimmer, "Multimedia Programming for Windows", Windcrest/McGraw-Hill, 370 p., 1994.

11 Steve Rimmer, "Windows Bitmapped Graphics", Windcrest/ McGraw-Hill, 381 p., 1993.

12 William H. Murray ,III Chris H. Pappas, "Windows 3.1 Programming", McGraw-Hill, 805 p., 1992