MIDI

(Musical Instrument Digital Interface)

โดย

1. นายธีระพงศ์ วัฒนพิทยกุล 33100152
2. นายอภิชาติ แสงธนวณิชย์ 33100489
3. นายอาคม บุษยกุล 33100520

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต
สาขาวิศวกรรมอิเล็คทรอนิคส์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2536

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

033347

สารบัญ

บทคัดย่อ

บทที่ 1 บทนำ 1

บทที่ 2 ทฤษฎีของระบบ MIDI 4

ข้อกำหนดของระบบมาตรฐาน MIDI 5

ความหมายของข้อความต่างๆในแต่ละ CHANNEL 9

บทที่ 3 การออกแบบและการสร้าง 16

บทที่ 4 การทดลองและผลการทดลอง 25

บทที่ 5 บทสรุปและวิจารณ์ผล 38

คู่มือการใช้งาน MIDI PROJECT 40

กิตติกรรมประกาศ

หนังสืออ้างอิง

APPENDIX

MIDI -MUSICAL INSTRUMENT DIGITAL INTERFACE- SHEET

IMPLEMENTATION SHEET K-11 SYNTHESIZER

PROJECT MIDI FILE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MIDI SYSTEM

โดย 1. นายธีระพงศ์ วัฒนพิทยกุล 33100152
2. นายอภิชาติ แสงธนวณิชย์ 33100489
3. นายอาคม บุษยกุล 33100520

อาจารย์ที่ปรึกษา รศ.ดร. มนัส สังวรศิลป์

บทคัดย่อ

ปริญญานิพนธ์นี้ เสนอการนำเครื่องคอมพิวเตอร์มาช่วยในงานทางศิลปการดนตรีซึ่งเป็นศาสตร์ที่มีความละเอียดอ่อนอย่างมาก โดยมีจุดประสงค์ให้เป็นประโยชน์กับผู้ที่ต้องการศึกษาทางด้านดนตรีหรือช่วยงานผู้ที่มีความชำนาญในงานด้านนี้อยู่แล้ว โครงงานนี้จะทำให้ไมโครคอมพิวเตอร์ทำงานควบคุม ซินธิไซเซอร์ให้ออกเสียงดนตรีตามที่โปรแกรมไว้ ลักษณะการโปรแกรมทำได้โดยบันทึกโน๊ตของเครื่องดนตรีแต่ละชิ้นเข้าไปแล้วจึงส่งข้อมูลทั้งหมดให้ซินธิไซเซอร์เล่นเป็นเพลงพร้อมกันตามโน๊ตของเครื่องดนตรีแต่ละชิ้นที่ได้โปรแกรมไว้ ในขณะที่มีการรับข้อมูลเข้าไปไมโครคอมพิวเตอร์สามารถส่งข้อมูลเก่าที่บันทึกไว้แล้วเล่นพร้อมกันได้ รวมทั้งการปรับปรุงแก้ไขข้อมูลดังกล่าวในขณะที่บันทึกด้วย นอกจากนั้นไมโคร-คอมพิวเตอร์ยังทำหน้าที่เป็นกลองอิเล็คทรอนิคส์ ซึ่งผู้ใช้สามารถโปรแกรมจังหวะกลองแบบต่างๆ ด้วย software ที่พัฒนาขึ้นมานี้มี utilities ช่วยในการแก้ไขข้อมูล ลบ ก๊อปปี้ และเลื่อนระดับเสียง เป็นต้น นอกจากนี้ยังมีส่วนของควอนไตซ์ (quantize) ซึ่งทำหน้าที่วิเคราะห์ข้อมูลที่เล่นโดยนักดนตรีให้ลงจังหวะได้ถูกต้อง ช่วยให้ประสิทธิภาพในการบันทึกสมบูรณ์ยิ่งขึ้น รายละเอียดต่างๆรวมทั้งส่วนที่เป็นฮาร์ดแวร์ และซอฟท์แวร์ ได้นำเสนออย่างละเอียดแล้วในปริญญานิพนธ์นี้

Abstract

In this thesis, we introduce the application of computer in art of music. The objectives are the utilities for ones interested in these branches and others professional. This project works in force of microprocessor and synthesizer that have been programmed before. This program will record music note of each track then send them simultaneously. As we record from synthesizer, the microcomputer can playback and we can edit the better note concomittantly. Moreover, microcomputer can use as DRUM MACHINE that user can determine the rythm in form of GUI (Graphic User Interface). Finally, the developed software provides many utilities such as EDIT, COPY, DELETE, QUANTIZE etc. which are suitable for interested ones. Details are recommended in this thesis.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1 บทนำ

เมื่อมนุษย์เริ่มสนใจเทคโนโลยี จะเป็นการแน่นอนว่าทุกอย่างจะดำเนินต่อไปและพัฒนาไปเรื่อย ๆ แต่นั่นเป็นเรื่องของอนาคต สำหรับตอนนี้เราจะมาดูความก้าวหน้าทางด้านดนตรีซึ่งเป็นการแน่ที่ว่าทุกคนย่อมมีเสียงเพลงอยู่ในหัวใจบ้างไม่มากก็น้อย ในสมัยก่อนไม่มีเครื่องดนตรีที่ใช้ไฟฟ้า แต่ในปัจจุบันเครื่องดนตรีแทบทุกชิ้นจะเป็นเครื่องที่ควบคุมด้วยไฟฟ้า เช่น กีตาร์ไฟฟ้า , คีย์บอร์ด เป็นต้น โดยในการเล่นเครื่องดนตรีแต่ละชิ้นก็ต้องฝึกฝนอย่างมากเพื่อให้ชำนาญในเครื่องดนตรีชิ้นนั้นๆ และเมื่อคอมพิวเตอร์เข้ามามีบทบาททางด้านดนตรี เราสามารถเพิ่มประสิทธิภาพของเครื่องดนตรีคือสามารถทำลูกเล่นได้มากขึ้นดังนั้นคนรุ่นใหม่จึงควรทำความรู้จักกับ 'MIDI' เพราะเป็นการใช้คอมพิวเตอร์ผสมผสานกับเครื่องดนตรี

ประวัติของมิดี้ - มีที่มาจากหลายๆประเทศเช่น สหรัฐอเมริกา,ยุโรป และญี่ปุ่นเป็นต้นในช่วงก่อนปี พ.ศ .2513 ได้มีเครื่องสังเคราะห์เสียงโดยใช้ระบบอิเลคทรอนิคส์(ซินธิไซเซอร์) ซึ่งมีระบบควบคุมแรงดันเป็น 1 volt/Octave (Octave คือระยะห่างของเสียงเป็น 8 ขั้นคู่เสียง) ได้มีการผลิตออกมาจนแทบเป็นสากล ซึ่งดูเหมือนกับว่าเป็นการเข้าถึงยุคใหม่ของความ Compatibility ของเครื่องดนตรีจากต่างโรงงานที่ผลิตกันเช่น มีซินธิไซ-เซอร์ A และ B ซึ่งผลิตจากต่างบริษัทกัน A ก็สามารถใช้แรงดันจาก B มาควบคุมได้

เมื่อซินธิไซเซอร์ชนิดเล่นได้หลายโน๊ต (Polyphonic ซินธิไซเซอร์)ถือกำเนิดขึ้นก็ยังใช้หลักการคล้ายๆ กับระบบควบคุมแรงดัน คือจะใช้คอมพิวเตอร์จัดการกับตัวสร้างระบบควบคุมแรงดันหลายๆ ตัว ซึ่งประกอบด้วยเครื่องสร้างความถี่ขยายเสียงและกรองความถี่หลายๆตัวเช่นกัน แต่เพื่อให้มีประสิทธิภาพสูงสุดในการจัดการระบบข้างต้นจะต้องทำด้วยซอฟแวร์ซึ่งจะดีกว่าการใช้อุปกรณ์อิเลคทรอนิคส์ในการควบคุมแรงดัน ดังนั้นระบบควบคุมแรงดันที่ใช้อุปกรณ์อิเลคทรอนิคส์จึงหมดไป ขณะนั้นก็ไม่มีวิธีที่เชื่อมต่อระหว่างซินธิไซเซอร์จากต่างผู้ผลิตกัน

บางบริษัทก็ทำให้ผู้ใช้มองเห็นข้อจำกัดในเชื่อมโยงเครื่องจากต่างบริษัทผลิต คือได้พัฒนาระบบบัสซึ่งสามารถขยายได้ มีระบบบัสแบบอนุกรมซึ่งมีข้อดีคือมีราคาไม่แพง และแบบขนานคือมีความเร็วสูง แต่ทั้งหมดที่กล่าวมาเป็นลิขสิทธิ์ของบริษัทที่พัฒนาเพียงผู้เดียว การเชื่อมโยงระหว่างเครื่องดนตรีจากที่ทำได้ยากก็เป็นไม่มีทางทำได้เลย ในเดือนธันวาคม ปี พ.ศ.2525 บริษัท SEQUENTIAL CIRCUITS, Inc (เป็นผู้ผลิต Polyphonic ซินธิไซ-เซอร์ตัวแรกที่ยอมรับกันอย่างกว้างขวาง) ได้ส่งซินธิไซเซอร์ตัวแรกของพวกเขาออกสู่ท้องตลาดคือ PROPHET 600 หนึ่งในความน่าสนใจของอุปกรณ์ชิ้นนี้คือมีพอร์ทเชื่อมต่ออนุกรมอยู่ DAVE SMITH ผู้ที่เป็นประธานบริษัทให้ชื่อว่า UNIVERSAL SYNTHESIZER INTERFACE (USI) และในงานอุตสาหกรรมเครื่องดนตรีในปีเดียวกัน ฝ่ายเทคนิคของ SEQUENTIAL YAMAHA , ROLAND และอีกหลายๆบริษัทในสหรัฐและญี่ปุ่น ได้มาประชุมเพื่อหามาตรฐานในการเชื่อมต่อระหว่างเครื่องดนตรี ได้มีความเห็นสอดคล้องกันคือ โครงสร้างของระบบสื่อสารจะมีความใกล้เคียงกับ USI ของ DAVE SMITH มาก และให้มีความเป็นกลางในเรื่องของความเร็ว ความง่ายในการใช้งานและราคาถูก

ในเดือนมิถุนายน พ.ศ.2526 PROPHET 600 สามารถเชื่อมโยงกับ YAMAHA DX-7 (เป็นซินธิไซเซอร์ซึ่งใช้หลักการสังเคราะห์เสียงแบบ FM (FREQUENCY MODULATION)) โดยการสร้างเสียงจากแหล่งกำเนิด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เนิดเสียงกระแสสลับหลายๆตัวมาผสมกัน แต่เรื่องนี้ไม่ได้รับความสนใจเท่าที่ควร เดือน สิงหาคม พ.ศ.2526 เรื่องนี้ได้ถูกนำมาพูดถึงอีกโดย SEQUENTIAL YAMAHA , ROLAND , KORG และ KAWAIได้ตกลงเรียกระบบนี้ว่า MIDI 1.0 ซึ่งเป็นประวัติศาสตร์ของระบบมิดี้

ฮาร์ดแวร์ของระบบมิดี้

คุณลักษณะที่สำคัญของมิดี้ก็คือ มีสายต่อและข้อต่อราคาไม่แพงนัก เป็นพอร์ทสื่อสารแบบอนุกรม และยังมีความเร็วคล้ายคลึงกับ RS-232 คือประกอบด้วยบิทที่เริ่มต้น 1 บิท ,บิทหยุด 1 บิท และบิทข้อมูลอีก8 บิท รวมเป็น 10 บิท โดยทีช่วงเวลาบิทละ 32 $^{+}$Sรวม320 $^{+}$S ซึ่งจะเป็นการสื่อสารแบบอะซิงโครนัส ด้วยความเร็ว 31.25 KBaud

ส่วนที่ไม่เหมือน RS-232 (ซึ่งใช้แรงดันแบบ 2 ขั้ว) มิดี้มีกระแส 1.5 mA ไหลวนซึ่งผ่าน Optoisolator ที่จุดสุดท้ายของการรับสัญญาณเข้ามา และสาเหตุที่ต้องใช้Optoisilator ในระบบที่ออกแบบมาให้ราคาถูกที่สุด เพราะสามารถป้องกันกระแสลงกราวด์ ซึ่งเป็นปัญหาโดยทั่วไปของระบบเครื่องเสียง

สำหรับขั้วต่อที่มิดี้ใช้แบบ DIN 5เข็ม มีตัวเมียอยู่บนอุปกรณ์มิดี้ ตัวผู้ติดต่ออยู่กับสายต่อ ขั้วต่อแบบ DIN นี้ราคาไม่แพงและเมื่อเสียบต่อกันแล้วสามารถล็อคตัวเองได้ จึงสะดวกในการนำไปใช้งานในที่ต่างๆ ทำให้ขั้วต่อแบบนี้มีความสมบูรณ์ในการใช้งานโดยทั่วไปและในห้องอัดเสียงขั้วต่อนี้สามารถแทนได้ด้วย ขั้วต่อแบบ XLR ซึ่งเป็นขั้วต่อระดับมืออาชีพ เมื่ออุปกรณ์นั้นใช้งานบนเวทีกลางแจ้ง

สายที่ใช้เชื่อมโยงในมิดี้เป็นสายชิลด์(shield) ตีเกลียว มีความยาวสายไม่เกิน 50 ฟุต ที่ปลายสายทั้ง 2 ข้างเป็นขั้วต่อ DIN 5 เข็มตัวผู้ การต่อป้องกันกระแสลงกราวด์ สังเกตที่ว่าไม่มีเข็มใดของขั้วต่อ MIDHN ที่ลงกราวด์เลย และในขณะที่ขั้วที่ 2 ของ MIDI-OUT จะต่อลงกราวด์ ซึ่งเป็นลักษณะการต่อที่พิเศษของระบบมิดี้ และเมื่อมีการเชื่อมต่อระหว่างอุปกรณ์ก็จะไม่มีระบบกราวด์ร่วมกันเลย

การใช้งานที่น่าสนใจของระบบมิดี้มากที่สุดคือ การเขียนเพลงประกอบภาพยนตร์โดยปกติ นักแต่งเพลงเมื่อแต่งเพลงเสร็จก็จำเป็นต้องจ้างนักดนตรีเล่นจริงๆและเมื่อเล่นดนตรีบันทึกเสียงเรียบร้อยแล้วผู้แต่งเพลงต้องนำเพลงมาเพื่อหาข้อผิดพลาด ถ้าเกิดมีความผิดพลาดเกิดขึ้นก็ต้องทำการอัดใหม่ เมื่อแก้ไขเรียบร้อยแล้วก็นำมาใส่ในตัวภาพยนตร์และต้องจัดให้เพลงอยู่ส่วนที่เหมาะสมของภาพยนตร์เรื่องนั้น คือให้ตรงตามอารมณ์ของภาพยนตร์ ทั้งหมดเป็นเรื่องที่ยุ่งยากและต้องใช้ค่าใช้จ่ายสูงเพราะต้องอัดในสตูดิโอที่มีราคาแพง แต่ปัจจุบันซินธิไซเซอร์สามารถสร้างเสียงดนตรีให้เหมือนจริงได้เกือบทุกชนิด ดังนั้นการทำเพลงภาพยนตร์สามารถใช้ระบบมิดี้และอุปกรณ์มิดี้ ทำแทนได้ทั้งหมด อีกทั้งการแก้ไขบทเพลงก็ทำได้ง่ายเพราะเป็นข้อมูลแบบดิจิตอล ค่าใช้จ่ายในการทำงานก็ถูกลงเป็นอย่างมากเพราะไม่ต้องเสียค่าสตูดิโอแพงๆ และไม่ต้องเสียค่าจ้างนักดนตรีหลายๆคน

ในการแสดงดนตรีในปัจจุบันก็มีการนำระบบมิดี้มาใช้มากขึ้นในรูปแบบของซีเควนเซอร์เพราะขณะที่เล่นดนตรีสดๆการที่จะเล่นให้สมบูรณ์ 100% ให้เสียงทุกเสียงออกมาครบตามต้นฉบับนั้นเป็นไปได้ยากและต้องความสามารถของนักดนตรีสูง แต่ถ้าเราอัดการเล่นดนตรีบางส่วนที่มีความซับซ้อนลงในซีเควนเซอร์ก็จะทำให้การแสดงดนตรีมีความคล่องตัวขึ้นคือไม่ต้องพะวงกับส่วนที่ยุ่งยากของเพลงเพราะเราใช้ซีเควนเซอร์เล่นให้แทน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

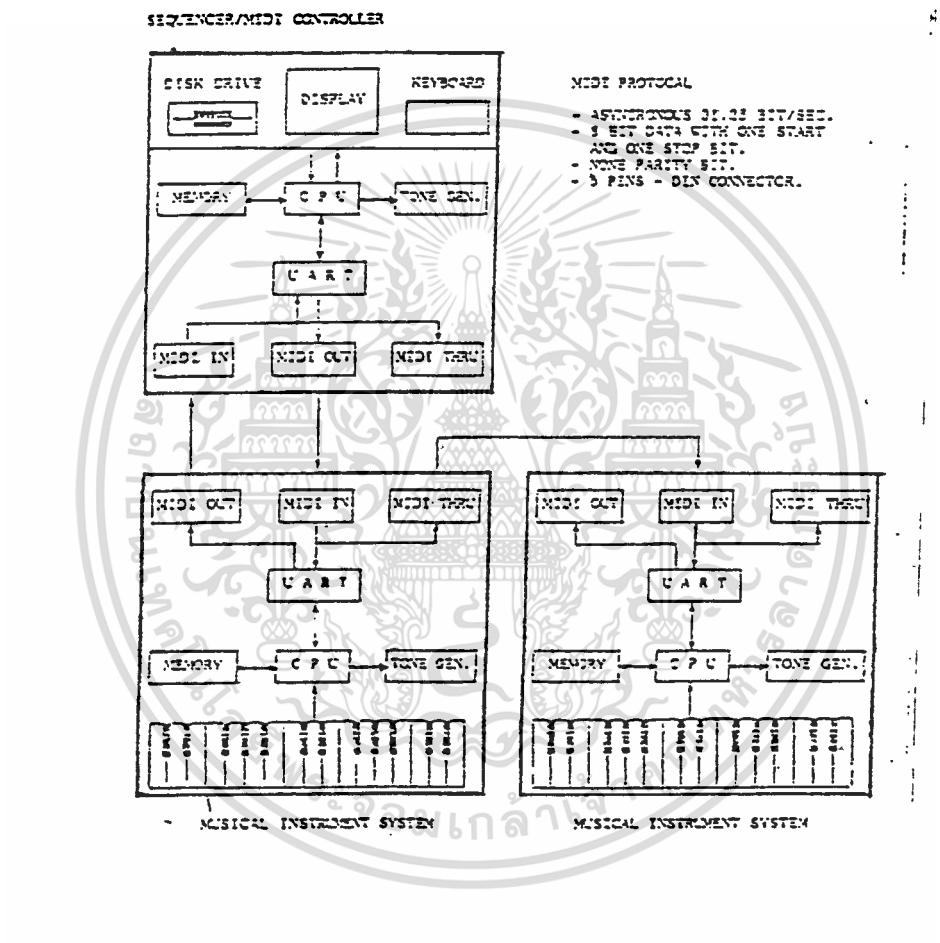
การอัดข้อมูลการเล่นดนตรีส่วนที่ซับซ้อนก็สามารถทำได้โดยการเล่นในจังหวะช้าๆก่อน แล้วค่อยปรับความเร็วให้เท่ากับที่ใช้จริงดังนั้นเวลาเล่นจริงๆพร้อมซีเควนเซอร์จะได้เสียงที่สมบูรณ์แบบ

ระบบมิดี้สามารถสร้างความสำราญให้กับทุกๆคนได้โดยที่เสียค่าใช้จ่ายน้อย สามารถที่จะสร้างสตูดิโอขั้นเยี่ยมในบ้านได้อย่างง่ายดาย เราเรียกว่า HOME STUDIO ซึ่งจำเป็นต้องมีอุปกรณ์เพียงเล็กน้อย ได้แก่ คอมพิวเตอร์ , CARD-อินเตอร์เฟส และ แหล่งกำเนิดเสียงอาทิเช่น ซินธิไซเซอร์,ซาวด์โมดูล หรือ ดรัมแมชชีน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2 ทฤษฎีของระบบ MIDI

MIDI ย่อมาจาก "MUSICAL INSTRUMENT DIGITAL INTERFACE" เป็นระบบมาตรฐานสำหรับการสื่อสารข้อมูลระหว่างเครื่องดนตรี โดยจะมีการสื่อสารในรูปแบบของAsynchronous Serial I/O ซึ่งสามารถเขียนแผนภาพแสดงการทำงานได้ดังนี้



รูปที่ 1 แสดงการสื่อสารของระบบ MIDI

จากรูปอุปกรณ์ควบคุม อันได้แก่ SEQUENSER ซึ่งอาจจะเป็น คอมพิวเตอร์ จะทำหน้าที่เป็นตัวควบคุมการทำงานของเครื่องดนตรีต่างๆ โดยเป็นตัวจัดลำดับในการส่งรหัสข้อมูลมีดี้ ซึ่งจะมีขั้นตอนการทำงานดังนี้

1.SEQUENCER จัดส่งรหัส MIDI ไปยังเครื่องดนตรีชิ้นที่ 1 โดยใช้ Opto-isolator ทำหน้าที่คัปปลิ้งสัญญาณซึ่งจะส่งไปที่ UART และสัญญาณอีกส่วนหนึ่งจะส่งไปที่บัฟเฟอร์เพื่อส่งผ่านข้อมูลไปยังเครื่องดนตรีตัวอื่นเรียกว่า "MIDI THRU"

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.UART จะทำหน้าที่เปลี่ยนข้อมูลแบบขนานเพื่อนำไปประมวลผลและเมื่อมีการส่งข้อมูลจากเครื่องดนตรีไปยัง SEQUNCER เพื่อบันทึกข้อมูล UART จะทำหน้าที่เปลี่ยนข้อมูลแบบขนานเป็นแบบอนุกรมก่อนส่งสัญญาณออกไป

ถ้าไม่อาศัยข้อมูลจากคอมพิวเตอร์เมื่อนักดนตรีเล่นเครื่องดนตรีชิ้นที่ 1 ส่วนควบคุมของเครื่องดนตรีชิ้นนี้จะส่งรหัส MIDI ออกไป และคอมพิวเตอร์จะรับข้อมูลนี้มาทำกระบวนการต่างๆอาทิเช่น การเก็บข้อมูลลงในดิสค์ ซึ่งในลักษณะนี้จะคล้ายกับการบันทึกเสียงแต่แทนที่จะเก็บในลักษณะของสัญญาณเสียงคอมพิวเตอร์จะเก็บในรูปแบบของรหัส MIDI และเมื่อต้องการเล่นซ้ำ (PLAY BACK) ทางคอมพิวเตอร์ก็จะส่งรหัส MIDI กับไปยังเครื่องดนตรีและเครื่องดนตรีจะทำงานตามลักษณะของข้อมูลเดิม หรือจะเรียกว่าเป็นสร้างเสียงขึ้นมาใหม่ (REGENERATE) นั่นเอง

จะเห็นได้ชัดว่าเมื่อเครื่องดนตรีสามารถสื่อสารภายใต้มาตรฐานเดียวกัน แล้วเราสามารถนำไปประยุกต์ใช้งานได้อย่างกว้างขวาง เช่น

-สามารถควบคุมเครื่องดนตรีได้ทีละหลายๆเครื่อง อันเนื่องจากเครื่องดนตรีทุกชิ้นสื่อสารด้วยภาษาเดียวกัน

-สามารถส่งรหัส MIDI ไปควบคุมกลองอิเลคทรอนิค (DRUM MACHINE) เพื่อให้มีจังหวะตรงกับเครื่องดนตรีชิ้นๆ (Synchronize)ได้

-อุปกรณ์แต่งเสียง (SOUND EFFECT) ต่างๆสามารถควบคุมเสียงด้วยรหัส MIDI

-ใช้ในการสร้างเสียง (VOICE QUALITY) ของเครื่องดนตรีประเภทซินธิไซเซอร์ (VOICE-EDITING SOFTWARE)

-ข้อมูลจากเครื่องดนตรีเครื่องหนึ่งอาจจะนำไปใช้กับอีกเครื่องหนึ่งได้ (DATATRANSFER)

-ถ้าใช้การเก็บรหัส MIDI แทนที่จะเก็บสัญญาณเสียง เราสามารถเล่นซ้ำได้ดังกล่าวไปแล้วซึ่งคล้ายกับการบันทึกเสียง (TAPELESS RECORDING)

-ประโยชน์จากการสื่อสาร เช่นสามารถส่งข้อความ MIDI ผ่านโมเด้ม (MODEM) ได้ ดังนั้นไม่ว่าจะอยู่ที่ไหนก็ตาม นักดนตรีก็ยังคงสามารถส่งดนตรีของเขาไปยังห้องอัดเสียงได้ทางโทรศัพท์โดยไม่มีความผิดเพี้ยนเลยเนื่องจากการส่งเป็นโค๊ดรหัส

สำหรับเครื่องดนตรีที่ไม่ใช้ไฟฟ้า (ACOUSTIC) เมื่อเราสามารถทำให้เครื่องดนตรีกำเนิดรหัสข้อมูล MIDIเราก็สามารถใช้พ่วงกับเครื่องดนตรีชิ้นอื่นด้วยระบบของ MIDI ได้ด้วย

ข้อกำหนดของระบบมาตรฐาน MIDI

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

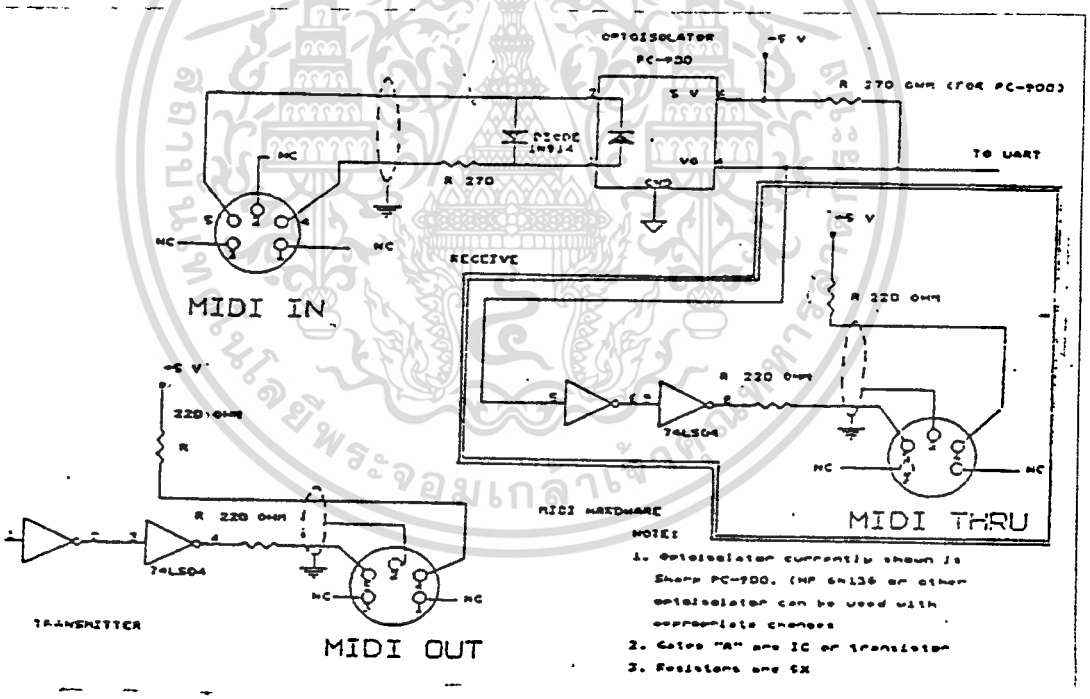
-ข้อกำหนดโดยทั่วไป

MIDI เป็นระบบมาตรฐานที่จะให้ ซินธิไซเซอร์ (SYNTHESIZER) , ซีเควนเซอร์ (SEQUENCER) , คอมพิว-

เตอร์ , อุปกรณ์สร้างจังหวะ ฯลฯ สามารถติดต่อกันได้ อุปกรณ์ที่ใช้ระบบมิดี้ มักจะรับส่งข้อความในระบบมิดี้ได้โดย ทางด้านรับจะรับข้อความ (message) ที่เป็นรหัสมิดี้แล้วทำงานตามความหมายของข้อความนั้น อุปกรณ์ทางฮาร์ดแวร์ที่สำคัญประกอบด้วย OPTO-ISOLATOR , UNIVERSAL ASYNCHRONOUS RECIVER-TRANSMITTER (UART) ส่วนทางด้านส่งก็จะส่งรหัสมิดี้ออกไป โดยมี UART และตัวขับสัญญาณ (line driver) เป็นองค์ประกอบสำคัญ

-ฮาร์ดแวร์

การสื่อสารจะเป็นแบบอซิงโครนัส โดยใช้ความเร็วในการรับ-ส่งที่ 31.25 kBaud มีบิทเริ่มและบิทหยุด (start-stop bit) อย่างละ 1 บิท บิทข้อมูล 8 บิท รวมทั้งหมด 10 บิท ต่อข้อมูล 1 ไบท์ซึ่งใช้เวลาส่ง 320 ไมโครวินาที มีวงจรเป็นดังรูปที่ 2 ซึ่งเป็นรูปของ MIDI HARDWARE



รูปที่ 2 MIDI HARDWARE

จากรูปกระแสในลูป = 1.5mA โดยที่โลจิก 0 จะมีกระแสไหล ทางด้านรับควรมี OPTO-ISOLATOR เป็นตัวคัปปลิ้งสัญญาณ ซึ่งเป็นชนิดการทำงานที่ความเร็วสูงเช่น Sharp PC-900 เวลาในการสวิทช์ควรน้อยกว่า 2 ไมโครวินาที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

-อุปกรณ์สำหรับเชื่อมต่อ (CONNECTOR)

เป็นแบบ DIN 5 ขา (แจ็คกลม) ตัวผู้ ขาที่ใช้คือขา 5,4 ส่วนขา 2 ต่อลงกราวด์นอกจากนั้นไม่ใช้ซึ่งจะถูกปล่อยลอยไว้

ความยาวของสาย (MIDI CABLE)ไม่เกิน 15 เมตร หรือ 50 ฟุต และควรมีการพันเกลียว (twist) เพื่อลดสัญญาณรบกวน

ชื่อของคอนเนคเตอร์ : MIDI IN(ข้อมูลเข้า) , MIDI OUT(ข้อมูลออก) , MIDI'THRU (การส่งผ่านข้อมูลที่ได้มาจาก MIDI IN ให้กับเครื่องดนตรีชิ้นอื่นๆ)

-รูปแบบของข้อมูล (DATA FORMAT)

แต่ละข้อความในระบบ MIDI มักมีหลายไบท์โดยประกอบด้วยไบท์สถานะ (STATUS BYTE) ตามด้วยไบท์ข้อมูล (DATA BYTE) อีก 1-2 ไบท์ ยกเว้นข้อความเกี่ยวกับเวลา (REAL-TIME MESSAGE) และข้อความพิเศษ (EXCLUSIVE MESSAGE) ซึ่งจะกล่าวถึงภายหลัง

ประเภทของข้อความจะแบ่งออกเป็น 2 ประเภทใหญ่ๆคือ แชนแนล กับ ระบบ

1.**ข้อความเกี่ยวกับแชนแนล** - ในไบท์สถานะจะใช้ 4 บิท เป็นตัวกำหนดแชนแนลของไบท์สถานะ (สามารถกำหนดได้ 16 แชนแนล) ทางด้านรับจะตอบสนองเฉพาะไบท์สถานะที่มีแชนแนลตรงกับแชนแนลของตนเท่านั้น ซึ่งข้อความเกียวกับแชนแนลมี 2 ประเภทคือ

VOICE - ใช้ควบคุมเสียง ข้อความเกี่ยวกับเรื่องเสียงจะถูกส่งผ่านทางแชนแนลเสียง(VOICE CHANNEL)

MODE - ใช้กำหนดลักษณะของการตอบสนองต่อข้อความที่เกี่ยวกับเรื่องของเครื่องดนตรีโดยจะส่งมาทางแชนแนลพื้นฐาน (BASIC CHANNEL) ของเครื่องดนตรี

2.**ข้อความเกี่ยวกับระบบ** - ข้อความเกี่ยวกับระบบจะไม่มีการกำหนดแชนแนล ดังนั้นไม่ว่าจะด้านรับจะเป็นแชนแนลใดก็จะตอบสนองต่อข้อความเกี่ยวกับระบบเสมอ ซึ่งแยกได้เป็น 3 ประเภท

-ข้อความทั่วไป (COMMON MESSAGE) เป็นข้อความสำหรับทุกอุปกรณ์ในระบบ

-ข้อความเกี่ยวกับเวลา จะมีแต่ไบท์สถานะเท่านั้น ไม่มีไบท์ข้อมูล สำหรับข้อความประเภทนี้จะส่งเมื่อใดก็ได้ แม้แต่จะส่งคั่นระหว่างไบท์ของข้อความอื่นๆ

-ข้อความพิเศษ (EXCLUSIVE MESSAGE) มีหลายไบท์ข้อมูล และจะจบด้วยไบท์ที่เรียกว่าEND OF EXCLUSIVE (EOX) หรือไบท์สถานะอื่นๆเป็นข้อความที่ใช้เฉพาะกับอุปกรณ์ของผู้ผลิตที่กำหนดไว้ในรหัสประจำเครื่องของแต่ละบริษัทผู้ผลิตหรือ IDENTIFICATION CODES-("ID CODES") เท่านั้นถ้าอุปกรณ์ทางด้านรับมีรหัสประจำเครื่องไม่ตรงกับข้อความพิเศษที่ได้ก็จะไม่สนใจไบท์ข้อมูลที่ตามมาซึ่งผู้ผลิตแต่ละรายสา-มารถกำหนดรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แบบและความหมายข้อมูลของตัวเองได้ โดยส่งเป็นข้อความพิเศษที่มีรหัสประจำเครื่องของตน (ซึ่งเป็นรหัสเดียวกันกับทางด้านรับ)

-ประเภทของข้อมูล (DATA TYPES)

จะแบ่งเป็น 2 ประเภทคือ ไบท์สถานะและไบท์ข้อมูล

1.ไบท์สถานะ - มี 8 บิท โดยค่าของบิทที่มีค่านัยสำคัญสูงสุดจะถูกเซ็ทเป็น 1 ซึ่งไบท์สถานะนี้จะเป็นตัวบ่งบอกประเภทของข้อความ และบอกให้รู้ถึงความหมายของไบท์ข้อมูลที่ตามมา

สถานะที่กำลังทำงาน (RUNNING STATUS) สำหรับข้อความเกี่ยวกับเสียงและโหมดเท่านั้นเมื่อด้านรับได้รับไบท์สถานะและทำขบวนการต่างๆ ด้านรับจะคงยังอยู่ในสถานะนั้นจนกว่าจะได้รับไบท์สถานะใหม่ที่ไม่เหมือนเดิม ดังนั้นถ้ามีการส่งไบท์สถานะเดิมซ้ำมาอีกครั้ง อาจทำเมื่อต้องการแก้ไขไบท์ข้อมูลที่ผิด โดยการส่งข้อมูลใหม่ภายใต้สถานะที่กำลังทำงานข้อความที่สมบูรณ์ต้องประกอบด้วยไบท์ข้อมูลที่ถูกต้องตามข้อกำหนด

สถานะที่กำลังทำงานจะหยุดเมื่อมีไบท์สถานะใหม่เข้ามา เว้นแต่ถ้ามีข้อความเกี่ยวกับเวลาเข้ามา ก็จะเป็นเพียงการคั่นจังหวะสถานะที่กำลังทำงานแต่เพียงชั่วคราวเท่านั้น

สถานะที่ไม่สามารถทำงานได้ (UNIMPLEMENT STATUS) ทางด้านรับจะไม่สนใจไบท์สถานะที่ไม่สามารถทำงานตามได้ และจะไม่สนใจไบท์ข้อมูลที่ตามมา

สถานะที่ไม่มีการกำหนดความหมาย (UNDEFINED STATUS) จะไม่มีการใช้ไบท์สถานะที่ไม่มีความหมาย ซึ่งควรระวังในช่วงขณะที่เปิดเครื่องและปิดเครื่องอาจมีการส่งข้อมูลที่ผิดพลาดได้

2.ไบท์ข้อมูล - จะถูกส่งตามหลังไบท์สถานะ โดยมีไบท์ข้อมูลประมาณ 1-2 ไบท์(ยกเว้นข้อมูลเกี่ยวกับเวลา) ไบท์ข้อมูลมี 8 บิท โดยบิทที่มีค่านัยสำคัญสูงสุดจะถูกรีเซ็ทเป็น 0 ซึ่งจะมีจำนวนของไบท์ข้อมูลที่มีได้เป็นไปตามไบท์สถานะที่ถูกกำหนดตายตัว

-โหมดต่างๆ ในแต่ละแชนแนล (CHANNEL MODE)

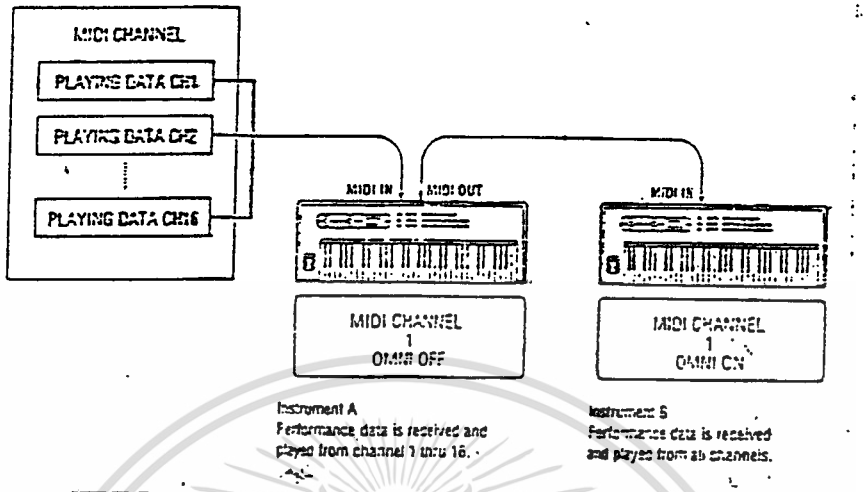
ซินธิไซเซอร์จะประกอบด้วยส่วนที่สร้างเสียง (SOUND GENERATOR) เสียงที่เกิดขึ้นเรียกว่า VOICE การกำหนดเสียงเป็นกระบวนการเกี่ยวกับข้อมูลพวกการเล่นโน๊ต (NOTE ON) กับการเลิกเล่นโน๊ต (NOTE OFF) จากคีย์บอร์ดซึ่งเสียงที่มีโน๊ตต่างๆจะถูกเล่นได้ในจังหวะที่ถูกต้อง

ข้อความเกี่ยวกับโหมด (MODE MESSAGE) ใช้กำหนดลักษณะการสร้างเสียงของเครื่องดนตรีประเภทซินธิไซเซอร์ ได้แก่ OMNI(ON/OFF) , POLY และ MONO

MONO ON - จะเป็นการกำหนดการกำเนิดเสียงในแชนแนลเสียงให้มีได้ 1 เสียงเท่านั้นในเวลาเดียวกัน (MONO PHONIC)

MONO OFF (POLY ON) - ซินธิไซเซอร์ที่อยู่ในโหมดนี้สามารถสร้างเสียงได้พร้อมๆกันทีละหลายๆเสียง (POLY PHONIC) ซึ่งทั้งนี้จะขึ้นอยู่กับความสามารถในการ
สร้างเสียงของตัวซินธิไซเซอร์ด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3 แผนภาพแสดงการทำงานในโหมดต่างๆ

สำหรับ OMNI ถ้า OMNI ON ตัวรับจะสามารถรับข้อความเกี่ยวกับเสียงได้จากทุกแชนแนลเสียงโดยไม่มีความแตกต่างของแชนแนล ถ้า OMNI OFF ด้านรับจะรับเฉพาะข้อความเกี่ยวกับเสียงที่มีแชนแนลตรงกับของตนเท่านั้น

ดังนั้นทางด้านรับที่ถูกกำหนดเป็นแชนแนลพื้นฐานจะมีลักษณะการทำงานได้ 4 โหมดดังนี้

โหมด 1 - OMNI ON , POLY

โหมด 2 - OMNI ON , MONO

โหมด 3 - OMNI OFF , POLY

โหมด 4 - OMNI OFF , MONO

ทางด้านส่งและรับจะทำงานได้ทีละ 1 โหมดเท่านั้นในเวลาเดียวกันและปกติจะส่งและรับในโหมดเดียวกัน

ด้านรับจะรับรู้ข้อความเกี่ยวกับโหมดทางแชนแนลพื้นฐานเท่านั้น ส่วนข้อความเกี่ยวกับเสียงอาจได้รับทางแชนแนลพื้นฐานหรือแชนแนลอื่นๆก็ได้ซึ่งเรียกรวมกันว่าแชนแนลเสียงซึ่งโดยทั่วไปมักกำหนดให้แชนแนลแรก (0) เป็นแชนแนลพื้นฐาน

-ความหมายของข้อความต่างๆในระบบ MIDI

1.ข้อความที่เกี่ยวข้องกับเสียงในแต่ละแชนแนล (CHANNEL VOICE MESSAGE)

การเล่นโน๊ต (NOTE ON) - เป็นข้อความที่บอกให้รู้ถึงการเล่นโน๊ต ประกอบด้วย 3ไบท์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

033347

1.ไบท์สถานะคือ 1001nnnn - โดยที่ nnnn คือ หมายเลขแชนแนล 0-15

2.ไบท์ข้อมูลแรกจะบอกให้รู้ว่าโน๊ตไหนที่ถูกกด (NOTE NUMBER) มีค่าได้ตั้งแต่0-127 โดย 0 จะเป็นที่ต่ำที่สุด , 127 เป็นโน๊ตที่สูงที่สุด แต่โดยทั่วไปเครื่องดนตรีจะได้รับรู้โน๊ตได้น้อยกว่า เช่น ซินธีไซเซอร์ที่มี 5 ออคเตฟ จะรับรู้หมายเลขโน๊ตได้ตั้งแต่ 36-96 และหมายเลขโน๊ตของ โดกลาง (MIDDLE C) คือ 60

3.ไบท์ข้อมูลต่อมาจะบ่งบอกถึงน้ำหนักในการเล่นหรือความเร็วที่กดคีย์ (KEY VELOCITY) เหมือนกับการเล่นเปียโน ถ้ากดคีย์เปียโนด้วยความเร็วจะได้เสียงที่ดัง และถ้ากดช้าจะได้เสียงที่เบาความเร็วของคีย์มีค่าได้ตั้งแต่ 0-127 โดยความเร็ว 127 จะดังที่สุด และความเร็ว 0 จะไม่มีเสียงออกมา (NOTE OFF) และสำหรับเครื่องดนตรีที่ไม่มีผลตอบสนองต่อความเร็วของคีย์ก็สามารถกำหนดค่าความเร็วเฉลี่ยเท่ากับ 64 ได้

การเลิกเล่นโน๊ต (NOTE OFF) - เป็นข้อมูลที่บอกให้รู้ถึงการเลิกสร้างเสียง (เช่นเดียวกับการปล่อยนิ้วออกจากคีย์) ประกอบด้วย 3 ไบท์

1.ไบท์สถานะคือ 1000nnnn - โดยที่ nnnn คือหมายเลขแชนแนล

2.ไบท์ข้อมูลแรกบอกให้รู้ว่าให้เลิกสร้างเสียงโน๊ตตัวไหน (KEY NUMBER)

3.ความเร็วในการปล่อยคีย์ (KEY OFF VELOCITY) - เป็นไบท์ข้อมูลที่บอกถึงความเร็วในการปล่อยคีย์ และในกรณีที่อุปกรณ์ไม่มีคค่าตอบสนองความเร็วจะถูกเซ็ทให้มีค่าเป็น 64

เมื่อเครื่องดนตรีได้รับข้อความเกี่ยวกับการเล่นโน๊ตแล้ว เครื่องดนตรีชิ้นนั้นจะผลิตเสียงค้างอยู่จนกว่าจะได้รับข้อความเกี่ยวกับการเลิกเล่นโน๊ตของโน๊ตตัวนั้น แล้วจึงเลิกจะสร้างเสียง ดังนั้นเมื่อมีการส่งข้อความเกี่ยวกับการเล่นโน๊ตออกไปแล้ว จะต้องข้อความเกี่ยวกับการเลิกเล่นโน๊ตตามด้วยทุกครั้งมิฉะนั้นจะเกิดเสียงค้าง

การกำหนดความดันของคีย์แบบโพลีโฟนิค (POLY-PHONIC KEY PRESSURE) -คีย์บอร์ดในระบบ MIDI บางเครื่อง ไม่เพียงแต่ตอบสนองความเร็วของคีย์ได้เท่านั้นยังตอบสนองต่อน้ำหนัก (ความดัน) ของนิ้วที่กดลงไปอีกหลังจากที่กดคีย์ลงไปแล้ว ประกอบด้วย 3 ไบท์

1.ไบท์สถานะคือ 1010nnnn - nnnn คือหมายเลขแชนแนล

2.หมายเลขของโน๊ต

3.ความดันของคีย์ (KEY PRESSURE VALUE) มีค่าตั้งแต่ 0-127 โดย 0หมายถึงไม่มีความดันเลย และ 127 หมายถึง ความดันสูงสุด

ประโยชน์ของโพลีโฟนิคคีย์เพรชเชอร์ คือจะให้ความรู้สึกของเครื่องดนตรีได้มากขึ้นซึ่งระบบของโพลีโฟนิคคีย์เพรชเชอร์นี้จะมีระบบที่ซับซ้อนดังนั้นจึงพบในคีย์บอร์ดเพียงบางรุ่น

การกำหนดความดันรวม (OVERALL PRESSURE) - ต่างกับโพลีโฟนิคคีย์เพรชเชอร์ตรงที่ไม่สามารถบอกความดันแยกกันของแต่ละคีย์ที่ถูกกดได้ แต่จะบอกเป็นความดันเฉลี่ยของทุกๆคีย์ในเวลานั้นๆประกอบด้วยข้อมูล 2 ไบท์

1.ไบท์ข้อมูลคือ 1101nnnn - nnnn คือหมายเลขแชนแนล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.ความดันของแชนแนล (CHANNEL PRESSURE VALE) มีค่าตั้งแต่ 0-127โดยที่ 0 ไม่มีความดัน และ 127 มีความดันมากที่สุด

ข้อมูลเกี่ยวกับการควบคุม (CONTROL CHANGE) - ใช้ในการปรับแต่งเสียงสำหรับข้อมูลเกี่ยวกับการควบคุม ในปัจจุบันยังไม่มีมาตรฐานที่แน่นอนระหว่างบริษัทผู้ผลิต ประกอบด้วยข้อมูล 3 ไบท์

1.ไบท์สถานะคือ 1011nnnn

2.หมายเลขควบคุม (CONTROL NUMBER) - มีค่าตั้งแต่ 0-127 ดูความหมายได้จาก MIDI IMPLEMENTATION SHEET ของแต่ละบริษัทผู้ผลิต

3.ค่าที่จะให้แก่ตัวควบคุม (CONTROL VALUE)

การเอื้อนเสียง (PITCH BEND) - เป็นลักษณะของการเอื้อนเสียงจากสูงไปต่ำหรือจากต่ำไปสูงโดยที่ไม่ต้องเปลี่ยนตำแหน่งของคีย์ที่กด ประกอบด้วยข้อมูล 2 ไบท์

1.ไบท์สถานะคือ 1110nnnn

2.ค่าที่จะให้เอื้อน (PITCH BEND VALUE),โดยที่ค่าปกติมีค่า 64

ข้อมูลเกี่ยวกัยการโปรแกรมเสียง (PROGRAM CHANGE) หรือบางทีเรียกว่าPROGRAM SELECT จะบอกให้รู้ถึงการใช้โปรแกรมเสียงต่างๆในเครื่องดนตรี ประกอบด้วยข้อมูล 2 ไบท์

1.ไบท์สถานะคือ 1100nnnn

2.โปรแกรมเสียงที่ใช้ (SELECTED PROGRAM NUMBER) มีได้ตั้งแต่ 0 ถึง 127

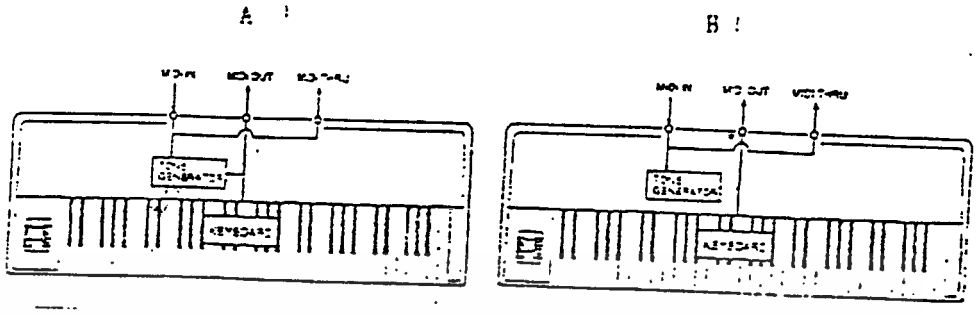
2.ข้อมูลเกี่ยวกับลักษณะการทำงานในแต่ละแชนแนล (CHANNEL MODE MESSAGES)

เป็นข้อความเกี่ยวกับโหมดในการทำงาน เป็นส่วนหนึ่งของข้อความเกี่ยวกับการควบคุมคีย์บอร์ดแบบโลคอล/รีโหมด (LOCAL/REMOTE KEYBOARD CONTROL) ใช้ในการพ่วงเครื่องดนตรีเข้าด้วยกันโดยควบคุมจาก SEQUENSER เครื่องเดียว

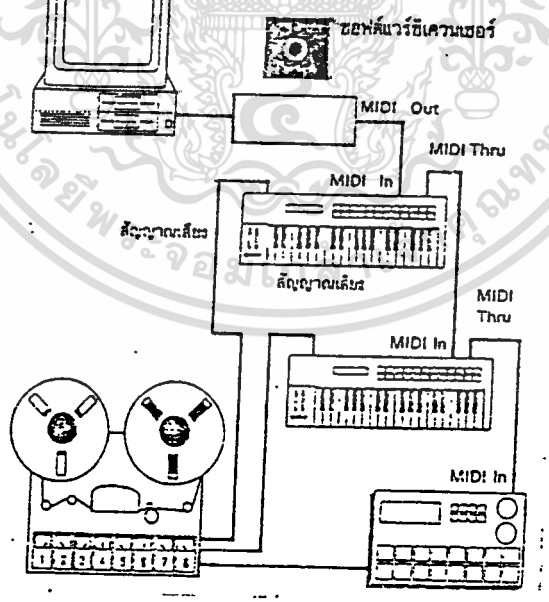
จากรูป (a) ถ้าเป็นการควบคุมแบบโลคอล (LOCAL CONTROL ON) ข้อมูลทางด้านเสียง (เช่นการเล่นโน๊ต) จากคีย์บอร์ดจะเข้าไปยังส่วนสร้างเสียงและข้อมูลเดียวกันนี้จะออกไปทาง MIDI OUT ด้วย(ข้อมูลจากมิดี้เอาท์ใช้ส่งให้กับอุปกรณ์อื่น) ดังนั้นในการพ่วงซินธิไซเซอร์เข้าด้วยกันดังรูปที่ 5 แสดงการเชื่อมต่อ เมื่อเล่นซินธิไซเซอร์ #1 จะมีเสียงออกมาทั้งซินธิไซเซอร์ #1 และซินธิไซเซอร์ #2 (ซินธิไซเซอร์ #2 เล่นตามข้อมูลที่ส่งมาจาก ซินธิไซเซอร์ #1)

จากรูป (b) เป็นกรณีการควบคุมแบบรีโหมด (LOCAL CONTROL OFF) จะต่างกับแบบ (a) ที่ว่าข้อมูลจากคีย์บอร์ดจะส่งไปที่ MIDI OUT โดยตรง (ไม่ส่งให้กับส่วนสร้างเสียง) ดังนั้นในรูปแสดงการเชื่อมต่อถ้าอยู่ในลักษณะการควบคุมแบบรีโหมด เมื่อกดคีย์ที่ซินธิไซเซอร์ #1 จะมีเสียงออกเฉพาะที่ ซินธิไซเซอร์ #2 เท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4 แผนภาพแสดงทางเดินของสัญญาณ



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5 แผนภาพแสดงการเชื่อมต่อของอุปกรณ์ MIDI

ข้อมูลเกี่ยวกับการควบคุมคีย์บอร์ดแบบ โลคอล/รีโหมด ประกอบด้วยข้อมูล 3 ไบท์

1.ไบท์สถานะคือ 1011nnnn (ซึ่งเป็นไบท์สถานะของข้อความเกี่ยวกับการควบคุม) -nnnn คือ แชนแนลพื้นฐาน

2.หมายเลขตัวควบคุม = 122

3.ถ้าเป็นแบบรีโหมดจะทำให้ไบท์นี้เป็น 0 , และถ้าเป็นแบบโลคอลจะเป็น 127

การเลิกเล่นโน๊ตทุกตัว (ALL NOTE OFF) - เป็นการหยุดการสร้างเสียง (หยุดทั้งแชนแนลโดยไม่สนใจว่าเป็นโน๊ตตัวใด) ประกอบด้วยข้อมูล 3 ไบท์

1.ไบท์สถานะคือ 1011nnnn 2 (เป็นไบท์สถานะของข้อความเกี่ยวกับการควบคุมนั่นเอง โดยที่ nnnn คือ หมายเลขแชนแนล)

2.หมายเลขตัวควบคุม = 123

3.เป็นไบท์ที่ไม่มีความหมายใดๆ เพียงแต่ใส่ให้ครบ 3 ไบท์เท่านั้น (DUMMY BYTE) เนื่องจากข้อความเกี่ยวกับการควบคุมนี้จะมีความยาว 3 ไบท์

การเลือกโหมด OMNI/POLY/MONO - ความหมายของแต่ละโหมดต่างๆได้อธิบายในช่วงต้นแล้วซึ่งทุครั้งที่เปลี่ยนโหมดจะเลิกเล่นโน๊ตทุกตัว รูปแบบของข้อความนี้ก็คล้ายกับการควบคุมอื่นๆ ประกอบด้วย

1.ไบท์สถานะคือ 1011nnnn

2.หมายเลขตัวควบคุมของแต่ละโหมด เป็นดังนี้

OMNI OFF = 124

OMNI ON = 125

MONO ON = 126

POLY ON = 127

3.ไบท์ที่ 3 จะให้เป็น 0 สำหรับตัวควบคุมหมายเลข 123,124,125,127 แต่ในกรณีของโหมดโมโน 126 ไบท์นี้จะแทนด้วยหมายเลขแชนแนล

ข้อมูลทั่วไปของระบบ (SYSTEM COMMON MESSAGE) สำหรับทุกๆ แชนแนล -เนื่องจากข้อมูลประเภทนี้เป็นข้อมูลของระบบ ดังนั้นจะไม่มีการกำหนดแชนแนล หรือนั่นก็คือทุกแชนแนลต้องตอบสนองต่อข้อมูลทั่วไปของระบบ

การชี้ตำแหน่งของเพลง (SONG POSITION POINTER) - เป็นการบอกให้รู้ถึงตำแหน่งของเพลงที่เล่นอยู่ การบอกตำแหน่งจะบอกว่าเพลงได้ดำเนินไปแล้วที่บีท (MIDI BEAT) ซึ่งบอกได้สูงสุด 16,384 บีท โดยที่ 1 บีทมีค่าเท่ากับโน๊ตขเบ็จสองชั้น 1 ตัว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเลือกเพลง (SONG SELECT)คล้ายๆกับการเลือกโปรแกมเสียงแต่แทนที่จะเป็นโปรแกรมเสียงจะเป็นโปรแกรมรูปแบบ (pattern) สั้นๆของเพลงแทน (ความ ยาวไม่กี่ห้อง) โดยมากจะใช้กลองอิเลคทรอนิคแต่ละเพลงคือแต่ละจังหวะ เช่น เพลงที่หนึ่ง (SONG1) เป็นจังหวะร็อค และเพลงที่สอง(SONG2) เป็นจังหวะวอลซ์

ข้อความพิเศษของระบบ (SYSTEM EXCLUSIVE) - ได้อธิบายในตอนต้นแล้ว

ข้อความเกี่ยวกับเวลาในระบบ (SYSTEM REAL TIME MESSAGE) สำหรับทุกแชนแนล ซึ่งเป็นข้อความที่ทำให้เครื่องดนตรีต่างๆเล่นอยู่ในจังหวะเดียวกัน (SYNCHRONIZED) เช่นในการพ่วงซินธิไซเซอร์กับกลองอิเลคทรอนิค เข้าด้วยกันจำเป็นอย่างยิ่งที่อุปกรณ์ทั้งสองนี้จะต้องทำงานอยู่ในจังหวะเดียวกัน

การรีเซ็ทระบบ (SYSTEM RESET) - เป็นข้อความที่บอกให้อุปกรณ์ระบบ MIDI ทุกชิ้นทำการรีเซ็ทตัวเอง (กลับไปอยู่ในสถานะแรกหลังจากเปิดไฟเข้าเครื่อง)

สัญญาณนาฬิกา (TIMING CLOCK) - ในการกำหนดจังหวะจะมีการส่งสัญญาณนาฬิกาออกไปด้วยอัตรา 24 ลูกต่อโน๊ตตัวดำหนึ่งตัว(ซึ่งจะถูกส่งออกไปอย่างต่อเนื่อง)เพื่อให้จังหวะอ้างอิงในระบบควรใช้สัญญาณนาฬิกาเดียวกัน (มีอุปกรณ์ส่งสัญญาณนาฬิกาเพียงตัวเดียว)

การกลับไปที่จุดเริ่มต้น (START FROM FIRST MEASURE) - เป็นข้อความที่บอกให้ SEQEUNSER และกลองอิเลคทรอนิค กลับไปที่จุดเริ่มต้นของเพลง แล้วเริ่มเล่นเพลงทันทีที่ได้รับสัญญาณนาฬิกา นั่นก็คือการบอกให้อุปกรณ์ทุกชิ้นเริ่มทำงานพร้อมกัน

การหยุด (STOP) - ข้อความนี้จะบอกให้หยุดการกระทำทุกอย่างที่มีผลมาจากการกำหนดจังหวะ (TIMING SENSITIVE)

การเริ่มเล่นต่อ (CONTINUE START) - เป็นการบอกให้อุปกรณ์เล่นต่อจากตำแหน่งเพลงที่เล่นครั้งสุดท้าย

การตรวจสอบสภาวะการทำงาน (ACTIVE SENSING) - เป็นข้อความที่ใช้แก้ปัญหาตัวส่งและตัวรับถูกตัดขาดออกจากกัน (เช่นสายหลุด) ซึ่งก่อเกิดความผิดพลาด เช่น ส่งข้อความให้เล่นโน๊ตออกไปแล้วเกิดสายหลุด เมื่อทางด้านส่งส่งข้อความให้เลิกเล่นโน๊ตตามไปก็จะไม่ถึงด้านรับ ทำให้เกิดความเสียหายมากในการแสดงสด ซึ่งการทำงานของแอคทีฟเซ็นซิ่งจะทำงานโดยการส่งข้อความที่เรียกว่า"แอคทีฟเซ็นซิ่ง" ออกไปเรื่อยๆเมื่อทางด้านรับได้รับ "แอคทีฟเซ็นซิ่ง" แล้วถ้าในระยะเวลาสั้นๆไม่ได้รับข้อความนี้เข้ามาอีกก็จะหยุดสร้างเสียง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่กล่าวมาแล้วทั้งหมดเป็นความหมายของข้อความในระบบ MIDI ทั้งหมด แต่โดยทั่วไปแล้วเครื่องดนตรีแต่ละชิ้นจะไม่สามารถตอบสนองต่อข้อความเหล่านี้ได้ทุกข้อความ ซึ่งความสามารถในการตอบสนองต่อข้อความต่างๆของแต่ละอุปกรณ์ จะดูได้จาก MIDI IMPLEMENTATION SHEET ของอุปกรณ์นั้นๆ



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3 การออกแบบและการสร้าง

-รายละเอียดเกี่ยวกับโครงงาน

โครงงานนี้จะทำให้ MICROCOMPUTER ทำงานควบคุมเครื่องดนตรีต่างๆให้เล่นเพลงตามที่ต้องการโดยการบันทึกการเล่นของเครื่องดนตรีแต่ละชิ้นเข้าไปก่อน หรืออาจโปรแกรมโน๊ตดนตรีเข้าไปทีละ Track โดยสามารถโปรแกรมหรือบันทึกได้ถึง 128 Track แล้วจึง ส่งข้อมูลทั้งหมดให้เครื่องดนตรีเล่นพร้อมกัน โดยในขณะที่มีการรับข้อมูลเข้าไป COMPUTER สามารถส่งข้อมูลเก่าที่เก็บไว้แล้วออกไปพร้อมกันด้วย รวมทั้งการปรับปรุงแก้ไขข้อมูลส่วนดังกล่าวในขณะที่บันทึกด้วย (เช่นการเลื่อนระดับเสียง(TRANSPOSE) , การกำหนดความดัง(SET VOLUME) , การกำหนดเสียง(SET VOICE) , การกำหนดแทรค (SET TRACK) โดยจะส่งไปให้แทรคที่ต้องการ)

นอกจากนั้น ไมโครคอมพิวเตอร์ยังทำหน้าที่เป็น กลองอิเลคทรอนิค (DRUMMACHINE) ด้วยซึ่งผู้ใช้สามารถโปรแกรมผ่านมอนิเตอร์ได้อย่างสะดวก

การแก้ไขข้อมูลที่บันทึกเข้าไปก็สามารถกระทำได้โดยสะดวก ด้วยวิธีการแสดงผล(DISPLAY) เป็นรูปของการกดคีย์ หรือการเล่นขณะนั้นๆ และยังสามารถทำการลบ,การCOPY และการเลื่อนระดับเสียงได้เช่นกัน นอกจากนี้ยังมีส่วนของการควอนไตซ์ (QUANTIZE) ซึ่งจะทำหน้าที่วิเคราะห์ข้อมูลที่เล่นจากนักดนตรีว่าควรจะมีการลงจังหวะอย่างไรถึงจะถูกต้อง รวมถึงการปรับแต่งข้อมูลอื่น ๆ อีกซึ่งช่วยทำให้มีประสิทธิภาพในการบันทึกข้อมูลสมบูรณ์ยิ่งขึ้น และยังสามารถแก้ไข,ลบ,ปรับระดับความดังของโน๊ตที่เล่นเข้าไปได้อีกด้วย

โดยสรุปก็คือ MICROCOMPUTER จะต้องทำหน้าที่เป็น SEQUENCER จัดลำดับข้อมูลสื่อสารระหว่างเครื่องดนตรี กับ นักดนตรี ให้มีประสิทธิภาพมากที่สุด และใช้งานได้ง่าย

-ฮาร์ดแวร์ของ MIDI CARD

หลักการทำงานที่สำคัญของ MIDI CARD อยู่ที่ Z80B-SIO/0 ซึ่งจะทำหน้าเป็นSERIAL-I/O โดยเราจะทำการโปรแกรมให้ SIO ทำงานในโหมดอินเทอร์รัพท์ (INTERRUPT MODE) ตามรูปของวงจรรูปที่ 9

Z80B-SIO จะรับความถี่ในการมอดบนคริสตอล (CRYSTAL) ขนาด 2 MHz ซึ่งต่อร่วมอยู่กับ IC เบอร์7400 ในการส่งข้อมูลจะมีIC 74LS04 ทำหน้าที่เป็นบัฟเฟอร์โดยจะมีตัวความต้านทาน (RESISTOR) ขนาด 220 โอห์ม ซึ่งจะ MATCH กับค่าอิมพีแดนซ์ของMIDI IN ตามค่ามาตรฐานพอดี

ในการอ้างถึง PORT ของ IBM/PC ถึง Z80B-SIO จะใช้ 74LS138 2 ตัวร่วมกับแนนท์เกท 7400 โดยจะต้องสร้าง IORQ ให้กับ Z80B-SIO เนื่องจากการจัดขาของ Z80B-SIO ไม่ตรงกับพอร์ทของ IBM-PC

สำหรับโครงงานนี้จะเลือกใช้ทั้งสอง CHANNEL ของ Z80B-SIO เพื่อให้สามารถอ้างแชนแนลได้ 32 แชนแนลและใช้หมายเลขของคอนโทรลพอร์ท (CONTROL PORT) คือ 2B1h สำหรับ แชนแนล 1-16 และ 2A1h สำหรับแชนแนล 17-32 และดาต้าพอร์ท (DATA PORT) คือ 2B0h สำหรับแชนแนล 1-16 และ 2A0h สำหรับแชนแนล 17-32 Z80B-SIO จะถูกโปรแกรมให้รับส่งข้อมูลด้วย BAUD RATE FACTOR = 64 ซึ่งจะได้ความถี่ใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การมอดดูเลชั่น = 2 MHz/64 = 31.25 KBaud พอดี และมีขนาดข้อมูล 8 BIT รวมกับ START/STOP BIT อีกอย่างละ 1 BIT

ส่วนประกอบอีกส่วนหนึ่งคือ 8253 ซึ่งเป็น PROGRAMMABLE INTERVAL TIMERโดยจะนำมาใช้ทำฐานเวลาให้กับระบบ สำหรับตัว 8253 เองจะมีวงจรนับ (COUNTER) อยู่ 3 ตัวที่เป็นอิสระต่อกันได้แก่ C0,C1 และ C2 ที่ใช้ใน MIDI CARD นี้เราจะใช้ C0 กับ C1 โดยเราจะกำหนดให้ C1 ทำงานในโหมด 3 คือจะผลิต SQUARE WAVE ซึ่งในโหมดนี้ เราจะใช้ในการนำมาหารความถี่ลงมาจากความถี่อินพุทแล้วป้อนให้ที่ขา C1 (ดังนั้นที่ขาอินพุทของ C0 จะมีความถี่เท่ากับ 1 MHz โดยการหาร2) และที่ C0 ก็จะทำงานที่โหมด 3 เช่นกันแต่ค่าที่ใช้ในการหารความถี่ช่วงนี้จะเป็นค่ามาจากผู้ใช้ (USER) เพื่อใช้กำหนดเป็นฐานเวลาของความเร็วของจังหวะ (TEMPO - TEMPO = 120 หมายถึง ในเวลา1 นาที จะประกอบด้วย โน๊ตตัวดำ 120 ตัว)จากนั้นนำค่าความถี่ที่ได้จาก C0 ไปทริกที่ IORQ3 ส่วน Z80B-SIO ที่ขาของ INT จะต่อเข้ากับ IORQ2 เพื่ออินเทอร์รัพท์ให้เกิดการรับส่งข้อมูลจากเครื่องดนตรี

ความจำเป็นที่ต้องใช้ 8253 นั้น เพราะเนื่องจากเครื่องดนตรีบางชนิดไม่มีการส่งรหัสClock ออกมาดังนั้นจึงจำเป็นที่จะต้องสร้างฐานเวลาขึ้นมาเป็นตัวควบคุมการ บันทึกและการส่งข้อมูล

คอนเนคเตอร์ในส่วนการติดต่อกับอุปกรณ์ MIDI จะใช้สาย DIN 5 ขาตามมาตรฐานของ MIDI แต่ในส่วนที่ติดต่อกับ IBM/PC จะใช้ DB CONNECT 9 PIN เพื่อความสะดวกในการวิจัย

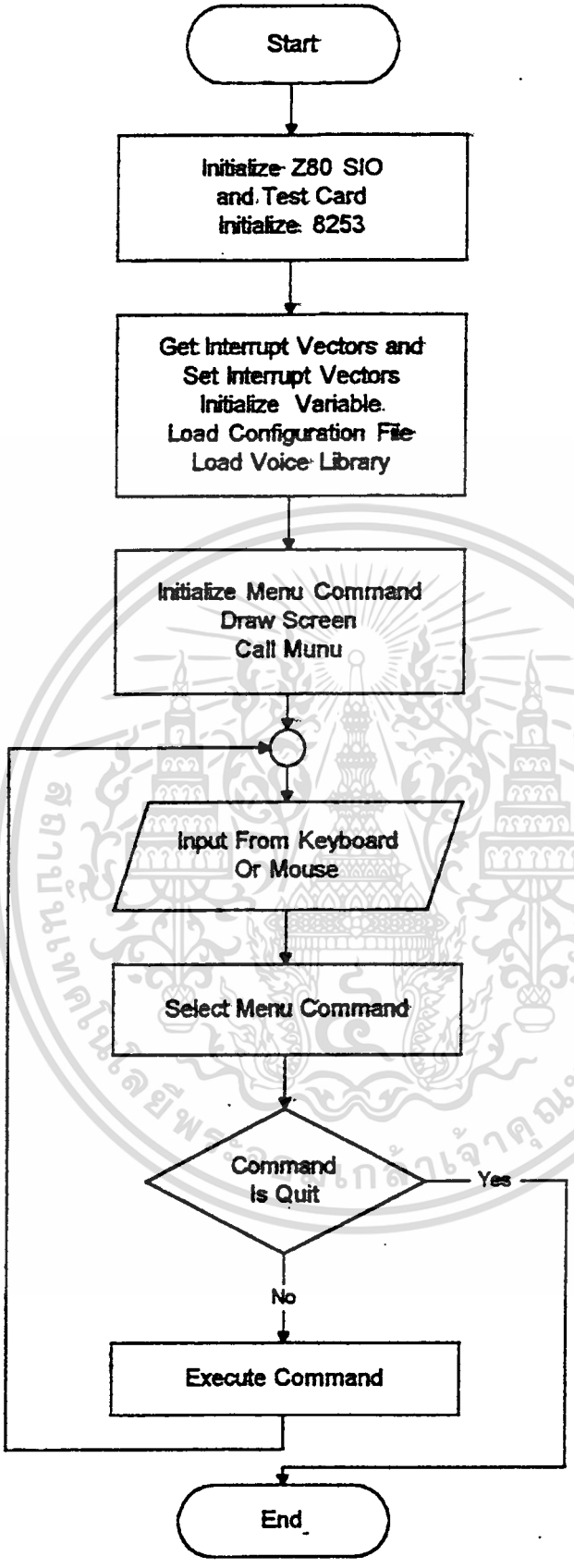
-ซอฟท์แวร์

ในโครงงานนี้ ใช้ ภาษา C ในการเขียนโปรแกรม และใช้ Borland C++ เป็น Compiler เนื่องจากโปรแกรมมีขนาดใหญ่ไม่สามารถ Compile ได้โดย Turbo C นอกจากนี้ Borland C++ ยังมี Inline Assembler ซึ่งทำให้สามารถเขียน Source Code Assembly 8088 ร่วมกับ Source Code ภาษา C ได้ทันที และภาษา C ยังสามารถติดต่อกับ Hardware ระดับล่างได้เป็นอย่างดีอีกด้วย การเขียนโปแกรมในการสร้างDATA และการส่งข้อมูล จึงสามารถทำได้สะดวก

ต้องทำความเข้าใจกับคำว่า Track และ Channel ก่อน Track คือรูปแบบหรือโน๊ตหรือ ข้อมูลของการเล่นนั่นเอง ส่วน Channel คือ ช่องของเครื่องดนตรีต่างๆ ที่สามารถ Generate เสียงขึ้นมาได้โดยไม่ขึ้นแก่กัน ดังนั้นเราสามารถส่งข้อมูลหลายๆ Track ไปที่ Channel เดียวกันได้ และโดยเฉพาะเมื่อเล่นโน๊ตไม่ซ้ำกันในระหว่างTrack ที่ส่งข้อมูลออก Channel เดียวกันนั้น แต่ก็ควรระวังข้อมูลบางชนิดที่จะมีผลต่อทุกตัวโน๊ตที่เล่นอยู่ เช่น Control Change ตัวอย่างที่เห็นได้ชัดคือ All Note Off จะเป็นการเลิกเล่นโน๊ตที่ On อยู่ All Note Off จะส่งผลถึงโน๊ตของ Track ที่ส่งออกที่Channel เดียวกันด้วย

การเก็บ DATA ของโปรแกรมที่เขียนขึ้นมาใช้ในโครงงานนี้ จะใช้การ Swap ลง File แล้วเมื่อต้องการใช้ข้อมูลนั้นก็จะทำการจอง Memory จากระบบขึ้นมา และ Load ข้อมูลจาก Swap File มาใส่ Memory File ที่เก็บอยู่นั้นชื่อ Track.0,Track.1,...โดย ส่วนขยายของชื่อไฟล์เป็นหมายเลขแทรค ลบ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6 Flowchart ของ โปรแกรมหลัก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-ส่วน SEQUENSER

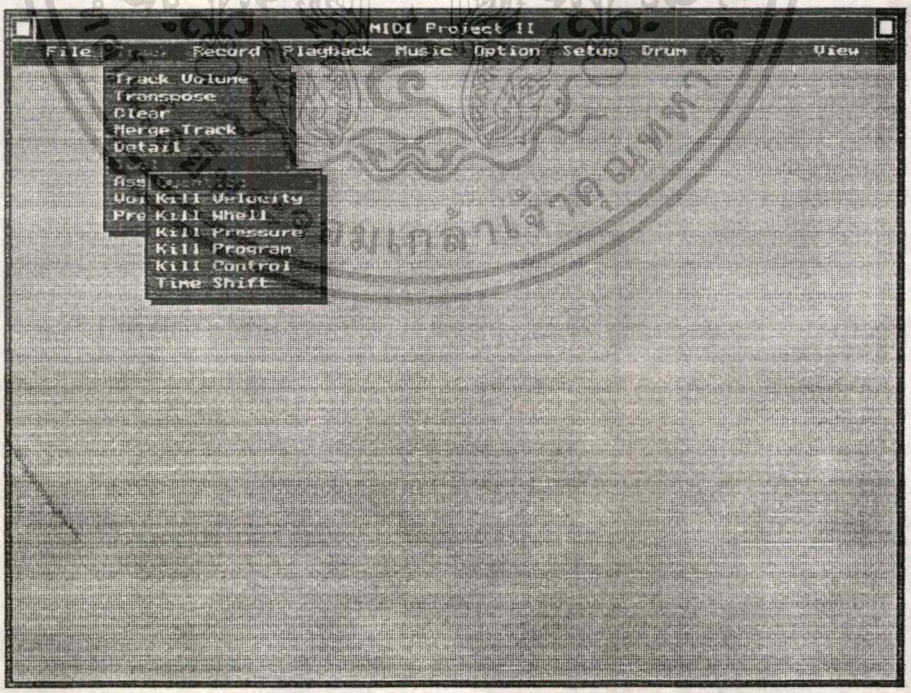
เป็นส่วนของการจัดลำดับการส่งข้อมูล ทำให้เครื่องดนตรีสังเคราะห์เสียงขึ้นมาได้พร้อม ๆ กันหลาย ๆ เสียงโดยจะมี Pointer เป็นตัวชี้ข้อมูลของแต่ละแทรค ในตอนเริ่มต้น Pointer จะชี้อยู่ที่ข้อมูลเวลา จะนำค่าเวลาที่ Pointer ชี้ได้ไปเปรียบเทียบกับค่าเวลาของระบบ (ซึ่งจะเพิ่มค่าเองจาก Interrupt Routine ของ IORQ 3) ว่ามีค่าเท่ากันหรือน้อยกว่าหรือไม่ หากใช่ ก็หมายความว่าถึงเวลาที่จะต้องส่งข้อมูลนั้นแล้ว ก็จะเลื่อน Pointer ไป 3 Byte ตอนนี้ Pointer ก็จะชี้อยู่ที่ ข้อมูลที่ต้องทำการส่ง และก็จะนำข้อมูลมา ปรับแต่งตาม Patameter ต่าง ๆ แล้วแล้วทำการส่งข้อมูลที่ Pointer ชี้อยู่นั้นไปแล้วเลื่อน Pointer ไปเรื่อย ๆ จนกว่า Pointer จะไปชี้อยู่ที่ FF ก็จะหยุดส่งข้อมูลแล้วเลื่อน Pointer ไปอีก 1 Byte ตำแหน่งที่ Pointer ชี้อยู่เป็นข้อมูลเวลา แล้วโปรแกรมก็จะทำเช่นนี้ไปอีกจนกว่าจะครบทุก Track

-ส่วน DRUM MACHINE

ตามปกติของการเล่นกลองแล้ว จะเล่นเป็นรูปแบบที่ ซ้ำเดิมเป็นส่วนมาก ซึ่งในที่นี้จะขอเรียกว่า Pattern เราสามารถที่จะสร้าง Pattern ขึ้นมาหลาย ๆ Pattern แล้วนำมาจัดเรียงกันให้เป็นดังที่เราต้องการได้

-ส่วนปรับปรุงข้อมูล

โปรแกรมส่วนนี้เป็นโปรแกรมที่ใช้ปรับปรุงข้อมูลให้ดีขึ้น เช่น การ Quantize การลบข้อมูลบางชนิดออก โดยอยู่ใน Menu Track / Tool ซึ่งจะเห็นว่ามีดังรูปที่ 7 และ รูปที่ 8 และตัวอย่างของการใช้ Tool / Quantize / 1/4 แสดงให้เห็นดังรูปที่ 9

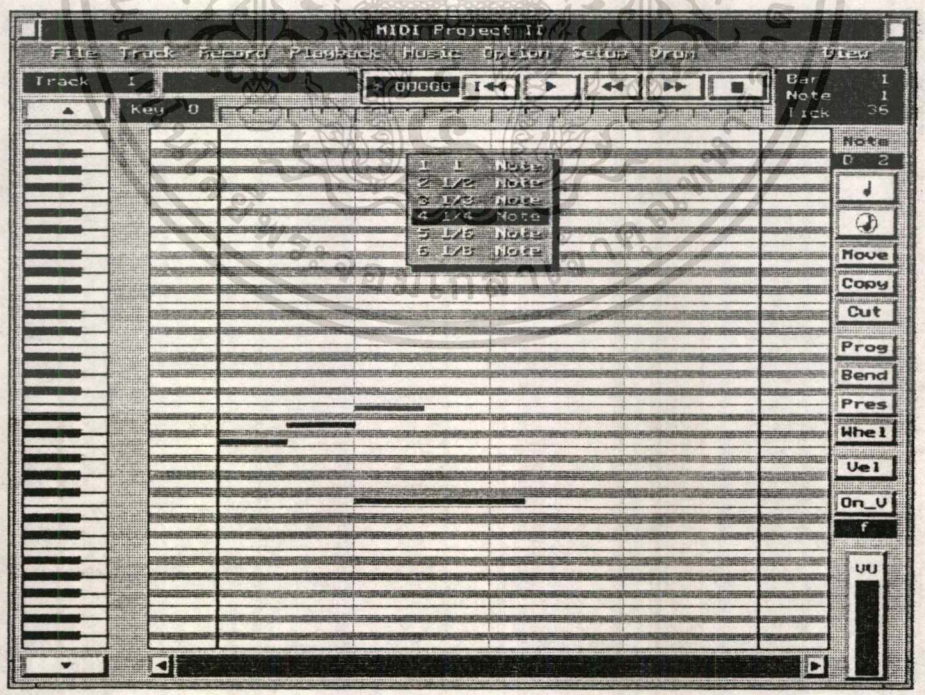


รูปที่ 7 Menu Track / Tool

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 8 Menu Tool ใน Step Record



รูปที่ 9 ตัวอย่างของการ Quantize ช่วยจัดจังหวะโน๊ตให้ถูกต้อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-หน่วยความจำ (MEMORY)

เมื่อเริ่มรันโปรแกรม ก็จะทำการจอง Memory ไว้ 64 kByte เป็น Temporary Memory เพื่อใช้ในการ Load ,Save ,Edit ,Record และใช้ในการปรับแต่งข้อมูลต่างๆ ข้อมูลที่ใช้ในการ Setup Parameter ต่างๆ จะใช้ Variable และข้อมูล MIDI นั้น จะใช้การจอง Memory จาก Dos เข้ามา และใช้ Pointer เป็นตัวชี้ เมื่อไม่มีความจำเป็น ที่จะต้องใช้ข้อมูล ก็จะทำการเก็บลง Disk ในส่วนที่เป็น Temporary แล้วคืน Memory สู่ระบบ และเนื่องจาก โปรแกรมจะกินเนื้อที่ใน Memory ประมาณ 250 kB ดังนั้นในการรันโปรแกรมควรมี Conventional Memory เหลืออยู่อย่างน้อย 400 kB

-FORMAT ของข้อมูล

เราจะเก็บข้อมูลไว้ใน Memory โดยใช้ Pointer เป็นตัวชี้ ซึ่งจะมีขนาดตามจำนวณข้อมูล แต่เมื่อยังไม่มีความต้องการใช้ข้อมูล จะทำการ Swap ข้อมูลลงเก็บใน Disk แล้วทำการคืน Memory สู่ระบบ

ซึ่งลักษณะการเก็บข้อมูลจะเก็บข้อมูลเวลาก่อน (3 Byte) แล้วตามด้วยข้อมูลMIDI แล้วตามด้วย FF เป็นการบอกว่า ข้อมูล MIDI ที่เวลานี้หมดแล้ว แล้วตามด้วยข้อมูลชุดต่อไป เมื่อหมดข้อมูล จะตามด้วย FFFF โดยเวลาจะเรียงกันอยู่จากมากไปน้อย

-ซอฟท์แวร์ ส่วนที่ทำงานร่วมกับ ฮาร์ดแวร์

ซอร์ฟแวร์ของโครงงานนี้จะทำงานประสานกับฮาร์ดแวร์หลายๆส่วนในระบบซึ่งได้แก่

-คีย์บอร์ด - ซึ่งใช้ในการติดต่อกับระบบ ทั้งในขณะที่เล่นเพลงและ SETUP PARAMETER ต่างๆของเพลง รวมทั้งการ LOAD,SAVE และ การแก้ไข(EDIT)

-จอภาพ - จะใช้แสดงผลข้อมูลต่างๆของระบบ และที่สำคัญก็คือการแสดงผลเวลาหรือหมายเลขบีท (BEAT) ของระบบซึ่งใช้กราฟิกโหมดในการแสดงผล

-INT 1Ch - ไทม์เมอร์ ทิค อินเทอร์รัพท์ (TIMER TICK INTERRUPT) ซึ่งจะเกิดขึ้น 18.2 ครั้งใน 1 วินาที ซึ่งเราจะใช้โปรแกรมย่อย (ROUTINE) นี้ในการส่งแอ๊คทีฟ เซ็นซ์ (ACTIVE SENSE) ลงไปในอินเทอร์รัพท์นี้.

-MIDICARD-8253 INTERRUPT - จะเป็นฐานเวลาสำหรับเพลงที่จะถูก SET ให้มีขนาด 96 CLOCK/โน๊ตตัวดำ ผ่านทาง IORQ3

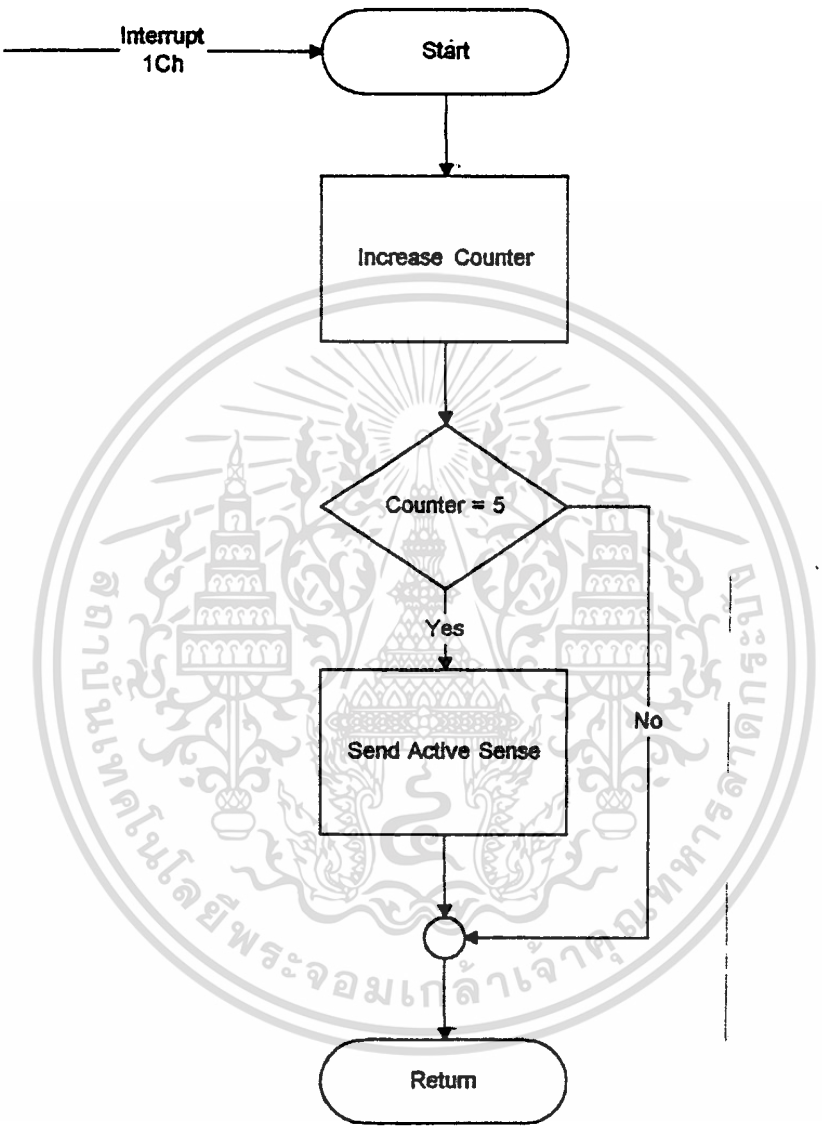
-MIDI CARD-SIO INTERRUPT - จะ INTERRUPT เมื่อมีการส่งข้อมูลเข้าในระบบโดยจะ INTERRUPT ผ่านทาง IORQ2

-อินเทอรรัพท์หมายเลข 1Ch

เนื่องจากความจำเป็นของ MIDI CONTROLLER ที่จะต้องทำการส่ง ACTIVE SENSE ให้กับอุปกรณ์ในระบบทุกตัว ในขณะที่ไม่มีการส่งข้อมูล MIDI ทุกๆ 300 ms ดังนั้นเราจะใช้ INT 1Ch นี้ และเนื่องจากการ INTERRUPT นี้จะใช้เวลา 18.2 ครั้งต่อวินาที ดังนั้นหมายความว่า ถ้าเกิด INT 1Ch 5 ครั้ง โดยไม่มีการส่งข้อมูลเลยก็จะส่งสัญญาณ ACTIVE SENSE ออกไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อย่างไรก็ตามขณะที่เราส่งข้อมูล MIDI ออกไปเราต้องทำการรีเซ็ทของวงจรนับการ INTERRUPT ของ INT 1Ch ให้เป็นศูนย์ใหม่เพื่อการเริ่มนับใหม่ทุกครั้งด้วย



รูปที่ 10 Flow Chart ของการส่ง Active Sense

-IORQ3 8253-MIDI CARD

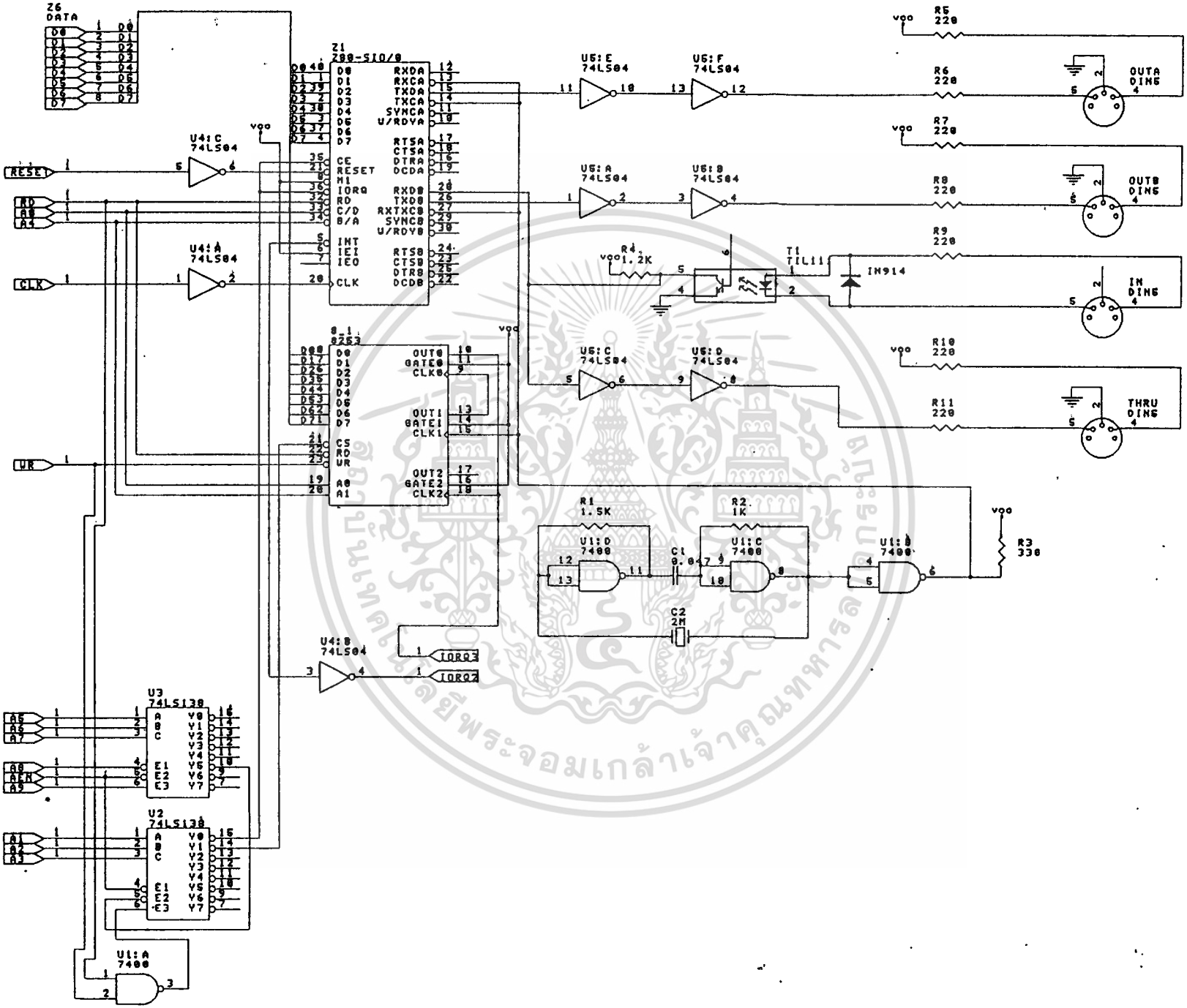
ขณะที่เล่นเพลงเราจะต้องมีเวลาที่แน่นอน ซึ่งเราสามารถกำหนดให้มีความเร็วตามมาตรฐาน MIDI คือ 24 CLOCK/โน๊ตตัวดำ ซึ่ง INTERRUPT ตัวนี้จะทำการเพิ่มเวลาของระบบแล้วส่งรหัสที่หมายถึงCLOCK ในที่นี้คือรหัส F8h ออกไปด้วยทุก 4 ครั้งที่มีการ Interrupt ของ 8253 และ IORQ3 จะทำการแสดงผล เวลาในหน่วย บีท ซึ่งในที่นี้คือ 96 ครั้ง จะมีการแสดงโน๊ตหนึ่งครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-IORQ2 SIO-MIDI CARD

เราได้โปรแกรมให้ Z80-SIO/0 รับข้อมูลจาก MIDI ใน Interrrupt Modeและติดต่อกับคอมพิวเตอร์ผ่านทาง IRQ2 เมื่อมีการ Interrupt เข้ามาทาง IRQ2 ก็จะมีการรับข้อมูลเข้ามาครั้งหนึ่ง ซึ่งจะนำข้อมูลที่อ่านมาจาก Z80-SIO/0 เก็บไว้ใน Queue ที่มีขนาด 64 Byte และเป็น Queue แบบวน และโปรแกรมหลักจะนำข้อมูลจาก Queuue นี้ไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 11 วงจรของ MIDI Interface CARD ที่ใช้ในโครงงานนี้

บทที่ 4 การทดลองและผลการทดลอง

ได้มีการนำ MIDI Card ที่สร้างขึ้นและโปรแกรมที่เขียนขึ้นนี้ไปทดลองใช้กับเครื่องดนตรีประเภท Synhesizer หลายเครื่องและคอมพิวเตอร์ที่มีความเร็วต่าง ๆ กัน ดังนี้

Computer

- PC-AT 80286-16
- PC-AT 80386 SX-16
- PC-AT 80386 DX-25
- PC_AT 80386 DX-33
- PC-AT 80486 DX-33
- PC-AT AMD-80486 DX-40
- PC-AT 80486 DX2-66

Synthesizer

- KAWAI K1 II
- KAWAI K11
- Technics KN650
- Roland JV30

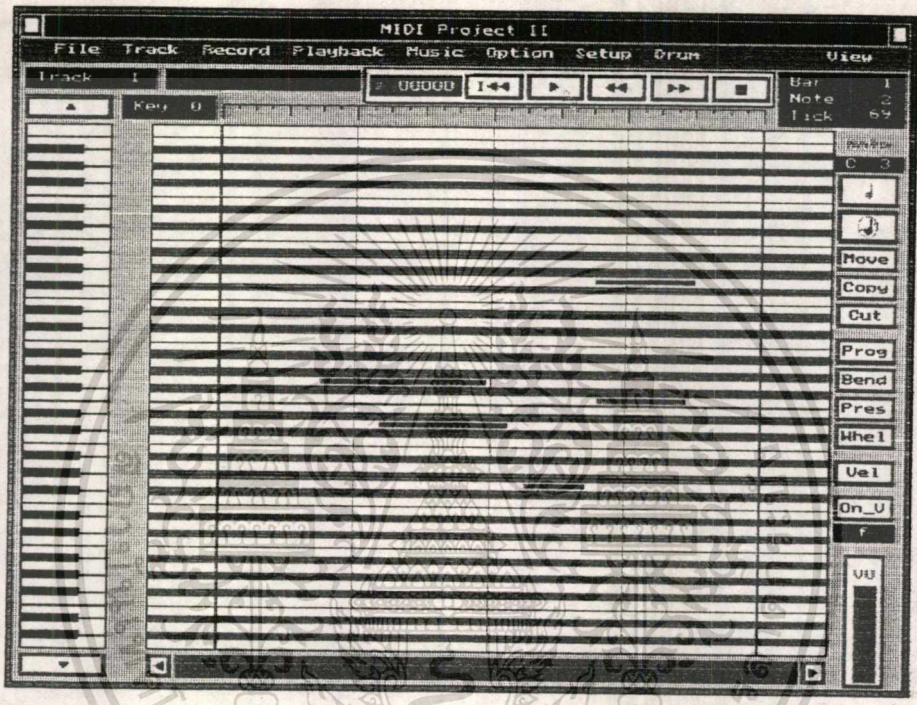
ปรากฏว่า สามารถใช้งานได้ดีในทุกเครื่องดนตรี และ คอมพิวเตอร์ โดยให้ เอาท์พุทใน จังหวะที่เท่ากัน แต่เมื่อจำนวณ แทรค ที่ถูกใช้มีมาก และข้อมูลที่ถูกส่งออกในเวลาเดียวกันมีมาก จะทำให้มีการกระตุกของเพลงบ้างเนื่องจาก Protocal ของ MIDI เอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



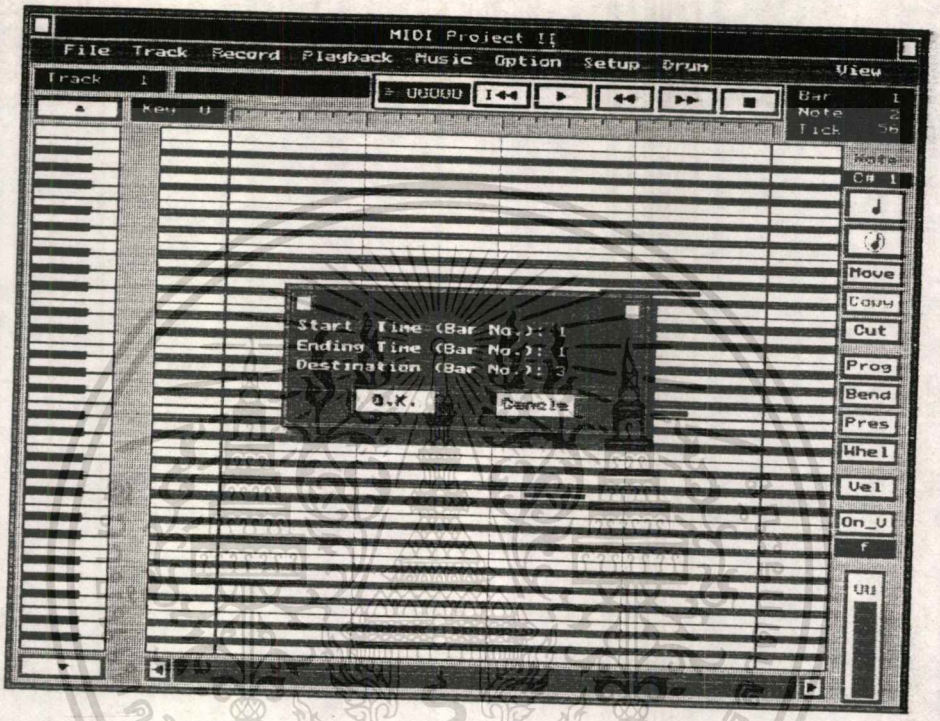
รูปแสดง Menu ของโปรแกรมหลัก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



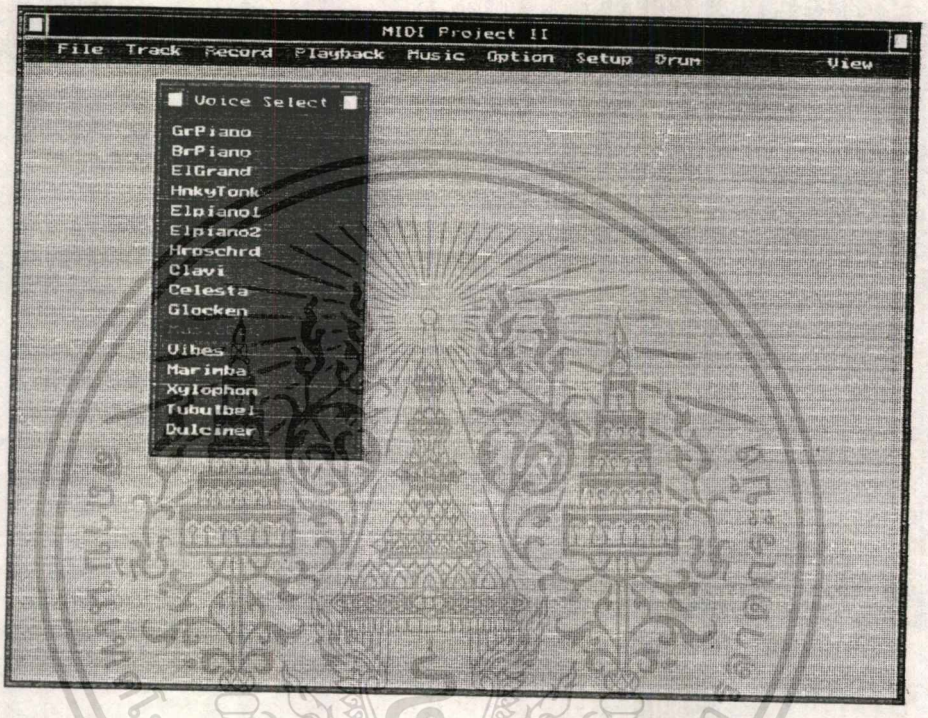
รูปแสดง ลักษณะการใส่โน๊ตของ Step Record

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



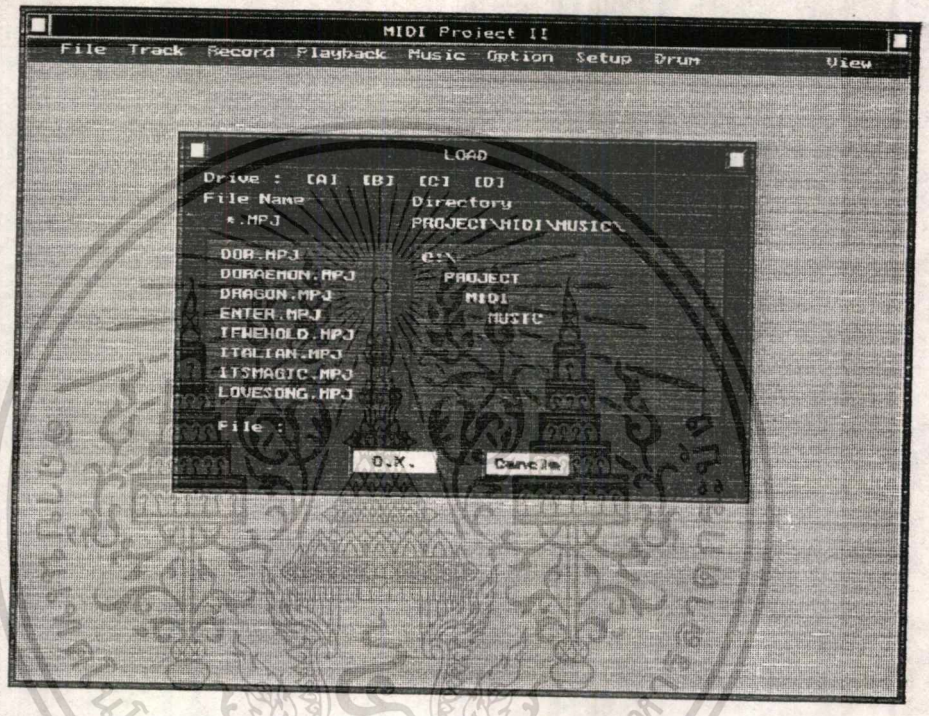
รูปแสดง การคัดลอกโน๊ตระหว่างห้องของ Step Record

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



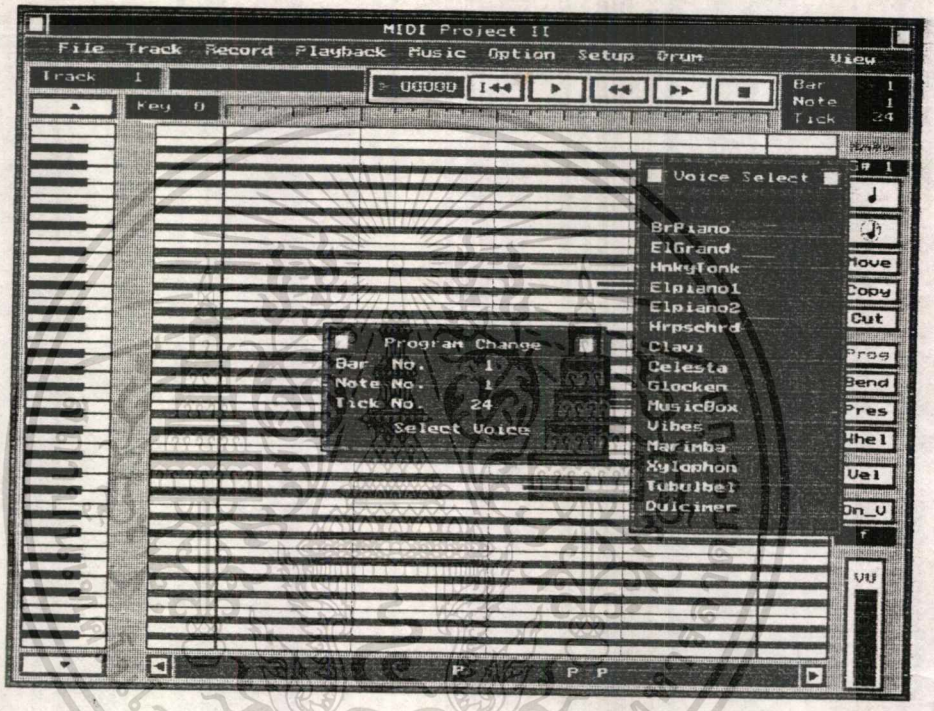
รูปแสดงการเลือกเสียงของเครื่องดนตรี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



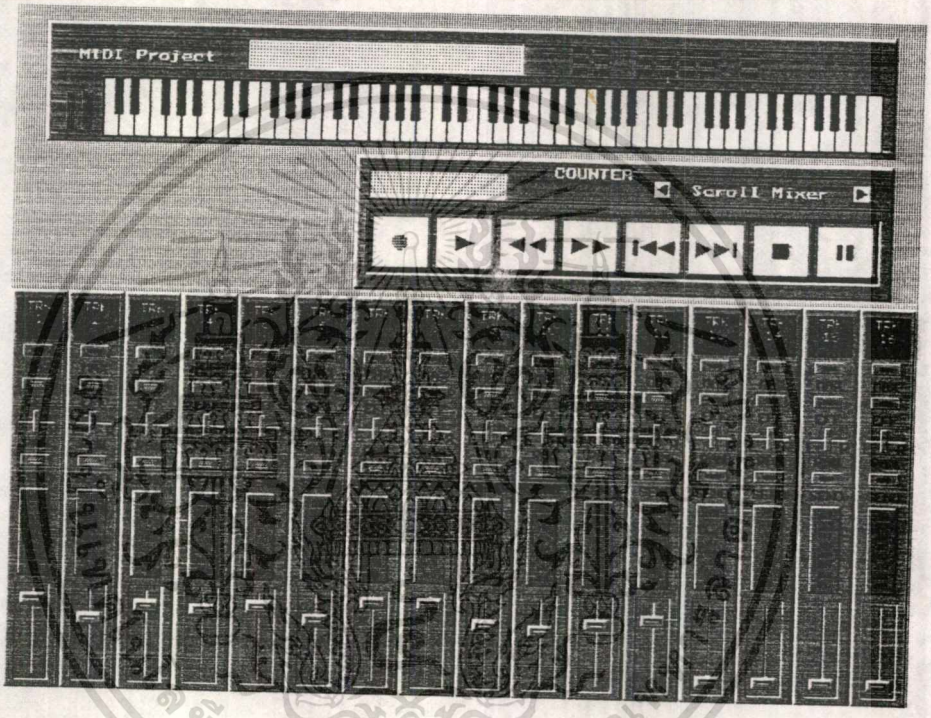
รูปแสดงการ Load เพลงที่ทำการบันทึกไว้ขึ้นมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้



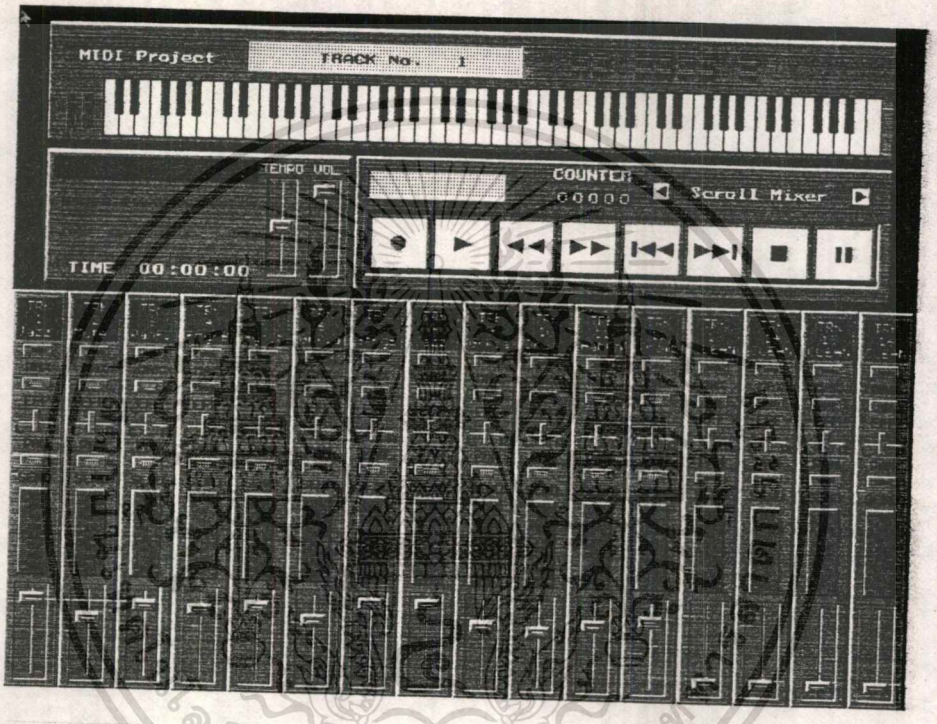
รูปแสดการ เปลี่ยนเสียงเครื่องดนตรีในระหว่างการเล่นโน๊ต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



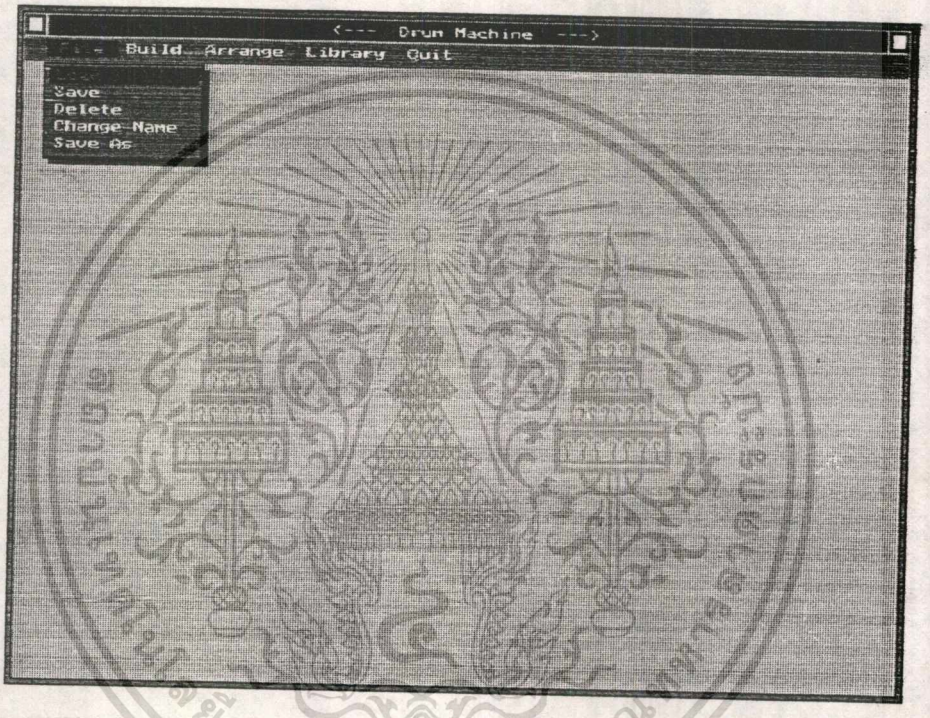
รูปของ Realtime Record

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



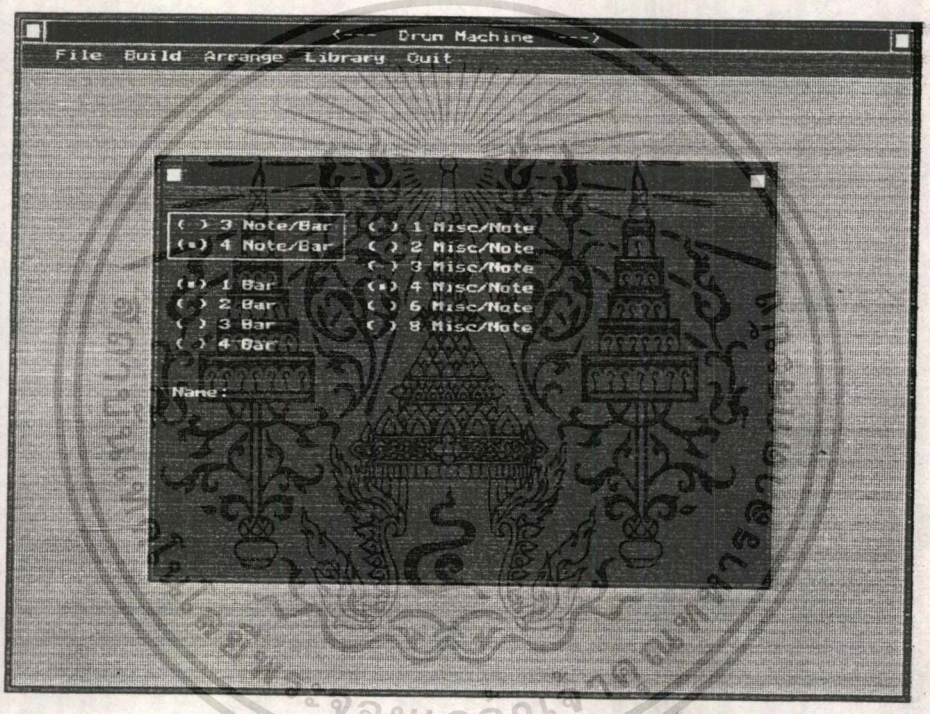
รูปของ Playback Screen

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



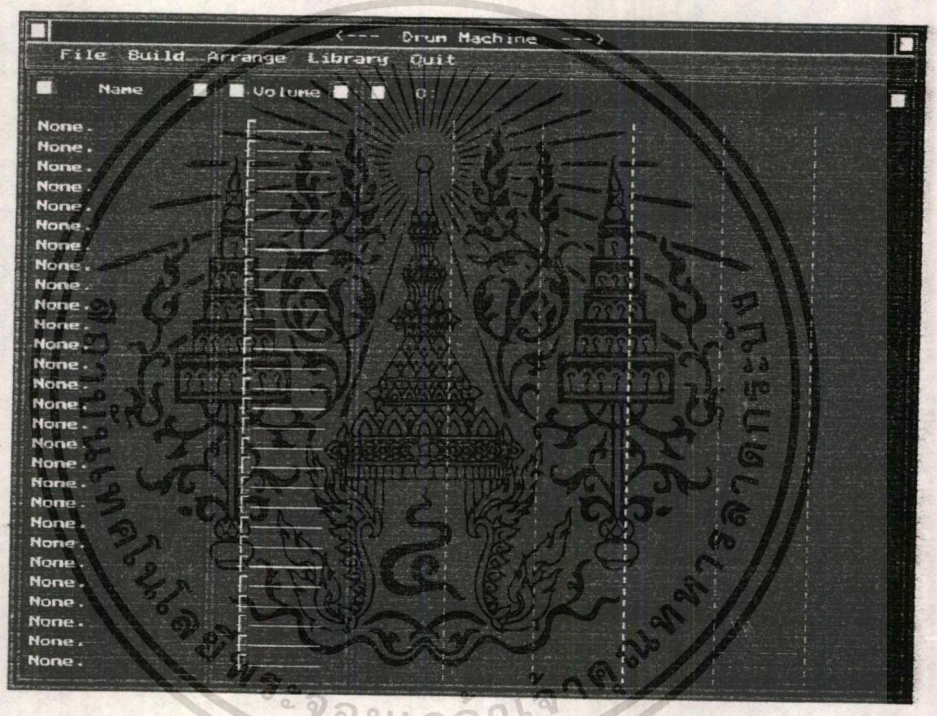
รูป Menu ของ Drum Machine

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



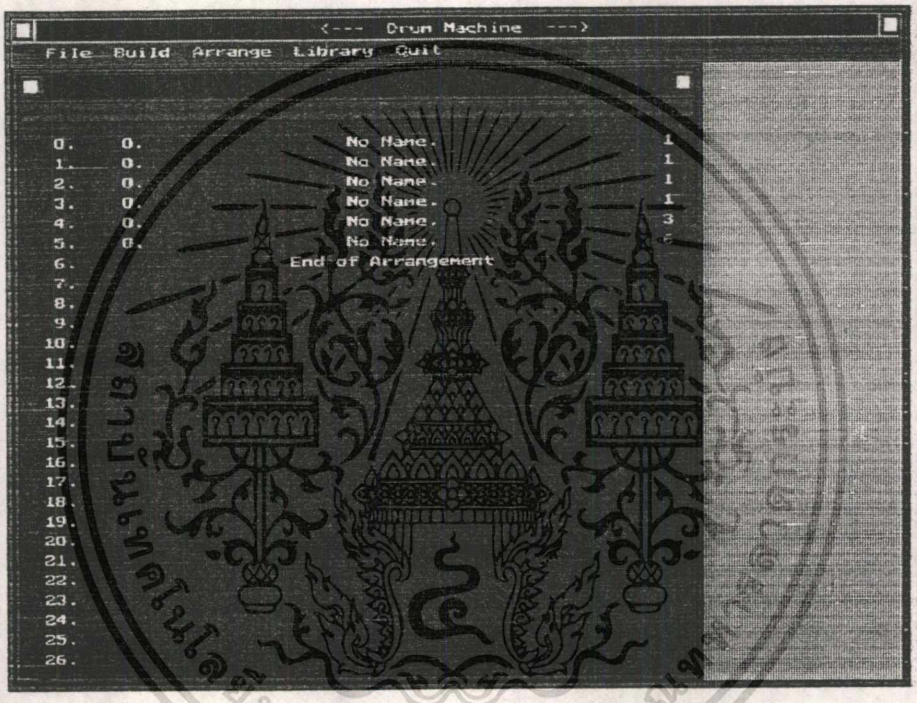
รูปแสดงการ Setup Pattern ของ Drum Machine

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดงการสร้าง Pattern ของ Drum Machine

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดงการนำ Pattern ของ Drum Machine มารวมกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5 สรุปและวิจารณ์ผล

1. MIDI Card และ โปรแกรม MIDIPROJ

โปรแกรมที่เรียนขึ้นใน โครงงานนี้ สามารถ โปรแกรมโน๊ตได้ในรูปแบบของTablature ของ Keyboard ยังไม่เป็น โน๊ต บรรทัด 5 เส้นซึ่งเป็นสากลกว่าแต่ก็สามารถใช้งานได้แล้วโดยการบันทึกข้อมูลเข้าไปทีละ Track (ยังไม่สามารถบันทึกข้อมูลคราวละหลาย Track หรือ หลาย Channel ได้) สามารถปรับแต่งข้อมูลโดยการใช้ Tool ที่มีให้ ซึ่งทำให้ข้อมูลดีขึ้นตามต้องการได้ สามารถใช้ Editor ที่สร้างขึ้นมาในการใช้ ลบข้อมูลที่เล่นผิดพลาดได้ เพิ่มข้อมูลที่เล่นไม่ทันได้ เปลี่ยนความหนักเบาของการเล่นโน๊ตได้ ปรับแต่งการเล่นที่ไม่ลงจังหวะดนตรี (อาจมีการเหลื่อมล้ำนิดหน่อย) ให้ลงจังหวะที่ถูกต้อง จากการบันทึกจากการเล่นจริงได้

แต่เนื่องจากมาตรฐานการส่งข้อมูลของ MIDI เอง ที่กำหนดไว้ว่า Baud Rateเป็น 32.15 kBaud และเป็นแบบ Asynchronous ทำให้การส่งข้อมูล แต่ละ Byte ใช้เวลาประมาณ 300S การเปิดเสียงโน๊ต 1 ตัวต้องส่งข้อมูล 3 Byte ก็จะใช้เวลาประมาณ 1 mS ดังนั้นการเปิดเสียงโน๊ต 20 ตัวก็จะใช้เวลาถึง 20 mS ซึ่งเวลาขนาดนี้หูของคนเราก็สามารถฟังออกได้ ดังนั้น การส่งโน๊ตหลายๆตัวพร้อม,ๆ กันจะทำให้เพลงกระตุกได้ แต่การกำหนดให้จังหวะของเพลงเร็วนั้นไม่ได้ทำให้เพลงกระตุก เนื่องจาก การเพิ่มความเร็วของเพลง เป็นการเพิ่มความถี่ในการ Interrupt CPU ที่ IRQ 3 เท่านั้นเองซึ่งเป็นการเปลี่ยนค่าเวลาของระบบเท่านั้นเอง

ในโครงงานนี้เป็นเพียงการศึกษาและการทดลองสร้าง Application เพื่อใช้ในการแต่งเพลง ซึ่งผลที่ออกมาดีพอสมควร แต่ก็คงไม่สามารถเปรียบเทียบกับโปรแกรมของต่างประเทศที่ทำออกมาขายและใช้ในห้องอัดจริงๆ ได้

ในโครงงานนี้ใช้ Z80-SIO เป็นตัว Interface ซึ่งเป็น USART ที่ Design มาเพื่อใช้กับ Micro Processor ตระกูล Z80 เมื่อนำมาใช้กับ PC ทำให้มีปัญหาในเรื่องของความ Compatible ซึ่งจริง ๆ แล้วเราสามารถใช้ UART เบอร์ใดก็ได้ (เช่น 8250,8251 ซึ่งเป็นของบริษัท Intel เอง) ที่มีความ Compatible มากกว่า ส่วนเหตุผลที่เลือกใช้ Z80-SIO นั้นเป็นเพราะ Z80-SIO มี USART 2 ชุดใน Chip เดียวกันและ Synthesizer ที่ใช้ในวิทยานิพนธ์นี้เป็น KAWAI - K11 ซึ่งสร้างมาให้รับ Input ได้ 32 Channel โดยการแบ่งเป็น 2 Port

ประโยชน์ที่ได้จากโครงงานนี้

-เราสามารถควบคุมเครื่องดนตรีให้เล่นเพลงตามที่ต้องการได้ ทำให้บุคคลทั่วไปที่มีความรู้พื้นฐานทางทฤษฎีดนตรีสามารถแต่งเพลงและสร้างงานทางด้านดนตรีได้ และเหมาะกับผู้สนใจและเริ่มหัดดนตรีใหม่ นอกจากนี้ยังช่วยให้นักดนตรีสร้างสรรงานทางดนตรีได้อย่างรวดเร็วและมีประสิทธิภาพมากกว่าความสามารถที่มีอยู่ในตัวนักดนตรีเอง

-ทำให้การแต่งเพลง สามารถทำได้ง่ายกว่าเดิมมาก ซึ่งผู้แต่งก็ไม่จำเป็นที่ต้องมีความสามารถทางด้านดนตรีสูง และไม่จำเป็นต้องเล่นดนตรีได้หลายๆประเภทในการแต่งเพลงหนึ่งๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-ทำให้ประหยัดเครื่องดนตรีและอุปกรณ์ลงไปมากเนื่องจาก เพียงใช้ไมโครคอมพิวเตอร์และซินธิไซเซอร์ชุดเดียวก็สามารถทำหน้าที่แทนเครื่องดนตรีได้เกือบทุกชนิด

-เป็นพื้นฐานในการศึกษาถึง ระบบ MIDI ซึ่งเป็นระบบมาตรฐานทั่วโลก และยังเป็นพื้นฐานในการพัฒนาเครื่องดนตรีต่างๆด้วย

-เราสามารถประยุกต์ใช้ระบบการส่งข้อมูลให้ไปควบคุมการทำงานอื่นๆได้เช่น การควบคุมแสงไฟตามจังหวะ หรือระบบที่ซับซ้อนกว่านี้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คู่มือการใช้งาน โปรแกรม MIDI PROJECT

Content

1. Software and Hardware Requirement

2. Installing Program

3. Getting Start

4. Song Writing

5. Playback

6. Drum Machine

1. Software and Hardware Requirement

โปรแกรมนี้เป็นโปรแกรมที่ทำงานร่วมกับ MIDI CARD ได้ ดังนั้นก่อนอื่นต้องติดตั้งMIDI CARD เข้ากับเครื่อง-PC เสียก่อน (อย่างน้อยควรเป็น 386 DX-33 และจอที่ใช้ต้อง Support VGA ด้วย เท่านั้น และต้องติดตั้ง Mouse ด้วยเพราะในหลาย Function ของโปรแกรมนี้จะต้องใช้ Mouse Control เท่านั้น) และหลังจากติดตั้ง MIDI CARD แล้ว สามารถตรวจสอบการทำงานของ CARD ได้โดยต่อ MIDI CABLE เข้ากับ เครื่องดนตรีที่มีช่อง MIDI เรียกโปรแกรม TEST.EXE จากแผ่นโปรแกรม และจะต้องเห็น ตัวเลขที่รอบขวาบนของจอภาพเปลี่ยนค่าตลอดเวลา โดยที่ ตัวเลขทางขวาเปลี่ยนทุกๆ 1/192 วินาที ส่วนเลขทางด้านซ้ายมือ จะเปลี่ยนเมื่อเลขทางขวามือเปลี่ยนจาก 95 เป็น 0 และหากเครื่องดนตรีที่นำมาต่อด้วยสามารถส่งรหัส Active Sensing ได้ ก็จะเห็น คำว่า Active Sense ทางด้าน รอบซ้ายบนของจอภาพกระพริบ และเมื่อ มีการส่งรหัส MIDI จากเครื่องดนตรี ก็จะแสดงให้เห็นว่าส่งรหัสอะไรมา

โปรแกรม MIDI PROJECT จะประกอบด้วย File ต่างๆ ดังต่อไปนี้

- MIDIPROJ.EXE

  เป็น File โปรแกรมหลักที่จะเรียกใช้ MIDI CARD และเรียก File อื่นๆ ขึ้นมาใช้

- TEST.EXE

  ใช้สำหรับการตรวจสอบการทำงานของ MIDI CARD และ ตรวจสอบข้อมูลที่มาจาก เครื่องดนตรีที่ต่ออยู่ด้วย

- *.VCS

  เป็น Library ที่ใช้เก็บรหัสของเสียงของเครื่องดนตรีต่าง

- MIDI2.CFG

  เป็น Configuration File (หากมี)

- \DRIVER\EGAVGA.BGI

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- \DRIVER*.CHR

เป็นกราฟิกไดรเวอร์ของภาษา C ที่ใช้ในการเขียนโปรแกรม

- \EXAMPLE*.MPJ

เป็นไฟล์ที่ใช้ในการเก็บเพลงที่บันทึกไว้

MIDI CARD ที่นำมาต่อด้วยหากเป็น CARD รุ่นแรก (สร้างโดย อุทัย เกียรติวิกรัย) ก็จะสามารถใช้กับโปรแกรมนี้ได้เหมือนกัน แต่จะมี 1 Input / 1 Output แต่หากเป็น CARD รุ่นที่สอง ก็จะมี 1 Input / 2 Output หรือ 32 แชนแนล

หากต้องการที่จะรันโปรแกรมให้มีผลดี ควรรันโปรแกรมจาก HARD DISK และหลังจาก Install โปรแกรมแล้วต้องมีที่ว่างใน HARD DISK อย่างน้อย 1 Mb เพราะโปรแกรมจะมีการ Swap DATA ลงใน Disk ในที่ๆ Set ให้เป็น.Temporary Path

2. Installing Program

1.) สร้าง Directory MIDIPROJ ขึ้นมา (จะใช้ชื่ออื่นก็ได้) และใน Directory MIDIPROJ ให้สร้าง Sub-Directory ขึ้นอีก 3 Sub-Directory ดังนี้

- TEMP

เป็น Temporary Path ใช้สำหรับการ Swap DATA และ ส่งผ่านค่า Parameter ระหว่างโปรแกรม

- DRIVER

ใช้สำหรับเก็บ Graphics Driver

- EXAMPLE

เก็บตัวอย่างโปรแกรมเพลง

2.) Copy File ต่อไปนี้ไว้ที่ Directory MIDIPROJ

- MIDIPROJ.EXE
- TEST.EXE
- DRUM.EXE
- *.VCS
- PERCSTD.LIB

Copy File EGAVGA.BGI และ *.CHR ไปที่ MIDIPROJ\DRIVER

Copy File *.MPJ ไปไว้ที่ MIDIPROJ\EXAMPLE

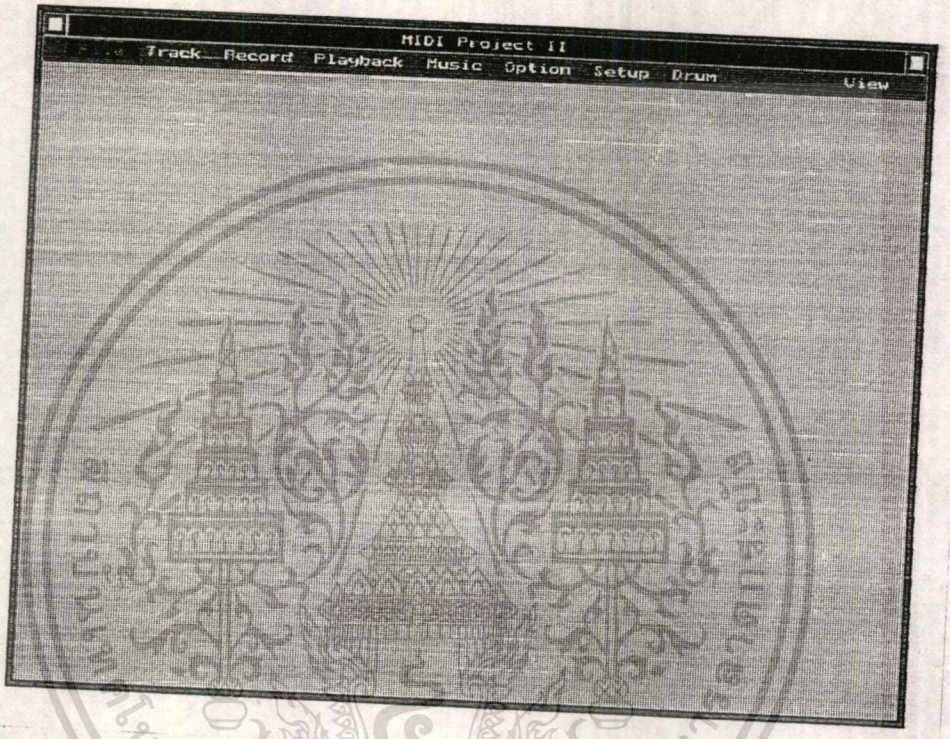
และเมื่อทำการติดตั้งโปรแกรมเรียบร้อยแล้ว , ติดตั้ง MIDICARD เรียบร้อยแล้วให้ต่อ MIDI CABLE เข้ากับ เครื่องดนตรี และทดลองรันโปรแกรมได้เลย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. Getting Start

เรียก MIDIPROJ

โปรแกรมจะมีหน้าตาดังรูปที่ 1



รูปที่ 1 Main Menu

ใน Main Menu จะประกอบไปด้วย

1). File ใน Menu File ก็จะประกอบไปด้วย

- Load โหลดโปรแกรมเพลงจาก Disk (จากไฟล์ *.MPJ)
- Save จัดเก็บโปรแกรมเพลง ลง Disk
- New Reset ข้อมูลต่างๆ ให้อยู่ในสภาพเดียวกับที่เปิดเครื่องขึ้นมา
- Comment เขียนข้อความพิเศษเป็นหมายเหตุของเพลง
- Os shell กลับสู่ DOS ชั่วคราว
- Quit จบโปรแกรม

2). Track ใน Menu Track ประกอบด้วย

- Track Volume ใช้ Set ระดับความดังของเสียงใน Track ปัจจุบัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- Transpose ใช้ในการเลื่อนระดับเสียงของแต่ละ Track
- Clear ใช้ในการ Reset Parameter ต่างๆของ Track นั้นให้กลับเป็น เหมือนตอนเปิดเครื่อง และลบ Note ที่มีอยู่ทั้งหมดทิ้งไป
- Merge Track ใช้รวมโน๊ตของ 2 Track เข้าด้วยกัน
- Detail เขียนหมายเหตุของแต่ละ Track ไว้
- Tool ใช้ในการปรับแต่งข้อมูลต่างๆ และใน Menu Tool ประกอบด้วย
  - Quantize ใช้0ในการจัดโน๊ตต่างๆให้ลงจังหวะที่ถูกต้อง
  - Kill Velocity ใช้ Set ระดับความดังของ โน๊ตทุกตัวให้ดังเท่ากันหมด (Velocity = 127)
  - Kill Whell ใช้ลบ Pitch Bend
  - Kill Pressure ใช้ลบข้อมูล Channel Pressure และ Polyphonic Key Pressure After Touch
  - Kill Program ใช้ลบข้อมูล Program Change
  - Kill Control ใช้ลบข้อมูล Control Change
  - Time Shift ใช้ในการเลื่อนเวลาในการเล่นโน๊ตออกไปให้ช้าลง หรือเลื่อนให้เร็วขึ้นได้
  - Copy ใช้ในการคัดลอกข้อมูลจาก Track หนึ่งไปยัง Track หนึ่ง
  - Assign Channel ใช้ในการกำหนด Channel ที่จะส่งข้อมูลของ แต่ละ Track
- Voice Select ใช้สำหรับการเลือกเสียงที่ต้องการสำหรับ Track นั้น
- Present Track ใช้ในการกำหนด Track ปัจจุบัน

3). Record Menu Record ใช้สำหรับการบันทึกโน๊ต ทั้งจากการเล่นจริง และการ โปรแกรม ซึ่งสามารถแก้ไขการเล่นผิดพลาดได้ด้วย ใน Menu Record ประกอบด้วย

- Step ใช้สำหรับการโปรแกรมโน๊ตและการแก้ไขโน๊ต ซึ่งจะเป็นในลักษณะคล้าย ๆ กับ Tablature ของ Keyboard
- Realtime ใช้ในการบันทึกโน๊ตจากการเล่นดนตรีจริง

4). Playback ใช้สำหรับการเล่นโน๊ตทุกๆ Track ที่เราเล่นเข้าไป รวมทั้ง สามารถที่จะควบคุมความสมดุล ของความดัง , ค่อยของเครื่องดนตรีแต่ละTrack ด้วย โดยลักษณะการควบคุมจะเหมือนกับการควบคุม Mixer และการสั่งให้โปรแกรมเล่นเพลงก็ใช้รูปแบบของ Tape Deck ในการควบคุม

5). Music ใช้ในการกำหนด Parameter ต่าง ๆ ของเพลง ใน Menu Music ประกอบด้วย

- Tempo ใช้กำหนดความเร็วของเพลง โดยมีค่าเป็น Quater Note / Min โดยมี Default เป็น 120
- Transpose ใช้ในการเลื่อนโน๊ตขึ้นหรือลง เป็น Semi Tone มีค่าได้ตั้งแต่ -36 ถึง 36

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

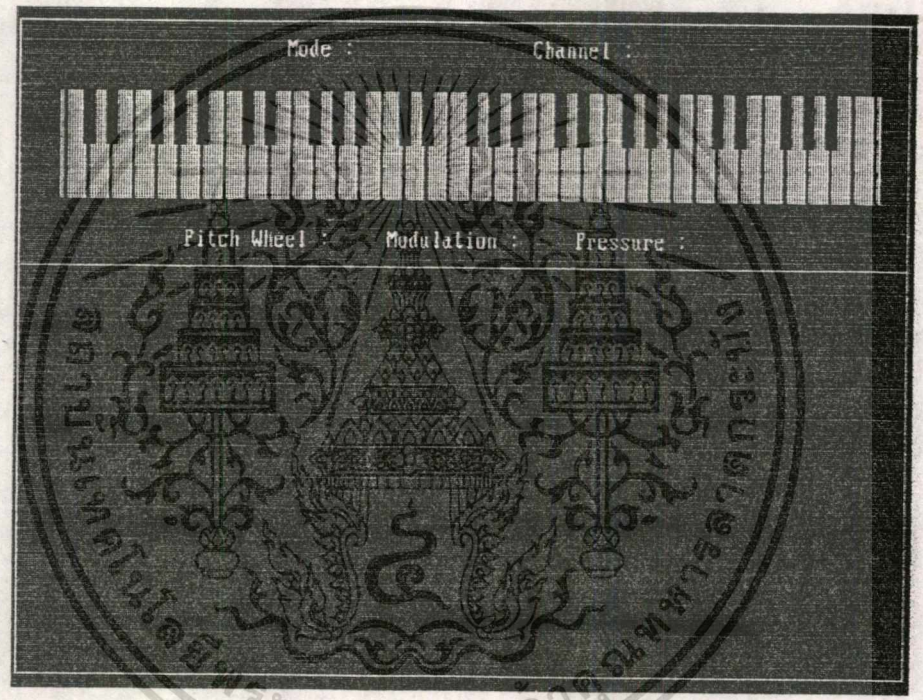
- Note / Bar กำหนดว่าจะให้ใน 1 ห้องเพลง มีความยาวเท่ากับ 3 Quater Note หรือ 4 Quater Note

6). Option ใช้ในการเซ็ตค่า Parameter ต่าง ๆ ก่อนเข้าโปรแกรม เช่น Directory ของ Swap File

7). Drum เรียก Drum Machine มาใช้งาน

8). View เป็น MIDI DATA Analyzer ดูลักษณะของข้อมูลที่เข้ามาทาง MIDI Port มีประโยชน์ในการตรวจสอบการทำงานของโปรแกรมและอุปกรณ์ พ่วงต่อ

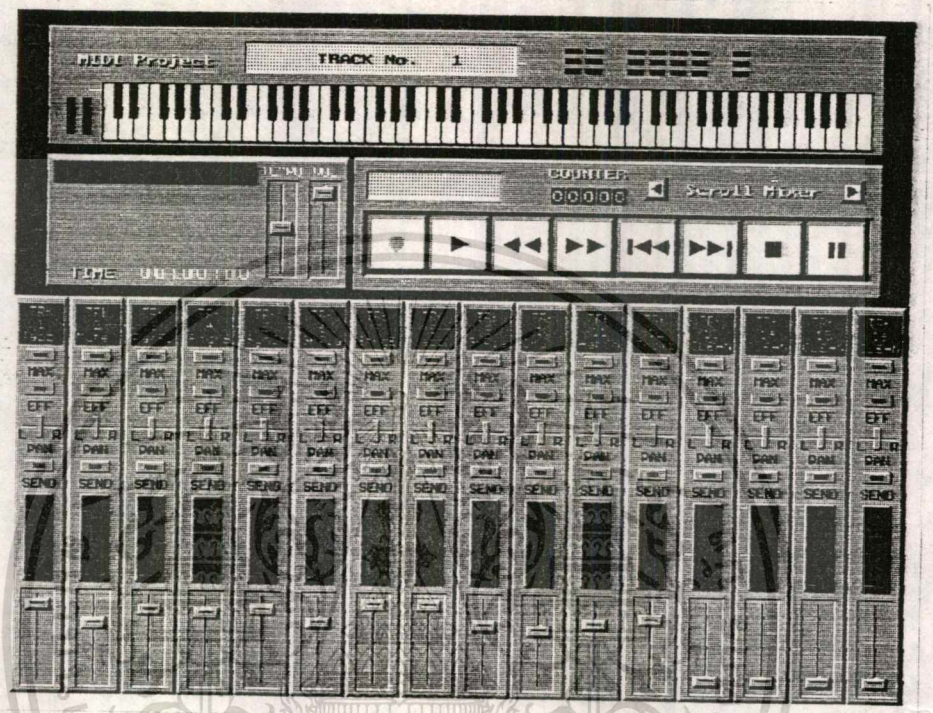
ลองเลื่อน Menu ไปที่ View กด Enter กด y , y หน้าจอจะขึ้นดังรูปที่ 2



รูปที่ 2 MIDI DATA Analyzer

ถ้าเครื่องดนตรีที่มาต่อด้วยเป็นเครื่องที่ส่ง Active Sense ด้วย จะเห็น คำว่าActive Sense ที่มุมซ้ายของจอ กระพริบและเห็นว่ามีระหัส FE เข้ามาเรื่อย ๆทดลองกด Key ที่ Keyboard ของ เครื่องดนตรีนั้นจะเห็นว่า มีรหัส MIDI เข้ามา เมื่อลองปล่อย Key ก็จะเห็นรหัส MIDI เข้ามาเช่นกัน และที่มุมขวาของจอจะเห็นว่า ค่า Timer มีค่าเพิ่มขึ้นเรื่อย ๆ ถ้าเป็นดังนี้ แสดงว่า โปรแกรมและ MIDI CARD และเครื่องดนตรีทำงานได้ปกติ ทดลอง Load File ที่อยู่ใน Example ขึ้นมา แล้วเลือก Menu "Play" หน้าจอจะขึ้นมาตามรูปที่ 3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3 Playback (Mixer And Deck Control)

หลังจากนั้นใช้ Mouse ไปกดปุ่มที่ control Deck จะได้ผลเหมือนกับที่เคยเล่นTape หรือ Compact Disc เสียงที่ Set ไว้อาจไม่ถูกต้องเพราะหมายเลขของเสียงในเครื่องดนตรีแต่ละเครื่องอาจไม่เหมือนกัน และ Channel ของกลอง ก็อาจไม่ตรงกัน (Reference ตาม KAWAI K11 - GM Bank - Synthesizer ที่เขียนติดไว้ว่า GS หรือ GM จะมีชุดเสียงที่ตั้งไว้ตามมาตรฐาน GS ,Channel กลอง เป็น Channel 10) ผู้ใช้อาจต้องสร้าง Library ของเครื่องดนตรีขึ้นนั้นขึ้นมา (ซึ่งจะกล่าวในภายหลัง) นอกจากเสียงอาจไม่ตรงกันแล้วเสียงของ กลองก็อาจไม่ตรงกันด้วยก็ได้ (เช่นในโปรแกรมสั่งให้เล่น Bass Drum แต่เครื่องดนตรีนั้นกลับเล่น Snare Drum ออกมา เป็นต้น)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

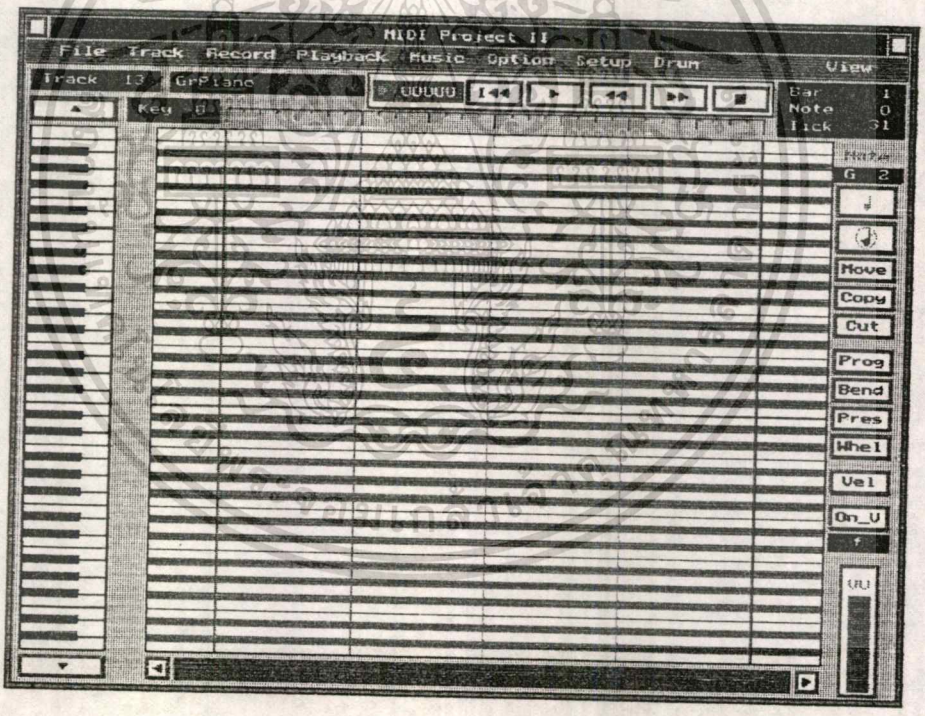
4. Song Writing

การที่จะเก็บโน๊ตเข้าไปในโปรแกรมทำได้ครั้งละ 1 Track เท่านั้น แต่เราก็สามารถ เก็บเข้าไปทีละ Track แล้วสั่งให้เล่นพร้อม ๆ กันได้ และการที่จะบันทึกโน้ตทำได้2 วิธี คือการเล่นเข้าไปจริง ๆ โดยการเลือก Menu Record แล้วเลือก Menu ย่อย Realtime Record หรือ การโปรแกรมเป็นโน๊ตเข้าไปโดยโน๊ตที่โปรแกรมเข้าไปนั้นจะเป็นในลักษณะของ Tablature Keyboard เลือกจาก Keyboard ว่าจะกด Key ใด แล้วใช้ Mouse ลากโน๊ตไปในตำแหน่งที่ตรงกับจังหวะของโน๊ตนั้น

1). Step Record

ก่อนอื่นให้เลือก Menu Track,Present Track เพื่อเลือก Track ที่ต้องการจะทำการบันทึกเสียก่อน

เมื่อเลือก Menu นี้หน้าจอจะเป็นดังรูปที่ 4



รูปที่ 4 Step Record Screen

ส่วนประกอบที่สำคัญมีดังนี้

1. Keyboard Index เราจะลากโน๊ตโดยเทียบกับKeyboardนี้ ในที่นี้แสดงเพียง 5 Octave คือสามารถแสดงได้เพียงโน๊ตหมายเลข 36 ถึง 96 (รหัส MIDI ซึ่งจริง ๆ แล้วมีตั้งแต่ 0-127) แต่จะมี Icon สำหรับเปลี่ยน

เอกสารนี้มีลิขสิทธิ์ สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Octave ได้อยู่ด้านบนและล่างของ Keyboard นี้ โดยมี Display บอกไว้ว่าตอนนี้ Shift ไปกี่ Octave ในส่วนนี้เราสามารถใช้ Mouse มา Click ได้โดยจะมีการเล่นโน๊ตตามที่ Click ไป (ใช้ Check ว่า โน๊ตที่จะใส่ตรงกับที่เราต้องการหรือเปล่า ทั้งยังแสดงให้เห็นว่าเล่นโน๊ตตัวไหนอยู่ในขณะที่สั่งให้โปรแกรมเล่นเพลงอีกด้วย)

2. Tablature เป็นที่สำหรับใส่โน๊ตหรือ Control ต่าง ๆ โดยแสดงครั้งละ 1 ห้อง จะมีเส้นทึบเป็นตำแหน่งของ จุดเริ่มต้นและจุดสุดท้าย และจะแสดงโน๊ตของห้องก่อนหน้าและห้องถัดไปอีก ครึ่ง Quater Note มีเส้นบางตามยาวแบ่งจังหวะ แต่ละเส้นห่างกัน 1 Quater Note เมื่อ Cursor ของ Mouse อยู่ในส่วนนี้ cursor ของ Mouse ก็จะกลายเป็นเส้นตรงตามแนวตั้งและตามแนวขวางตัดกันที่ตำแหน่ง Cursor และเส้นตามแนวนอนจะชี้ไปถึง Keyboard และเส้นตามแนวตั้งจะชี้ไปยัง Scale Note ด้านบน เพื่อบอกระดับเสียงและตำแหน่งที่จะให้เล่นโน๊ตนั้น โดยจะมี Stats Bar เป็นตัวบอกอีกที ด้านล่างของ Tablature จะมี Scroll Bar สำหรับการเลื่อน Tablature ไปทีละ 1 ห้อง

3. Mode Selector ใช้สำหรับเลือกโหมดที่จะทำงานใน Tablature ว่าจะเป็นการ ใส่โน๊ต , ลบโน๊ต , Copy , Move , Cut , แทรก Control ต่าง ๆ ลงในระหว่างการเล่นโน๊ตนั้น รวมถึงการเลือกระดับความดังของ โน๊ตที่จะใส่ลงไปด้วย รวมถึงการเปลี่ยนระดับความดังของ โน๊ตได้ด้วย

4. Tape Deck Controller สั่งให้โปรแกรมเล่นเพลงได้เหมือนกับเล่น Tape ธรรมดา โดยขณะเล่นจะมี VU Meter แสดงระดับความดังของโน๊ตใน Track ที่กำลัง Edit อยู่ด้วย

การบันทึกโน๊ตทำได้ 2 วิธีดังนี้

การเขียนโน๊ต

1. เลือก Mode ใส่โน๊ต (หากอยู่ใน Mode นี้แล้วก็ไม่ต้องเลือกอีก)
2. เลือกความดังของโน๊ตที่จะใส่ (หากความดังอยู่ในระดับที่ต้องการแล้วก็ไม่ต้องเลือกอีก)
3. เลื่อน Mouse ไปยังตำแหน่งของ โน๊ตที่ต้องการ
4. Click ปุ่มซ้ายมือของ Mouse ค้างไว้
5. ลาก Mouse ไปทางขวา จนได้ความยาวของโน๊ตตามต้องการ (สามารถลากข้าม ห้อง ได้ และลากกลับคืนมาทางซ้ายได้)
6. ปล่อยปุ่ม Mouse จะได้โน๊ตตามที่ต้องการ ลักษณะโน๊ตเป็นดังรูปที่ 5

การลบโน๊ตที่ไม่ต้องการออก

1. เลือก Mode ลบโน๊ต (หากอยู่ใน Mode นี้แล้วก็ไม่ต้องเลือกอีก)
2. เลื่อน Mouse ให้อยู่ตรงกับโน๊ตที่ต้องการที่จะลบ
3. Click ปุ่มซ้ายมือของ Mouse แล้วปล่อยออก
4. โน๊ตที่ไม่ต้องการก็จะหายไป

การ Copy โน๊ตจากห้องหนึ่งไปยังอีกห้องหนึ่ง

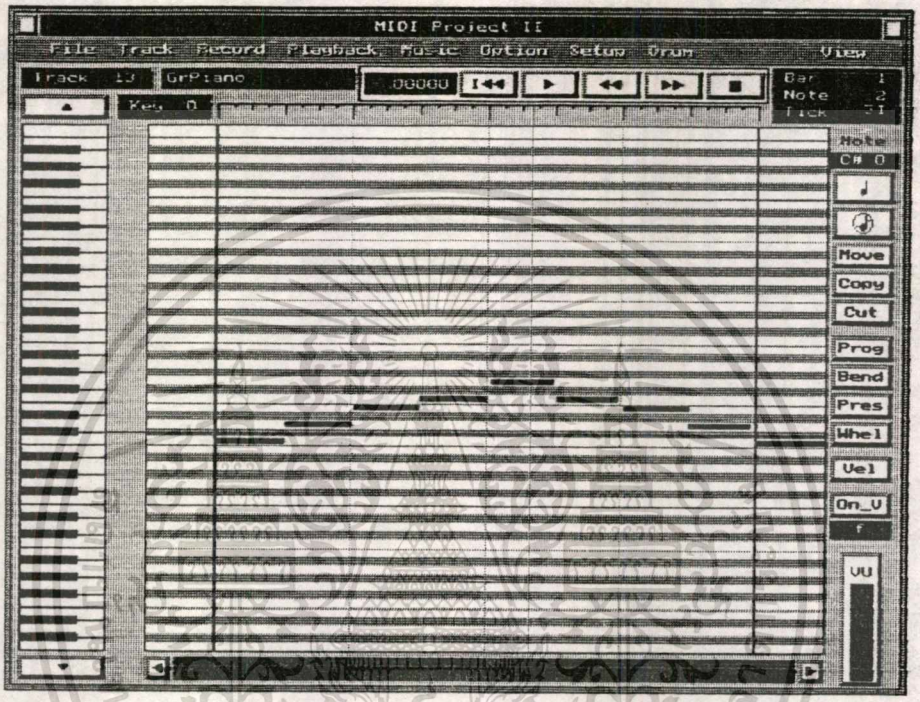
1. เลือก Mode Copy (หากอยู่ใน Mode นี้แล้วก็ไม่ต้องเลือกอีก)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. เลื่อน Mouse ให้อยู่ใน Tablature

3. Click ปุ่มทางซ้ายมือของ Mouse แล้วปล่อยออก

4. หน้าจอจะขึ้นมาดังรูปที่ 6



รูปที่ 5 ลักษณะของโน๊ตที่ได้จากการใช้ Step Record

5. โปรแกรมจะถามห้องที่จะเริ่มต้น Copy แล้วถามห้องสุดท้ายที่ต้องการ Copy ล้วจะถามว่า Block มานี้จะ Copy ไปที่ห้องใดเป็นห้องแรก หาก ไม่ต้องการที่จะ Copy ให้เลือก Cancle หากตกลงให้ O.K.

การเปลี่ยนเสียงของเครื่องดนตรีก่อนเล่นโน๊ตตัวต่อไป

1. เลือก Mode Prog (หากอยู่ใน Mode นี้แล้วก็ไม่ต้องเลือกอีก)

2. เลื่อน Mouse ให้ Cursor ของ Mouse อยู่หลังจากจุดเริ่มต้นของโน๊ตตัว สุดท้ายก่อนการเปลี่ยนเสียง

3. Click ปุ่มทางซ้ายมือของ Mouse แล้วปล่อยออก

4. หน้าจอจะขึ้นมาดังรูปที่ 7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6 หน้าจอของ Mode Copy



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 7 การเปลี่ยนเสียงขณะเล่นโน๊ตอยู่

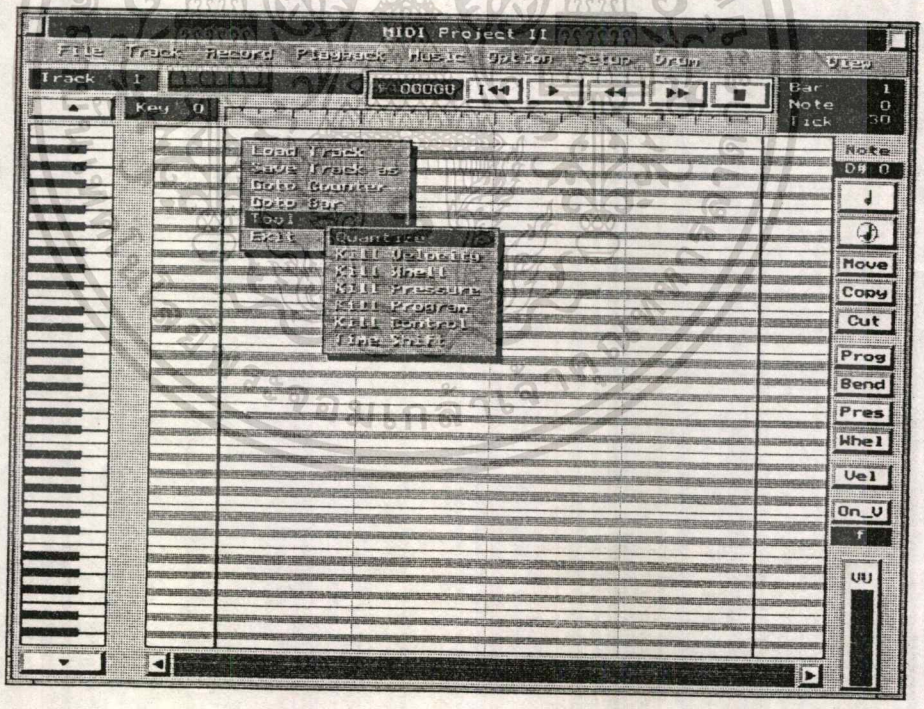
5. เลือกเสียงตามที่ต้องการ (ดู Voice Select)
6. ที่ Scroll Bar จะขึ้นมาบอกว่าที่ตำแหน่งนั้นมีการเปลี่ยนเสียง

การเปลี่ยนระดับความดังของโน๊ต

1. เลือก Mode Vel (หากอยู่ใน Mode นี้แล้วก็ไม่ต้องเลือกอีก)
2. เลือกความดังของโน๊ตที่ต้องการจะเปลี่ยนความดัง (หากความดังอยู่ในระดับที่ต้องการแล้วก็ไม่ต้องเลือกอีก)
3. เลื่อน Mouse ให้อยู่ตรงกับโน๊ตที่ต้องการที่จะเปลี่ยนระดับความดังของเสียง
4. Click ปุ่มทางซ้ายมือของ Mouse แล้วปล่อยออก

การออกจาก Step Record

เลื่อน Mouse ไปอยู่ใน Tablature แล้ว Click ปุ่มทางขวาของ Mouse แล้วปล่อย จะขึ้น Menu ดังรูปที่ 8



รูปที่ 8 Menu ใน Step Record

เลือก Quit ก็จะออกจาก Step Record และโน๊ตที่โปรแกรมไว้จะถูกเก็บลงFile Track.xxx โดยที่ xxx เป็นหมายเลข Track - 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. Realtime Record

ก่อนอื่นให้เลือก Menu Track,Present Track เพื่อเลือก Track ที่ต้องการจะทำการบันทึกเสียก่อน

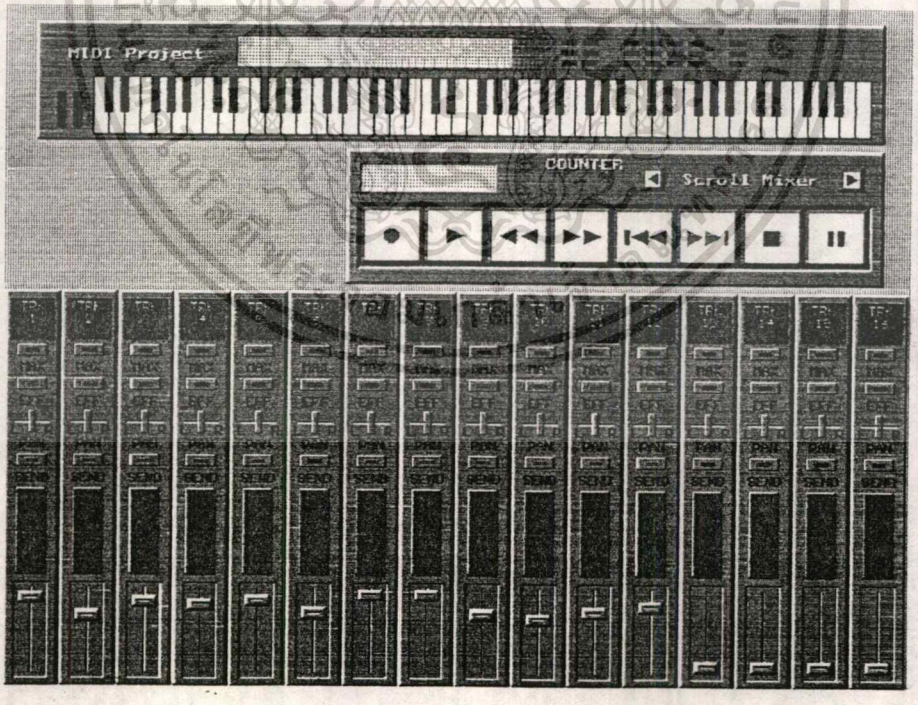
ก่อนการเลือก Realtime Record ควรที่จะ Set เครื่องดนตรีที่นำมาต่อด้วย ให้เป็น Local Control off เสียก่อนเพราะโปรแกรมจะ รับ DATA จากเครื่อง ดนตรีนั้นมา Process แล้ว Return ข้อมูลนั้นคืนสู่เครื่องดนตรี พร้อมกับ DATA ของ Track อื่นที่มีอยู่ออกไปด้วยด้วย

เมื่อเลือก Menu Realtime Record หน้าจอจะขึ้นดังรูปที่ 9

เมื่อเริ่ม Record โปรแกรมจะมี Count in ให้ เพื่อที่จะให้เริ่มต้นได้ถูกต้อง หากมีการเล่นโน๊ตออกมา โน๊ตทั้งหมดจะถูก Shift ไปที่เวลา 0 และจะมีการส่งข้อมูลเดิมของ Track ที่กำลังทำการบันทึกนั้น ออกไปด้วย และขณะที่บันทึกอยู่ก็จะมีเสียง เคาะจังหวะให้

และเมื่อบันทึกเสร็จแล้วให้กด Keyboard (ของ Computer) Key ใดก็ได้การบันทึกจะสิ้นสุดลง และข้อมูลที่ได้จะถูกนำไปรวมกับข้อมูลเดิมที่มีอยู่ แต่หากไม่ต้องการให้ข้อมูลนั้นทับข้อมูลเดิม ให้เลือก Menu Track , Clear ก่อนข้อมูลเดิมจะหายไป

บางครั้งเพื่อความแน่นอนในการบันทึกแบบ Realtime Record นี้ เราอาจSet ให้โปรแกรมเล่นในจังหวะที่ ช้า ๆ ก่อนได้ แล้วจึงทำการเปลี่ยนจังหวะให้เร็วเท่าปกติก็ได้



รูปที่ 9 Realtime Record

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการบันทึกทั้งสองแบบนี้ สามารถใช้ร่วมกันได้โดยอาจใช้ Realtime Recordก่อน แล้วใช้ Step Record เขียนโน๊ตในส่วนที่ไม่สามารถเล่นได้ ก็ได้ หรือใช้

Step Record ในการลบโน๊ตที่เล่นผิดแล้วใส่โน๊ตที่ถูกต้องเข้าไปได้ หรือลบโน๊ตที่ไม่ต้องการออกได้จะทำให้การบันทึกสมบูรณ์ยิ่งขึ้น

5. Playback

หลังจากที่บันทึกไปแล้ว เราจะนำเพลงนั้นมาเล่นกัน โดยการเลือก Menu Playback หน้าจอจะขึ้นมาดังรูปที่ 10

ส่วนประกอบที่สำคัญในหน้าจอนี้ ประกอบด้วย

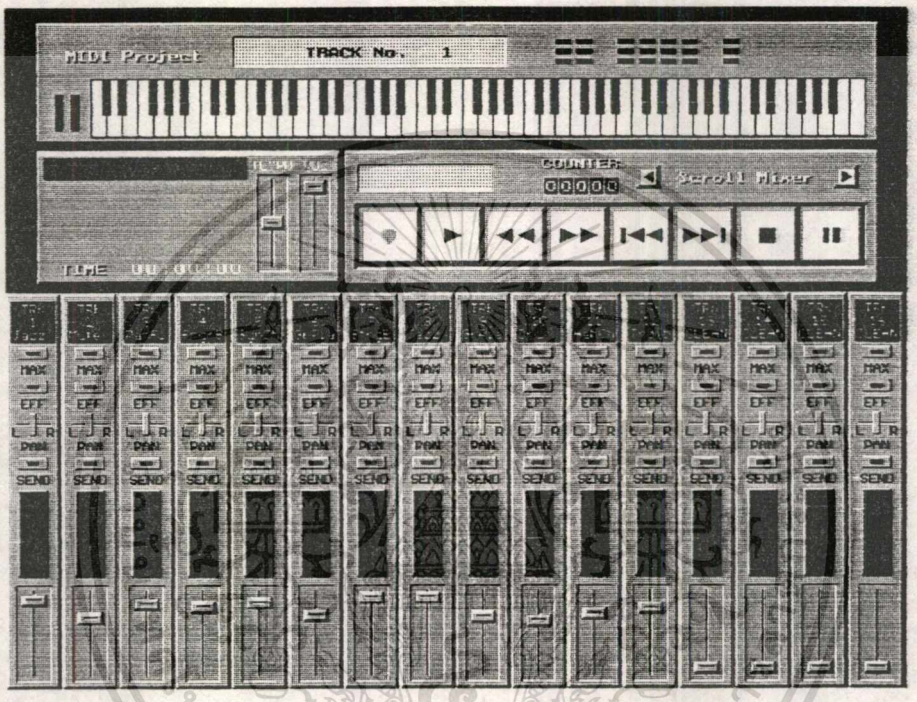
1). Keyboard Monitor จะแสดงโน๊ตของ Track ที่เลือกมาให้แสดง ในขณะที่เล่นอยู่ ซึ่งสามารถเลือกได้โดยการ เลื่อน Mouse Cursor ไปยังส่วน Header ของ Mixer ที่ต้องการจะให้แสดงแล้ว Click ปุ่มทางซ้ายมือของ Mouse แล้วปล่อย Monitor ก็จะแสดงโน๊ตที่กำลังเล่นอยู่ของ Track นั้น

2). Main Control ใช้เปลี่ยนความดังของทุก Track และใช้เปลี่ยน ความเร็วในการ Playback ด้วย

3). Tape Deck ควบคุมการเล่นของเพลงได้เหมือนกับการใช้ Tape ทั่วไป

4). Mixer ส่วนนี้มีความสำคัญมาก เพราะเป็นส่วนที่ใช้ควบคุม Balance ของเพลงทั้งหมด เราจะใช้ในการกำหนดความดังของแต่ละ Track ควบคุมให้ส่งหรือไม่ส่งข้อมูลของ Track นั้น กำหนดให้เสียงของ Track นั้น Pan ออกไปด้านซ้ายหรือขวาได้ และ MAX คือการ Force ให้เล่นโน๊ตของ Track นั้นด้วยความดังสูงที่สุด โน๊ตทุกตัวจะเล่นดังเท่ากันหมด และจะสามารถควบ คุมความดังได้โดย การควบคุมที่ Volume ของ Mixer ได้อย่างเดียว และใน Mixer นี้จะมี VU Meter คอยบอกระดับ ความดังของโน๊ตใน แต่ละ Track อีกด้วย ในที่นี้แสดงไว้เพียง 16 Track แต่เราก็สามารถ เลื่อน Mixer ได้จาก Scroll Mixer ใน Tape Deck Controller

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10 Playback

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. Drum Machine

Drum Machine เป็นโปรแกรมช่วยให้การโปรแกรมกลองง่ายขึ้นมาก ซึ่งจากพื้นฐานของการตีกลองจริง ๆ แล้ว เป็นรูปแบบที่ซ้ำ ๆ กันเสียเป็นส่วนมาก ดังนั้นเราสามารถที่จะโปรแกรมเป็นจังหวะสำเร็จเตรียมไว้แล้วนำมาประกอบกันได้

ใน Drum Machine จะเรียกใช้โดย MIDIPROJ ผ่านทาง Menu Drum แล้วDrum Machine จะมี Menu ของตัวเอง ในที่นี้จะกล่าวเฉพาะส่วนที่สำคัญเท่านั้น เนื่องจากยังอยู่ในช่วงของการพัฒนาโปรแกรม



รูปที่ 11 Drum Machine

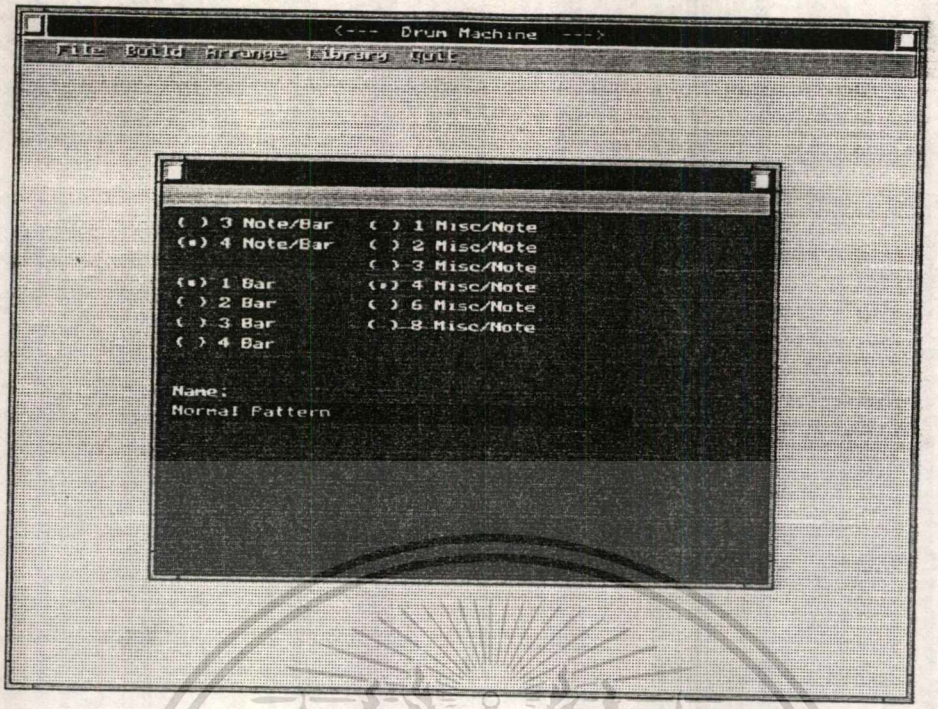
1). Menu Pattern / Setup

ในแต่ละ Pattern จะต้องมี Parameter บอกว่า Pattern ที่เราจะสร้างขึ้นมา เป็นอย่างไร ใน Pattern นั้นมีกี่ห้อง ห้องหนึ่งประกอบไปด้วยกี่จังหวะ ในแต่ละจังหวะจะแบ่งแยกย่อยออกเป็นกี่ส่วน

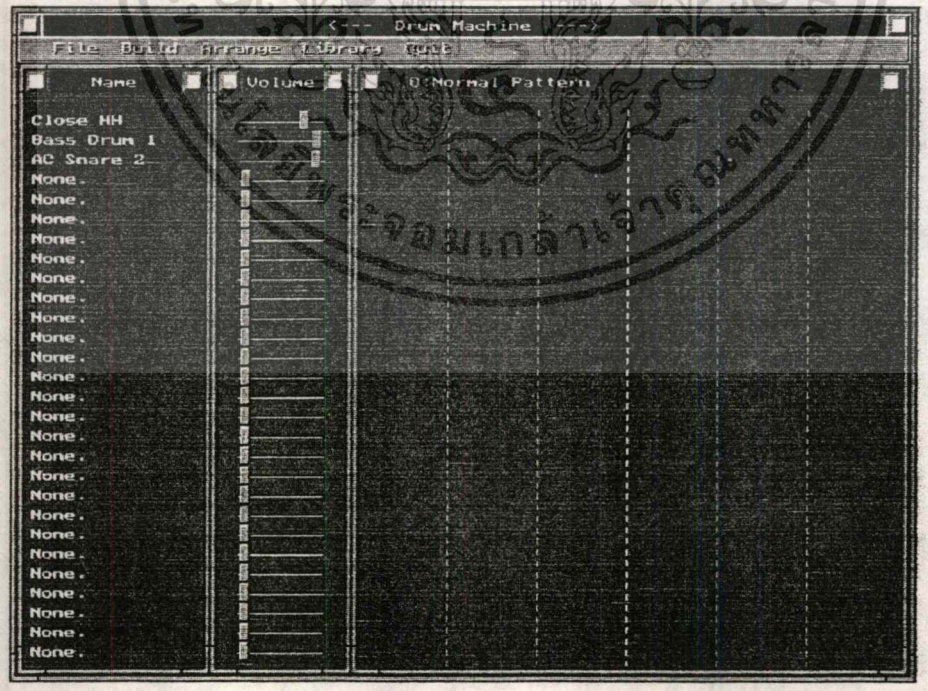
2). Menu Pattern / Build

เป็นการสร้าง Pattern ขึ้นมา ลักษณะของ Screen ที่แสดงออกมาจะเป็น Matrix ให้เลือกว่าที่ ตรงนี้จะให้ตีกลองชิ้นนี้หรือเปล่า ซึ่ง ในรูปที่ 13 แสดงไว้ จะเห็นได้ว่าประกอบไปด้วย 3 Window คือ ชื่อชิ้นกลองที่จะนำมาใช้ , Volume ของกลองชิ้นนั้น และ Edit Screen ที่เป็น Matrix ซึ่งการ สร้าง Patternทำได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

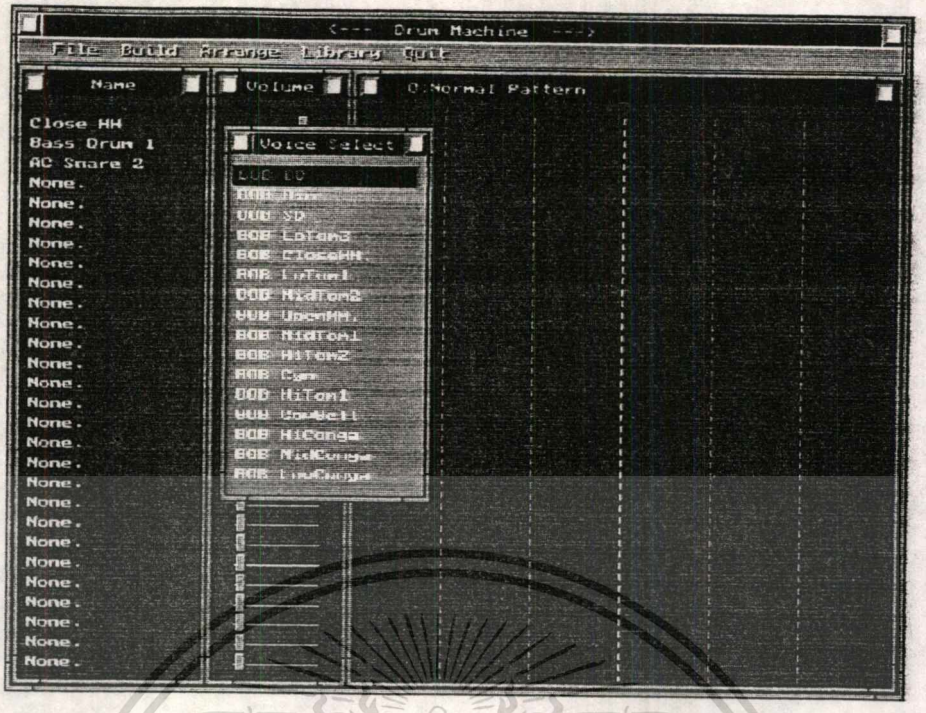


รูปที่ 12 การ Setup Pattern



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 13 หน้าจอของส่วนสร้าง Pattern



รูปที่ 14 การเลือก Percussion ที่จะนำมาประกอบเป็น Pattern

1). เลือกชิ้นกลองโดยการเลื่อน Mouse ไปยัง Window แรกเลือกตำแหน่งว่าจะให้เป็นกลองชิ้นที่เท่าไร Click Mouse โปรแกรมจะมี Library ของกลอง ขึ้นมาให้เลือกดังรูปที่ 14

2). กำหนดความดังของกลองได้โดยใช้ Mouse ไปเลื่อน Volume ที่ Window Volume

3). เลื่อน Mouse ไปยังตำแหน่งที่ต้องการ Click Mouse หรือ ใช้ Arrow Key เลื่อนให้ Highlite ไปอยู่ตามตำแหน่งที่ต้องการแล้ว กด Space Bar ซึ่งจะ Toggle ไปมาระหว่างมีค่า และ ไม่มีค่า

4). สามารถทดสอบ Pattern ที่สร้างขึ้นมาได้โดยการกด Enter และโปร แกรมได้โดยการกด Key ใด ๆ

5). กด Esc เพื่อออกจาก Pattern / Build

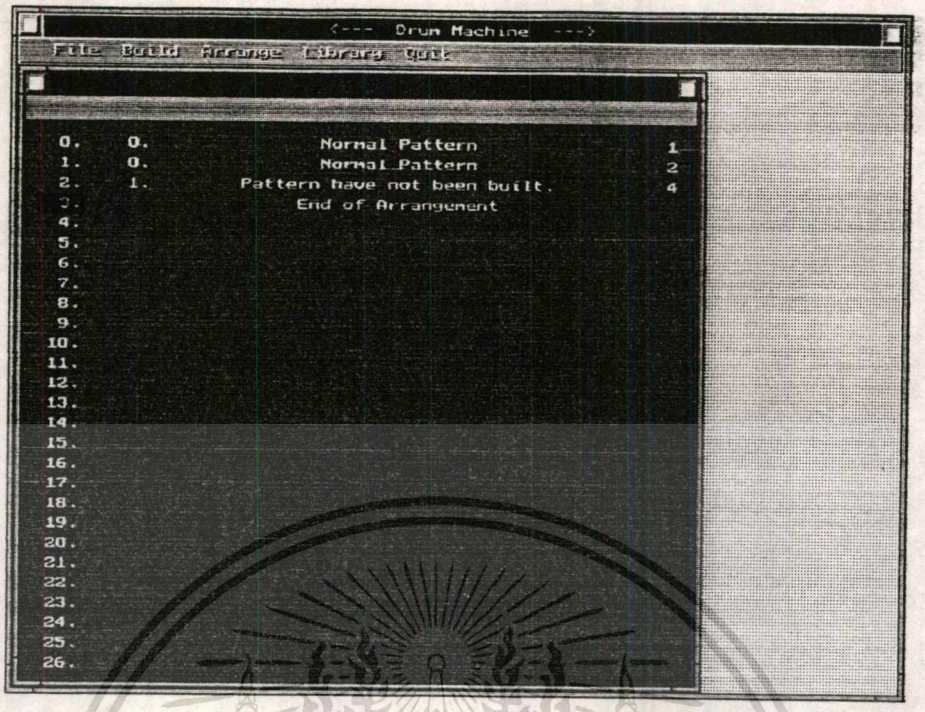
3). Menu Arrange / Edit

เมื่อเลือก Menu Arrage / Edit หน้าจอจะขึ้นดังรูปที่ 15

เราจะเลือก Pattern ที่ต้องการเล่นเรียงลำดับกันไปได้ และสามารถกำหนดได้ว่า จะให้เล่น Pattern นี้กี่ครั้ง การแทรกช่องว่างเพื่อเพิ่มเติมจังหวะทำได้โดยการ เลื่อน Highlite ให้อยู่ที่ลำดับที่ ต้องการแทรก กด Insert

หากโปรแกรมมาเกินก็สามารถลบได้เช่นกันโดยการเลื่อน Highlite ให้ตรงกับลำดับที่ต้องการลบ กด Del และเมื่อต้องการออกจาก Drum Machine ก็จะทำได้โดยการกด Alt-X หรือเลือก Menu Quit / Quit Program โปรแกรมจะ Generate Code MIDI ได้ด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 15 Arrange Pattern

หมายเหตุ

Usage : DRUM [Channel [Track [Tempo]]]

- Channel = Channel Build to Output File
- Track = Extension Of Output File (Track.xxx) xxx = Track
- Tempo = Frequency of Rythm (Quater Note / Min)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

1.ขอขอบคุณ รศ.ดร มนัส สังวรศิลป์ ที่สนับสนุนการทำ MIDI PROJECT มาด้วยดีตลอด

2.ขอขอบคุณ อ.ประภากร สุวรรณะ ในการเอื้อเฟื้อสถานที่ในการทำงาน

3 ขอขอบคุณ อ.ชนิษฐา แซ่ตั้ง และรุ่นพี่ห้อง อ.พลผดุง ผดุงกุล ที่ช่วยให้คำปรึกษา

4.ขอขอบคุณ นาย ธำรงค์ศักดิ์ โกสิยตระกูล ที่ให้กำลังใจมาตลอด

5.ขอขอบคุณเพื่อนร่วมห้อง PROJECT ที่รับฟังดนตรีไร้ขีดจำกัด

6.ขอขอบคุณ นาย ไพโรจน์ วิวัฒนวานิช ที่แนะนำ MICROSOFT WORD และ VISIO

7 ขอขอบคุณ นาย ศุภฤกษ์ มานิตพรสุทธิ์ ที่เอื้อเฟื้อ PRINTER โดยมิคิดมูลค่า

8.ขอขอบคุณเพื่อนๆ ทุกคนที่ช่วยเพิ่มประสบการณ์ในการทำงาน

9 ขอขอบคุณน้องๆ MUSIC ที่ช่วยเป็นกำลังงานและกำลังใจอย่างสูง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนังสืออ้างอิง

1.รายงาน THESIS ของกลุ่ม MIDI PROJECT ปีการศึกษา 2533 ของนาย อุทัย เกียรติวิกรัย และ นาย ประสิทธิ์ วรฉัตราวณิช

2.JOHN UFFENBACK , " MICROCOMPUTERS AND MICROPROCESSSORS " , PRINTICE-HALL,1991

3.หนังสือเรื่อง การอินเทอร์เฟส IBM PC ของ PHISICS CENTER

4 MIDI -MUSICAL INSTRUMENT DIGITAL INTERFACE- ,RICHARD EVERS,EDITOR

5 คู่มือการใช้งานของ K-11 SYNTHESIZER

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

APPENDIX

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MIDI – Musical Instrument Digital Interface

Richard Evers, Editor



The idea started slow, but rapidly picked up speed.

Throughout the ages, man has always strived for new and exiting ways in which to amuse himself. Man is the master amuser. One way in which amusement was found, and still is today, is with music. In the beginning, there is a hint that the first musical instruments were rocks, producing a somewhat pleasing sound when smashed together in whatever unison they could muster. Soon, wood was found to possess the enthralling capacity to produce various sounds when struck. As time passed, man developed the art of cutting and forming pieces of wood to produce other truly unique sounds. From this simple beginning, we now have every instrument from the bongo drum to the grand piano. Over thousands of years of patient evolution, man has developed the most beautiful art of all – music.

This unique beginning is a primer to enable you to gain easy entrance into todays world of digital music synthesis, a world far removed from what music has been in past. Although the music synthesizer is capable of cloning the sounds of all man made instruments, it also has the ability to allow or disallow the limitations imposed on the instrument by mans capabilities. With a statement sure to be written off as sheer personal judgement, I will state that the finest of todays music synthesizers are capable of mimicking the sounds of man made instruments to such an extent that the synthetic counterpart will be preferred. But, as with all new concepts, many people will strive to retain the older, more familiar yet less refined methods of past.

Today, in what is known as the age of the microprocessor, the true " State Of The Art " (what does that mean?) phase of our existence, analog systems still persist in great numbers. Although the digital movement has taken all forms of music reproduction by storm, ie. the compact disk, digital recorded record albums and, of course, music sythesis, analog is still with us. Time will be the sole test of our loyalty to a friend long past its prime. Achieving results that were once thought impossible using analog logic now appear common place due the advent of todays digital high technology. But, justly enough, within a few years travel, our current high tech will also be considered obsolete. The sad fate of time. Obsolescence.

To get on track, a few years ago, when digital synthesis was beginning to reach its apex, a very obvious road block materialized. There were no standards to meet, no industry standard that would allow various components to be easily connected together for use. The manufacturers were producing terrific products, but the buyer was restricted to one name brand for add ons. A restricting situation.

To solve this problem, a few key people in the music sythesis industry started thinking, talking, and attracting attention along this line. The idea started slow, but rapidly picked up speed. MIDI, The Musical Instrument Digital Interface, was the solution presented to combat this problem.

The MIDI system is a series of guidelines that manufacturers should meet in order to ensure the compatibility of their products with others. As with all set guidelines, MIDI is not perfect and does actually impose limitations at times. Without coming down too hard at first, let me explain the guidelines.

The system is based on the almost current 8 bit technology, mainly because the designers were trying to produce a system that would easily fit into the budget of most starving artists. They chose a universal cable connection of the 5 pin DIN plug, using only 3 of the 5 pins. Pin #2 is ground, with pin #'s 4 and 5 being used for a current loop.

" The interface operates at 31.25 kilobaud (±1%), asynchronous, with a start bit, 8 data bits (D0 to D7), and a stop bit. This makes a total of 10 bits for a period of 320 microseconds per serial byte. "

The above paragraph was lifted directly from the MIDI 1.0 Specifications, Document No. MIDI-1.0, Dated August 5, 1983.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

With each MIDI equipped unit, you will notice either two or three connectors for MIDI use. There is the MIDI In, MIDI Out, and possibly, MIDI Through (American spelling Thru). MIDI In is the connector for data coming into the unit from other MIDI equipped units on line. This can include drum and rhythm machines, extra keyboards, electric guitar and drum interfaces, controllers and sequencers, or extra synthesizers.

A total of 16 channels, numbered 1 through 16, are allowed under MIDI but more than 16 instruments can be on line at any one time. By assigning duplicate channel numbers to units, or having some units respond to all messages (see Omni Mode), vast numbers of units can be serviced. One main limitation to this provision is the extended line length required, which produces added line loss and a greater time delay with transmitted and received data. By reading more than a few articles on this subject, one fact becomes apparent. When using more than a couple of MIDI units on line, noticeable delays appear in what is deemed simultaneous sound reproduction. But, I have recently read an outstanding interview with a person well qualified in the field. He stated that the time delay between units was not completely the fault of the MIDI implementation. In his opinion, the rate at which the units process the information provided leads to the noticeable time lag. Without greater experience in the subject, it is hard to take a side.

MIDI Out is the data being transmitted from the MIDI equipped unit. MIDI Through is a copy of the information currently being passed into the unit through MIDI In. This feature, if provided, gives the user the ability to slave units together. According to MIDI specs, the cable length between units cannot exceed 50 feet, using shielded, twisted cable with Pin #2 connected to the shielding at both ends.

The transmitted data is quite easy to identify. All transmitted information is 8 bits in length, with the Status bytes having the high bit set and data bytes having the high bit clear. As we will soon discuss, the Status byte has been provided to control the system. Within the Status class will be found all commands necessary to determine what the system will do, when it will do it, and how it should be done. The Data bytes are really the workers of the system. Not only do they carry information such as the note value to be played, but they also labour under the rule of the Status bytes to ensure that all commands are understood. For the purpose of this article, the eight bit byte is arranged from bits 7 to 0, with bit 7 acting as the high bit.

The charts below have been prepared to show all the possible byte values encountered while working with the MIDI system. Complete explanations will follow.

Table Of Notes Corresponding To Data Values Expressed In Hex

	Notes											
Octave	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
–	$00	$01	$02	$03	$04	$05	$06	$07	$08	$09	$0A	$0B
0	$0C	$0D	$0E	$0F	$10	$11	$12	$13	$14	$15	$16	$17
1	$18	$19	$1A	$1B	$1C	$1D	$1E	$1F	$20	$21	$22	$23
2	$24	$25	$26	$27	$28	$29	$2A	$2B	$2C	$2D	$2E	$2F
3	$30	$31	$32	$33	$34	$35	$36	$37	$38	$39	$3A	$3B
4	$3C	$3D	$3E	$3F	$40	$41	$42	$43	$44	$45	$46	$47
5	$48	$49	$4A	$4B	$4C	$4D	$4E	$4F	$50	$51	$52	$53
6	$54	$55	$56	$57	$58	$59	$5A	$5B	$5C	$5D	$5E	$5F
7	$60	$61	$62	$63	$64	$65	$66	$67	$68	$69	$6A	$6B
8	$6C	$6D	$6E	$6F	$70	$71	$72	$73	$74	$75	$76	$77
9	$78	$79	$7A	$7B	$7C	$7D	$7E	$7F				

Status Bytes With Messages

Bit Pattern	Decimal Value	Hex Value	Data Bytes	Status Byte Message
1000 xxxx	128 to 143	$80 to $8F	2	Note Off
1001 xxxx	144 to 159	$90 to $9F	2	Note On
1010 xxxx	160 to 175	$A0 to $AF	2	Polyphonic Key Pressure After Touch
1011 xxxx	176 to 191	$B0 to $BF	2	Control Change
1011 xxxx	176 to 191	$B0 to $BF	2	Select Channel Mode
1100 xxxx	192 to 207	$C0 to $CF	1	Program Change
1101 xxxx	208 to 223	$D0 to $DF	1	Channel Pressure After Touch
1110 xxxx	224 to 239	$E0 to $EF	2	Pitch Wheel Change
1111 0000	240	$F0	variable	System Exclusive
1111 0001	241	$F1	–	Unimplemented
1111 0010	242	$F2	2	Song Position Pointer
1111 0011	243	$F3	1	Song Select
1111 0100	244	$F4	–	Unimplemented
1111 0101	245	$F5	–	Unimplemented
1111 0110	246	$F6	0	Tune Request
1111 0111	247	$F7	0	End Of Exclusive
1111 1000	248	$F8	0	Timing Clock
1111 1001	249	$F9	–	Unimplemented
1111 1010	250	$FA	0	Start
1111 1011	251	$FB	0	Continue
1111 1100	252	$FC	0	Stop
1111 1101	253	$FD	–	Unimplemented
1111 1110	254	$FE	0	Active Sensing
1111 1111	255	$FF	0	System Reset

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-- After that unrelenting barrage of MIDI information, I feel that it is only fitting to supply you with a bit more of a detailed description. To begin, the frequency data bytes, hex values $00 to $7F, are obviously the note values as expressed through almost 10 complete octaves. This is one drawback of the system. 7 bits of resolution is not sufficient to allow graduations in smaller increments than semitones. One recent speculation made regarding this point stated that future revisions of the system specifications should allow note increments in graduations of cents, 1/100's of a semitone. A possible but difficult to implement idea.

Status Messages, in contrast to the note values, require more than a little explanation before becoming coherent to any degree. Therefore, there are two different Status messages that can be encountered, The Channel Message and the System Message.

The Channel Message

A Channel message is one that states a message and a specific unit number to which the message is addressed. The lower 4 bits, bits 0–3, are used for this identification. In the charts shown above, plus the descriptions to follow, this has been signified by the xxxx in place of the lower nibble in each appropriate bit pattern. This 4 bit channel designation also explains the limitation of this system to 16 channels (decimal range 0–15).

There are two types of channel messages; Voice and Mode. Voice messages control each separate units voices, which are sent over the voice channels. The Mode message defines the instruments response to the Voice messages, sent over the instruments basic channel. The Mode message can control such things as Local Control On and Off, All Notes Off, Omni Mode On or Off, Mono Mode On/Poly Mode Off, or Poly Mode On/Mono Mode Off. See message Select Channel Mode for more information.

The System Message

System messages are ones that do not pertain to any one specific channel. Being the elite of the system entitles them to speak and except all to listen, immediately. Within the realm of the System message falls the Common, Real-Time, and Exclusive messages.

Common messages are directed to all units, regardless of channel number. The Common messages are comprised of Song Position Pointer, Song Select, Tune Request, and End Of Exclusive. Each Common message consists of a single byte, of which an in depth explanation will follow promptly.

Real-Time messages are also intended to be heard by all units in the system, at any time, even during the transmission of other data by a Status byte. A Real-Time messa[illegible], i.[illegible]ue in which can be tested for and ignored by the units [illegible] action if the information is desired. Real-Time messages are comprised of the Timing Clock, Start, Continue, Stop, Active Sensing, plus System Reset. Explanations will follow.

There is only one Exclusive Message, of which carries one Status byte, plus any length of data bytes following. It is terminated by either the Common message End Of Exclusive (EOX), or any other Status byte. An Exclusive Message is one that has been incorporated to identify the manufacturer of each piece of equipment, plus allow the manufacturer to transmit whatever message they please. A form of personalized service, finally.

Status Byte Explanations

Message:Note Off
Pattern: 1000 xxxx

The Note Off Message, used in conjunction with the Note On message, determine when each note will start and stop for the channel affected. Following this byte are always two data bytes. The first byte determines which note within a specific octave is to be affected, as can be demonstrated in the note chart above. The second byte is the Note Off Velocity. This value is used in synthesizers capable of ADSR (attack, decay, sustain, release), to set the rate of release from the sustained level. In synthesizers not so enabled, the release is immediate.

Message:Note On
Pattern: 1001 xxxx

As specified above, the Note On command is used to determine which note is stuck on a specified channel. The Note On Velocity, the second data byte, has a unique use this time. It determines the loudness setting of the note value struck. In units capable of ADSR, this level of loudness will be attained within a user predetermined time limit. If the unit is incapable of ADSR, the level will be reached immediately. The maximum loudness setting, considering we are working with the lower 7 of 8 bits, is a decimal value of 127. A value of 0 would be the same as turning the Note Off. According to specs, a value of 1 is equivalent to "triple pianissimo", very quite. A value of 127 (maximum), is "triple forte", very loud. The middle of the scale is 64, which is somewhere between "mezzo–piano" and "mezzo–forte" – middle scale. If the unit is incapable of various velocity settings, a value of 64 is used and transmitted.

Message:Polyphonic Key Pressure After Touch
Pattern: 1010 xxxx

Polyphonic Key Pressure After Touch is really just what is implied. The new value of key pressure attained after touching it. If the value has been set before to reflect the key pressure exerted, and the user strikes the key once again, this new pressure must be reflected. This status byte has two data bytes following. The first byte is the note stuck, with the second value byte representing the pressure. The values of pressure range from 0, no pressure at all, to 127, bashed through the keyboard

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Message:Control Change
Pattern: 1011 xxxx

Control Change is a status byte that is transmitted whenever a controller mechanism is adjusted. Controller mechanisms in this context refer to foot pedals, knobs (pots), modulation wheels, sliders, and switches. Two data bytes are used after the status message to reflect this change. The first byte indicates which controller was affected, while the second byte determines the new value attained. There are four ranges allowed for various controller mechanism used, as shown below.

Value Range	Controller Type
$00 to $3F	Continuous (ie. Foot Pedals, Knobs, Modulation Wheels, Sliders)
$40 to $5F	Switches
$60 to $7A	Presently Undefined
$7B to $7F	Channel Mode Messages (see Select Channel Mode following)

The first range, Continuous, often requires a more subtle range of graduations in comparison to say a switch. Either a switch is off or on, 0 or greater than 0. Continuous devices such as a slider can graduate across the scale in as large or small increments as desired, therefore a very subtle range is required. For this reason, the Control Change message allows for three data bytes if necessary. As stated before, the first data byte determines the controller affected, with the second data byte determining the value to set. This acts as the High Byte, or increment of 256 Lower Bytes. By now you know that the third byte is the Low Byte, thereby allowing a range of 14 bits, or 16,384 graduations. The Low Byte is not mandatory, therefore it can be left off if not required.

Message:Select Channel Mode
Pattern: 1011 xxxx

Select Channel Mode is a unique message that allows alterations to the way a MIDI unit will respond to, and transmit MIDI channel messages. Two data bytes are used to allow this transformation, with the first data byte having a limited range of $7A to $7F. This limitation is imposed due to the fact that two different messages share the same bit pattern (see Control Change above). The second data byte is used in conjunction with the first to achieve the result desired, as shown below.

Byte 1	Byte 2	Result
$7A	$00	Local Control Off
$7A	$7F	Local Control On
$7B	$00	All Notes Off
$7C	$00	Omni Mode Off, All Notes Off
$7D	$00	Omni Mode On, All Notes Off
$7E	$0x	Mono Mode On, Poly Mode Off, All Notes Off (x = number channels)
$7F	$00	Poly Mode On, Mono Mode Off, All Notes Off

The term Poly refers to Polyphonic sound reproduction, which is the ability for the unit to allow more than one note to be played simultaneously. Mono refers to Monophonic sound reproduction, which is reproduction of sound that only allows one note to be played during any one time period.

Omni is a term that is used to describe the ability for a unit to respond to all system messages, or only ones addressed to it's basic channel. When Omni is Off, the unit will only listen for messages addressed to itself. When Omni is On, it will listen and act upon every message coming over the bus.

Local Control is the ability for a unit to act through, or bypass it's own circuitry for the generation of sound. With Local Control On, the synthesizer will act like it normally does. With Local Control Off, the keyboard will still produce MIDI data as it is played, but the synthesizer will not produce any sound to compliment the data. This feature was incorporated to allow keyboards to control instruments other than the synthesizer attached.

Message:Program Change
Pattern: 1100 xxxx

Program Change is used with synthesizers that have banks of memory set aside for various user chosen sounds. Often times these sounds are assigned by changing the bank or patch number currently in use. Program Change allows a reflection of a change in this bank number for the channel affected, with one data byte assigned for the bank number chosen. The bank number can have a value of 0 to 127.

Message:Channel Pressure After Touch
Pattern: 1101 xxxx

The Channel Pressure After Touch is an interesting feature that determines the average pressure values for the unit at any instant. With this average pressure determined, a desired variation of the overall timbre value or volume of the instrument can be calculated by determining the deviation from the average required. The single data byte allowed can have a value of 0, no pressure, to 127, as much pressure as possible all around.

Message:Pitch Wheel Change
Pattern: 1110 xxxx

This message is one that allows a reflection of any change in the setting of the pitch wheel. As with Continuous Controller Mechanisms such as sliders, a fairly large scale is required to reflect variations in the pitch wheel setting. For this purpose, 14 bits of resolution have been provided. One odd point to note about the MIDI system specs at this point. With the Control Change message, a resolution of 14 bits was available if required. The data bytes were read High to Low. For a Pitch Wheel Change, 14 bits are also provided, but they are read Low to High. This poorly thought out variation could lead to confusion in writing code if taken for granted. So much for conventions.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Message:System Exclusive
Pattern: 1111 0000

A System Exclusive message is one that I find to be refreshing. It allows identification of the manufacturer, and also allows the same to get his two bits in. A System Exclusive message is comprised of the first data byte signifying the identification number assigned to the manufacturer, as per the IMA (International MIDI Association in California); and as many bytes following as the manufacturer requires to tell his story. This message is terminated either by an End Of Exclusive Message (EOX – 1111 0111), or any other Status byte that happens by. Below can be found a partial list of manufacturers ID codes, as supplied by the IMA.

Manufacturer	ID
Sequential Circuits Inc.	$01
Big Briar	$02
Octavew/Plateau	$03
Moog Music	$04
Passport Designs	$05
Lexicon	$06
Bon Tempi	$20
S.I.E.L.	$21
Kawai	$40
Roland	$41
Korg	$42
Yamaha	$43

Message:Song Position Pointer
Pattern: 1111 0010

The Song Position Pointer is a 14 bit value that allows a record to be kept of the number of beats since the start of a song session. This has been incorporated for use with synthesizers equipped with a sequencer (digital recorder), or for a rhythm machine. With this feature enabled, a flag can be set to allow music to be played from a specific location within the song. The two data bytes are read Low to High.

Message:Song Select
Pattern: 1111 0011

As with the Song Position Pointer, Song Select is also meant to be used with synthesizers equipped with a sequencer, or for a rhythm machine. Song Select uses one data byte to select which song or note sequence is to be played after a Start message has been received.

Message:Tune Request
Pattern: 1111 0110

Tune Request is a throw back into the age of Analog synthesizers. This single Status byte, no data bytes, is used to request a tuning of the Analog synthesizers oscillators.

Message:End Of System Exclusive (EOX)
Pattern: 1111 0111

The EOX message is a single Status byte, no data bytes, that flags when a System Exclusive Message has been completed.

Message:Timing Clock
Pattern: 1111 1000

The Timing Clock Message is one that can be used to synchronize all sequencers and/or rhythm machines on line. The clock transmits its message at a rate of 6 messages per beat. As stated earlier, this type of message will appear at a regular intervals, regardless of the current state of other Status messages.

Message:Start
Pattern: 1111 1010

As before, this Status message is intended for use with a synthesizer equipped with a sequencer or rhythm machine. This message will inform the sequencer/rhythm machine to begin playing a pre-arranged song or note sequence from the beginning. There is only one Status Byte, no data bytes, for this message. See the Song Select Message for a little more information regarding Start.

Message:Continue
Pattern: 1111 1011

As before, the Continue message has been incorporated for use with synthesizers equipped with a sequencer or rhythm machine. This message is used to restart the current song sequence after receiving the next Timing Clock message. The sequence is then picked up from the next position in the Song Position Pointer. This message is flagged by the user pressing the Continue button on the sythesizer. As before, this message carries one Status byte and no data bytes.

Message:Stop
Pattern: 1111 1100

Again, the Stop message is for synthesizers equipped with a sequencer or rhythm machine. When received, this message tells the sequencer to stop playing its current sequence. This message carries one Status byte and no data bytes.

Message:Active Sensing
Pattern: 1111 1110

This message is one that is transmitted by any MIDI instrument on line, powered up, but not actively involved in anything. This message is transmitted once every 300 milliseconds if

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

there is no activity on the MIDI bus. One Status byte and no data bytes.

Message: System Reset
Pattern: 1111 1111

The System Reset message performs exactly as you might expect. It tells all MIDI instruments on line to perform a power up sequence to return them to a freshly powered up state.

Retrospect

The MIDI system, as has been expressed throughout this article, is a series of well thought of specifications. Through each manufacturers implementation, musicians can use instruments of different origin and expect predictable results. This is the basic flaw in the industry today. MIDI on paper appears quite explicit in its goals. But soon after implementation, the manufacturers discovered many minute points not completely taken into consideration by MIDI Version 1.0. Today, it is this problem that confronts every musician who considers the step into the world of music synthesis.

In partial explanation, manufacturers immediately found problems implementing MIDI 1.0 following its release. Although each manufacturer tried their best to work the MIDI system within the boundaries of their machines, hindsight informed us of the inevitable. The manufacturers did not work together to make sure the systems were compatible. Each operated within their own collective vacuum, producing equipment meeting untested specifications, without considering the fact that MIDI 1.0 might be vague enough to allow multiple interpretations.

When the first MIDI machines were released on the market, problems became apparent immediately. Although many of the machines were compatible, some subtle to extreme cases of incompatibility did exist. And, as with any manufacturer faced with high R&D costs, plus further misinterpretations, many of the problems went unresolved. Take for example the Yamaha DX7 music synthesizer. It's a great machine, yet its MIDI implementation is not completely compatible with say a Roland or a Korg. In this relatively new field of digital music synthesis, it is not hard to find people who have personally become victims in this rush to compatibility.

Although I may appear to hold conflicting opinions regarding the MIDI, this is not so. MIDI in general does have its fine points. But I think that it is time for the manufacturers to get together and try to work their problems out. The specifications do not have to be altered just yet, just clarified down to the finest detail. Following this, each manufacturer can regroup to produce low cost true MIDI updates for the machines currently out. The longer this move is neglected, the greater the chance of more permanent problems. So much for a bit of sage advice.

In Summation

In closing, I would like to reflect on possible extensions to the MIDI 1.0 specifications. Due to the fact that 16 and 32 bit chips have dropped in price, implementations using these chips could be considered. With these chips, high speed communications, multi tasking, and unbelievable control over vast amounts of RAM and ROM could be taken advantage of. Further to this, we have had a chance to work with the MIDI 1.0 for quite a while now. This time of reflection has enabled musicians from all over to discover most of the weak points inherent in the system. With these points in mind, plus the technology available to us today, a true implementation could be performed, with one problem. The market has already been flooded with MIDI 1.0 equipped units. A revision at this stage would make everything else obsolete, as far as current thought allows. Due to this single fact, the MIDI specifications will not be allowed revision on a radical scale for some time to come. Although it may be hard to accept this twist of fate, the human side of this story requires consideration. Musical technology is slated to stagnate for the next few years, after which time will come an age of radical change. What our dreams are composed of today is the reality of tomorrow.

Finally, I would like to thank a few people who have helped me understand MIDI more than I ever thought possible. They include Vera Barycky, who supplied me with vast amounts of hard to get information that otherwise would be inaccessible. My father, Ted Evers, for his continuous stream of information and knowledge, and my brother, John Evers, for all his related experience working within the field, from which I was able to extract some particularly critical information.

References

The MIDI Handbook	by Paul Vytas
MIDI Specification 1.0	by the IMA in California
The Roland Juno–60 Manual	by Roland
The Roland JX–3P Implementation	by Roland
The Yamaha RX15 Digital Rhythm Programmer Manual	by Yamaha
Keyboard Magazine	June 84 & July 85
Mix Magazine	August 1984
Computing Now	December 1984
A stack of IMA newsletters	by the IMA in California

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Digital Synthesizer

Model : K11

MIDI Implementation Chart

Date : Jan. 20 1993
Version : 1.0

Function		Transmission	Reception	Remarks
Basic Channel	Default	X	A1–16, B1–16	Data is stored even after power is OFF.
	Changed	1–16 (Common to A,B)	A1–16each, B1–16each	
Mode	Default	X	mode 3	
	Messages	X	**********	
	Altered	**********		
Note Number		0–127	0–127	
	Sound range	**********	0–127	
Velocity	Note ON	○	○	
	Note OFF	8n, V=64	9n. V=0, 8n. V=0–127	
After Touch	For key	X	X	
	For channel	○	○	
Pitch bender		○	○ (14bit)	
Control Changes	1	○	○	Modulation
	6	X	○	Data entry
	7	○	○	Volume
	10	○	○	Panpot
	11	X	○	Expression
	64	○	○	Hold 1 (Sustain)
	67	X	○	Soft pedal
	69	X	○	Hold 2 (Sustain)
	91	○	○ (LO/HI)	Effect
	120	○	○	All Sound OFF
	121	○	○	Reset All Controllers
	100, 101	X	○	RPN, LSB, MSB
Program Change		○	0–127	
		0–127	0–127	
System exclusive		○	○ *1)	Receive only IN A
System Common	: Song position	X	X	
	: Song Select	X	X	
	: Tune	X	X	
System real time	: Clock	X	X	
	: Commands	X	X	
Others	: Local ON/OFF	X	○	
	: All notes OFF	○	○	
	: Active Sensing	○	○	
	: Reset	X	X	

Notes

1) Turn On or OFF by Exclusive
Bank select by Exclusive
RPN #0=Pitch Bender sensitivity
#1=Master fine tuning
#2=Master coarse tuning. Values are given by Data entry

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของลิขสิทธิ์ทุกครั้งที่มีการนำไปใช้

Mode 1 : OMNI ON, POLY Mode 2 : OMNI ON, MONO

Mode 3 : OMNI OFF, POLY Mode 4 : OMNI OFF, MONO

○ : Yes
X : No

```
/*
            M I D I    P R O J E C T

        MIDI Group Are ..

        Teerapong Wattanapittayakul 33100152
        Apichat   Saengtanawanich   33100489
        Arkhom    Busayakul         33100520

        Advisor ..

        Assc.Prof.Dr. Manas Sungworasilp        */

int cardtype;

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <dos.h>
#include <dir.h>
#include <alloc.h>
#include <graphics.h>
#include <string.h>
#include <time.h>
#include "midi.h"
#include "mouse.h"

boolean mouse_installed=false,mouse_active=false;

void far interrupt ( *Save_IRQ2)(void);
void far interrupt ( *Save_IRQ3)(void);
void far interrupt ( *Save_Int1C)(void);

boolean card_enable=false;
char track_name[128][12];
unsigned char far *ready_data[128],*ready_data_node[128],*track,*track_node
                ,*temp1,*temp1_node,*start;
unsigned char transpose_level=0,drum_ch=9,track_switch[128][3],channel_pan[32],
             track_vcs[128],track_vol[128],track_ch[128];
unsigned char present_pgm[32],track_present_pgm[128];
unsigned char *pgm_stack;int stack_point=0;
int  track_in_use[128],point_track_in_use=0;
long int note_on[128][2]; // use for step editor
int track_trans[128];
int tempo_rate=120;
int present_track=0;
boolean funckey;
int text_width=8;
int text_tall=8;
unsigned int main_timer;unsigned char main_tick;// 1 timer = 96 tick
unsigned int counter,old_count;
boolean pull_down_active=false;
boolean midiin_access;
boolean sending_data;
char mainmenu_command[main_menu_no][10]
   =("File","Track","Record","Playback","Music"
```

เอกสารนี้จัดทำขึ้นไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        ,"Option","Setup","Drum","View"}
       ,comment_text[3][25];
    int mainmenu_xpos[main_menu_no]
       ={30,78,134,198,278,334,398,454,576};
    int main_menu_pos=0,popup_menu_no=-1;
    unsigned char midiin_que[64],first_que=0,last_que=0;
    unsigned char midiout_que[2][1024];int out_head[2],out_tail[2];
    int noteperbar=4,step_type=0;
    int percent_tempo=50,main_volume=100;

    int VU_value,VU_reach,note_on_no,active_count=0;
    char mixer_VU[128][2],track_note_on[128];
    int mixer_index=0;
    time_t s_time,old_end_time,end_time;
    unsigned int t_hour,t_min,t_sec;
    boolean update_enable=false,monitor=false;

    lib *vcs_lib[9],pgm_chng;
    int present_lib=0;


    color_struct color;
    option_struct option;
    int edit_mode=0;
    struct track_header{
     unsigned char track_switch[3],
                   track_vcs,
                   track_vol,
                   track_ch;
     int           track_trans;
     char          track_name[12];
     }track_header;

   struct file_header{
     unsigned char transpose_level,
                  drum_ch,
                  channel_pan[32];
     int          tempo_rate,
                  noteperbar,
                  step_type,
                  mixer_index;
          }file_header;


   void all_done(boolean rim,boolean show_time,boolean vu);
   void quit(void);

   unsigned char pull_down(pulldownmenu *active_menu);
   void re_channel(unsigned char *data,unsigned char channel);

   void load_lib(void){
          int i,c,cont_count;
     char filename[40],temp[40],ch;
     FILE *flib;

     for(i=0;i<9;i++)
       if(strcmp(option.vcslib_name[i],"")){
         sprintf(filename,"%s%s",option.lib_path,option.vcslib_name[i]);
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        if((flib=fopen(filename,"rb"))==NULL){
          }
        else{
          vcs_lib[i]=getmem(sizeof(lib));
          vcs_lib[i]->max=0;
          c=0;
          do{
            ch=fgetc(flib);
            if((ch!=13)&&(ch!=10))
              temp[c++]=ch;
            }while((ch!=13)&&(ch!=10));
          temp[c]=0;
          do{ch=fgetc(flib);}while((ch==13)||(ch==10));
          if(!strcmp(temp,"Voice Library")){//This File is Voice Library File
            c=0;
            do{
              ch=fgetc(flib);
              if((ch==13)||(ch==10)) c++;
              }while(c<2);
            temp[0]=fgetc(flib);
            temp[1]=fgetc(flib);
            temp[2]=fgetc(flib);
            temp[3]=0;
            fgetc(flib);
            fgetc(flib);
            vcs_lib[i]->max=atoi(temp);
            for(cont_count=0;cont_count<vcs_lib[i]->max;cont_count++){
              vcs_lib[i]->number[cont_count]=0;
              if(!feof(flib)){
                for(c=0;c<3;c++){
                  ch=fgetc(flib);
                  switch(c){
                    case 0:vcs_lib[i]->number[cont_count]+=((ch-48)*100);break;
                    case 1:vcs_lib[i]->number[cont_count]+=((ch-48)*10);break;
                    case 2:vcs_lib[i]->number[cont_count]+=(ch-48);break;
                    }
                  }
                c=0;
                ch=fgetc(flib);
                do{
                  ch=fgetc(flib);
                  if((ch!=13)&&(ch!=10))
                    vcs_lib[i]->name[cont_count][c++]=ch;
                  }while((ch!=13)&&(ch!=10));
                vcs_lib[i]->name[cont_count][c]=0;
                ch=fgetc(flib);
                }
              }
            }
          fclose(flib);
          }
        }
    }

  void midiout(int channel,unsigned char midi_data){
    char buffer_status;
    unsigned data_port,control_port;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น รวมทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    switch(channel){
      case 0:data_port=0x2B0;
             control_port=0x2B1;
             break;
      case 1:data_port=0x2A0;
             control_port=0x2A1;
             break;
      }
     do{
      outportb(control_port,0);
      buffer_status=inportb(control_port)&0x04;
      }while(buffer_status==0);
     outportb(data_port,midi_data);
     }

   void put_out(int channel,unsigned char midi_data){
     midiout_que[channel][out_tail[channel]++]=midi_data;
     if(out_tail[channel]==1024)out_tail[channel]=0;
     }

   void push_pgm(unsigned char channel,unsigned char pgm_no){
     *(unsigned int*)(pgm_stack+stack_point*5)=main_timer;
     *(unsigned char*)(pgm_stack+stack_point*5+2)=main_tick;
     *(unsigned char*)(pgm_stack+stack_point*5+3)=channel;
     *(unsigned char*)(pgm_stack+stack_point*5+4)=pgm_no;
     stack_point++;
     }

   void pop_pgm(unsigned int time,unsigned char tick){
     unsigned channel,pgm_no;
     if(stack_point>0){
       while((*(unsigned int*)(pgm_stack+(stack_point-1)*5)<time)||
             ((*(unsigned int*)(pgm_stack+(stack_point-1)*5)==time)&&
             ((*(unsigned char*)(pgm_stack+(stack_point-1)*5+2))<=tick)){
         channel=*(unsigned char*)(pgm_stack+(stack_point-1)*5+3);
         pgm_no=*(unsigned char*)(pgm_stack+(stack_point-1)*5+4);
         midiout(channel/2,Pgm_chng|channel);
         midiout(channel/2,pgm_no);
         stack_point--;
         }
       }
     }

   void send_out(void){
     int i;

     for(i=0;i<2;i++)
       while(out_head[i]!=out_tail[i]){
         midiout(i,midiout_que[i][out_head[i]]);
         out_head[i]++;
         if(out_head[i]==1024)out_head[i]=0;
         }
     }

   void interrupt active_sense(__CPPARGS){
     active_count++;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(active_count==5){
      midiout(0,Active_sense);
      if(cardtype==1)midiout(1,Active_sense);
      active_count=0;
      }
    Save_Int1C();
    }

  void interrupt midiin(__CPPARGS){
    unsigned char temp,port21h;

    temp=inportb(0x2B0);
    if((!sending_data)&&(temp!=Active_sense)&&(temp!=0xFB)){
      if((temp&0x80)==0x80)
        midiout(present_track/16,((temp&0xF0)|((track_ch[present_track])%16)));
      else
        midiout(present_track/16,temp);
        }
    if((temp!=Active_sense)&&(midiin_access)){
      midiin_que[last_que]=temp;
      last_que++;
      if(last_que>=64) last_que=0;
      }
    Save_IRQ2();
    asm sti;
    port21h=inportb(0x21)&0xFB;
    outportb(0x21,port21h);
    }

  void interrupt inc_timer(__CPPARGS){
    unsigned char port21h;

    port21h=inportb(0x21)&0xF7;
    outportb(0x21,0xFF);
    main_tick++;
    if(main_tick==95){
      main_tick=0;
      main_timer++;
      }
    if(main_tick%4==0){
      midiout(0,0xF8);
      midiout(1,0xF8);
      }
    if((main_tick%8)==0) update_enable=true;
    Save_IRQ3();
    asm sti;
    inportb(0x21);
    outportb(0x21,port21h);
    }

  unsigned char read_que(void){
    unsigned char ch;

    ch=midiin_que[first_que];
    midiin_que[first_que++]=0xFF;
    if(first_que==64) first_que=0;
    return(ch);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}

void init_graph(void){
  int gd=DETECT,gm,err_code;

  screen_saver(0);
  initgraph(&gd,&gm,option.graphics_driver_path);
  err_code=graphresult();
  if(err_code!=grOk){
    printf("Graphics error: %s\n", grapherrormsg(err_code));
    printf("Press any key to halt:");
    getch();
    printf("\n\n%s\n",option.graphics_driver_path);
    exit(1); /* terminate with an error code */
    }
  text_width=textwidth(" ");
  text_tall=textheight(" ");
  }

boolean endof_data(unsigned char *data){
  if((*(unsigned int*)data==0xFFFF)&&(*(data-1)==0xFF))
    return(true);
  else
    return(false);
  }

void insert(unsigned char *data,int num){
  unsigned char *node;
  int i;

  node=data;
  do{
    if(!endof_data(data))data++;
    }while(!endof_data(data));
  data+=num;
  data+=3;
  do{
    *data=*(data-(num));
    data--;
    }while(data>=node+num);
  for(i=0;i<num;i++)
    *(node+i)=0xF0;
  }

void del(unsigned char *data,int num){
  unsigned char *node;
  unsigned int i,j;

  node=data;
  i=0;
  while(!endof_data(node+i)){
    i++;
    }
  i=i-num+2;
  for(j=0;j<i;j++)
    *(data+j)=*(data+j+num);
  data=node;
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }

  void *find_next(unsigned char *data){
    if(!endof_data(data)) data+=3;
    if((*data!=0xFF)&&(!endof_data(data))){
      while(*data!=0xFF)data++;
      data++;
      }
    if(!endof_data(data))
      return(data);
    else
      return(NULL);
    }

  void *find_pred(unsigned char *data,unsigned char *start){
    if(data!=start){
      data--;
      do{
        data--;
        }while((*data!=0xFF)||(*(data-1)==0xFF));
      data++;
      return(data);
      }
    else
      return(NULL);
    }

  void return_time(unsigned char *data,unsigned int *note,unsigned char *tick){
    *note=*(unsigned int*)data;
    *tick=*(unsigned char*)(data+2);
    }

  void *find_time(unsigned char *data,unsigned char *start,unsigned int note_count,unsigned
  char int_tick,boolean en_insert){
    unsigned char *node,tick_1;
    unsigned int time_1;
    boolean end=false;

    node=data;
    if((!endof_data(node))&&(*(node+2)+96*(unsigned long)(*(unsigned int*)node))<(unsigned l
  ong)(note_count*96+int_tick)){
      // Search Forward First
      do{
        if(!endof_data(node))node++;
        do{
          if(*node!=0xFF)node++;
          if((*(node-1)==0xFF)&&(*(unsigned int*)(node)==0xFFFF))
            end=true;
          }while((*node!=0xFF)&&(!end));
        if(!end)node++;
        return_time(node,&time_1,&tick_1);
        }while( ( *(node+2)+96*(unsigned long)(*(unsigned int*)node) < int_tick+96*(unsigned
  long)(note_count))&&(!end));
      }
    else if((node!=start)&&(*(node+2)+96*(unsigned long)(*(unsigned int*)node))>(unsigned lo
  ng)(note_count*96+int_tick)){
      // Search Reward
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      end=false;
      node++;
      do{
        if(node!=start)node-=2;
        do{
          if(node==start)
            end=true;
          if((!end)&&(*node!=0xFF))node--;
          }while((*node!=0xFF)&&(!end));
        if((*(node-1)==0xFF)&&(!end)){
          //node--;
          }
        else if(!end)
          node++;
        }while( ( *(node+2)+96*(unsigned long)(*(unsigned int*)node) > int_tick+96*(unsigned
 long)(note_count))&&(!end));
      if(*(node+2)+96*(unsigned long)(*(unsigned int*)node)<int_tick+96*(unsigned long)(note
_count)){
        node+=3;
        do{
          node++;
          }while(*(node-1)!=0xFF);
        }
      }
    if((unsigned long)((*(unsigned int*)node)*96+(*(node+2)))==(unsigned long)(note_count*96
+int_tick)){
      // Find
      return(node);
      }
    else
      if(en_insert){
        insert(node,4);
        *(unsigned int*)node=note_count;
        *(node+2)=int_tick;
        *(node+3)=0xFF;
        return(node);
        }
      else
        return(node);
    }

  void wait_input(void){
    do{
      }while(first_que==last_que);
    }

  char assign_VU_value(unsigned char velocity){
    return((velocity+8)/16);
    }

  void key_pos(unsigned char note_num,int *x,int *y){
    *x=((note_num-24)/12)*77+5;
    *y=35;
    switch(note_num%12){
      case  0:*x+=0; break;
      case  1:*x+=5; *y=20;break;
      case  2:*x+=11;break;
```

เอกสารนี้จัดทำขึ้นไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      case  3:$x+=16;$y=20;break;
      case  4:$x+=22;break;
      case  5:$x+=33;break;
      case  6:$x+=38;$y=20;break;
      case  7:$x+=44;break;
      case  8:$x+=49;$y=20;break;
      case  9:$x+=55;break;
      case 10:$x+=60;$y=20;break;
      case 11:$x+=66;break;
      }
    }

  void rec_key_pressed(unsigned char note_num,boolean pushed){
    int x,y,col;

    if((note_num>23)&&(note_num<109)){
      key_pos(note_num,&x,&y);
      if(pushed)
        if(y==20)col=15;
        else     col=0;
      else
        if(y==20)col=0;
        else     col=15;
      setfillstyle(1,col);
      bar(x-2+62,y-2+50,x+1+62,y+1+50);
      }
    }

  void repaint_key(void){
    unsigned char i;

    for(i=24;i<109;i++)
      rec_key_pressed(i,false);
    }

  void send_data(void){
    int i;
    unsigned int time;
    unsigned char tick,temp;

    for(i=0;i<point_track_in_use;i++)
      if(ready_data[track_in_use[i]]!=NULL){
        return_time(ready_data[track_in_use[i]],&time,&tick);
      while((time<main_timer)||((main_timer==time)&&(tick<=main_tick))){
        ready_data[track_in_use[i]]+=3;
        if(track_switch[track_in_use[i]][0])
        do{
          switch($ready_data[track_in_use[i]]&0xF0){
            case Note_on  :if(present_pgm[track_ch[track_in_use[i]]]!=track_present_pgm[trac
k_in_use[i]]){
                           present_pgm[track_ch[track_in_use[i]]]=track_present_pgm[track_
in_use[i]];
                           put_out(track_ch[track_in_use[i]]/16,Pgm_chng|(track_ch[track_i
n_use[i]]%16));
                           put_out(track_ch[track_in_use[i]]/16,track_present_pgm[track_in
_use[i]]);
                           }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i]]
++);
                        do{
                          if((monitor)&&(track_in_use[i])==present_track)
                            rec_key_pressed(*ready_data[track_in_use[i]],true);
                          put_out(track_ch[track_in_use[i]]/16,track_trans[track_in_use[i
]]+*ready_data[track_in_use[i]]++);
                          if(track_switch[track_in_use[i]][2]){
                            temp=(unsigned char)(((track_vol[track_in_use[i]]*127/100)*ma
in_volume)/100);
                            ready_data[track_in_use[i]]++;
                            }
                          else
                            temp=(unsigned char)((((track_vol[track_in_use[i]]*(*ready_da
ta[track_in_use[i]]++))/100)*main_volume)/100);
                          track_note_on[track_in_use[i]]++;
                          if(mixer_VU[track_in_use[i]][0]<assign_VU_value(temp))
                            mixer_VU[track_in_use[i]][0]=assign_VU_value(temp);
                          put_out(track_ch[track_in_use[i]]/16,temp);
                          }while((*ready_data[track_in_use[i]]&0x80)!=0x80);
                        break;
        case Note_off :put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i]]
++);
                        do{
                          if((monitor)&&(track_in_use[i])==present_track)
                            rec_key_pressed(*ready_data[track_in_use[i]],false);
                          put_out(track_ch[track_in_use[i]]/16,track_trans[track_in_use[i
]]+*ready_data[track_in_use[i]]++);
                          put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i
]]++);
                          if(track_note_on[track_in_use[i]]>0)track_note_on[track_in_use[
i]]--;
                          if(track_note_on[track_in_use[i]]==0) mixer_VU[track_in_use[i]]
[0]=0;
                          }while((*ready_data[track_in_use[i]]&0x80)!=0x80);
                        break;
        case Pgm_chng ://put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i
]]++);
                        ready_data[track_in_use[i]]++;
                        //put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i
]]++);
                        push_pgm(track_ch[track_in_use[i]],track_present_pgm[track_in_use
[i]]);
                        track_present_pgm[track_in_use[i]]=*ready_data[track_in_use[i]]++
;
                        break;
        case Ctrl_chng:put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i]]
++);
                        do{
                          put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i
]]++);
                          put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i
]]++);
                          }while((*ready_data[track_in_use[i]]&0x80)!=0x80);
                        break;
        default       :do{
                          put_out(track_ch[track_in_use[i]]/16,*ready_data[track_in_use[i
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
]]++);
                          }while((*ready_data[track_in_use[i]]&0x80)!=0x80);
                        break;
          }
        }while(*ready_data[track_in_use[i]]!=0xFF);
      else
      while(*ready_data[track_in_use[i]]!=0xFF){
        ready_data[track_in_use[i]]++;
        }
      ready_data[track_in_use[i]]++;
      return_time(ready_data[track_in_use[i]],&time,&tick);
      }
    }
  }

void init_sio(void){
  unsigned char testsio;

  outportb(0x2B1,0);
  testsio=inportb(0x2B1);
  if(testsio == 0xFF){
    putchar(0x07);
    printf("MIDI Card Error\n\nCannot Record or Playback\n\n");
    }
  else card_enable=true;

  asm cli;
  asm nop;
  outportb(0x2B1,0x18);asm nop;
  outportb(0x2B1,0x04);asm nop;
  outportb(0x2B1,0xC4);asm nop;
  outportb(0x2B1,0x03);asm nop;
  outportb(0x2B1,0xC1);asm nop;
  outportb(0x2B1,0x05);asm nop;
  outportb(0x2B1,0x68);asm nop;
  outportb(0x2B1,0x01);asm nop;
  outportb(0x2B1,0x18);asm nop;
  outportb(0x2A1,0x18);asm nop;
  outportb(0x2A1,0x04);asm nop;
  outportb(0x2A1,0xC4);asm nop;
  outportb(0x2A1,0x03);asm nop;
  outportb(0x2A1,0xC1);asm nop;
  outportb(0x2A1,0x05);asm nop;
  outportb(0x2A1,0x68);asm nop;
  outportb(0x2A1,0x01);asm nop;
  outportb(0x2A1,0x18);asm nop;
  asm sti;

  }

void program_8253(int tempo_rate){
  unsigned long int divider,div1,div2;

  divider=(unsigned int)10416u/tempo_rate*60;
  div1=divider%256;
  div2=(unsigned char)(divider/256);
  outportb(0x2B3,0x76);
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    outportb(0x2A3,2);
    outportb(0x2A3,0);
    //outportb(0x2B3,0x40);
    outportb(0x2B3,0x36);
    outportb(0x2A2,div1);
    outportb(0x2A2,div2);
    //outportb(0x2B3,0x00);
    }

/* Extend Function */

void save_track(void){
  char *name,*command;
  unsigned long size=0;
  FILE *save_fp;
  boolean en_thistime;
  int c;

  name=getmem(100);
  command=getmem(100);
  sprintf(name,"%strack.%d",option.temporary_path,present_track);
  sprintf(command,"del %s",name);
  save_fp=fopen(name,"wb");
  if(save_fp){
    track=track_node;
    while(*(unsigned char*)track!=0xFF){
      fputc(*(track++),save_fp);
      size++;
      }
    fputc(*track++,save_fp);
    size++;
    if(*(unsigned char*)(track+2)==0xF0){
      while(*track==0xF0)
        track++;
      }
    //while(!endof_data(track)){
    while(*(unsigned int*)track!=0xFFFF){
      en_thistime=false;
      c=3;
      while(*(unsigned char*)(track+c)!=0xFF){
        if(*(unsigned char*)(track+c)!=0xF0)
          en_thistime=true;
        c++;
        }
      if(en_thistime){
        fputc(*track++,save_fp);
        fputc(*track++,save_fp);
        fputc(*track++,save_fp);
        size+=3;
        while(*(unsigned char*)track!=0xFF){
          if(*(unsigned char*)track!=0xF0){
            fputc(*track++,save_fp);
            size++;
            }
          else track++;
          }
        fputc(*track++,save_fp);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            size++;
            }
          }
        fputc(0xFF,save_fp);
        fputc(0xFF,save_fp);
        size+=2;
        fclose(save_fp);
        }
      else{
        message(208,200,"Error!","Option Error program aboart");
        quit();
        }
      if(size==3) system(command);
      farfree(command);
      farfree(name);
      }

    void load_track(void){
      char *name;
      FILE *load_fp;
      int i;

      name=getmem(100);
      sprintf(name,"%strack.%d",option.temporary_path,present_track);
      track=track_node;
      load_fp=fopen(name,"rb");

      if(load_fp){
        track_node=track;
        for(i=0;i<3;i++)
          *(track++)=fgetc(load_fp);
        do{
          *(track++)=fgetc(load_fp);
          }while(!endof_data(track-3));
        fclose(load_fp);
        track=track_node;
        }
      else{
        *(track++)=0xFF;
        *(track++)=0xFF;
        *(unsigned int*)track=0xFFFF;
        track=track_node;
        }
      farfree(name);
      }

    void find_start_track(void){
      track=track_node;
      if(*track!=0xFF)
        do{
          track++;
          }while(*track!=0xFF);
      track++;
      }

    void find_time_data(unsigned int time,unsigned char tick){
      int i;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    for(i=0;i<128;i++){
      if(ready_data[i]!=NULL){
        ready_data[i]=find_time(ready_data[i],(unsigned char*)(ready_data_node[i]+1),time,ti
ck,false);
        }
      }
    }


void perform_data(boolean p_track,unsigned int start_time,unsigned char start_tick){
  int old_track,done,trk_no;
  char name[100];
  struct ffblk ffblk;
  unsigned size,count;

  sprintf(name,"%sTRACK.*",option.temporary_path);
  old_track=present_track;
  done=findfirst(name,&ffblk,0);
  while(!done){
    trk_no=atoi((char*)(&ffblk.ff_name[6]));
    if(((p_track)||(trk_no!=old_track))&&(((unsigned char)(ffblk.ff_name[6]))<58)){
      track_in_use[point_track_in_use++]=trk_no;
      present_track=trk_no;
      load_track();
      find_start_track();
      size=0;
      while(!endof_data((unsigned char*)(track+size))){
        size++;
        }
      if(size!=0){// All Condition is OK;
        size+=3;
        ready_data_node[trk_no]=getmem(size+8);
        ready_data[trk_no]=ready_data_node[trk_no];
        *ready_data[trk_no]++=0xFF;
        for(count=0;count<size;count++){
          *(ready_data[trk_no]+count)=*(track+count);
          }
        ready_data[trk_no]=find_time(ready_data[trk_no],(unsigned char*)(ready_data_node[t
rk_no]+1),start_time,start_tick,false);
        }
      }
    done=findnext(&ffblk);
    }
  present_track=old_track;
  }

void destroy_data(void){

  while(point_track_in_use!=0){
    point_track_in_use--;
    farfree(ready_data_node[track_in_use[point_track_in_use]]);
    ready_data_node[track_in_use[point_track_in_use]]=ready_data[track_in_use[point_track_
in_use]]=NULL;
    }
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void lib_browser(int *number,char *name,int xpos,int ypos){
  int save_color,i,return_no,no,x_mouse,y_mouse,x1,y1;
  struct fillsettingstype save_fill;
  void far *ptr,*ptr1,*ptr2;
  boolean end,khit;
  char ch;

  end=false;
  save_color=getcolor();
  getfillsettings(&save_fill);
  return_no=0;
  no=0;
  ptr=getmem(imagesize(xpos,ypos,xpos+150,ypos+224+40));
  ptr1=getmem(imagesize(0,0,150,15));
  getimage(xpos,ypos,xpos+150,ypos+224+40,ptr);
  frame(xpos,ypos,xpos+150,ypos+224+40,false);
  write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_color);
  setcolor(color.menu_text_color);
  for(i=0;i<16;i++){
    outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28,vcs_lib[present_lib]
->name[i]);
    }
  setcolor(color.active_menu_text_color);
  setfillstyle(1,1);
  do{
    khit=false;
    x_mouse=xpos+30;
    y_mouse=ypos+14*no+color.frame_width+28+7;
    set_mouse_position(x_mouse,y_mouse);
    x1=xpos+color.frame_width+4;
    y1=ypos+color.frame_width+25+no*14;
    getimage(x1,y1,x1+16*text_width,y1+13,ptr1);
    bar(x1,y1,x1+16*text_width,y1+13);
    outtextxy(xpos+color.frame_width+8,ypos+14*no+color.frame_width+28,vcs_lib[present_lib
]->name[return_no]);
    do{
      do{
        mouse_position(&x_mouse,&y_mouse);
        }while((!kbhit())&&(!clicked())&&(y_mouse>=ypos+14*no+color.frame_width+28)&&(y_mo
use<=ypos+14*no+color.frame_width+28+13));
      khit=true;
      if(!kbhit()){
        if(y_mouse<ypos+14*no+color.frame_width+21){
          ch=UP_KEY;
          funckey=true;
          }
        else if(y_mouse>ypos+14*no+color.frame_width+21){
          ch=DOWN_KEY;
          funckey=true;
          }
        }
      }while((!kbhit())&&(!khit)&&(!clicked()));
    if(leftb_pressed()){
      end=true;
      ch=RETURN_KEY;
      funckey=false;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }
  else if(rightb_pressed()){
    end=true;
    ch=ESC_KEY;
    funckey=false;
    }
  else if(kbhit()){
    ch=readfunc();
    }
  putimage(x1,y1,ptr1,COPY_PUT);
  if(funckey)switch((unsigned char)ch){
    case UP_KEY   :if(return_no>0){
                     return_no--;
                     if(no>0)no--;
                     else{// scroll screen
                       y1=ypos+color.frame_width+25;
                       ptr2=getmem(imagesize(x1,y1,x1+16*text_width,y1+15*14));
                       getimage(x1,y1,x1+16*text_width,y1+15*14,ptr2);
                       putimage(x1,y1+14,ptr2,COPY_PUT);
                       setcolor(color.menu_text_color);
                       setfillstyle(1,color.frame_color);
                       bar(x1,y1,x1+16*text_width,y1+14);
                       outtextxy(xpos+color.frame_width+8,ypos+color.frame_width+28,vcs_
lib[present_lib]->name[return_no]);
                       setfillstyle(1,1);
                       setcolor(color.active_menu_text_color);
                       farfree(ptr2);
                       }
                     }
                   break;
    case DOWN_KEY :if((return_no<vcs_lib[present_lib]->max-1)&&
                   (vcs_lib[present_lib]->number[return_no+1]>=0)){
                     return_no++;
                     if(no<15) no++;
                     else{//scroll screen
                       y1=ypos+color.frame_width+25+14;
                       ptr2=getmem(imagesize(x1,y1,x1+16*text_width,y1+15*14));
                       getimage(x1,y1,x1+16*text_width,y1+15*14,ptr2);
                       putimage(x1,y1-14,ptr2,COPY_PUT);
                       setcolor(color.menu_text_color);
                       setfillstyle(1,color.frame_color);
                       bar(x1,y1+14*14,x1+16*text_width,y1+15*14);
                       outtextxy(xpos+color.frame_width+8,ypos+14*no+color.frame_width+2
8,vcs_lib[present_lib]->name[return_no]);
                       setfillstyle(1,1);
                       setcolor(color.active_menu_text_color);
                       farfree(ptr2);
                       }
                     }
                   break;
    case PAGEUP_KEY:
                   if((return_no<16)&&(return_no==no)){
                     return_no=0;
                     no=0;
                     }
                   else{
                     if(return_no>=16)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        return_no-=16;
                      else{
                        no=0;
                        return_no=0;
                        }
                      frame(xpos,ypos,xpos+150,ypos+224+40,false);
                      write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_co
lor);
                      setcolor(color.menu_text_color);
                      if(return_no-no<16){
                        no=0;
                        }
                      for(i=0;i<16;i++){
                        outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28
,vcs_lib[present_lib]->name[return_no-no+i]);
                        }
                      setcolor(color.active_menu_text_color);
                      }
                    break;
     case PAGEDOWN_KEY:
                    if(return_no+16<vcs_lib[present_lib]->max){
                      return_no+=16;
                      frame(xpos,ypos,xpos+150,ypos+224+40,false);
                      write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_co
lor);
                      setcolor(color.menu_text_color);
                      if(return_no+no+16>vcs_lib[present_lib]->max)
                        no=16-(vcs_lib[present_lib]->max-return_no);
                      for(i=0;i<16;i++)
                        outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28
,vcs_lib[present_lib]->name[return_no-no+i]);
                      setcolor(color.active_menu_text_color);
                      }
                    else{
                      no+=vcs_lib[present_lib]->max-return_no-1;
                      return_no=vcs_lib[present_lib]->max-1;
                      }
                    break;
     case CTRL_PGUP:no=0;
                    return_no=0;
                    frame(xpos,ypos,xpos+150,ypos+224+40,false);
                    write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_colo
r);
                    setcolor(color.menu_text_color);
                    for(i=0;i<16;i++)
                      outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28,v
cs_lib[present_lib]->name[return_no-no+i]);
                    setcolor(color.active_menu_text_color);
                    break;
     case CRTL_PGDN:return_no=vcs_lib[present_lib]->max-1;
                    if(return_no<15)
                      no=return_no;
                    else
                      no=15;
                    frame(xpos,ypos,xpos+150,ypos+224+40,false);
                    write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_colo
r);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

```
                    setcolor(color.menu_text_color);
                    for(i=0;i<16;i++)
                      outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28,v
cs_lib[present_lib]->name[return_no-no+i]);
                    setcolor(color.active_menu_text_color);
                    break;
        }
      else{
        if((ch==RETURN_KEY)||(ch==ESC_KEY)) end=true;
        else if((ch>='1')&&(ch<='8')){
          if(strcmp(option.vcslib_name[ch-49],"")){
            present_lib=ch-49;
            return_no=0;
            no=0;
            frame(xpos,ypos,xpos+150,ypos+224+40,false);
            write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_color);
            setcolor(color.menu_text_color);
            for(i=0;i<16;i++){
              outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28,vcs_lib[pres
ent_lib]->name[i]);
              }
            setcolor(color.active_menu_text_color);
            }
          }
        }
      }while(!end);
    setcolor(save_color);
    setfillstyle(save_fill.pattern,save_fill.color);
    if(ch==RETURN_KEY){
      *number=vcs_lib[present_lib]->number[return_no];
      strcpy(name,vcs_lib[present_lib]->name[return_no]);
      }
    else if(ch==ESC_KEY) *number=-1;
    putimage(xpos,ypos,ptr,COPY_PUT);
    farfree(ptr1);
    farfree(ptr);
    setcolor(save_color);
    setfillstyle(save_fill.pattern,save_fill.color);
    }

  void panpot(unsigned char channel,unsigned char value){
    midiout(channel/16,Ctrl_chng|(channel%16));
    midiout(channel/16,0x0A);
    if(value<127)
      midiout(channel/16,value);
    else
      midiout(channel/16,127);
    }

  void effect(int channel,boolean high){
    midiout(channel/16,Ctrl_chng|(channel%16));
    midiout(channel/16,0x5B);
    if(high)
      midiout(channel/16,127);
    else
      midiout(channel/16,0);
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void set_synth(void){
  int i,j;

  if(cardtype==1)j=32;
  else j=16;
  for(i=0;i<j;i++){
    effect(i,track_switch[i][1]);
    panpot(i,channel_pan[i]);
    }
  }

void all_note_off(void){
  int i;
  unsigned char j;

  for(i=0;i<2;i++)
    if((cardtype==1)||(i==0))
      for(j=0;j<16;j++){
        midiout(i,Ctrl_chng|j);
        midiout(i,0x7B);
        midiout(i,0);
        midiout(i,Ctrl_chng|j);
        midiout(i,0x40);
        midiout(i,0);
        }
  }

void channel_note_off(int track_no){
  int i,j;

  i=track_ch[track_no]/16;
  j=track_ch[track_no]%16;
  midiout(i,Ctrl_chng|j);
  midiout(i,0x7B);
  midiout(i,0);
  }

void set_header(void){
  int i;
  file_header.transpose_level=transpose_level;
  file_header.drum_ch=drum_ch;
  for(i=0;i<32;i++)
    file_header.channel_pan[i]=channel_pan[i];
  file_header.tempo_rate=tempo_rate;
  file_header.noteperbar=noteperbar;
  file_header.step_type=step_type;
  file_header.mixer_index=mixer_index;
  }

void save_header(void){
  int i;

  transpose_level=file_header.transpose_level;
  drum_ch=file_header.drum_ch;
  for(i=0;i<32;i++)
    channel_pan[i]=file_header.channel_pan[i];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  tempo_rate=file_header.tempo_rate;
  noteperbar=file_header.noteperbar;
  step_type=file_header.step_type;
  mixer_index=file_header.mixer_index;
  }

void set_track_header(int track_no){
  int i;

  for(i=0;i<3;i++)
    track_header.track_switch[i]=track_switch[track_no][i];
  track_header.track_vcs=track_vcs[track_no];
  track_header.track_vol=track_vol[track_no];
  track_header.track_ch=track_ch[track_no];
  track_header.track_trans=track_trans[track_no];
  strcpy(track_header.track_name,track_name[track_no]);
  }

void save_track_header(int track_no){
  int i;

  for(i=0;i<3;i++)
    track_switch[track_no][i]=track_header.track_switch[i];
  track_vcs[track_no]=track_header.track_vcs;
  track_vol[track_no]=track_header.track_vol;
  track_ch[track_no]=track_header.track_ch;
  track_trans[track_no]=track_header.track_trans;
  strcpy(track_name[track_no],track_header.track_name);
  }

/* Working Function */

void startup(void){
  unsigned char port21h,test_card;

  reset_mouse();
  program_8253(tempo_rate);
  init_sio();
  Save_Int1C=getvect(0x1C);
  Save_IRQ2=getvect(0x0A);
  Save_IRQ3=getvect(0x0B);
  setvect(0x0B,inc_timer);
  setvect(0x0A,midiin);
  setvect(0x1C,active_sense);
  port21h=inportb(0x21);
  port21h&=0xF3;
  outportb(0x21,port21h);
  if(card_enable){
    outportb(0x2A0,0xFE);
    delay(1);
    outportb(0x2A1,0);
    test_card=inportb(0x2A1)&0x04;
    if(test_card==0)cardtype=0;
    else cardtype=1;
    if(cardtype){
      printf("MIDI Card  Type II\n");
      printf("   32 Channel Enable.\n");
```

เอกสารนี้ใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }
      else{
        printf("MIDI Card  Type I\n");
        printf("   16 Channel Enable.\n");
        }
      printf("  .   Press any key to continue.");
      readfunc();printf("\n");
      }
    }

  void init_data(void){
    int i;

    for(i=0;i<128;i++){
      track_vcs[i]=0;
      track_vol[i]=0;
      track_trans[i]=0;
      if(cardtype==1)track_ch[i]=i%32;
      else track_ch[i]=i%16;
      ready_data[i]=NULL;
      ready_data_node[i]=NULL;
      mixer_VU[i][0]=0;
      mixer_VU[i][1]=0;
      track_note_on[i]=0;
      track_switch[i][0]=false;
      track_switch[i][1]=false;
      track_switch[i][2]=false;
      strcpy(track_name[i],"");
      track_present_pgm[i]=255;
      }
    for(i=0;i<32;i++){
      channel_pan[i]=64;
      present_pgm[i]=255;
      }
    for(i=0;i<10;i++)
      strcpy(comment_text[i],"");
    midiin_access=false;
    sending_data=false;
    }

  void main_screen(void){
    int i;

    i=color.frame_color;
    color.frame_color=color.screen_color;
    frame(0,0,639,479,true);
    color.frame_color=i;
    write_text(0,639,10,"MIDI Project II",color.title_text_color);
    for(i=0;i<main_menu_no;i++){
      write_menu_command(mainmenu_xpos[i],26,mainmenu_command[i],0,false);
      }
    }

  void quantize(void){
    unsigned int time,old_time;
    int i;
    unsigned char tick,old_tick,ch,misc;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    ch=pull_down(&quantize_menu);
    if((ch==RETURN_KEY)&&(!funckey)){
      switch(quantize_menu.command_no){
        case 0:misc=96;break;
        case 1:misc=48;break;
        case 2:misc=32;break;
        case 3:misc=24;break;
        case 4:misc=16;break;
        case 5:misc=12;break;
        }
      load_track();
      track=track_node;
      while(*track!=0xFF)track++;
      track++;
      old_time=0xFFFF;
      old_tick=0xFF;
      while((!endof_data(track))&&(track!=NULL)){
        return_time(track,&time,&tick);
        if((tick%misc)!=0){
          if(tick%misc>misc/2)
            tick=tick-(tick%misc)+misc;
          else
            tick=tick-(tick%misc);
          }
        if(tick==96){
          tick=0;
          time++;
          }
        if((time!=old_time)||(tick!=old_tick)){
          *(unsigned int*)track=time;
          *(unsigned char*)(2+track)=tick;
          old_time=time;
          old_tick=tick;
          track+=3;
          while(*track!=0xFF)track++;
          track++;
          }
        else{
          track--;
          for(i=0;i<4;i++){
            *track++=0xF0;
          }
          while(*track!=0xFF)track++;
          track++;
          }
        }
      save_track();
      }
    }

  void kill_velocity(void){
    //char temp[4]="127";
    unsigned char velocity=127;
    load_track();
    track=track_node;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    while(*track!=0xFF){
      track++;
      }
    track++;
    while(!endof_data(track)){
      track+=3;
      while(*track!=0xFF){
        if(((*track)&0xF0)==Note_on){
          track++;
          do{
            track++;
            *track=velocity;
            track++;
            }while(((*track)&0x80)!=0x80);
          }
        else{
          do{
            track++;
            }while(((*track)&0x80)!=0x80);
          }
        }
      track++;
      }
    save_track();
    }

  void kill(int Message){
    unsigned char *start;
    int i,num_del;

    load_track();
    track=track_node;
    while(*track!=0xFF){
      track++;
      }
    track++;
    start=track;
    while(!endof_data(track)){
      track+=3;
      num_del=0;
      while(*track!=0xFF){
        if(((*track)&0xF0)==Message){
          do{
            *track++=0xF0;
            num_del++;
            }while(((*track)&0x80)!=0x80);
          }
        else{
          do{
            track++;
            }while(((*track)&0x80)!=0x80);
          }
        }
      track++;
      track=find_pred(track,start);
      if(!track)track=start;
      if((track==start)&&(*(track+3+num_del)==0xFF)){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        //track--;
        for(i=0;i<3;i++)
          *track++=0xF0;
        while(*track!=0xFF)track++;
        *track++=0xF0;
        start=track;
        }
      else if(*(track+3+num_del)==0xFF){
        track--;
        for(i=0;i<4;i++){
          *track++=0xF0;
          }
        while(*track!=0xFF)track++;
        track++;
        }
      else{
        track+=3;
        while(*track!=0xFF)track++;
        track++;
        }
      }
    save_track();
    }

  void kill_whell(void){
    kill(Pitch_wheel);
    }

  void kill_pressure(void){
    kill(Key_Press);
    kill(Channel_press);
    }

  void kill_program(void){
    kill(Pgm_chng);
    }

  void kill_control(void){
    kill(Ctrl_chng);
    }

  void time_shifter(void){
    int  time_shf;
    char tick_shf,ch;
    long int tmp;
    unsigned int time;
    unsigned char tick;
    char temp[6];

    open_window(150,150,400,208,1,"Shift Note");
    setcolor(color.normaltext_color);
    outtextxy(160,175,"Enter Value");
    outtextxy(160,189,"x 1/96 Quater Note :");
    strcpy(temp,"");
    ch=readtext(336,189,temp,5,5,false);
    tmp=atol(temp);
    if((tmp==0)&&(ch==RETURN_KEY)&&(!funckey))//Error
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      message(180,200," Error "," Bad Input Value ");
    else if((ch==RETURN_KEY)&&(!funckey)){
      time_shf=(int)(tmp/96);
      tick_shf=(char)(tmp%96);
      load_track();
      track=track_node;
      find_start_track();
      while(!endof_data(track)){
        return_time(track,&time,&tick);
        tmp=(long int)((time+time_shf)*96+((unsigned char)tick+(signed char)tick_shf));
        time=(unsigned int)(tmp/96);
        tick=(unsigned char)(tmp%96);
        *(unsigned char*)(track+2)=tick;
        *(unsigned int*)track=time;
        track+=3;
        while(*track!=0xFF)track++;
        track++;
        }
      save_track();
      }
    release_window(1);
    }

  void drum_machine(void){
    char command[100];
    unsigned char port21h;

    setvect(0x1C,Save_Int1C);
    setvect(0x0A,Save_IRQ2);
    setvect(0x0B,Save_IRQ3);
    mouse_cursor_off();
    reset_mouse();
    closegraph();
    farfree(track_node);
    sprintf(command,"CD %s",option.program_path);
    command[strlen(command)-1]=0;
    system(command);
    sprintf(command,"DRUM");
    system(command);
    track_node=getmem(65535u);
    init_graph();
    reset_mouse();
    program_8253(tempo_rate);
    init_sio();
    Save_Int1C=getvect(0x1C);
    Save_IRQ2=getvect(0x0A);
    Save_IRQ3=getvect(0x0B);
    setvect(0x0B,inc_timer);
    setvect(0x0A,midiin);
    setvect(0x1C,active_sense);
    port21h=inportb(0x21);
    port21h&=0xF3;
    outportb(0x21,port21h);
    main_screen();
    }
  void view(void){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    char command[100];
    unsigned char port21h;

    setvect(0x1C,Save_Int1C);
    setvect(0x0A,Save_IRQ2);
    setvect(0x0B,Save_IRQ3);
    mouse_cursor_off();
    reset_mouse();
    closegraph();
    farfree(track_node);
    sprintf(command,"CD %s",option.program_path);
    command[strlen(command)-1]=0;
    system(command);
    sprintf(command,"TEST");
    system(command);
    track_node=getmem(65535u);
    init_graph();
    reset_mouse();
    program_8253(tempo_rate);
    init_sio();
    Save_Int1C=getvect(0x1C);
    Save_IRQ2=getvect(0x0A);
    Save_IRQ3=getvect(0x0B);
    setvect(0x0B,inc_timer);
    setvect(0x0A,midiin);
    setvect(0x1C,active_sense);
    port21h=inportb(0x21);
    port21h&=0xF3;
    outportb(0x21,port21h);
    main_screen();
    }

void load(void){

  FILE *fp,*fp_temp;
  char fname[100];
  unsigned char ch;
  int i,j,k,track_no,track_num;
  unsigned long int size,c;

  i=load_save(fname,LOAD);
  if(i==0){
    //strcat(fname,name);
    if((fp=fopen(fname,"rb"))==NULL){
      message(200,200,"ERROR"," ERROR Opening File ");
      }
    else{
      sprintf(fname,"IF EXIST %sTRACK.$ DEL %sTRACK.$",option.temporary_path
         ,option.temporary_path);
      system(fname);
      init_data();
      for(i=0;i<20;i++)
        fgetc(fp);
      ch=fgetc(fp);
      if(ch==26){
        for(i=0;i<3;i++){
          j=0;
```

เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            k=0;
            while(j<2){
              ch=fgetc(fp);
              if((ch!=13)&&(ch!=10))
                comment_text[i][k++]=ch;
              else j++;
              }
            comment_text[i][k++]=0;
            }
          ch=fgetc(fp);
          if(ch==165){
            fread(&file_header,sizeof(file_header),1,fp);
            save_header();
            program_8253(tempo_rate);
            fread(&track_num,sizeof(track_num),1,fp);
            for(i=0;i<track_num;i++){
              track_no=(int)fgetc(fp);
              fread(&track_header,sizeof(track_header),1,fp);
              save_track_header(track_no);
              fread(&size,sizeof(size),1,fp);
              sprintf(fname,"%strack.%d",option.temporary_path,track_no);
              fp_temp=fopen(fname,"wb");
              for(c=0;c<size;c++)
                fputc(fgetc(fp),fp_temp);
              fclose(fp_temp);
              }
            }
          }
        else{
          message(185,350,"ERROR","  -- Not MIDI Project File --  ");
          }
        fclose(fp);
        }
      }
    main_screen();
    }

void save(void){
  FILE *fp;
  char fname[80],backup_name[80],command[128];
  int i,c;
  unsigned long int size;

  i=load_save(fname,SAVE);
  if(i==0){
    strcpy(backup_name,fname);
    c=0;
    while(c<strlen(backup_name)&&(backup_name[c]!='.'))c++;
    backup_name[c]=0;
    strcat(backup_name,".BAK");
    sprintf(command,"IF EXIST %s COPY %s %s > NULL",fname,fname,backup_name);
    system(command);
    }
  if((i==0)&&((fp=fopen(fname,"wb"))==NULL)){
    message(200,200," ERROR "," ERROR Opening File ");
    }
  else if(i==0){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      perform_data(true,0,0);
      fprintf(fp,"MIDI Project File.\r\n");
      fputc(26,fp);
      fprintf(fp,"%s\r\n%s\r\n%s\r\n",comment_text[0],comment_text[1],comment_text[2]);
      fputc(165,fp);
      set_header();
      fwrite(&file_header,sizeof(file_header),1,fp); //Save Main Setting
      fwrite(&point_track_in_use,sizeof(point_track_in_use),1,fp);
      for(i=0;i<point_track_in_use;i++){ //Save Track Setting and MIDI Data
        fputc((unsigned char)track_in_use[i],fp);
        set_track_header(track_in_use[i]);
        fwrite(&track_header,sizeof(track_header),1,fp);
        size=1;
        while(!endof_data((unsigned char*)ready_data_node[track_in_use[i]]+(unsigned int)siz
e))size++;
        size+=2;
        fwrite(&size,sizeof(size),1,fp);
        fwrite(ready_data_node[track_in_use[i]],(unsigned int)size,1,fp);
        }
      destroy_data();
      fclose(fp);
      }
    main_screen();
    }

  void new_all(void){
    char *command;
    boolean ok;

    command=getmem(100);
    ok=prompt_execute(320,200,"New","Are you sure to Delete All Track ?");
    if(ok){
      sprintf(command,"IF EXIST %sTRACK.* DEL %sTRACK.*",option.temporary_path
             ,option.temporary_path);
      system(command);
      init_data();
      }
    farfree(command);
    }

  void detail_screen(void){
    int i;

    setfillstyle(1,8);
    bar(20,40,230,190-7*14);
    setcolor(color.normaltext_color);
    for(i=0;i<3;i++){
      outtextxy(25,45+14*i,comment_text[i]);
      }
    }

  void comment(void){
    void far *ptr;
    unsigned char ch;
    int line_no=0,save_color[2];
    ptr=getmem(imagesize(20,40,230,190));
```

เอกสารนี้มีไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    getimage(20,40,230,190,ptr);
    detail_screen();
    save_color[0]=color.readtext_background_color;
    save_color[1]=color.normaltext_color;
    color.readtext_background_color=8;
    color.normaltext_color=7;
    do{
      ch=readtext(25,45+14*line_no,comment_text[line_no],24,24,true);
      if(funckey) switch(ch){
        case UP_KEY  :if(line_no>0)line_no--;break;
        case DOWN_KEY:if(line_no<2)line_no++;break;
        }
      //else if((ch==RETURN_KEY)&&(line_no<2)) line_no++;
      }while((funckey)||((ch!=ESC_KEY)&&(ch!=RETURN_KEY)));
    putimage(20,40,ptr,COPY_PUT);
    farfree(ptr);
    color.readtext_background_color=save_color[0];
    color.normaltext_color=save_color[1];
    }

  void track_volume(void){
    void far *ptr;
    char *temp,ch;
    int x1,y1,x2,y2,v,i;
    boolean error=false;

    x1=100;y1=80;x2=344;y2=140;
    temp=getmem(100);
    ptr=getmem(imagesize(x1,y1,x2+4,y2+4));
    getimage(x1,y1,x2+4,y2+4,ptr);
    setfillstyle(1,color.shadow);
    bar(x1+4,y1+4,x2+4,y2+4);
    frame(x1,y1,x2,y2,false);
    write_text(x1,x2,y1+10,"Volume",color.title_text_color);
    sprintf(temp,"Setting Volume for Track %3d",present_track+1);
    setcolor(color.normaltext_color);
    outtextxy(x1+color.frame_width+8,y1+25,temp);
    outtextxy(x1+color.frame_width+8+80,y1+40,"Enter 0-100");
    sprintf(temp,"%d",track_vol[present_track]);
    ch=readtext(x1+color.frame_width+100+80,y1+40,temp,3,3,false);
    if((strlen(temp)>0)&&(strlen(temp)<4)&&(!funckey)&&(ch==RETURN_KEY)){
      for(i=0;i<strlen(temp);i++)
        if((*(temp+i)<48)||(*(temp+i)>57)) error=true;
      if(!error){
        v=atoi(temp);
        if(v>100) error=true;
        }
      if(!error){
        track_vol[present_track]=v;
        }
      }
    putimage(x1,y1,ptr,COPY_PUT);
    farfree(ptr);
    farfree(temp);
    }
  void track_transpose(void){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    char temp[50],ch;
    boolean check=true;
    int i;

    open_window(320-140,200,320+120,260,1,"Transpose");
    setcolor(color.normaltext_color);
    sprintf(temp,"Set Transpose For Track %3d",present_track+1);
    outtextxy(320-140+20,200+25,temp);
    outtextxy(320-140+36,200+25+14,"Value [-36 - 36]:");
    itoa(track_trans[present_track]-transpose_level,temp,10);
    ch=readtext(320+40,200+25+14,temp,3,3,false);
    if (!(funckey)&&(ch==RETURN_KEY)){
      for(i=0;i<strlen(temp);i++){
        if((i!=0)&&((temp[i]<48)||(temp[i]>57)))
          check=false;
        else if(!(((temp[i]>=48)&&(temp[i]<=57))||(temp[i]=='-')))
          check=false;
        }
      if(check)
        track_trans[present_track]=atoi(temp)+transpose_level;
      else
        message(226,270," ERROR ","  Invalid Number  ");
      }
    release_window(1);
    }

  void clear_track(void){
    boolean ok;
    char command[80];

    sprintf(command,"Clear Track %d",present_track+1);
    ok=prompt_execute(320,220,command,"     Are you sure?     ");
    if(ok){
      sprintf(command,"IF EXIST %s\TRACK.%d DEL %s\TRACK.%d",
         option.temporary_path,present_track,option.temporary_path,present_track);
      system(command);
      }
    }

  void merge(unsigned char *node1,unsigned char *node2,unsigned char *dest_node){
    unsigned char *n1,*n2,*dest,*temp,tick,tick1,tick2,old_tick=0xFF;
    unsigned int time,time1,time2,old_time=0xFFFF;

    n1=node1;
    n2=node2;
    dest=dest_node;
    while(*n1!=0xFF)n1++;n1++;
    while(*n2!=0xFF)
      *dest_node++=*n2++;
    n2++;
    if(endof_data(n1))n1=NULL;
    if(endof_data(n2))n2=NULL;
    while((n1)||(n2)){
      return_time(n1,&time1,&tick1);
      return_time(n2,&time2,&tick2);
      if(n2&&((time1>time2)||((time1==time2)&&(tick1>tick2)))){
        temp=n2;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        time=time2;
        tick=tick2;
        n2=find_next(n2);
        }
      else{
        temp=n1;
        time=time1;
        tick=tick1;
        n1=find_next(n1);
        }
      temp+=3;
      if((time!=old_time)&&(tick!=old_tick)){
        *dest++=0xFF;
        *(unsigned int*)dest=time;
        dest+=2;
        *dest++=tick;
        }
      while(*temp!=0xFF){
        *dest++=*temp++;
        }
      }
    *dest++=0xFF;
    *(unsigned int*)dest=0xFFFF;
    }

  void merge_track(void){
    int save_t,line=0,source,dest;
    char argument[2][6],ch;
    boolean end=false,ok=false;

    strcpy(argument[0],"");
    strcpy(argument[1],"");
    open_window(200,200,420,280,1,"Merge Track");
    setcolor(color.normaltext_color);
    outtextxy(215,225,"Source      Track :");
    outtextxy(215,239,"Destination Track :");
    box(310-70,253,310-6,270,true);
    box(310+6,253,310+70,270,true);
    write_text(310-70,310-6,258,"O.K.",7);
    write_text(310+6,310+70,258,"Cancle",7);
    do{
      if((line==0)||(line==1)){
        ch=readtext(375,225+14*line,argument[line],3,3,true);
        }
      else{
        if(line==2){
          box(310-70,253,310-6,270,true);
          write_text(310-70,310-6,258,"O.K.",color.active_text_color);
          }
        else if(line==3){
          box(310+6,253,310+70,270,true);
          write_text(310+6,310+70,258,"Cancle",color.active_text_color);
          }
        ch=readfunc();
        if(line==2){
          box(310-70,253,310-6,270,true);
          write_text(310-70,310-6,258,"O.K.",7);
```

เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }
      else if(line==3){
        box(310+6,253,310+70,270,true);
        write_text(310+6,310+70,258,"Cancle",7);
        }
      }
    if(funckey)
      switch(ch){
        case SHF_TAB_KEY :if(line>0)line--;
                          else line=3;
                          break;
        case UP_KEY      :if(line>0)line--;
                          else line=3;
                          break;
        case DOWN_KEY    :if(line<3)line++;
                          else line=0;
                          break;
        }
    else
      switch(ch){
        case TAB_KEY     :if(line<3)line++;
                          else line=0;
                          break;
        case RETURN_KEY  :if(line==2){
                            end=true;
                            ok=true;
                            box(310-70,253,310-6,270,false);
                            write_text(310-70,310-6,258,"O.K.",color.active_text_color);
                            delay(200);
                            box(310-70,253,310-6,270,true);
                            write_text(310-70,310-6,258,"O.K.",7);
                            delay(50);
                            }
                          else if(line==3){
                            end=true;
                            box(310+6,253,310+70,270,false);
                            write_text(310+6,310+70,258,"Cancle",color.active_text_color);
                            delay(200);
                            box(310+6,253,310+70,270,true);
                            write_text(310+6,310+70,258,"Cancle",7);
                            delay(50);
                            }
                          else line++;
                          break;
        case ESC_KEY     :end=true;
                          box(310+6,253,310+70,270,false);
                          write_text(310+6,310+70,258,"Cancle",color.active_text_color);
                          delay(200);
                          box(310+6,253,310+70,270,true);
                          write_text(310+6,310+70,258,"Cancle",color.normaltext_color);
                          delay(50);
                          break;
        }
    }while(!end);
  release_window(1);
  if(ok){
    save_t=present_track;
```

เอกสารนี้จัดเก็บไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    source=atoi(argument[0])-1;
    dest=atoi(argument[1])-1;
    present_track=source;
    perform_data(true,0,0);
    if(ready_data[source]==NULL)
      *(unsigned long int*)ready_data_node[source]=0xFFFFFFFFul;
    if(ready_data[dest]==NULL)
      *(unsigned long int*)ready_data_node[dest]=0xFFFFFFFFul;
    merge(ready_data_node[source],ready_data_node[dest],track_node);
    track=track_node;
    re_channel(track,track_ch[source]);
    save_track();
    destroy_data();
    present_track=save_t;
    }
  }

void track_comment(void){
  }

void re_channel(unsigned char *data,unsigned char channel){
  unsigned char *temp;

  temp=data;
  while(*temp!=0xFF)temp++;
  temp++;
  while(!endof_data(temp)){
    temp+=3;
    do{
      if((*temp&0x80)==0x80){
        *temp=(*temp&0xF0)|(channel%16);
        temp++;
        }
      else
        do{
          temp++;
          }while((*temp&0x80)!=0x80);
      }while(*temp!=0xFF);
    temp++;
    }
  }

void assign_channel(void){
  char text[40];
  void far *ptr;
  unsigned char ch;
  boolean ok=true;
  int i;

  ptr=getmem(imagesize(200,200,444,284));
  getimage(200,200,444,284,ptr);
  setfillstyle(1,color.shadow);
  bar(204,204,444,284);
  frame(200,200,440,280,false);
  write_text(200,440,210,"Assign Channel",color.title_text_color);
  setcolor(color.normaltext_color);
  sprintf(text,"Assign Track %d To",present_track+1);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    outtextxy(245,230,text);
    sprintf(text,"Channel [1-%d]",cardtype*16+16);
    outtextxy(240,250,text);
    text[0]=0;
    ch=readtext(360,250,text,2,2,false);
    for(i=0;i<strlen(text);i++)
      if((text[i]<48)||(text[i]>57))ok=false;
    if((ok)&&(ch==RETURN_KEY)){
      if((atoi(text)>0)&&(atoi(text)<cardtype*16+16)){
        track_ch[present_track]=atoi(text)-1;
        load_track();
        re_channel(track_node,track_ch[present_track]);
        save_track();
        }
      }

    //readfunc();
    putimage(200,200,ptr,COPY_PUT);
    farfree(ptr);
    }

  void voice_select(void){
    int no;
    char name[16];

    lib_browser(&no,name,100,50);
    if(no!=-1){
      track_vcs[present_track]=(char)no;
      strcpy(track_name[present_track],name);
      midiout(track_ch[present_track]/16,Pgm_chng|(track_ch[present_track]%16));
      midiout(track_ch[present_track]/16,track_vcs[present_track]);
      }
    }

  void assign_track(void){
    void far *ptr;
    char track_no[4];
    int i;
    boolean err;

    err=false;
    itoa(present_track+1,track_no,10);
    ptr=getmem(imagesize(220,200,420,250));
    getimage(220,200,420,250,ptr);
    frame(220,200,420,250,false);
    write_text(220,420,210,"Present Track",color.title_text_color);
    setcolor(color.normaltext_color);
    outtextxy(238,228,"Enter Track No.");
    readtext(366,228,track_no,3,3,false);
    for(i=0;i<strlen(track_no);i++){
      if((track_no[i]<48)||(track_no[i]>57))
        err=true;
      }
    i=atoi(track_no);
    if((!err)&&(i>=1)&&(i<129))
      present_track=i-1;
    else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      message(220,280,"Error","Invalid Track Number");
    putimage(220,200,ptr,COPY_PUT);
    farfree(ptr);
    }

  void keyboard(int x,int y,int style,int color){
    int numpoint,polypoint[16];

    polypoint[1]=y    ;polypoint[0]=x+2;
    polypoint[3]=y+25 ;polypoint[2]=x+2;
    polypoint[5]=y+25 ;polypoint[4]=x;
    polypoint[7]=y+40 ;polypoint[6]=x;
    polypoint[9]=y+40 ;polypoint[8]=x+9;
    polypoint[11]=y+25;polypoint[10]=x+9;
    polypoint[13]=y+25;polypoint[12]=x+6;
    polypoint[15]=y   ;polypoint[14]=x+6;
    switch (style) {
          case DO     :polypoint[12]=x+9;polypoint[14]=x+9;break;
          case ME     :polypoint[0]=x;polypoint[2]=x;
          }
    numpoint = 8;
    setcolor(0);
    setfillstyle(1,color);
    fillpoly(numpoint,polypoint);
  }

  void one_octave(int x_pos,int y_pos){

    keyboard(x_pos   ,y_pos,2,15);
    keyboard(x_pos+11,y_pos,0,15);
    keyboard(x_pos+22,y_pos,1,15);
    keyboard(x_pos+33,y_pos,2,15);
    keyboard(x_pos+44,y_pos,0,15);
    keyboard(x_pos+55,y_pos,0,15);
    keyboard(x_pos+66,y_pos,1,15);
  }

  void seven_octave(int x_pos,int y_pos){
    int i,j,save_color;
    struct fillsettingstype fill_info;

    save_color=getcolor();
    getfillsettings(&fill_info);
    setfillstyle(1,7);
    setcolor(0);
    bar(x_pos-40,y_pos-40,x_pos+558,y_pos+44);
    setfillstyle(1,0);
    bar(x_pos-1,y_pos-1,x_pos+549,y_pos+40);
    for(i=0;i<=6;i++){
      one_octave(x_pos+77*i,y_pos);
      }
    setfillstyle(1,15);
    bar(x_pos+539,y_pos,x_pos+548,y_pos+39);
    setfillstyle(1,8);
    bar(x_pos-26,y_pos+10,x_pos-21,y_pos+35);
    bar(x_pos-15,y_pos+10,x_pos-10,y_pos+35);
    setcolor(0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    write_text(x_pos+30,x_pos+30,y_pos-20,"MIDI Project",15);
    setfillstyle(1,14);
    bar(x_pos+100,y_pos-30,x_pos+294,y_pos-9);
    setcolor(15);
    line(x_pos+100,y_pos-9,x_pos+294,y_pos-9);
    line(x_pos+294,y_pos-30,x_pos+294,y_pos-9);
    setcolor(8);
    line(x_pos+100,y_pos-30,x_pos+294,y_pos-30);
    line(x_pos+100,y_pos-30,x_pos+100,y_pos-9);
    setfillstyle(1,8);
    for(i=0;i<=2;i++)
      for(j=0;j<=8;j++)
        if((j!=2)&&(j!=7)) bar(x_pos+330+j*15,y_pos-28+i*7,
                               x_pos+340+j*15,y_pos-25+i*7);
    setcolor(15);
    line(x_pos-40,y_pos-40,x_pos-40 ,y_pos+44);
    line(x_pos-39,y_pos-39,x_pos-39 ,y_pos+43);
    line(x_pos-40,y_pos-40,x_pos+558,y_pos-40);
    line(x_pos-39,y_pos-39,x_pos+557,y_pos-39);
    setcolor(8);
    line(x_pos-40 ,y_pos+44,x_pos+558,y_pos+44);
    line(x_pos-39 ,y_pos+43,x_pos+557,y_pos+43);
    line(x_pos+558,y_pos-40,x_pos+558,y_pos+44);
    line(x_pos+557,y_pos-39,x_pos+557,y_pos+43);
    setfillstyle(fill_info.pattern,fill_info.color);
    setcolor(save_color);
    }



void step_mode(void){

    box(585,113,627,135,true);
    setfillstyle(1,0);
    setcolor(0);
    pieslice(606,127,0,360,2);
    line(608,119,608,127);
    box(585,138,627,160,true);
    pieslice(606,127+25,0,360,2);
    line(608,144,608,152);
    setcolor(12);
    circle(606,149,7);
    line(600,152,612,146);
    box(585,163,627,180,true);
    setcolor(8);
    outtextxy(590,168,"Move");
    box(585,183,627,200,true);
    outtextxy(590,188,"Copy");
    box(585,203,627,220,true);
    outtextxy(594,208,"Cut");
    box(585,228,627,245,true);
    outtextxy(590,233,"Prog");
    box(585,248,627,265,true);
    outtextxy(590,253,"Bend");
    box(585,268,627,285,true);
    outtextxy(590,273,"Pres");
    box(585,288,627,305,true);
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น รวมทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    outtextxy(590,293,"Whel");
    box(585,313,627,330,true);
    outtextxy(594,318,"Vel");
    box(585,338,627,355,true);
    outtextxy(590,343,"On_V");
    }

  int edit_mode_chng(int y_mouse){
    int command=-1;

    if((y_mouse>113)&&(y_mouse<135)){
      command=Put_note;
      }
    else if((y_mouse>138)&&(y_mouse<160)){
      command=Del_note;
      }
    else if((y_mouse>163)&&(y_mouse<180)){
      command=Move;
      }
    else if((y_mouse>183)&&(y_mouse<200)){
      command=Copy;
      }
    else if((y_mouse>203)&&(y_mouse<220)){
      command=Cut;
      }
    else if((y_mouse>228)&&(y_mouse<245)){
      command=Prog;
      }
    else if((y_mouse>248)&&(y_mouse<265)){
      command=Bend;
      }
    else if((y_mouse>268)&&(y_mouse<285)){
      command=Pressure;
      }
    else if((y_mouse>288)&&(y_mouse<305)){
      command=Whell;
      }
    else if((y_mouse>313)&&(y_mouse<330)){
      command=Chng_velocity;
      }
    return(command);
    }

  void draw_keyboard(void){
    int i,j;

    setfillstyle(1,15);
    bar(10,82,70,451);
    setfillstyle(1,15);
    bar(100,82,100+(noteperbar+1)*96,451);
    setcolor(0);
    rectangle(10,82,70,451);
    rectangle(99,82,101+(noteperbar+1)*96,451);
    for(i=0;i<61;i++){
      j=i%12;
      setfillstyle(1,0);
      setcolor(0);
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      if((j==4)||(j==11)){
        line(10,445-i*6,70,445-i*6);
        }
      if((j==1)||(j==3)||(j==6)||(j==8)||(j==10)){
        line(10,448-i*6,70,448-i*6);
        bar(10,450-i*6,50,445-i*6);
        }
      setfillstyle(1,7);
      setcolor(7);
      if((j==4)||(j==11)){
        line(100,445-i*6,100+(noteperbar+1)*96,445-i*6);
        }
      if((j==1)||(j==3)||(j==6)||(j==8)||(j==10)){
        line(100,448-i*6,100+(noteperbar+1)*96,448-i*6);
        bar(100,450-i*6,100+(noteperbar+1)*96,445-i*6);
        }
      }
    setcolor(8);
    line(147,82,147,451);
    line(148,82,148,451);
    line(147+96*noteperbar,82,147+96*noteperbar,451);
    line(148+96*noteperbar,82,148+96*noteperbar,451);
    setcolor(7);
    for(i=1;i<noteperbar;i++){
      line(148+i*96,82,148+i*96,451);
      }
    }

  void key_step(void){
    int i;
    char *temp;

    temp=getmem(100);
    setcolor(8);
    line(148,68,148+noteperbar*96,68);
    for(i=0;i<=noteperbar;i++)
      line(148+i*96,68,148+i*96,78);
    for(i=0;i<noteperbar;i++)
      line(196+i*96,68,196+i*96,75);
    for(i=0;i<noteperbar*2;i++)
      line(172+i*48,68,172+i*48,72);
    for(i=0;i<noteperbar*4;i++)
      line(160+i*24,68,160+i*24,70);
    /*
    for(i=0;i<4;i++)
      line(196+i*96,68,196+i*96,75);
    for(i=0;i<8;i++)
      line(172+i*48,68,172+i*48,72);
    for(i=0;i<16;i++)
      line(160+i*24,68,160+i*24,70);
    */
    box(10,62,70,78,true);
    setcolor(8);
    outcharxy(38,67,30);
    box(10,455,70,471,true);
    outcharxy(38,460,31);
    setfillstyle(1,9);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    bar(120,455,80+(noteperbar+1)*96,471);
    setcolor(15);
    rectangle(120,455,80+(noteperbar+1)*96,471);
    setcolor(color.shadow);
    line(120,455,120,471);
    line(120,455,80+(noteperbar+1)*96,455);
    box(100,455,116,471,true);
    setcolor(8);
    outcharxy(105,460,17);
    box(84+(noteperbar+1)*96,455,100+(noteperbar+1)*96,471,true);
    outcharxy(89+(noteperbar+1)*96,460,16);
    setfillstyle(1,0);
    setcolor(color.normaltext_color);
    bar(10,43,100,57);
    sprintf(temp,"Track %3d",present_track+1);
    outtextxy(18,46,temp);
    bar(108,43,244,57);
    outtextxy(112,46,track_name[present_track]);
    //outtextxy(112,46,vcs_lib[track_vcs[present_track]]->name[track_vcs[present_track]]);
    bar(542,41,630,57+14+8);
    outtextxy(550,44,   "Bar     1");
    outtextxy(550,56,"Note    0");
    outtextxy(550,68,"Tick    0");
    bar(78,62,142,78);
    outtextxy(86,66,"Key  0");
    setfillstyle(1,15);
    bar(248,41,538,65);
    setcolor(0);
    rectangle(248,41,538,65);
    setfillstyle(1,8);
    bar(252,46,316,60);
    setcolor(12);
    outcharxy(256,50,16);
    setcolor(14);
    outtextxy(272,50,"00000");
    for(i=0;i<5;i++)
      box(319+i*44,44,359+i*44,62,true);
    setcolor(8);
    temp[0]='I';temp[1]=17;temp[2]=17;temp[3]=0;// Skip back
    outtextxy(327,50,temp);
    temp[0]=16;temp[1]=0;                       // Play
    outtextxy(379,50,temp);
    temp[0]=17;temp[1]=17;temp[2]=0;            // Rewind
    outtextxy(419,50,temp);
    temp[0]=16;temp[1]=16;                      // Forward
    outtextxy(463,50,temp);
    temp[0]='■';temp[1]=0;                      // Stop
    outtextxy(511,50,temp);
    setcolor(12);
    outtextxy(590,88,"Note");
    step_mode();
    farfree(temp);
    }

void return_note(char no,char *note){
    if(no>=0){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      switch(no%12){
        case  0:strcpy(note,"C   ");break;
        case  1:strcpy(note,"C#  ");break;
        case  2:strcpy(note,"D   ");break;
        case  3:strcpy(note,"D#  ");break;
        case  4:strcpy(note,"E   ");break;
        case  5:strcpy(note,"F   ");break;
        case  6:strcpy(note,"F#  ");break;
        case  7:strcpy(note,"G   ");break;
        case  8:strcpy(note,"G#  ");break;
        case  9:strcpy(note,"A   ");break;
        case 10:strcpy(note,"A#  ");break;
        case 11:strcpy(note,"B   ");break;
        }
      if(no<60){
        note[2]='-';
        note[3]=(char)((59-no)/12+49);
        }
      else{
        note[3]=(char)((no-60)/12+48);
        }
      }
    else
      strcpy(note,"Err.");
    }

  int check_mouse(void){
    int command,x_mouse,y_mouse,i;
    char temp[40];

    mouse_position(&x_mouse,&y_mouse);
    command=-1;
    if((x_mouse>10)&&(x_mouse<70)&&(y_mouse>62)&&(y_mouse<78)){
      // Keyboard Shift UP is pressed
      box(10,62,70,78,false);
      setcolor(12);
      outcharxy(40,67,30);
      do{}while(leftb_pressed());
      box(10,62,70,78,true);
      setcolor(8);
      outcharxy(38,67,30);
      command=Shift_up;
      }
    else if((x_mouse>10)&&(x_mouse<70)&&(y_mouse>455)&&(y_mouse<471)){
      // Keyboard Shift DOWN is pressed
      box(10,455,70,471,false);
      setcolor(12);
      outcharxy(40,460,31);
      do{}while(leftb_pressed());
      box(10,455,70,471,true);
      setcolor(8);
      outcharxy(38,460,31);
      command=Shift_dn;
      }
    else if((x_mouse>100)&&(x_mouse<116)&&(y_mouse>445)&&(y_mouse<471)){
      box(100,455,116,471,false);
      setcolor(12);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
   outcharxy(105,460,17);
   do{}while(leftb_pressed());
   box(100,455,116,471,true);
   setcolor(8);
   outcharxy(105,460,17);
   command=Left_bar;
   }
 else if((x_mouse>84+(noteperbar+1)*96)&&(x_mouse<100+(noteperbar+1)*96)
       &&(y_mouse>445)&&(y_mouse<471)){
   box(84+(noteperbar+1)*96,455,100+(noteperbar+1)*96,471,false);
   setcolor(12);
   outcharxy(89+(noteperbar+1)*96,460,16);
   do{}while(leftb_pressed());
   box(84+(noteperbar+1)*96,455,100+(noteperbar+1)*96,471,true);
   setcolor(8);
   outcharxy(89+(noteperbar+1)*96,460,16);
   command=Right_bar;
   }
 else if((y_mouse>44)&&(y_mouse<62)){
   for(i=0;i<5;i++)
     if((x_mouse>319+i*44)&&(x_mouse<359+i*44)){
       box(319+i*44,44,359+i*44,62,false);
       setcolor(12);
       switch(i){
         case 0:temp[0]='I';temp[1]=17;temp[2]=17;temp[3]=0;// Skip back
                outtextxy(328,50,temp);
                break;
         case 1:temp[0]=16;temp[1]=0;                       // Play
                outtextxy(380,50,temp);
                break;
         case 2:temp[0]=17;temp[1]=17;temp[2]=0;            // Rewind
                outtextxy(420,50,temp);
                break;
         case 3:temp[0]=16;temp[1]=16;temp[2]=0;            // Forward
                outtextxy(464,50,temp);
                break;
         case 4:temp[0]='■';temp[1]=0;                      // Stop
                outtextxy(512,50,temp);
                break;
         }
       do{}while(leftb_pressed());
       box(319+i*44,44,359+i*44,62,true);
       setcolor(8);
       switch(i){
         case 0:temp[0]='I';temp[1]=17;temp[2]=17;temp[3]=0;// Skip back
                outtextxy(327,50,temp);
                command=Skip_back;
                break;
         case 1:temp[0]=16;temp[1]=0;                       // Play
                outtextxy(379,50,temp);
                command=Play;
                break;
         case 2:temp[0]=17;temp[1]=17;temp[2]=0;            // Rewind
                outtextxy(419,50,temp);
                command=Rewind;
                break;
         case 3:temp[0]=16;temp[1]=16;temp[2]=0;            // Forward
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                outtextxy(463,50,temp);
                command=Forward;
                break;
          case 4:temp[0]='■';temp[1]=0;                    // Stop
                outtextxy(511,50,temp);
                command=Stop;
                break;
          }
        i=5;
      }
    }
  else if((x_mouse>10)&&(x_mouse<70)&&(y_mouse>82)&&(y_mouse<451)){
    command=Example;
    }
  else if((x_mouse>585)&&(x_mouse<627)){
    command=Mode;
    }
  return(command);
  }

void update_screen(unsigned long int bar_no,int key_shift){
  int i,x1,y1,x2,y2;
  unsigned int note;unsigned char tick,*start,*save_track,*temp_ptr;

  draw_keyboard();
  setfillstyle(1,9);
  bar(120,455,80+(noteperbar+1)*96,471);
  save_track=track;
  track=track_node;
  do{
    track++;
    }while(*(track-1)!=0xFF);
  start=track;
  for(i=0;i<128;i++){
    note_on[i][0]=-1;
    }
  if(!endof_data(track)){
    return_time(track,&note,&tick);
    if(note<bar_no*noteperbar)
      do{
        temp_ptr=track;
        temp_ptr+=3;
        if(*temp_ptr!=0xFF);
        do{
          switch((*temp_ptr)&0xF0){
            case Note_on  :temp_ptr++;
                           do{
                             note_on[*temp_ptr][0]=note;
                             note_on[*temp_ptr][1]=tick;
                             temp_ptr+=2;
                             }while(((*temp_ptr)&0x80)!=0x80);
                           break;
            case Note_off :temp_ptr++;
                           do{
                             note_on[*temp_ptr][0]=-1;
                             temp_ptr+=2;
                             }while(((*temp_ptr)&0x80)!=0x80);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        break;
          default      :if(*temp_ptr!=0xFF)
                          do{
                            temp_ptr++;
                            }while(((*temp_ptr)&0x80)!=0x80);
                        break;
          }
         }while(*temp_ptr!=0xFF);
       track=find_next(track);
       if(track) return_time(track,&note,&tick);
       }while((track)&&(note*96+tick<(bar_no*noteperbar-1)*96+48));
    }
  track=save_track;

  if(bar_no!=0) track=find_time(track,start,(unsigned int)(bar_no*noteperbar-1),48,false);
  else track=find_time(track,start,0,0,false);
  return_time(track,&note,&tick);

  if((!endof_data(track))&&(note<(bar_no+1)*noteperbar)){
    do{
      track+=3;
      do{
        switch(*track&0xF0){
          case Note_on  :track++;
                         note_on[*(track)][0]=note;
                         note_on[*(track)][1]=tick;
                         track+=2;
                         break;
          case Note_off :track++;
                         y1=447-1-(*track-36)*6+key_shift*72;
                         y2=447+2-(*track-36)*6+key_shift*72;
                         x1=(int)((note_on[(int)*track][0]-bar_no*noteperbar)*96+note_on[(
int)*track][1]+148);
                         if(x1<100)
                          x1=100;
                         x2=(int)((note-noteperbar*bar_no)*96+tick+148);
                         if(x2>100+96*(noteperbar+1))x2=100+96*(noteperbar+1);
                         if((y2>82)&&(y1<451)){
                           setfillstyle(1,9);
                           bar(x1,y1,x2,y2);
                           }
                         note_on[*track][0]=-1;
                         track+=2;
                         break;
          case Pgm_chng :track+=2;
                         setcolor(15);
                         outtextxy((int)(144+96*(note-noteperbar*bar_no)+tick),460,"P");
                         break;
          default       :do{
                           track++;
                           }while(((*track)&0x80)!=0x80);
                         break;
          }
        }while(*track!=0xFF);
      track++;
      return_time(track,&note,&tick);
      }while((note*96+tick<(bar_no+1)*96*noteperbar+48)&&(!endof_data(track)));
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }
  setfillstyle(1,9);
  for(i=0;i<128;i++)
    if(note_on[i][0]!=-1){
      y1=447-1-(i-36)%6+key_shift%72;
      y2=447+2-(i-36)%6+key_shift%72;
      x1=(int)((note_on[i][0]-bar_no%noteperbar)%96+note_on[i][1]+148);
      if(x1<100) x1=100;
      if((y2>82)&&(y1<451))
        bar(x1,y1,148+96%noteperbar+48,y2);
      }
  track=save_track;
  }

void update_counter(unsigned long int counter){
  char counter_text[6];
  int i;

  sprintf(counter_text,"%5u",counter);
  for(i=0;i<5;i++)
    if(counter_text[i]==' ')counter_text[i]='0';
  setfillstyle(1,8);
  bar(252,46,316,60);
  if(counter%2==0)
    setcolor(12);
  else
    setcolor(8);
  outcharxy(256,50,16);
  setcolor(14);
  outtextxy(272,50,counter_text);
  }

void keypressed(int note_no,int key_shift){
  int x,y1,y2,col;

  y1=447-1-(note_no-36)%6+key_shift%72;
  y2=447+2-(note_no-36)%6+key_shift%72;
  if((note_no%12==1)||(note_no%12==3)||(note_no%12==6)||(note_no%12==8)||(note_no%12==10))
{
    x=40;
    col=15;
    }
  else{
    x=60;
    col=0;
    }
  setfillstyle(1,col);
  if((y1>82)&&(y2<451))
    bar(x,y1,x+3,y2);
  }

void unpressed(int note_no,int key_shift){
  int x,y1,y2,col;
  y1=447-1-(note_no-36)%6+key_shift%72;
  y2=447+2-(note_no-36)%6+key_shift%72;
  if((note_no%12==1)||(note_no%12==3)||(note_no%12==6)||(note_no%12==8)||(note_no%12==10))
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
    x=40;
    col=0;
    }
  else{
    x=60;
    col=15;
    }
  setfillstyle(1,col);
  if((y1>82)&&(y2<451))
    bar(x,y1,x+3,y2);
  }

void draw_VU_step(void){
  int i;

  box(593,380,621,473,true);
  setcolor(12);
  outtextxy(600,390,"VU");
  setfillstyle(1,8);
  bar(604-4,406-4,611+4,465+4);
  setfillstyle(1,0);
  for(i=0;i<8;i++){
    bar(604,406+8*i,611,409+8*i);
    }
  }

void check_VU_value(unsigned char value){
  int temp;

  if(value>110)      temp=8;
  else if(value> 95) temp=7;
  else if(value> 80) temp=6;
  else if(value> 68) temp=5;
  else if(value> 54) temp=4;
  else if(value> 38) temp=3;
  else if(value> 16) temp=2;
  else if(value>  1) temp=1;
  else temp=0;

  if(temp>VU_reach)VU_reach=temp;
  }

void interrupt ( *Save_IC_VU)(void);

void update_VU_step(void){
  int i;

  if(VU_value<VU_reach){
    for(i=VU_value+1;i<=VU_reach;i++){
      if(++VU_value>4) setfillstyle(1,LIGHTRED);
      else setfillstyle(1,LIGHTGREEN);
      bar(604,406+8*(8-VU_value),611,409+8*(8-VU_value));
      }
    if((VU_value==VU_reach)&&(VU_reach>0)) VU_reach--;
    }
  else if(VU_value>VU_reach){
```

เอกสารนี้สงวนลิขสิทธิ์ ใช้ได้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น มีความผิดให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    setfillstyle(1,0);
    bar(604,406+8*(8-VU_value),611,409+8*(8-VU_value));
    VU_value--;
    }
  update_enable=false;
  }

void interrupt update_VU(__CPPARGS){
  update_VU_step();
  Save_1C_VU();
  }

void install_VU(void){
  Save_1C_VU=getvect(0x1C);
  setvect(0x1C,update_VU);
  }

void uninstall_VU(void){
  setvect(0x1C,Save_1C_VU);
  }

void copy_note(unsigned int copy_time_s,unsigned int copy_time_e,unsigned int time_shift){
  unsigned int time_1,time_2,i,j,c=0;
  unsigned char tick_1,tick_2,old_time=0xFFFFu,old_tick=0,n_on[128];
  unsigned char *ctemp,*ctemp_node,*temp_ptr_node,*temp_ptr,*save_pos;
  boolean found,receive;

  for(i=0;i<128;i++)
    n_on[i]=0;
  time_1=copy_time_s;
  tick_1=0;
  time_2=copy_time_e;
  tick_2=0;
  track=find_time(track,start,time_1,tick_1,false);
  ctemp_node=getmem(16383u);
  ctemp=ctemp_node;
  return_time(track,&time_1,&tick_1);
  *ctemp++=0xFF;
  while((!endof_data(track))&&((time_1<time_2)||((time_1==time_2)&&(tick_1<tick_2)))){
    *(unsigned int*)ctemp=time_1;ctemp+=2;
    *(unsigned char*)ctemp=tick_1;ctemp++;
    track+=3;
    while(*track!=0xFF){
      receive=true;
      if((*track&0xF0)==Note_on)
        n_on[c++]=*(track+1);
      else if(((*track)&0xF0)==Note_off){
        found=false;
        for(i=0;i<c;i++){
          if(*(track+1)==n_on[i]){
            found=true;
            for(j=i;j<c-1;j++){
              n_on[j]=n_on[j+1];
            }
            c--;
          }
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        if(!found){
          receive=false;
          track+=3;
          }
        }
      if(receive)
        *ctemp++=*track++;
      }
    track++;
    *ctemp++=0xFF;
    return_time(track,&time_1,&tick_1);
    }
  for(i=0;i<c;i++){
    save_pos=track;
    found=false;
    do{
      return_time(track,&time_1,&tick_1);
      track+=3;
      do{
        if(((*track&0xF0)==Note_off)&&(*(track+1)==n_on[i]))
          found=true;
        else track++;
        }while((*track!=0xFF)&&(!found));
      track++;
      }while(!found);
      if((old_time==time_1)&&(old_tick==tick_1))
        ctemp--;
      else{
        *(unsigned int*)ctemp=time_1;
        *(unsigned char*)(ctemp+2)=tick_1;
        ctemp+=3;
        }
      *ctemp++=Note_off|(track_ch[present_track]/2);
      *ctemp++=n_on[i];
      *ctemp++=64;
      *ctemp++=0xFF;
    track=save_pos;
    }
  *(unsigned int*)ctemp=0xFFFFu;
  ctemp=ctemp_node+1;
  while(!endof_data(ctemp)){
    return_time(ctemp,&time_1,&tick_1);
    time_1+=(unsigned int)time_shift;
    *(unsigned int*)ctemp=time_1;
    ctemp+=3;
    while(*ctemp!=0xFF){ctemp++;}
    ctemp++;
    }
  track=track_node;
  temp_ptr_node=getmem(65535u);
  temp_ptr=temp_ptr_node;
  while(!endof_data(track))
    *temp_ptr++=*track++;
  *(unsigned int*)temp_ptr=0xFFFFu;
  track=track_node;
  merge(ctemp_node,temp_ptr_node,track);
  track=track_node;
```

เอกสารนี้ เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  while(*track!=0xFF)track++;
  track++;
  start=track;
  farfree(temp_ptr_node);
  farfree(ctemp_node);
  }

void step_record(void){
  int i,x_mouse,y_mouse,x1,y1,x2,y2,xmax;
  unsigned char set_velocity[9]={1,16,32,48,64,80,95,110,127};
  char  force[9][4]={"ppp","pp","p","mp","m","mf","f","ff","fff"};
  void far *ptr1,*ptr2,*ptr3,*ptr4,*ptr5,*ptr6;
  int bar_count=0,note_count=0,int_tick=0,key_shift=0,pgm_no;
  char note_no=60,temp[20],ch;
  unsigned char channel,trans,noteon_velocity,*save_pos,*temp_ptr;
  unsigned int time_1,time_2,copy_time_s=0,copy_time_e=4;
  unsigned int time_shift;
  unsigned char tick_1,tick_2;
  unsigned int counter=0;boolean stop=false;
  boolean  chk_del,rew,exit;
  int note_on_no;

  exit=false;
  edit_mode=0;
  channel=track_ch[present_track];
  trans=track_trans[present_track];
  noteon_velocity=set_velocity[6];
  setfillstyle(1,0);
  bar(585,358,627,370);
  setcolor(15);
  outtextxy(606-4*strlen(force[6]),360,force[6]);// f
  for(i=0;i<128;i++){
    note_on[i][0]=-1;
    track_present_pgm[i]=track_vcs[i];
    }
  key_step();
  perform_data(false,0,0);
  ptr1=getmem(imagesize(0,0,639,0));
  ptr2=getmem(imagesize(0,0,0,479));
  load_track();
  find_start_track();
  start=track;
  set_mouse_position(200,250);
  x_mouse=148+note_count*96+int_tick;
  y_mouse=447-(note_no-36)*6-key_shift*72;
  set_mouse_position(x_mouse,y_mouse);
  setfillstyle(1,12);
  setcolor(12);
  pieslice(606,127,0,360,2);
  line(608,119,608,127);
  draw_VU_step();
  VU_reach=0;
  update_screen(0,key_shift);
  popup_menu[0].x_pos=220;
  popup_menu[0].y_pos=150;
  do{
    mouse_position(&x_mouse,&y_mouse);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if((x_mouse>=148)&&(x_mouse<100+(noteperbar)%96+48)
     &&(y_mouse>=85)&&(y_mouse<451)&&(!kbhit()))
  do{
    if(rightb_pressed()){
      do{}while(rightb_pressed());
      ch=pull_down(&step_menu);
      do{}while(clicked());
      if((!funckey)&&(ch==RETURN_KEY)){
        switch(step_menu.command_no){
          case 0:break;// Load Track
          case 1:break;// Save Track as
          case 2:bar_count=(unsigned)counter/noteperbar;
                 update_screen(bar_count,key_shift);
                 sprintf(temp,"%5d",bar_count+1);
                 setfillstyle(1,0);
                 bar(542+32,44,630,52);
                 setcolor(color.normaltext_color);
                 outtextxy(582,44,temp);
                 mouse_position(&x_mouse,&y_mouse);
                 break;// Goto Counter
          case 3:break;// Goto Bar number
          case 4:save_track();
                 switch(popup_menu[0].command_no){
                   case 0:quantize();break;
                   case 1:kill_velocity();break;
                   case 2:kill_whell();break;
                   case 3:kill_pressure();break;
                   case 4:kill_program();break;
                   case 5:kill_control();break;
                   case 6:time_shifter();break;
                   }
                 load_track();
                 find_start_track();
                 start=track;
                 update_screen(bar_count,key_shift);
                 break;// Tool
          case 5:popup_menu[0].x_pos=100;
                 popup_menu[0].y_pos=112;
                 exit=true;
                 break;// Exit
          }
        }
      }
    getimage(10,y_mouse,100+(noteperbar+1)%96,y_mouse,ptr1);
    getimage(x_mouse,68,x_mouse,451,ptr2);
    setcolor(12);
    line(10,y_mouse,100+(noteperbar+1)%96,y_mouse);
    line(x_mouse,68,x_mouse,451);
    setfillstyle(1,0);
    bar(582,54,630,79);
    bar(582,100,630,110);
    note_no=110-((y_mouse-1)/6)+12*key_shift;
    note_count=(int)(x_mouse-148)/96;
    int_tick=(x_mouse-148)%96;
    sprintf(temp,"%d",note_count);
    setcolor(color.normaltext_color);
    outtextxy(614,56,temp);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          sprintf(temp,"%2d",int_tick);
          outtextxy(606,66,temp);
          return_note(note_no,temp);
          outtextxy(590,101,temp);
          do{}while((!moved())&&(!clicked())&&(!kbhit()));
          putimage(10,y_mouse,ptr1,COPY_PUT);
          putimage(x_mouse,68,ptr2,COPY_PUT);
          if(kbhit()){
            ch=readfunc();
            if(funckey){
              y_mouse=y_mouse-y_mouse%6+3;
              x_mouse=x_mouse-x_mouse%6+4;
              switch(ch){
                case UP_KEY   :if(y_mouse>87) y_mouse-=6;break;
                case DOWN_KEY :if(y_mouse<447)y_mouse+=6;break;
                case LEFT_KEY :if(x_mouse>148)x_mouse-=6;break;
                case RIGHT_KEY:if(x_mouse<532)x_mouse+=6;break;
                }
              set_mouse_position(x_mouse,y_mouse);
              }
            else{
              switch(ch){
                case ' '       :save_pos=track;
                                sending_data=true;
                                for(i=0;i<128;i++){
                                  note_on[i][0]=-1;
                                  }
                                note_on_no=0;
                                track=find_time(track,start,bar_count*noteperbar,0,false);
                                return_time(track,&time_1,&tick_1);
                                time_2=noteperbar*bar_count;
                                VU_reach=0;
                                VU_value=0;
                                midiout(track_ch[present_track]/16,Pgm_chng|(track_ch[present
_track]%16));
                                midiout(track_ch[present_track]/16,track_vcs[present_track]);
                                main_timer  = noteperbar*bar_count;
                                main_tick   = 0;
                                while(!kbhit()){
                                  if((time_1<main_timer)||((time_1==main_timer)&&(tick_1<=mai
n_tick))){
                                    track+=3;
                                    do{
                                      switch(*track&0xF0){
                                        case Note_on :midiout(track_ch[present_track]/16,*(tr
ack++));
                                                      do{
                                                        note_on[(int)*track][0]=0;
                                                        note_on_no++;
                                                        keypressed((int)*track,key_shift);
                                                        midiout(track_ch[present_track]/16,tr
ack_trans[present_track]+*(track++));
                                                        check_VU_value(*track);
                                                        midiout(track_ch[present_track]/16,*(
track++));
                                                        }while(((*track)&0x80)!=0x80);
                                                      break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                case Note_off:midiout(track_ch[present_track]/16,*(tr
_ack++));
                                                do{
                                                  note_on[(int)*track][0]=-1;
                                                  if(note_on_no>0)note_on_no--;
                                                  if(note_on_no==0) VU_reach=0;
                                                  unpressed((int)*track,key_shift);
                                                  midiout(track_ch[present_track]/16,tr
ack_trans[present_track]+*(track++));
                                                  midiout(track_ch[present_track]/16,*(
track++));
                                                  }while(((*track)&0x80)!=0x80);
                                                break;
                                default         :do{
                                                  midiout(track_ch[present_track]/16,*(
track++));
                                                  }while(((*track)&0x80)!=0x80);
                                                break;
                                    }
                                  }while(*track!=0xFF);
                                 track++;
                                 return_time(track,&time_1,&tick_1);
                                 }
                                 if(update_enable)update_VU_step();
                                 }
                                if(kbhit()) readfunc();
                                midiout(track_ch[present_track]/16,Ctrl_chng|(track_ch[presen
t_track]%16));
                                midiout(track_ch[present_track]/16,0x7B);
                                midiout(track_ch[present_track]/16,0);
                                sending_data=false;
                                for(i=0;i<128;i++)
                                 if(note_on[i][0]!=-1)
                                   unpressed(i,key_shift);
                               track=save_pos;
                               VU_reach=0;
                               VU_value=0;
                               draw_VU_step();
                               break;
            case 's'          :if(!endof_data(track)){
                                 return_time(track,&time_1,&tick_2);
                                 }
                               else{
                                 save_pos=track;
                                 track=find_pred(track,start);
                                 return_time(track,&time_1,&tick_2);
                                 }
                               save_track();
                               load_track();
                               do{
                                 track++;
                                 }while(*track!=0xFF);
                               track++;
                               find_start_track();
                               start=track;
                               track=find_time(track,start,time_1,tick_1,false);
                               break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      }
     }
    }
   if(leftb_pressed()&&(edit_mode==Put_note)){
    note_no=110-((y_mouse-1)/6)+12*key_shift;
    mouse_position(&x_mouse,&y_mouse);
    x2=x_mouse;
    time_1=(unsigned int)(bar_count*noteperbar+(int)((x_mouse-148)/96));
    tick_1=(unsigned char)((x_mouse-148)%96);
    y1=447-1-(note_no-36)*6+key_shift*72;
    y2=447+2-(note_no-36)*6+key_shift*72;
    x1=x_mouse;
    xmax=x1;
    ptr3=getmem(imagesize(0,0,(noteperbar)*96+96,6));
    ptr4=getmem(imagesize(0,0,8,8));
    ptr5=getmem(imagesize(0,0,8,8));
    ptr6=getmem(imagesize(0,0,8,8));
    getimage(x1,y1,100+(noteperbar)*96+48,y2,ptr3);
    getimage(x1,74,x1+7,81,ptr4);
    setcolor(8);
    outcharxy(x1,74,16);
    getimage(x2-6,74,x2+7-6,81,ptr5);
    setcolor(12);
    outcharxy(x2-6,74,17);
    do{
      i=x_mouse;
      mouse_position(&x_mouse,&y_mouse);
      time_2=(unsigned int)(bar_count*noteperbar+(int)((x_mouse-148)/96));
      tick_2=(unsigned char)((x_mouse-148)%96);
      if((time_2*96+tick_2>time_1*96+tick_1)&&(time_2/noteperbar>bar_count)){
        bar_count++;
        update_screen(bar_count,key_shift);
        x_mouse=148;
        set_mouse_position(x_mouse,y_mouse);
        x1-=96*noteperbar;
        if(x1<100) x1=100;
        sprintf(temp,"%5d",bar_count+1);
        setfillstyle(1,0);
        bar(542+32,44,630,52);
        setcolor(color.normaltext_color);
        outtextxy(582,44,temp);
        xmax=x_mouse;
        getimage(x1,y1,100+(noteperbar)*96+48,y2,ptr3);
        }
      if((x_mouse<148)&&(time_2*96+tick_2>time_1*96+tick_1)){
        bar_count--;
        update_screen(bar_count,key_shift);
        x_mouse=96*noteperbar+147;
        set_mouse_position(x_mouse,y_mouse);
        x1=(time_1-bar_count*noteperbar)*96+tick_1+148;
        if(x1<100) x1=100;
        sprintf(temp,"%5d",bar_count+1);
        setfillstyle(1,0);
        bar(542+32,44,630,52);
        setcolor(color.normaltext_color);
        outtextxy(582,44,temp);
        getimage(x1,y1,100+(noteperbar)*96+48,y2,ptr3);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        xmax=100+(noteperbar)%96+48;
        }
      if(x_mouse>xmax)xmax=x_mouse;
      if((x_mouse!=i)&&(x_mouse>=x1)&&(x_mouse<100+(noteperbar)%96+48)){
        putimage(x2-6,74,ptr5,COPY_PUT);
        x2=x_mouse;
        getimage(x2-6,74,x2+7-6,81,ptr5);
        setcolor(12);
        outcharxy(x2-6,74,17);
        setfillstyle(1,0);
        bar(582,54,630,79);
        note_count=(int)(x2-148)/96;
        int_tick=(x2-148)%96;
        sprintf(temp,"%d",note_count);
        setcolor(color.normaltext_color);
        outtextxy(614,56,temp);
        sprintf(temp,"%2d",int_tick);
        outtextxy(606,66,temp);
        putimage(x1,y1,ptr3,COPY_PUT);
        setfillstyle(1,9);
        bar(x1,y1,x2,y2);
        }
      }while(leftb_pressed());
    if(time_2%96+tick_2>time_1%96+tick_1){
      track=find_time(track,start,time_1,tick_1,true);
      track+=3;
      while(*track!=0xFF){
        track++;
        }
      insert(track,3);
      *(track++)=Note_on|track_ch[present_track];
      *(track++)=note_no;
      *(track++)=noteon_velocity;
      track++;
      track=find_time(track,start,time_2,tick_2,true);
      save_pos=track;
      track+=3;
      while(*track!=0xFF){
        track++;
        }
      insert(track,3);
      *(track++)=Note_off|track_ch[present_track];
      *(track++)=note_no;
      *(track++)=64;
      track=save_pos;
      }
    putimage(x2-6,74,ptr5,COPY_PUT);
    putimage(x1,74,ptr4,COPY_PUT);
    farfree(ptr6);
    farfree(ptr5);
    farfree(ptr4);
    farfree(ptr3);
    }
  else if(leftb_pressed()&&((edit_mode==Del_note)||(edit_mode==Chng_velocity))){
    do{}while(leftb_pressed());
    mouse_position(&x_mouse,&y_mouse);
    time_1=bar_count*noteperbar+(x_mouse-148)/96;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          tick_1=(x_mouse-148)%96;
          track=find_time(track,start,time_1,tick_1,false);
          temp_ptr=track;
          chk_del=false;
          do{
            temp_ptr=find_pred(temp_ptr,start);
            return_time(temp_ptr,&time_1,&tick_1);
            if(!temp_ptr)chk_del=true;
            if((time_1)=bar_count*noteperbar)&&(temp_ptr)){
              save_pos=temp_ptr;
              temp_ptr+=3;
              do{
                switch((*temp_ptr)&0xF0){
                  case Note_off :temp_ptr++;
                                 do{
                                   if(*temp_ptr==note_no)chk_del=true;
                                   temp_ptr+=2;
                                   }while(((*temp_ptr)&0x80)!=0x80);
                                 break;
                  case Note_on  :temp_ptr++;
                                 do{
                                   if(*temp_ptr==note_no){
                                    if(edit_mode==Chng_velocity){
                                     temp_ptr++;
                                     *temp_ptr=noteon_velocity;
                                     chk_del=true;
                                     }
                                    else if(edit_mode==Del_note){
                                     del(temp_ptr,2);
                                     if(((*temp_ptr)&0x80)==0x80){
                                      temp_ptr--;
                                      if((((*temp_ptr)&0xF0)==Note_on)&&(((*(temp_ptr+1
)&0x80)==0x80))
                                       del(temp_ptr,1);
                                      temp_ptr=save_pos;
                                      if(*(temp_ptr+3)==0xFF)
                                        del(temp_ptr,4);
                                      }
                                     // Go to Delete Note_off Event
                                     chk_del=false;
                                     do{
                                       save_pos=temp_ptr;
                                       temp_ptr+=3;
                                       do{
                                         switch(((*temp_ptr)&0xF0)){
                                           case Note_off :temp_ptr++;
                                                          do{
                                                            if(*temp_ptr==note_no){
                                                              del(temp_ptr,2);
                                                              if(((*temp_ptr)&0x80)==0x8
0){
                                                                temp_ptr--;
                                                                if(((*temp_ptr)&0xF0)==N
ote_off)
                                                                  del(temp_ptr,1);
                                                                temp_ptr=save_pos;
                                                                if(*(temp_ptr+3)==0xFF)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                                    del(temp_ptr,4);
                                                   }
                                                  chk_del=true;
                                                  }
                                                 else temp_ptr+=2;
                                                 }while((((*temp_ptr)&0x80)!=
0x80)&&(!chk_del));

                                                 break;
                                        default  :do{
                                                  temp_ptr++;
                                                  }while(((*temp_ptr)&0x80)!=0
x80);

                                                 break;
                                        }
                                       }while((*temp_ptr!=0xFF)&&(!chk_del));
                                      temp_ptr++;
                                      //temp_ptr=find_next(temp_ptr);
                                      }while(!chk_del);
                                     update_screen(bar_count,key_shift);
                                     }
                                    }
                                   else{
                                    temp_ptr+=2;
                                    }
                                   }while((((*temp_ptr)&0x80)!=0x80)&&(!chk_del));
                                  break;
                         default  :do{
                                   temp_ptr++;
                                   }while(((*temp_ptr)&0x80)!=0x80);
                                  break;
                         }
                        }while((*temp_ptr!=0xFF)&&(!chk_del));
                      temp_ptr++;
                      temp_ptr=find_pred(temp_ptr,start);
                      }
                    }while((time_i>=bar_count*noteperbar)&&(!chk_del));
                   track=save_pos;
                   }
                 else if((leftb_pressed())&&(edit_mode==Prog)){
                   mouse_position(&x_mouse,&y_mouse);
                   x1=220;
                   y1=220;
                   x2=420;
                   y2=310;
                   ptr3=getmem(imagesize(x1,y1,x2+4,y2+4));
                   getimage(x1,y1,x2+4,y2+4,ptr3);
                   setfillstyle(1,color.shadow);
                   bar(x1+4,y1+4,x2+4,y2+4);
                   frame(x1,y1,x2,y2,false);
                   write_text(x1,x2,y1+10,"Program Change",color.title_text_color=15);
                   setcolor(color.normaltext_color);
                   outtextxy(x1+10,y1+25,"Bar  No.");
                   outtextxy(x1+10,y1+39,"Note No.");
                   outtextxy(x1+10,y1+53,"Tick No.");
                   sprintf(temp,"%5d",bar_count+1);
                   outtextxy(x1+82,y1+25,temp);
                   sprintf(temp,"%5d",note_count);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            outtextxy(x1+82,y1+39,temp);
            sprintf(temp,"%5d",int_tick);
            outtextxy(x1+82,y1+53,temp);
            write_text(x1,x2,y1+70,"Select Voice",color.normaltext_color);
            lib_browser(&pgm_no,temp,x2+20,100);
            track=find_time(track,start,bar_count*noteperbar+note_count,int_tick,true);
            temp_ptr=track;
            temp_ptr+=3;
            insert(temp_ptr,2);
            *(temp_ptr++)=Pgm_chng|track_ch[present_track];
            *temp_ptr=(unsigned char)(pgm_no);
            putimage(x1,y1,ptr3,COPY_PUT);
            update_screen(bar_count,key_shift);
            farfree(ptr3);
            }
          else if((leftb_pressed())&&(edit_mode==Copy)){
            do{}while(leftb_pressed());
            if(read_time_arg(&copy_time_s,&copy_time_e,&time_shift)){
              copy_note(copy_time_s,copy_time_e,(unsigned int)((unsigned int)time_shift-(uns
igned int)copy_time_s));
              update_screen(bar_count,key_shift);
              }
            }
          mouse_position(&x_mouse,&y_mouse);
        }while((!exit)&&(x_mouse>=148)&&(x_mouse<100+(noteperbar)*96+48)
          &&(y_mouse>=85)&&(y_mouse<451)&&(!clicked())&&(!kbhit()));
      else{
        mouse_cursor_on();
        do{
          mouse_position(&x_mouse,&y_mouse);
          if(leftb_pressed()){
            mouse_cursor_off();
            switch(check_mouse()){
              case Shift_up :if(key_shift<3)key_shift++;
                             setfillstyle(1,0);
                             bar(78+32,62,142,78);
                             sprintf(temp,"%2d",key_shift);
                             setcolor(color.normaltext_color);
                             outtextxy(86+32,66,temp);
                             update_screen(bar_count,key_shift);
                             break;
              case Shift_dn :if(key_shift>-3)key_shift--;
                             setfillstyle(1,0);
                             bar(78+32,62,142,78);
                             sprintf(temp,"%2d",key_shift);
                             setcolor(color.normaltext_color);
                             outtextxy(86+32,66,temp);
                             update_screen(bar_count,key_shift);
                             break;
              case Left_bar :if(bar_count>0)bar_count--;
                             sprintf(temp,"%5d",bar_count+1);
                             setfillstyle(1,0);
                             bar(574+32,44,630,52);
                             setcolor(color.normaltext_color);
                             outtextxy(582,44,temp);
                             update_screen(bar_count,key_shift);
                             break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          case Right_bar:if(bar_count<16384)bar_count++;
                         sprintf(temp,"%5d",bar_count+1);
                         setfillstyle(1,0);
                         bar(574+32,44,630,52);
                         setcolor(color.normaltext_color);
                         outtextxy(582,44,temp);
                         update_screen(bar_count,key_shift);
                         break;
             case Skip_back:counter=0;
                         update_counter(counter);
                         break;
             case Play     :mouse_cursor_on();
                         sending_data=true;
                         save_pos=track;
                         track=start;
                         for(i=0;i<128;i++)
                           note_on[i][0]=-1;
                         find_time_data(counter,0);
                         note_on_no=0;
                         stop=false;
                         track=find_time(track,start,counter,0,false);
                         rew=false;
                         program_8253(tempo_rate);
                         main_timer=(unsigned int)counter;
                         main_tick=0;

                         while(!stop){
                           return_time(track,&time_1,&tick_1);
                           //if((unsigned long)((96*time_1)+tick_1)<=(unsigned long)((96
*main_timer)+main_tick)){
                           if((time_1<main_timer)||((time_1==main_timer)&&(tick_1<=main_
tick))){
                             track+=3;
                             do{
                               switch(*track&0xF0){
                                 case Note_on  :midiout(track_ch[present_track]/16,Pgm_c
hng|(track_ch[present_track]%16));
                                                midiout(track_ch[present_track]/16,track
_vcs[present_track]);
                                                midiout(track_ch[present_track]/16,*(tra
ck++));
                                                do{
                                                  keypressed((int)*track,key_shift);
                                                  note_on[(int)*track][0]=0;
                                                  note_on_no++;
                                                  midiout(track_ch[present_track]/16,tra
ck_trans[present_track]+*(track++));
                                                  check_VU_value((int)*track);
                                                  midiout(track_ch[present_track]/16,(un
signed char)((track_vol[present_track]*(*(track++)))/100));
                                                  }while(((*track)&0x80)!=0x80);
                                                break;
                                 case Note_off :midiout(track_ch[present_track]/16,*(tra
ck++));
                                                do{
                                                  unpressed((int)*track,key_shift);
                                                  note_on[(int)*track][0]=-1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                        if(note_on_no>0)note_on_no--;
                                        if(note_on_no==0) VU_reach=0;
                                        midiout(track_ch[present_track]/16,tra
ck_trans[present_track]+*(track++));
                                        midiout(track_ch[present_track]/16,*(t
rack++));
                                        }while(((*track)&0x80)!=0x80);
                                      break;
                          default      :do{
                                        midiout(track_ch[present_track]/16,*(t
rack++));
                                        }while(((*track)&0x80)!=0x80);
                                      break;
                           }
                         }while(*track!=0xFF);
                       track++;
                       }

                     if(counter!=main_timer){
                       counter=main_timer;
                       update_counter(counter);
                       }

                     if(kbhit()){
                       ch=readfunc();
                       if((!funckey)&&(ch==ESC_KEY)) stop=true;
                       }
                     mouse_position(&x_mouse,&y_mouse);
                     if((x_mouse<319)||(x_mouse>535)||(y_mouse<44)||(y_mouse>62)){
                       if(x_mouse<319) x_mouse=319;
                       if(x_mouse>535) x_mouse=535;
                       if(y_mouse<44)  y_mouse=44;
                       if(y_mouse>62)  y_mouse=62;
                       set_mouse_position(x_mouse,y_mouse);
                       }
                     if(leftb_pressed()){
                       mouse_cursor_off();
                       switch(check_mouse()){
                         case Stop     :stop=true;break;
                         case Forward  :program_8253(250);break;
                         case Rewind   :stop=true;rew=true;break;
                         case Skip_back:counter=0;
                                        update_counter(counter);
                                        stop=true;
                                        break;
                         case Play     :program_8253(tempo_rate);
                         }
                       mouse_cursor_on();
                       }
                     if(update_enable)update_VU_step();
                     send_data();
                     send_out();
                     }
                   draw_VU_step();
                   VU_value=0;
                   VU_reach=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    program_8253(tempo_rate);
                    all_note_off();
                    sending_data=false;
                    for(i=0;i<128;i++)
                      if(note_on[i][0]!=-1)
                        unpressed(i,key_shift);
                    setcolor(12);
                    outcharxy(256,50,16);
                    mouse_cursor_off();
                    if(!rew)break;
  case Rewind   :stop=false;
                    mouse_cursor_on();
                    do{
                      delay(30);
                      if(counter==0)stop=true;
                      else{
                        counter--;
                        update_counter(counter);
                        }
                      mouse_position(&x_mouse,&y_mouse);
                      if((x_mouse<319)||(x_mouse>535)||(y_mouse<44)||(y_mouse>62)){
                        if(x_mouse<319) x_mouse=319;
                        if(x_mouse>535) x_mouse=535;
                        if(y_mouse<44)  y_mouse=44;
                        if(y_mouse>62)  y_mouse=62;
                        set_mouse_position(x_mouse,y_mouse);
                        }
                      if(leftb_pressed()){
                        mouse_cursor_off();
                        if(check_mouse()==Stop)stop=true;
                        mouse_cursor_on();
                        }
                      }while(!stop);
                    mouse_cursor_off();
                    setcolor(12);
                    outcharxy(256,50,16);
                    break;
  case Forward  :stop=false;
                    mouse_cursor_on();
                    do{
                      delay(30);
                      if(counter>65534u)stop=true;
                      else{
                        counter++;
                        update_counter(counter);
                        }
                      mouse_position(&x_mouse,&y_mouse);
                      if((x_mouse<319)||(x_mouse>535)||(y_mouse<44)||(y_mouse>62)){
                        if(x_mouse<319) x_mouse=319;
                        if(x_mouse>535) x_mouse=535;
                        if(y_mouse<44)  y_mouse=44;
                        if(y_mouse>62)  y_mouse=62;
                        set_mouse_position(x_mouse,y_mouse);
                        }
                      if(leftb_pressed()){
                        mouse_cursor_off();
                        if(check_mouse()==Stop)stop=true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    mouse_cursor_on();
                    }
                  }while(!stop);
                mouse_cursor_off();
                setcolor(12);
                outcharxy(256,50,16);
                break;
case Example  :note_no=trans+110-((y_mouse-1)/6)+12*key_shift;
                if(note_no>=0){
                  sending_data=true;
                  midiout(channel/16,Note_on|(channel%16));
                  midiout(channel/16,note_no);
                  midiout(channel/16,noteon_velocity);
                  keypressed(note_no-trans,key_shift);
                  do{}while(leftb_pressed());
                  unpressed(note_no-trans,key_shift);
                  midiout(channel/16,Note_off|(channel%16));
                  midiout(channel/16,note_no);
                  midiout(channel/16,64);
                  sending_data=false;
                  }
                break;
case Mode     :if(edit_mode_chng(y_mouse)!=-1){
                  setcolor(0);
                  switch(edit_mode){
                    case Put_note:setfillstyle(1,0);
                                  pieslice(606,127,0,360,2);
                                  line(608,119,608,127);
                                  break;
                    case Del_note:setfillstyle(1,0);
                                  pieslice(606,127+25,0,360,2);
                                  line(608,144,608,152);
                                  setcolor(12);
                                  circle(606,149,7);
                                  line(600,152,612,146);
                                  break;
                    case Bend    :setcolor(8);
                                  outtextxy(590,253,"Bend");
                                  break;
                    case Whell   :setcolor(8);
                                  outtextxy(590,293,"Whel");
                                  break;
                    case Pressure:setcolor(8);
                                  outtextxy(590,273,"Pres");
                                  break;
                    case Prog    :setcolor(8);
                                  outtextxy(590,233,"Prog");
                                  break;
                    case Copy    :setcolor(8);
                                  outtextxy(590,188,"Copy");
                                  break;
                    case Move    :setcolor(8);
                                  outtextxy(590,168,"Move");
                                  break;
                    case Cut     :setcolor(8);
                                  outtextxy(594,208,"Cut");
                                  break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  case Chng_velocity:
                setcolor(8);
                box(585,313,627,330,true);
                outtextxy(594,318,"Vel");
                break;
  }
switch(edit_mode_chng(y_mouse)){
  case Put_note:box(585,113,627,135,false);
                setfillstyle(1,0);
                setcolor(0);
                pieslice(606,127,0,360,2);
                line(608,119,608,127);
                do{}while(leftb_pressed());
                box(585,113,627,135,true);
                setfillstyle(1,12);
                setcolor(12);
                pieslice(606,127,0,360,2);
                line(608,119,608,127);
                break;
  case Del_note:box(585,138,627,160,false);
                setfillstyle(1,0);
                setcolor(0);
                pieslice(606,127+25,0,360,2);
                line(608,144,608,152);
                setcolor(12);
                circle(606,149,7);
                line(600,152,612,146);
                do{}while(leftb_pressed());
                box(585,138,627,160,true);
                setfillstyle(1,12);
                setcolor(12);
                pieslice(606,127+25,0,360,2);
                line(608,144,608,152);
                setcolor(0);
                circle(606,149,7);
                line(600,152,612,146);
                break;
  case Bend    :box(585,248,627,265,false);
                setcolor(8);
                outtextxy(590,253,"Bend");
                do{}while(leftb_pressed());
                box(585,248,627,265,true);
                setcolor(12);
                outtextxy(590,253,"Bend");
                break;
  case Whell   :box(585,288,627,305,false);
                setcolor(8);
                outtextxy(590,293,"Whel");
                do{}while(leftb_pressed());
                box(585,288,627,305,true);
                setcolor(12);
                outtextxy(590,293,"Whel");
                break;
  case Pressure:box(585,268,627,285,false);
                setcolor(8);
                outtextxy(590,273,"Pres");
                do{}while(leftb_pressed());
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                box(585,268,627,285,true);
                setcolor(12);
                outtextxy(590,273,"Pres");
                break;
        case Prog    :box(585,228,627,245,false);
                setcolor(8);
                outtextxy(590,233,"Prog");
                do{}while(leftb_pressed());
                box(585,228,627,245,true);
                setcolor(12);
                outtextxy(590,233,"Prog");
                break;
        case Copy    :box(585,183,627,200,false);
                setcolor(8);
                outtextxy(590,188,"Copy");
                do{}while(leftb_pressed());
                box(585,183,627,200,true);
                setcolor(12);
                outtextxy(590,188,"Copy");
                break;
        case Move    :box(585,163,627,180,false);
                setcolor(8);
                outtextxy(590,168,"Move");
                do{}while(leftb_pressed());
                box(585,163,627,180,true);
                setcolor(12);
                outtextxy(590,168,"Move");
                break;
        case Cut     :box(585,203,627,220,false);
                setcolor(8);
                outtextxy(594,208,"Cut");
                do{}while(leftb_pressed());
                box(585,203,627,220,true);
                setcolor(12);
                outtextxy(594,208,"Cut");
                break;
        case Chng_velocity:
                box(585,313,627,330,false);
                setcolor(8);
                outtextxy(594,318,"Vel");
                do{}while(leftb_pressed());
                box(585,313,627,330,true);
                setcolor(12);
                outtextxy(594,318,"Vel");
                break;
      }
    edit_mode=edit_mode_chng(y_mouse);
    }
  else if((y_mouse>338)&&(y_mouse<355)){
    box(585,338,627,355,false);
    setcolor(12);
    outtextxy(590,343,"On_V");
    do{}while(leftb_pressed());
    box(585,338,627,355,true);
    setcolor(12);
    outtextxy(590,343,"On_V");
    ch=pull_down(&set_v);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        setcolor(8);
                        outtextxy(590,343,"On_V");
                        do{}while(leftb_pressed());
                        if((!funckey)&&(ch==RETURN_KEY)){
                          if(set_v.command_no!=9){
                            noteon_velocity=set_velocity[set_v.command_no];
                            setfillstyle(1,0);
                            bar(585,358,627,370);
                            setcolor(15);
                            outtextxy(606-4*strlen(force[set_v.command_no]),360,force
[set_v.command_no]);
                            }
                          else{
                            x1=400;x2=560;
                            y1=350;y2=420;
                            ptr3=getmem(imagesize(x1,y1,x2,y2));
                            getimage(x1,y1,x2,y2,ptr3);
                            frame(x1,y1,x2,y2,false);
                            write_text(x1,x2,y1+10,"Velocity",color.title_text_color)
;
                            write_text(x1,x2,y1+25,"Enter Velocity",color.normaltext_
color);
                            sprintf(temp,"%d",noteon_velocity);
                            readtext(x1+40,y1+42,temp,3,3,false);
                            putimage(x1,y1,ptr3,COPY_PUT);
                            //use x2 as temporary
                            x2=atoi(temp);
                            if((x2>0)&&(x2<128)){
                              noteon_velocity=x2;
                              setfillstyle(1,0);
                              bar(585,358,627,370);
                              setcolor(15);
                              outtextxy(606-4*strlen(temp),360,temp);
                              }
                            farfree(ptr3);
                            }
                          }
                        }
                      break;
                    }
                  mouse_cursor_on();
                  setcolor(12);
                  }
                }while(((x_mouse<148)||(x_mouse>=100+(noteperbar)*96+48)
                  ||(y_mouse<=85)||(y_mouse>=451))&&(!clicked())&&(!kbhit()));
              mouse_cursor_off();
              }
            if(kbhit())readfunc();
            }while(!exit);
          do{}while(clicked());
          save_track();
          farfree(ptr2);
          farfree(ptr1);
          destroy_data();
          main_screen();
          }
```

เอกสารนี้จัดทำขึ้นสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
// This routine support Playback and Realtime Record

void vol(int no,int level){
  setcolor(15);
  line(no*40+19-8,465-level/2-4,no*40+20+8,465-level/2-4);
  line(no*40+19-8,465-level/2-3,no*40+20+8,465-level/2-3);
  line(no*40+19-8,465-level/2-4,no*40+19-8,465-level/2+4);
  line(no*40+19-7,465-level/2-4,no*40+19-7,465-level/2+4);
  setcolor(8);
  line(no*40+20+8,465-level/2-4,no*40+20+8,465-level/2+4);
  line(no*40+20+7,465-level/2-3,no*40+20+7,465-level/2+3);
  line(no*40+19-8,465-level/2+4,no*40+20+8,465-level/2+4);
  line(no*40+19-7,465-level/2+3,no*40+20+7,465-level/2+3);
  setfillstyle(1,7);
  bar(no*40+19-8+2,465-level/2-4+2,no*40+20+8-2,465-level/2+4-2);
  setcolor(0);
  line(no*40+19-4,464-level/2,no*40+19+4,464-level/2);
  setcolor(15);
  line(no*40+19-4,465-level/2,no*40+19+4,465-level/2);
  }

void del_vol(int no,int level){
  setfillstyle(1,7);
  bar(no*40+19-8,465-level/2-4,no*40+20+8,465-level/2+4);
  setcolor(8);
  line(no*40+19,465-level/2-4,no*40+19,465-level/2+4);
  setcolor(15);
  line(no*40+20,465-level/2-4,no*40+20,465-level/2+4);
  }

void pan_vol(int no,int level){
  setfillstyle(1,7);
  bar(40*no+19-12,293-8,40*no+19+12,293+7);
  setcolor(0);
  settextstyle(2,0,0);
  outtextxy(40*no+5,292,"L");
  outtextxy(40*no+29,292,"R");
  settextstyle(0,0,0);
  setcolor(15);
  line(no*40+19-12,293,no*40+19+12,293);
  line(no*40+20,290,no*40+20,295);
  setcolor(8);
  line(no*40+19-12,292,no*40+19+12,292);
  line(no*40+19,290,no*40+19,295);
  setfillstyle(1,7);
  bar(no*40+19-2+(level-64)/8,293-8,no*40+19+2+(level-64)/8,293+7);
  setcolor(15);
  line(no*40+19-2+(level-64)/8,293-8,no*40+19+2+(level-64)/8,293-8);
  line(no*40+19-2+(level-64)/8,293-8,no*40+19-2+(level-64)/8,293+7);
  setcolor(8);
  line(no*40+19+2+(level-64)/8,293-8,no*40+19+2+(level-64)/8,293+7);
  line(no*40+19-2+(level-64)/8,293+7,no*40+19+2+(level-64)/8,293+7);
  }

void on_off_switch(int no,int func,boolean push,boolean on){
  int y;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    switch(func){
      case 1:y=320;break;
      case 2:y=266;break;
      case 3:y=242;break;
      }
    setfillstyle(1,7);
    bar(40*no+19-10,y-4,40*no+19+10,y+4);
    if(push)setcolor(15);
    else setcolor(8);
    line(40*no+19-10,y-4,40*no+19-10,y+4);
    line(40*no+19-10,y-4,40*no+19+10,y-4);
    if(push)setcolor(8);
    else setcolor(15);
    line(40*no+19+10,y-4,40*no+19+10,y+4);
    line(40*no+19-10,y+4,40*no+19+10,y+4);
    if(on)setfillstyle(1,12);
    else setfillstyle(1,0);
    bar(40*no+19-4,y-1,40*no+19+4,y+1);
    }

  void draw4a(int x1,int y1,int x2,int y2,int x3,int y3,int x4,int y4){
    int poly[8];

    poly[0]=x1;poly[1]=y1;
    poly[2]=x2;poly[3]=y2;
    poly[4]=x3;poly[5]=y3;
    poly[6]=x4;poly[7]=y4;
    fillpoly(4,poly);
    }

  void triangle(int x,int y,int direction,int col){
    setfillstyle(1,col);
    setcolor(col);
    if(direction==0) draw4a(x,y,x+10,y+5,x,y+10,x,y+11);
    else draw4a(x,y+5,x+10,y,x+10,y+11,x+10,y+10);
    }

  void deck_button(int no,int type,int func){

    // func 0 for playback
    // func 1 for record
    int col;

    box(251+45*no,140,294+45*no,180,type);
    setcolor(0);
    setfillstyle(1,0);
    switch(no){
      case 0:if(!func){setfillstyle(1,7);setcolor(7);}
             else{setfillstyle(1,12);setcolor(12);}
             pieslice(272,160,0,360,5);
             break;
      case 1:triangle(312,155,0,0);
             break;
      case 2:triangle(350,155,1,0);
             triangle(350+15,155,1,0);
             break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      case 3:triangle(395,155,0,0);
             triangle(395+15,155,0,0);
             break;
      case 4:triangle(444,155,1,0);
             triangle(444+13,155,1,0);
             bar(439,155,441,165);
             break;
      case 5:if(func==0)col=0;
               else col=7;
             triangle(483,155,0,col);
             triangle(483+13,155,0,col);
             bar(510,155,512,165);
             break;
      case 6:bar(537,155,547,165);
             break;
      case 7:bar(582,155,585,165);
             bar(589,155,592,165);
             break;
      }
    }

  int check_deck(void){
    int i,x,y,ret;

    ret=-1;
    mouse_position(&x,&y);
    if((y>140)&&(y<180))
      for(i=0;i<8;i++)
        if((x>251+i*45)&&(x<294+i*45)){
          ret=i;
          i=8;
          }
    return(ret);
    }

  void print_deck_status(int mode){

    setfillstyle(1,14);
    bar(251,111,345,127);
    setcolor(8);
    switch (mode){
      case RECORD  :outtextxy(259,116,"  RECORD");break;
      case PLAYBACK:outtextxy(259,116," PLAYBACK");break;
      case FASTPLAY:outtextxy(259,116," FASTPLAY");break;
      case REWIND  :outtextxy(259,116,"  REWIND");break;
      case FORWARD :outtextxy(263,116," FORWARD");break;
      case SKIPBACK:outtextxy(263,116,"SKIP BACK");break;
      case STOP    :outtextxy(259,116,"   STOP");break;
      case PAUSE   :outtextxy(263,116,"  PAUSE");break;
      case SKIPFORWARD:break;
      }
    }

  void deck(int func){
    int i;
    setfillstyle(1,7);
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    bar(241,100,620,194);
    setfillstyle(1,0);
    bar(241+8,130+8,620-9,189-7);
    setcolor(15);
    line(241,100,620,100);
    line(242,101,619,101);
    line(241,100,241,194);
    line(242,101,242,193);
    line(620-8,130+7,620-8,189-6);
    line(620-9,130+8,620-9,189-7);
    line(241+8,189-7,620-9,189-7);
    line(241+7,189-6,620-8,189-6);
    line(250+96,110,250+96,128);//status line
    line(250,128,250+96,128);//status line
    setcolor(8);
    line(620,100,620,194);
    line(619,101,619,193);
    line(241,194,620,194);
    line(242,193,619,193);
    line(241+8,130+8,620-9,130+8);
    line(241+7,130+7,620-8,130+7);
    line(241+8,130+8,241+8,189-7);
    line(241+7,130+7,241+7,189-6);
    line(250,110,250+96,110);//status line
    line(250,110,250,128);//status line
    for(i=0;i<8;i++)
      deck_button(i,1,func);
    setfillstyle(1,14);
    bar(251,111,345,127);
    write_text(410,410,106,"COUNTER",15);
    setfillstyle(1,0);
    //bar(500,111,550,127);
    write_text(450,606,116,"Scroll Mixer",15);
    box(450,111,466,127,true);
    outcharxy(455,116,17);
    box(590,111,606,127,true);
    outcharxy(595,116,16);
    }

  void mixer_counter(unsigned int counter,unsigned int old_count){
    char temp[5],temp1[5];
    int i;

    temp[4]=counter%10+48;
    temp[3]=(counter/10)%10+48;
    temp[2]=(counter/100)%10+48;
    temp[1]=(counter/1000)%10+48;
    temp[0]=(counter/10000)%10+48;
    temp1[4]=old_count%10+48;
    temp1[3]=(old_count/10)%10+48;
    temp1[2]=(old_count/100)%10+48;
    temp1[1]=(old_count/1000)%10+48;
    temp1[0]=(old_count/10000)%10+48;
    setcolor(14);
    setfillstyle(1,0);
    for(i=-2;i<=2;i++)
      if(temp[i+2]!=temp1[i+2]){
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        bar(410+i*11-4,120,410+i*11+4,130);
        outcharxy(411+i*11-4,122,temp[i+2]);
      }
    }

void mixer(int index,int mode){
  int i,j;
  char temp[10];

  setfillstyle(1,7);
  bar(0,200,638,479);
  for(i=0;i<16;i++){
    setcolor(8);
    line(i*40,479,i*40+38,479);
    line(i*40+1,478,i*40+37,478);
    line(i*40+38,200,i*40+38,479);
    line(i*40+37,201,i*40+37,478);
    setcolor(15);
    line(i*40,200,i*40+38,200);
    line(i*40+1,201,i*40+37,201);
    line(i*40,200,i*40,479);
    line(i*40+1,201,i*40+1,478);

    setcolor(8);
    line(i*40+19,410,i*40+19,470);
    setcolor(15);
    rectangle(i*40+19-12,406,i*40+19+12,474);
    setcolor(8);
    line(i*40+19-12,406,i*40+19-12,474);
    line(i*40+19-12,406,i*40+19+12,406);
    setcolor(15);
    line(i*40+20,410,i*40+20,470);
    vol(1,track_vol[i+index]);
    setfillstyle(1,8);
    bar(i*40+19-10,340,i*40+19+10,400);
    setcolor(15);
    line(i*40+19-10,340,i*40+19+10,340);
    line(i*40+19-10,340,i*40+19-10,400);
    setcolor(0);
    line(i*40+19-10,400,i*40+19+10,400);
    line(i*40+19+10,340,i*40+19+10,400);
    // Send Button
    settextstyle(2,0,0);
    outtextxy(40*i+20-12,326,"SEND");
    on_off_switch(i,1,true,track_switch[i+index][0]);
    // EFFECT
    setcolor(0);
    outtextxy(40*i+20-9,272,"EFF");
    on_off_switch(i,2,true,track_switch[i+index][1]);
    // MAX Note On Velocity
    setcolor(0);
    outtextxy(40*i+20-9,248,"MAX");
    on_off_switch(i,3,true,track_switch[i+index][2]);
    // Message
    setfillstyle(1,0);
    bar(i*40+19-15,206,i*40+19+15,234);
    setcolor(14);
```

เอกสารนี้เป็นการสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      outtextxy(i*40+5,205," TRK");
      itoa(i+index+1,temp,10);
      outtextxy(i*40+5+15-3*strlen(temp),205+9,temp);
      // Panpot
      setcolor(0);
      outtextxy(40*i+20-9,302,"PAN");
      pan_vol(i,channel_pan[track_ch[i+index]]);
      // VU
      for(j=0;j<8;j++){
        setfillstyle(1,0);
        bar(i*40+19-3,393-j*7,i*40+19+4,393-j*7+3);
        if((mode==PLAYBACK)||(mode==FASTPLAY))
          all_done(false,true,true);
        }
      }
    settextstyle(0,0,0);
    }

  void mixer_tname(int index){
    int i,j;
    char temp[10];

    settextstyle(2,0,0);
    setcolor(14);
    for(i=0;i<16;i++){
      if(ready_data_node[i+index]!=NULL){
        for(j=0;j<5;j++)
          temp[j]=track_name[i+index][j];
        temp[5]=0;
        outtextxy(i*40+5,205+18,temp);
        }
      else
        outtextxy(i*40+5,205+18,"CLEAN");
      }
    settextstyle(0,0,0);
    }

  void clean_VU(void){
    int i,j;

    setfillstyle(1,0);
    for(i=0;i<16;i++)
      for(j=0;j<8;j++)
        bar(i*40+19-3,393-j*7,i*40+19+4,393-j*7+3);
    for(j=0;j<128;j++){
      mixer_VU[j][0]=0;
      mixer_VU[j][1]=0;
      track_note_on[j]=0;
      }
    }

  int check_vol(int index){
    int x,y,i,ret;
    mouse_position(&x,&y);
    ret=-1;
    for(i=0;i<16;i++){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if((x>i*40+19-8)&&(x<i*40+20+8)&&(y>465-track_vol[i+index]/2-4)&&(y<465-track_vol[i+in
dex]/2+4))
    ret=i;
    }
  return(ret);
  }

void check_switch(int *no,int *func){
  int x,y,i;
  mouse_position(&x,&y);
  *no=-1;
  for(i=0;i<16;i++)
    if((x>i*40+19-10)&&(x<i*40+19+10)){
      *no=i;
      i=16;
      }
  if(*no!=-1)
    if(y>324)*func=-1;
      else if(y>=316)*func=1;
        else if(y>270)*func=-1;
          else if(y>=262)*func=2;
            else if(y>246)*func=-1;
              else if(y>=238)*func=3;
                else *func=-1;

  }

int check_pan(int index){
  int x,y,i,ret;

  ret=-1;
  mouse_position(&x,&y);
  for(i=0;i<16;i++){
    if((y>293-8)&&(y<293+7))
      if((x>=i*40+19-2+(channel_pan[track_ch[i+index]]-64)/16)
          &&(x<=i*40+19+2+(channel_pan[track_ch[i+index]]-64)/16)){
        ret=i;
        i=16;
        }
    }
  return(ret);
  }

void update_mixer_VU(int index){
  int i,j;

  for(i=0;i<16;i++){
    if(mixer_VU[i+index][1]<mixer_VU[i+index][0]){
      for(j=mixer_VU[i+index][1]+1;j<=mixer_VU[i+index][0];j++){
        if(j<=4)setfillstyle(1,10);
        else   setfillstyle(1,12);
        bar(i*40+19-3,400-j*7,i*40+19+4,400-j*7+3);
        }
      mixer_VU[i+index][1]=mixer_VU[i+index][0];
      mixer_VU[i+index][0]--;
      }
    else if(mixer_VU[i+index][1]>mixer_VU[i+index][0]){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        setfillstyle(1,0);
        bar(i*40+19-3,400-mixer_VU[i+index][1]*7,i*40+19+4,400-mixer_VU[i+index][1]*7+3);
        mixer_VU[i+index][1]--;
        }
      }
    update_enable=false;
    }

  void scroll_mixer(int mode){
    int x,y;

    mouse_position(&x,&y);
    if((y>=111)&&(y<=127)&&(((x>=450)&&(x<=466))||((x>=590)&&(x<=606)))){
      mouse_cursor_off();
      if((x>=450)&&(x<=466)){
        box(450,111,466,127,false);
        setcolor(12);
        outcharxy(455,116,17);
        do{
          if((mode==PLAYBACK)||(mode==FASTPLAY))
          all_done(false,true,true);
          }while(leftb_pressed());
        box(450,111,466,127,true);
        setcolor(0);
        outcharxy(455,116,17);
        if(mixer_index>0){
          mixer_index--;
          mixer(mixer_index,mode);
          mixer_tname(mixer_index);
          clean_VU();
          }
        }
      else if((x>=590)&&(x<=606)){
        box(590,111,606,127,false);
        setcolor(12);
        outcharxy(595,116,16);
        do{
          if((mode==PLAYBACK)||(mode==FASTPLAY))
          all_done(false,true,true);
          }while(leftb_pressed());
        box(590,111,606,127,true);
        setcolor(8);
        outcharxy(595,116,16);
        if(mixer_index<112){
          mixer_index++;
          mixer(mixer_index,mode);
          mixer_tname(mixer_index);
          clean_VU();
          }
        }
      mouse_cursor_on();
      }
    }

  void show_present_track(void){
    char txt[50];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    setfillstyle(1,14);
    bar(163,21,355,40);
    setcolor(8);
    sprintf(txt,"TRACK No. %3d",present_track+1);
    outtextxy(214,28,txt);
    }

  void chng_p_track(void){
    int x,y,i;

    mouse_position(&x,&y);
    for(i=0;i<16;i++){
      if((y>=206)&&(y<=234)){
        if((x>=i*40+4)&&(x<=i*40+34)){
          if(present_track!=mixer_index+i){
            present_track=i+mixer_index;
            repaint_key();
            show_present_track();
            }
          i=15;
          }
        }
      }
    }

  void del_mainvolume(int pos,int vol){
    setfillstyle(1,7);
    bar(190+pos*30-8,176-5-vol/2,190+pos*30+7,176+4-vol/2);
    setcolor(8);
    line(189+pos*30,176-5-vol/2,189+pos*30,176+4-vol/2);
    setcolor(15);
    line(190+pos*30,176-5-vol/2,190+pos*30,176+4-vol/2);
    }

  void mainvolume(int pos,int vol){
    setfillstyle(1,7);
    bar(190+pos*30-8,176-5-vol/2,190+pos*30+7,176+4-vol/2);
    setcolor(15);
    line(190+pos*30-8,176-5-vol/2,190+pos*30-8,176+4-vol/2);
    line(190+pos*30-7,176-4-vol/2,190+pos*30-7,176+3-vol/2);
    line(190+pos*30-8,176-5-vol/2,190+pos*30+7,176-5-vol/2);
    line(190+pos*30-7,176-4-vol/2,190+pos*30+6,176-4-vol/2);
    line(190+pos*30-4,176-vol/2,190+pos*30+3,176-vol/2);
    setcolor(8);
    line(190+pos*30+7,176-5-vol/2,190+pos*30+7,176+4-vol/2);
    line(190+pos*30+6,176-4-vol/2,190+pos*30+6,176+3-vol/2);
    line(190+pos*30-8,176+4-vol/2,190+pos*30+7,176+4-vol/2);
    line(190+pos*30-7,176+3-vol/2,190+pos*30+6,176+3-vol/2);
    line(190+pos*30-4,175-vol/2,190+pos*30+3,175-vol/2);
    }

  void playback_ctrl(void){
    int sav_col1,sav_col2;
    settextstyle(2,0,0);
    sav_col1=color.button_normal;
    sav_col2=color.bright;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    color.bright=15;
    color.button_normal=7;
    box(22,100,238,194,true);
    color.bright=sav_col2;
    color.button_normal=sav_col1;
    // Main_vol
    setcolor(15);
    line(230,185-68,230,185);
    line(230,185,230-21,185);
    line(230-10,185-65,230-10,185-4);
    outtextxy(230-18,104,"VOL");
    setcolor(8);
    line(230-21,185-68,230-21,185);
    line(230-21,185-68,230,185-68);
    line(230-11,185-65,230-11,185-4);
    // Tempo
    setcolor(15);
    line(200,185-68,200,185);
    line(200,185,200-21,185);
    line(200-10,185-65,200-10,185-4);
    outtextxy(200-25,104,"TEMPO");
    setcolor(8);
    line(200-21,185-68,200-21,185);
    line(200-21,185-68,200,185-68);
    line(200-11,185-65,200-11,185-4);
    setfillstyle(1,0);
    bar(27,105,170,120);
    settextstyle(0,0,0);
    setcolor(color.shadow);
    outtextxy(41,181,"TIME");
    setcolor(15);
    outtextxy(40 ,180,"TIME");
    outcharxy(90 ,180,'0');
    outcharxy(100,180,'0');
    outcharxy(110,180,':');
    outcharxy(120,180,'0');
    outcharxy(130,180,'0');
    outcharxy(140,180,':');
    outcharxy(150,180,'0');
    outcharxy(160,180,'0');
    mainvolume(0,percent_tempo);
    mainvolume(1,main_volume);
    }

void rimshot(void){

  midiout(drum_ch/16,Note_on|(drum_ch%16));
  midiout(drum_ch/16,37);
  midiout(drum_ch/16,84);

  midiout(drum_ch/16,Note_off|(drum_ch%16));
  midiout(drum_ch/16,37);
  midiout(drum_ch/16,64);
  }

void count_in(void){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    program_8253(tempo_rate);
    main_timer=0;
    main_tick=0;
    rimshot();
    do{}while((!kbhit())&&(main_timer<2));
    rimshot();
    do{}while((!kbhit())&&(main_timer<4));
    rimshot();
    do{}while((!kbhit())&&(main_timer<5));
    rimshot();
    do{}while((!kbhit())&&(main_timer<6));
    rimshot();
    do{}while((!kbhit())&&(main_timer<7));
    rimshot();
    do{}while((!kbhit())&&(main_timer<8));
    }

  void all_done(boolean rim,boolean show_time,boolean vu){
    send_data();
    send_out();
    if((update_enable)&&(vu))update_mixer_VU(mixer_index);
    if(counter!=main_timer){
      if(rim)rimshot();
      old_count=counter;
      counter=main_timer;
      mixer_counter(counter,old_count);
      }
    old_end_time=end_time;
    end_time=time(NULL);
    if((old_end_time!=end_time)&&(show_time)){
      t_hour=(int)(end_time-s_time)/3600;
      t_min=((int)(end_time-s_time)%3600)/60;
      t_sec=(int)(end_time-s_time)%60;
      setcolor(15);
      setfillstyle(1,7);
      bar(90,180,168,188);
      outcharxy(90 ,180,(t_hour/10)+48);
      outcharxy(100,180,(t_hour%10)+48);
      outcharxy(110,180,':');
      outcharxy(120,180,(t_min/10)+48);
      outcharxy(130,180,(t_min%10)+48);
      outcharxy(140,180,':');
      outcharxy(150,180,(t_sec/10)+48);
      outcharxy(160,180,(t_sec%10)+48);
      }
    }

  void realtime_record(void){
    unsigned char channel,message_type,temp,pgm_now,tick,noteon[128],note_no,note_vel;
    unsigned char *track_temp_node,*track_temp;
    unsigned int time,i,on_num=0,size=0;

    show_present_track();
    for(i=0;i<128;i++){
      noteon[i]=0;
      track_note_on[track_in_use[i]]=0;
      mixer_VU[i][0]=mixer_VU[i][1]=0;
```

เอกสารนี้จัดทำขึ้นเพื่อส่วนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  track_present_pgm[i]=track_vcs[i];
  }
old_count=12345;
message_type=Note_on;
sending_data=true;
channel=track_ch[present_track];
pgm_now=track_vcs[present_track];
cleardevice();
setfillstyle(1,color.screen_color);
bar(0,0,getmaxx(),getmaxy());
seven_octave(62,50);
deck(1);
mixer(mixer_index,STOP);
perform_data(true,0,0);
first_que=last_que=0;
midiin_access=true;
track=track_node;
time=65535u;tick=0;
*(unsigned long int*)track_node=0xFFFFFFFFlu;
count_in();
main_timer=0;main_tick=0;
do{
  do{
    if(first_que!=last_que){
      if((midiin_que[first_que]&0x80)==0x80){
        temp=read_que();
        if(temp==Exclusive){
          //midiin_access=false;
          message_type=Exclusive;
          }
        else if(temp==Endof_exc)
          //midiin_access=true;
          message_type=Endof_exc;
        else{
          //old_message=message_type;
          message_type=temp&0xF0;
          }
        }
      else{
      if(((main_timer!=time)||(main_tick!=tick))&&(message_type!=Exclusive)){
        time=main_timer;
        tick=main_tick;
        if(((message_type!=Ctrl_chng)||(midiin_que[first_que]!=0x7B))&&
           (message_type!=Exclusive)){
          *track++=0xFF;
          *(unsigned int*)track=main_timer;
          track+=2;
          *track++=main_tick;
          }
        }
      switch(message_type){
        case Note_on   :if(present_pgm[channel]!=pgm_now){
                          put_out(channel/16,Pgm_chng|channel%16);
                          put_out(channel/16,pgm_now);
                          }
                        put_out(channel/16,Note_on|channel%16);
                        note_no=read_que();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        wait_input();
                        note_vel=read_que();
                        if(note_vel==0){
                          put_out(channel/16,Note_off|channel%16);
                          *track++=Note_off|(channel%16);
                          *track=note_no;
                          if(track_note_on[present_track]>0)  track_note_on[present_trac
k]--;
                          if(track_note_on[present_track]==0) mixer_VU[present_track][0]
=0;
                          noteon[(int)*track]=0;
                          on_num--;
                          rec_key_pressed(track_trans[present_track]+*track,false);
                          put_out(channel/16,track_trans[present_track]+*track++);
                          *track=64;
                          put_out(channel/16,*track++);
                          }
                        else{
                          *track++=Note_on|(channel%16);
                          *track=note_no;
                          noteon[(int)*track]=1;
                          on_num++;
                          rec_key_pressed(track_trans[present_track]+*track,true);
                          put_out(channel/16,track_trans[present_track]+*track++);
                          *track=note_vel;
                          track_note_on[present_track]++;
                          if(mixer_VU[present_track][0]<assign_VU_value((char)((*track*m
ain_volume)/100)));
                            mixer_VU[present_track][0]=assign_VU_value((char)((*track*ma
in_volume)/100));
                          put_out(channel/16,(unsigned char)((track_vol[present_track]*(
*track++)/100)*main_volume)/100);
                          }
                        break;
        case Note_off  :temp=read_que();
                        if((noteon[(int)temp])&&(on_num>0)){
                          put_out(channel/16,Note_off|channel%16);
                          *track++=Note_off|(channel%16);
                          *track=temp;
                          if(track_note_on[present_track]>0)track_note_on[present_track]
--;
                          if(track_note_on[present_track]==0) mixer_VU[present_track][0]
=0;
                          noteon[(int)*track]=0;
                          on_num--;
                          rec_key_pressed(track_trans[present_track]+*track,false);
                          put_out(channel/16,track_trans[present_track]+*track++);
                          wait_input();
                          *track=read_que();
                          put_out(channel/16,*track++);
                          }
                        break;
        case Ctrl_chng :temp=read_que();
                        if(temp==0x7B){
                          wait_input();
                          read_que();
                          if(on_num>0){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        if((time!=main_timer)||(tick!=main_tick)){
                          time=main_timer;
                          tick=main_tick;
                          *track++=0xFF;
                          *(unsigned int*)track=main_timer;
                          track+=2;
                          *track++=main_tick;
                          }
                        put_out(channel/16,Ctrl_chng|(channel%16));
                        put_out(channel/16,0x7B);
                        put_out(channel/16,0);
                        *track++=Note_off|(channel%16);
                        for(i=0;i<128;i++)
                        if(noteon[i]){
                          *track++=(unsigned char)i;
                          *track++=64;
                          noteon[i]=0;
                          }
                        on_num=0;
                        track_note_on[present_track]=0;
                        }
                      }
                    else{
                      *track++=Ctrl_chng|(channel%16);
                      put_out(channel/16,Ctrl_chng|channel%16);
                      *track++=temp;
                      put_out(channel/16,temp);
                      wait_input();
                      *track=read_que();
                      put_out(channel/16,*track++);
                      }
                    break;
        case Pgm_chng  :*track++=Pgm_chng|(channel%16);
                    pgm_now=read_que();
                    *track++=pgm_now;
                    break;
        case Channel_press:
                    *track++=Channel_press|(channel%16);
                    put_out(channel/16,Channel_press|(channel%16));
                    *track=read_que();
                    put_out(channel/16,*track++);
                    break;
        default        :if(message_type!=Exclusive){
                      put_out(channel/16,message_type|channel%16);
                      *track++=message_type|(channel%16);
                      *track=read_que();
                      put_out(channel/16,*track++);
                      wait_input();
                      *track=read_que();
                      put_out(channel/16,*track++);
                      }
                    else{
                      do{
                        read_que();
                        }while((first_que!=last_que)&&((midiin_que[first_que]&0x80)!
=0x80));
                      }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        break;
                    }
                  }
                }
              }while((main_timer==time)&&(main_tick==tick));
              all_done(true,false,true);
            }while(!kbhit());
            all_note_off();
            readfunc();
            midiin_access=false;
            sending_data=false;
            *track++=0xFF;
            *(unsigned int*)track=0xFFFF;

            if(ready_data[present_track!=NULL]){
              track=track_node;
              while(*track!=0xFF)track++;
              track++;
              while(!endof_data(track+size))size++;
              size+=3;
              track_temp_node=getmem(size);
              track_temp=track_temp_node;
              track=track_node;
              for(i=0;i<size;i++)
                *track_temp++=*track++;
              merge(ready_data_node[present_track],track_temp_node,track_node);
              farfree(track_temp_node);
              }

            destroy_data();
            track=track_node;
            save_track();
            main_screen();
            }

          void screen_up(void){
            int i;
            void far *ptr;

            ptr=getmem(imagesize(28,0,430,9));
            for(i=0;i<24;i++){
              getimage(28,100+(i+1)*14,430,109+(i+1)*14,ptr);
              putimage(28,100+i*14,ptr,COPY_PUT);
              }
            farfree(ptr);
            setfillstyle(1,color.frame_color);
            bar(30,436,428,445);
            }

          void screen_down(void){
            int i;
            void far *ptr;

            ptr=getmem(imagesize(28,0,430,9));
            for(i=24;i>0;i--){
              getimage(28,100+(i-1)*14,430,109+(i-1)*14,ptr);
              putimage(28,100+i*14,ptr,COPY_PUT);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }
  farfree(ptr);
  setfillstyle(1,color.frame_color);
  bar(30,100,428,109);
  }

void time_detail(void){
  }

void detail_editor(void){
  int max_line=25,current_line=0,screen_line=0,i,x_pos,max_x;
  int start_x=34,start_y=100;
  unsigned char *start,ch,*ptr;
  unsigned int  time;
  unsigned char tick;
  char temp[255],misc[4];
  int save_bk_col;

  save_bk_col=color.readtext_background_color;
  color.readtext_background_color=color.frame_color;
  frame(24,54,434,460,true);
  window_(450,54,620,204);
  load_track();
  start=track;
  find_start_track();
  setcolor(color.normaltext_color);
  for(i=0;((i<max_line)&&(!endof_data(track)));i++){
    return_time(track,&time,&tick);
    track+=3;
    sprintf(temp,"%5d,%2d :",time,tick);
    while(*track!=0xFF){
      sprintf(misc," %2X",*track++);
      strcat(temp,misc);
      }
    sprintf(misc," %2X",*track++);
    strcat(temp,misc);
    outtextxy(start_x,start_y+current_line*14,temp);
    current_line++;
    }
  track=start;
  current_line=0;
  x_pos=0;
  do{
    ptr=track;
    max_x=0;
    ptr=track;
    ptr+=3;
    while(*(ptr+max_x)!=0xFF) max_x++;
    max_x++;
    if(x_pos==0){
      moveto(start_x+64,start_y+screen_line*14);
      }
    else{
      moveto(start_x+88+(x_pos-1)*24,start_y+screen_line*14);
      }
    cursor_on();
    mouse_cursor_on();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
ch=readfunc();
cursor_off();
mouse_cursor_off();
if(funckey){
  switch(ch){
    case LEFT_KEY  :if(x_pos>0){
                      x_pos--;
                      }
                    break;
    case RIGHT_KEY :if(x_pos<max_x){
                      x_pos++;
                      }
                    break;
    case UP_KEY    :if(current_line>0){
                      track=find_pred(track,start);
                      current_line--;
                      if(screen_line>0)
                        screen_line--;
                      else{
                        screen_down();
                        ptr=track;
                        return_time(ptr,&time,&tick);
                        ptr+=3;
                        sprintf(temp,"%5d,%2d :",time,tick);
                        while(*ptr!=0xFF){
                          sprintf(misc," %2X",*ptr++);
                          strcat(temp,misc);
                          }
                        sprintf(misc," %2X",*ptr++);
                        strcat(temp,misc);
                        outtextxy(start_x,start_y+screen_line*14,temp);
                        }
                      x_pos=0;

                      }
                    break;
    case DOWN_KEY  :current_line++;
                    track=find_next(track);
                    x_pos=0;
                    screen_line++;
                    if(screen_line==max_line){
                      screen_line--;
                      screen_up();
                      ptr=track;
                      return_time(ptr,&time,&tick);
                      ptr+=3;
                      sprintf(temp,"%5d,%2d :",time,tick);
                      while(*ptr!=0xFF){
                        sprintf(misc," %2X",*ptr++);
                        strcat(temp,misc);
                        }
                      sprintf(misc," %2X",*ptr++);
                      strcat(temp,misc);
                      outtextxy(start_x,start_y+screen_line*14,temp);
                      }
                    break;
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      }
    }while((funckey)||(ch!=ESC_KEY));
  color.readtext_background_color=save_bk_col;
  main_screen();
  }

void score_editor(void){
  }

void chng_ctrl(int mode){
  int x,y,save_y;
  int pos,vol;
  int m_pos=-1,*m_vol;

  mouse_position(&x,&y);
  pos=0;
  vol=percent_tempo;
  if((x>=190+pos*30-8)&&(x<=190+pos*30+7)&&(y>=176-5-vol/2)&&(y<=176+4-vol/2)){
    m_pos=0;
    m_vol=&percent_tempo;
    }
  else{
    pos=1;
    vol=main_volume;
    if((x>=190+pos*30-8)&&(x<=190+pos*30+7)&&(y>=176-5-vol/2)&&(y<=176+4-vol/2)){
      m_pos=1;
      m_vol=&main_volume;
      }
    }
  if(m_pos!=-1){
  mouse_cursor_off();
  set_mouse_position(x,176-(*m_vol/2));
  save_y=y;
    do{
      if((mode==PLAYBACK)||(mode==FASTPLAY))
        all_done(false,true,true);
      mouse_position(&x,&y);
      if(y!=save_y){
        del_mainvolume(m_pos,*m_vol);
        if(y<126){
          y=126;
          set_mouse_position(x,y);
          }
        else if(y>176){
          y=176;
          set_mouse_position(x,y);
          }
        save_y=y;
        *m_vol=(176-y)*2;
        mainvolume(m_pos,*m_vol);
        if(m_pos==0)
          program_8253(tempo_rate*(percent_tempo+50)/100);
        }
      }while(leftb_pressed());
    mouse_cursor_on();
    }
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void playback(void){
  int save_present_track;
  int mix_pos,x,y,i,func,mode=STOP,old_mode=STOP,old_old_mode=STOP,temp;
  unsigned char save_tick,save_tick1=0;
  unsigned int  save_timer,save_timer1=0;
  boolean exit=false,end_song;

  save_present_track=present_track;
  set_mouse_position(0,0);
  for(i=0;i<128;i++){
    mixer_VU[i][0]=0;
    mixer_VU[i][1]=0;
    track_note_on[i]=0;
    track_present_pga[i]=track_vcs[i];
    }
  counter=0;
  old_count=12345;
  cleardevice();
  seven_octave(62,50);
  deck(0);
  playback_ctrl();
  mixer_counter(counter,old_count);
  mixer(mixer_index,mode);
  sending_data=true;
  setcolor(8);
  outtextxy(194,28,"Transfering Data");
  perform_data(true,0,0);
  mixer_tname(mixer_index);
  show_present_track();
  set_synth();
  out_head[0]=0;
  out_tail[0]=0;
  out_head[1]=0;
  out_tail[1]=0;
  monitor=true;
  mouse_cursor_on();
  update_enable=false;
  do{
    if(rightb_pressed()){
      do{}while(rightb_pressed());
      exit=true;
      }
    if(leftb_pressed()){
      chng_p_track();
      scroll_mixer(mode);
      chng_ctrl(mode);
      temp=check_deck();
      if((temp!=-1)&&(temp!=0)){
        old_old_mode=old_mode;
        old_mode=mode;
        mode=temp;
        save_timer=main_timer;
        save_tick=main_tick;
        mouse_cursor_off();
        print_deck_status(mode);
        deck_button(temp,0,0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        do{
          }while(leftb_pressed());
        deck_button(temp,1,0);
        mouse_cursor_on();
        switch(mode){
          case PLAYBACK    :if(old_mode==PAUSE){
                              main_timer=save_timer1;
                              main_tick=save_tick1;
                              }
                            else if((old_mode!=PLAYBACK)&&(old_mode!=FASTPLAY)){
                              find_time_data(counter,0);
                              //program_8253(tempo_rate);
                              program_8253(tempo_rate*(percent_tempo+50)/100);
                              pop_pgm(counter,0);
                              main_timer=counter;main_tick=0;
                              }
                            else{
                              if(old_mode==FASTPLAY)
                                //program_8253(tempo_rate);
                                program_8253(tempo_rate*(percent_tempo+50)/100);
                              main_timer=save_timer;
                              main_tick=save_tick;
                              }
                            if(counter==0)
                              s_time=time(NULL);
                            break;
           case FORWARD     :if((old_mode==PLAYBACK)||(old_mode==FASTPLAY)){
                              clean_VU();
                              //program_8253(tempo_rate*2);
                              program_8253(2*tempo_rate*(percent_tempo+50)/100);
                              mode=FASTPLAY;
                              print_deck_status(mode);
                              main_timer=save_timer;
                              main_tick=save_tick;
                              }
                            else{
                              all_note_off();
                              clean_VU();
                              repaint_key();
                              }
                            break;
           case SKIPFORWARD :old_count=counter;
                            counter=noteperbar*(unsigned int)(counter/noteperbar)+noteperb
    ar;
                            find_time_data(counter,0);
                            mixer_counter(counter,old_count);
                            all_note_off();
                            clean_VU();
                            repaint_key();
                            if((old_mode==PLAYBACK)||(old_mode==FASTPLAY)){
                              mode=old_mode;
                              print_deck_status(mode);
                              main_timer=counter;
                              main_tick=0;
                              }
                            break;
           case SKIPBACK    :old_count=counter;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        counter=0;
                        all_note_off();
                        clean_VU();
                        repaint_key();
                        mixer_counter(counter,old_count);
                        mode=STOP;
                        print_deck_status(9);
                        break;
      case PAUSE       :if((old_mode!=PLAYBACK)&&(old_mode!=FASTPLAY))
                          print_deck_status(9);
                        if(old_mode==PAUSE){
                          mode=old_old_mode;
                          main_timer=save_timer1;
                          main_tick=save_tick1;
                          print_deck_status(mode);
                          }
                        else{
                          save_timer1=main_timer;
                          save_tick1=main_tick;
                          }
                        break;
      default          :all_note_off();
                        clean_VU();
                        repaint_key();
                        break;
      }
    if(mode==STOP)print_deck_status(9);
    }
  mix_pos=check_vol(mixer_index);
  if(mix_pos!=-1){
    mouse_cursor_off();
    set_mouse_position(mix_pos*40+19,465-track_vol[mix_pos+mixer_index]/2);
    while(leftb_pressed()){
      if((mode==PLAYBACK)||(mode==FASTPLAY))all_done(false,true,true);
      mouse_position(&x,&y);
      if(y>465)y=465;
      if(y<415)y=415;
      if(465-y!=track_vol[mix_pos+mixer_index]/2){
        del_vol(mix_pos,track_vol[mix_pos+mixer_index]);
        track_vol[mix_pos+mixer_index]=(465-y)*2;
        vol(mix_pos,track_vol[mix_pos+mixer_index]);
        }
      }
    mouse_cursor_on();
    }
  check_switch(&mix_pos,&func);
  if((mix_pos!=-1)&&(func!=-1)){
    mouse_cursor_off();
    if(track_switch[mix_pos+mixer_index][func-1]){
      track_switch[mix_pos+mixer_index][func-1]=false;
      if(mix_pos+mixer_index==present_track)repaint_key();
      }
    else
      track_switch[mix_pos+mixer_index][func-1]=true;
    on_off_switch(mix_pos,func,false,track_switch[mix_pos+mixer_index][func-1]);
    if((func==1)&&(!track_switch[mix_pos+mixer_index][0])){
      track_note_on[mix_pos+mixer_index]=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            channel_note_off(mix_pos+mixer_index);
            mixer_VU[mix_pos+mixer_index][0]=0;
            }
          if(func==2){
            effect(track_ch[mix_pos+mixer_index],track_switch[mix_pos+mixer_index][1]);
            }
          while(leftb_pressed()){
            if((mode==PLAYBACK)||(mode==FASTPLAY))all_done(false,true,true);
            }
          on_off_switch(mix_pos,func,true,track_switch[mix_pos+mixer_index][func-1]);
          mouse_cursor_on();
          }
        mix_pos=check_pan(mixer_index);
        if(mix_pos!=-1){
          mouse_cursor_off();
          set_mouse_position(mix_pos*40+19+(channel_pan[track_ch[mix_pos+mixer_index]]-64)/8
,293);
          while(leftb_pressed()){
            mouse_position(&x,&y);
            if(x<mix_pos*40+19-8){
              x=mix_pos*40+19-8;
              set_mouse_position(x,293);
              }
            if(x>mix_pos*40+19+8){
              x=mix_pos*40+19+8;
              set_mouse_position(x,293);
              }
            if(x!=40*mix_pos+19+(channel_pan[track_ch[mix_pos+mixer_index]]-64)/8){
              channel_pan[track_ch[mix_pos+mixer_index]]=(x-40*mix_pos-19)*8+64;
              pan_vol(mix_pos,channel_pan[track_ch[mix_pos+mixer_index]]);
              panpot(track_ch[mix_pos+mixer_index],channel_pan[track_ch[mix_pos+mixer_index]
]);
              }
            if((mode==PLAYBACK)||(mode==FASTPLAY))all_done(false,true,true);
            }
          mouse_cursor_on();
          }
        }
      end_song=true;
      for(i=0;i<point_track_in_use;i++){
        if(!endof_data(ready_data[track_in_use[i]]))end_song=false;
        }
      if(mode==REWIND){
        if(counter>0){
          old_count=counter;
          counter--;
          }
        else{
          print_deck_status(9);
          mode=STOP;
          }
        delay(30);
        mixer_counter(counter,old_count);
        }
      else if(end_song){
        if(mode!=STOP)
          print_deck_status(9);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        mode=STOP;
        clean_VU();
        }
      else if((mode==PLAYBACK)||(mode==FASTPLAY)){
        all_done(false,true,false);
        if(update_enable){
          mouse_position(&x,&y);
          if((y>322)&&(y<400)) mouse_cursor_off();
          update_mixer_VU(mixer_index);
          if((y>322)&&(y<400)) mouse_cursor_on();
          }
        }
      else if(mode==FORWARD){
        old_count=counter;
        counter++;
        delay(30);
        mixer_counter(counter,old_count);
        }
      if(kbhit())exit=((readfunc()==ESC_KEY)&&(!funckey));
      }while(!exit);
    monitor=false;
    all_note_off();
    destroy_data();
    sending_data=false;
    mouse_cursor_off();
    message(240,240,"Exiting","  Exiting  Playback  ");
    main_screen();
    present_track=save_present_track;
    }

  void tempo(void){
    void far *ptr;
    char *temp;
    int x1,y1,x2,y2,i,tempo_temp;
    boolean chk;

    x1=300;
    y1=100;
    x2=480;
    y2=160;
    ptr=getmem(imagesize(x1,y1,x2+4,y2+4));
    temp=getmem(100);
    getimage(x1,y1,x2+4,y2+4,ptr);
    setfillstyle(1,color.shadow);
    bar(x1+4,y1+4,x2+4,y2+4);
    frame(x1,y1,x2,y2,false);
    write_text(x1,x2,y1+10,"Tempo",color.title_text_color);
    setcolor(color.normaltext_color);
    sprintf(temp,"Current Tempo = %3d",tempo_rate);
    outtextxy(x1+color.frame_width+text_width,y1+25,temp);
    outtextxy(x1+color.frame_width+text_width+16,y1+39,"Set Tempo = ");
    sprintf(temp,"%d",tempo_rate);
    readtext(x1+color.frame_width+text_width+116,y1+39,temp,3,3,false);
    chk=true;
    if((strlen(temp)>0)&&(strlen(temp)<=3))
      for(i=0;i<strlen(temp);i++){
        if((temp[i]<48)||(temp[i]>57)) chk=false;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        mode=STOP;
        clean_VU();
        }
      else if((mode==PLAYBACK)||(mode==FASTPLAY)){
        all_done(false,true,false);
        if(update_enable){
          mouse_position(&x,&y);
          if((y>322)&&(y<400)) mouse_cursor_off();
          update_mixer_VU(mixer_index);
          if((y>322)&&(y<400)) mouse_cursor_on();
          }
        }
      else if(mode==FORWARD){
        old_count=counter;
        counter++;
        delay(30);
        mixer_counter(counter,old_count);
        }
      if(kbhit())exit=((readfunc()==ESC_KEY)&&(!funckey));
      }while(!exit);
    monitor=false;
    all_note_off();
    destroy_data();
    sending_data=false;
    mouse_cursor_off();
    message(240,240,"Exiting","  Exiting  Playback  ");
    main_screen();
    present_track=save_present_track;
    }

  void tempo(void){
    void far *ptr;
    char *temp;
    int x1,y1,x2,y2,i,tempo_temp;
    boolean chk;

    x1=300;
    y1=100;
    x2=480;
    y2=160;
    ptr=getmem(imagesize(x1,y1,x2+4,y2+4));
    temp=getmem(100);
    getimage(x1,y1,x2+4,y2+4,ptr);
    setfillstyle(1,color.shadow);
    bar(x1+4,y1+4,x2+4,y2+4);
    frame(x1,y1,x2,y2,false);
    write_text(x1,x2,y1+10,"Tempo",color.title_text_color);
    setcolor(color.normaltext_color);
    sprintf(temp,"Current Tempo = %3d",tempo_rate);
    outtextxy(x1+color.frame_width+text_width,y1+25,temp);
    outtextxy(x1+color.frame_width+text_width+16,y1+39,"Set Tempo = ");
    sprintf(temp,"%d",tempo_rate);
    readtext(x1+color.frame_width+text_width+116,y1+39,temp,3,3,false);
    chk=true;
    if((strlen(temp)>0)&&(strlen(temp)<=3))
      for(i=0;i<strlen(temp);i++){
        if((temp[i]<48)||(temp[i]>57)) chk=false;
```

เอกสารนี้ใช้สำหรับการที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }
      if(chk){
        tempo_temp=atoi(temp);
        if((tempo_temp<40)||(tempo_temp>250)) chk=false;
        else{
          tempo_rate=tempo_temp;
          //program_8253(tempo_rate);
          program_8253(tempo_rate*(percent_tempo+50)/100);
          }
        }
      putimage(x1,y1,ptr,COPY_PUT);
      farfree(temp);
      farfree(ptr);
      }

    void transpose(void){
      char temp[50],ch;
      boolean check=true;
      int i;

      open_window(320-140,200,320+120,260,1,"Transpose");
      setcolor(color.normaltext_color);
      sprintf(temp,"Set Transpose For All Track");
      outtextxy(320-140+20,200+25,temp);
      outtextxy(320-140+36,200+25+14,"Value [-36 - 36]:");
      itoa(transpose_level,temp,10);
      ch=readtext(320+40,200+25+14,temp,3,3,false);
      if (!(funckey)&&(ch==RETURN_KEY)){
        for(i=0;i<strlen(temp);i++){
          if((i!=0)&&((temp[i]<48)||(temp[i]>57)))
            check=false;
          else if(!(((temp[i]>=48)&&(temp[i]<=57))||(temp[i]=='-')))
            check=false;
          }
        if(check){
          for(i=0;i<128;i++)
            track_trans[i]-=transpose_level;
          transpose_level=atoi(temp);
          for(i=0;i<128;i++)
            track_trans[i]+=transpose_level;
          }
        else
          message(226,270," ERROR ","  Invalid Number  ");
        }
      release_window(1);
      }

    void change_color(void){
      }

    void directories(void){
      }

    void auto_quantize(void){
      }
    void auto_rewind(void){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  }

void VU_enable(void){
  }

void save_option(void){
  char filename[40];
  FILE *config;

  sprintf(filename,"%sMIDIPROJ.CFG",option.program_path);
  if((config=fopen(filename,"wb"))!=NULL){
    fwrite(&color,sizeof(color),1,config);
    fwrite(&option,sizeof(option),1,config);
    fclose(config);
    }
  }

void osshell(void){
  char *comspec;

  comspec=getenv("COMSPEC");
  closegraph();
  screen_saver(1);
  printf("\n\nType EXIT to return to MIDI Project\n\n");
  system(comspec);
  screen_saver(0);
  init_graph();
  main_screen();
  }

void quit(void){
  char command[100];

  farfree(track_node);
  setvect(0x1C,Save_Int1C);
  setvect(0x0A,Save_IRQ2);
  setvect(0x0B,Save_IRQ3);
  mouse_cursor_off();
  reset_mouse();
  closegraph();
  screen_saver(1);
  printf("\n\nMIDI Project.\n");
  printf("Program By.. Arkhom Busayakul\n");
  printf("                Electronics Part\n");
  printf("                   Faculty of Engineering  KMIT'L\n");
  printf("\nThank you for using MIDI Project!\n");
  outportb(0x2B3,0x32);
  outportb(0x2B3,0x72);
  sprintf(command,"IF EXIST %sTRACK.* DEL %sTRACK.*",option.temporary_path
          ,option.temporary_path);
  system(command);
  sprintf(command,"IF EXIST %sPATTERN.* DEL %sPATTERN.*",option.temporary_path
          ,option.temporary_path);
  system(command);
  sprintf(command,"IF EXIST %sDRUMMCHN.ARG DEL %sDRUMMCHN.ARG",option.temporary_path
          ,option.temporary_path);
  system(command);
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  exit(0);
  }

/* End of Working Function */

int main_menu_fx_m(void){
  int x_mouse,y_mouse,i;
  int temp;

  temp=-1;
  mouse_position(&x_mouse,&y_mouse);
  for(i=0;i<main_menu_no;i++){
    if((x_mouse>mainmenu_xpos[i]-8)&&(x_mouse<mainmenu_xpos[i]+8*strlen(mainmenu_command[i
])+8)){
      temp=i;
      i=main_menu_no;
      }
    }
  return(temp);
  }

void setmouse_cursor(int no){
  int x,y;

  x=mainmenu_xpos[no]-text_width+4*strlen(mainmenu_command[no]);
  y=34;
  set_mouse_position(x,y);
  }

void menu(void){
  int main_menu_pos=0,i,x1,y1,x2,y2,x_mouse,y_mouse,test;
  unsigned char ch;
  void far *ptr;

  do{
    x1=mainmenu_xpos[main_menu_pos]-text_width;
    y1=24;
    x2=mainmenu_xpos[main_menu_pos]+text_width
      *(strlen(mainmenu_command[main_menu_pos])+1);
    y2=26+text_tall+1;
    ptr=getmem(imagesize(x1,y1,x2,y2));
    getimage(x1,y1,x2,y2,ptr);
    setfillstyle(1,color.menu_active_color);
    bar(x1,y1,x2,y2);
    write_menu_command(mainmenu_xpos[main_menu_pos],26
          ,mainmenu_command[main_menu_pos],0,true);
    setfillstyle(1,0);
    if(!pull_down_active){
      do{
        setmouse_cursor(main_menu_pos);
        mouse_clicked=false;
        do{
          mouse_position(&x_mouse,&y_mouse);
          if(x_mouse<22) set_mouse_position(22,y_mouse);
          else if(x_mouse>616) set_mouse_position(616,y_mouse);
          test=main_menu_fx_m();
          }while((test==-1)||((!kbhit())&&(!leftb_pressed())&&(!rightb_pressed()))&&
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            ((test==main_menu_pos))));
        if(kbhit())
          mouse_clicked=false;
        else
          mouse_clicked=true;
        }while((!kbhit())&&(!mouse_clicked));
      if(!mouse_clicked)
        ch=readfunc();
      }
    else{
      if(pull_menu[main_menu_pos].num_item!=0){
        ch=pull_down(&pull_menu[main_menu_pos]);
        }
      else{
        setmouse_cursor(main_menu_pos);
        do{
          mouse_clicked=false;
          do{
            mouse_position(&x_mouse,&y_mouse);
            if(x_mouse<22) set_mouse_position(22,y_mouse);
            else if(x_mouse>616) set_mouse_position(616,y_mouse);
            test=main_menu_fx_m();
            }while((test==-1)||((!kbhit())&&(!leftb_pressed())&&(!rightb_pressed()))&&
               ((test==main_menu_pos))));
          if(kbhit())
            mouse_clicked=false;
          else
            mouse_clicked=true;
          }while((!kbhit())&&(!mouse_clicked));
        if(!mouse_clicked)
          ch=readfunc();
        }
      }
    putimage(x1,y1,ptr,COPY_PUT);
    if(mouse_clicked){ // mouse is moved or button is pressed
      if(leftb_pressed()){// select menu command and execute
        mouse_clicked=false;
        ch=RETURN_KEY;
        funckey=false;
        }
      else if(rightb_pressed()){ // cancel menu command not execute
        mouse_clicked=false;
        ch=ESC_KEY;
        funckey=false;
        }
      else{
        main_menu_pos=test;  // test = main_menu_fx_m()
        }
      }
    if((funckey)&&(!mouse_clicked)){
      switch(ch){
        case RIGHT_KEY :main_menu_pos++;
                        break;
        case LEFT_KEY  :main_menu_pos--;
                        break;
        }
      if(main_menu_pos<0)main_menu_pos=main_menu_no-1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(main_menu_pos==main_menu_no)main_menu_pos=0;
    }
  else{
    switch(ch){
      case ESC_KEY   :pull_down_active=false;
      default        :if((ch>96)&&(ch<123)&&(!mouse_clicked))
                        ch-=32;
                      for(i=0;i<main_menu_no;i++){
                        if(ch==mainmenu_command[i][0]){
                          main_menu_pos=i;
                          i=main_menu_no;
                          pull_down_active=true;
                          }
                        }
                      break;
    }
  }
putimage(x1,y1,ptr,COPY_PUT);
farfree(ptr);
if((!funckey)&&(ch==RETURN_KEY)){
  do{}while(leftb_pressed());
  if((pull_down_active)&&(pull_menu[main_menu_pos].num_item!=0)){
    //Execute menu command
    switch(main_menu_pos){
      case 0 :switch(pull_menu[main_menu_pos].command_no){
                case 0:load();break;
                case 1:save();break;
                case 2:new_all();break;
                case 3:comment();break;
                case 4:osshell();break;
                case 5:quit();break;
                }
              break;
      case 1 :switch(pull_menu[main_menu_pos].command_no){
                case 0:track_volume();break;
                case 1:track_transpose();break;
                case 2:clear_track();break;
                case 3:merge_track();break;
                case 4:track_comment();break;
                case 5:switch(popup_menu[0].command_no){
                         case 0:quantize();break;
                         case 1:kill_velocity();break;
                         case 2:kill_whell();break;
                         case 3:kill_pressure();break;
                         case 4:kill_program();break;
                         case 5:kill_control();break;
                         case 6:time_shifter();
                         }
                       break;//for popup menu
                case 6:assign_channel();break;
                case 7:voice_select();break;
                case 8:assign_track();break;
                }
              break;
      case 2 :switch(pull_menu[main_menu_pos].command_no){
                case 0:step_record();break;
                case 1:realtime_record();break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    case 2:detail_editor();break;
                    case 3:break;
                    }
                  break;
          case 4: switch(pull_menu[main_menu_pos].command_no){
                    case 0:tempo();break;
                    case 1:transpose();break;
                    }
                  break;
          case 5 :switch(pull_menu[main_menu_pos].command_no){
                    case 0:break;
                    case 1:break;
                    case 2:break;
                    case 3:break;
                    case 4:break;
                    case 5:break;
                    case 6:save_option();break;
                    }
                  break;
          }
        pull_down_active=false;
        mouse_clicked=false;
        ch=ESC_KEY;
        }
      else{
        if(pull_menu[main_menu_pos].num_item==0){
          //Execute main menu command
          switch(main_menu_pos){
            case 3:playback();break;
            case 7:drum_machine();break;
            case 8:view();
            }
          pull_down_active=false;
          }
        else{// First time you press Return key
          pull_down_active=true;
          }
      }
    }
  }while((!funckey)||(ch!=ALT_X_KEY));
}

void main(void){
  _stklen=65535u;
  startup();
  init_data();
  main_timer=0;
  main_tick=0;
  get_path();
  load_config();
  load_lib();
  init_graph();
  init_menu_command();
  main_screen();
  pgm_stack=getmem(1024);
  track_node=getmem(65535u);
  menu();
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
quit();
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <graphics.h>
#include <string.h>
#include <alloc.h>
#include <dos.h>
#include <dir.h>
#include "midi.h"
#include "mouse.h"

void far interrupt ( *Old_cursor_int)(void);

int blink_counter=0;
boolean cursor_flag=false;
void far *win_temp[10];int win_pos[10][2];
char cur_path[80]="",path[40];
pulldownmenu pull_menu[main_menu_no],popup_menu[2],set_v,step_menu,quantize_menu;
int mouse_clicked=false;

extern color_struct color;

void far *getmem(unsigned long size){
  char far *ptr;

  //if((ptr=(char far *)farcalloc(size,sizeof(char)))==NULL){
  if((ptr=(char far*)farmalloc(size))==NULL){
    closegraph();
    printf("Not enough memory !\n");
    printf("Program Abort\n");
    quit();
    }
  return(ptr);
  }

unsigned char readfunc(void){
  unsigned char temp;

  temp=getch();
  if(temp==0x00){
    temp=getch();
    funckey=true;
    }
  else
    funckey=false;
  return(temp);
  }

void screen_saver(int exe){
  int x,y,a;
  char far *vga_text;
  FILE *screen;
  char *filename,*command;
  unsigned char *screen_save;
  struct DOSERROR info;

  filename=getmem(100);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    command=getmem(100);
    sprintf(filename,"%s%s",option.temporary_path,"SCREEN.SAV");
    if(exe==0)
      screen=fopen(filename,"wb");
    else
      screen=fopen(filename,"rb");
    if(screen!=NULL){
      vga_text=MK_FP(0xB800,0);
      screen_save=getmem(4096);
      if(exe==1){
        x=fgetc(screen);
        y=fgetc(screen);
        gotoxy(x,y);
        fread(screen_save,4000,1,screen);
        }
      for(y=0;y<25;y++)
        for(x=0;x<80;x++)
          for(a=0;a<2;a++){
            if(exe==0)
              *(char*)(screen_save+y*160+x*2+a)=*(char*)(vga_text+y*160+x*2+a);
            else
              *(char*)(vga_text+y*160+x*2+a)=*(char*)(screen_save+y*160+x*2+a);
            }
      if(exe==0){
        x=wherex();
        y=wherey();
        fputc(x,screen);
        fputc(y,screen);
        fwrite(screen_save,4000,1,screen);
        }
      fclose(screen);
      if(exe==1){
        sprintf(command,"DEL %s",filename);
        system(command);
        }
      farfree(command);
      farfree(screen_save);
      }
    else{
      if(exe==0)
        perror("Unable to open file for writing");
      else
        perror("Unable to open file for reading");
      dosexterr(&info);
      printf("Extended DOS error information:\n");
      printf("   Extended error:     %d\n",info.de_exterror);
      printf("            Class:     %x\n",info.de_class);
      printf("           Action:     %x\n",info.de_action);
      printf("      Error Locus:     %x\n",info.de_locus);
      }
    farfree(filename);
    }
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
  void get_path(void){
    strcpy(path,"X:\\");
    path[0]='A'+getdisk();
    getcurdir(0,path+3);
```

```
    if(path[strlen(path)-1]!=0x5C)
      strcat(path,"\\");
    }

  void load_config(void){
    int i;
    FILE *config;

    if((config=fopen("MIDIPROJ.CFG","rb"))==NULL){
      color.frame_width=4;
      color.frame_title_color=0;
      color.border_color=7;
      color.bright=7;
      color.shadow=8;
      color.button_normal=15;
      color.button_pushed=7;
      color.screen_color=11;
      color.frame_color=2;
      color.bar_color=7;
      color.title_backgraound_color=2;
      color.title_text_color=15;
      color.menu_text_color=15;
      color.menu_hot_letter=14;
      color.menu_active_color=1;
      color.active_menu_text_color=7;
      color.normaltext_color=15;
      color.active_text_color=12;
      color.readtext_background_color=1;
      color.cursor_color=15;

      strcpy(option.graphics_driver_path,path);
      strcat(option.graphics_driver_path,"DRIVER\\");
      sprintf(option.temporary_path,"%s%s",path,"TEMP\\");
      strcpy(option.lib_path,path);
      strcpy(option.program_path,path);
      for(i=0;i<9;i++)
        strcpy(option.vcslib_name[i],"");
      strcpy(option.vcslib_name[0],"GM.VCS");
      strcpy(option.vcslib_name[1],"SP.VCS");
      strcpy(option.vcslib_name[2],"K1.VCS");
      strcpy(option.vcslib_name[3],"KN650.VCS");
      strcpy(option.vcslib_name[4],"DRUM.VCS");
      //strcpy(option.vcslib_name[8],"PERCSTD.LIB");
      option.auto_quantize=false;
      option.auto_rewind=true;
      option.VU_installed=true;
      }
    else{
      fread(&color,sizeof(color_struct),1,config);
      fread(&option,sizeof(option_struct),1,config);
      fclose(config);
      }
    }

void outcharxy(int xpos,int ypos,char ch){
  char temp[2];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    temp[1]=0;
    temp[0]=ch;
    outtextxy(xpos,ypos,temp);
    }

  void write(char ch){
    char temp[2];
    int save_color;

    save_color=getcolor();
    setcolor(color.shadow);
    temp[1]=0;
    temp[0]=ch;
    outtextxy(getx()+1,gety()+1,temp);
    setcolor(color.normaltext_color);
    outtext(temp);
    setcolor(save_color);
    }

  void write_text(int xpos_1,int xpos_2,int ypos,char *str,int col){
    int save_color,xpos;

    xpos=(xpos_1+xpos_2)/2-(text_width/2)*strlen(str);
    save_color=getcolor();
    setcolor(color.shadow);
    outtextxy(xpos+1,ypos+1,str);
    setcolor(col);
    outtextxy(xpos,ypos,str);
    setcolor(save_color);
    }

  void write_menu_command(int xpos,int ypos,char *str,int hot_no,boolean active){
    int save_color;

    save_color=getcolor();
    setcolor(color.shadow);
    outtextxy(xpos+1,ypos+1,str);
    if(active)
      setcolor(color.active_menu_text_color);
    else
      setcolor(color.menu_text_color);
    outtextxy(xpos,ypos,str);
    if(!active){
      setcolor(color.menu_hot_letter);
      outcharxy(xpos+text_width*hot_no,ypos,*(str+hot_no));
      }
    setcolor(save_color);
    }

  char readtext(int xpos,int ypos,char *gettext
              ,int max_char,int given_char,boolean edit){
    int  string_pos,pos,i;
    char temp[100],ch;
    unsigned char str_char[2];
    void far *ptr;
    boolean end=false;
    boolean first=true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 if(strlen(gettext)>max_char){
   closegraph();
   printf("Error in module Readtext\n");
   exit(0);
   }
 strcpy(temp,gettext);
 setfillstyle(1,color.readtext_background_color);
 bar(xpos-2,ypos-3,xpos+2+text_width*(given_char+1),ypos+2+text_tall);
 moveto(xpos,ypos);
 setcolor(color.normaltext_color);
 string_pos=strlen(temp);
 if(strlen(temp)<=given_char){
   outtext(temp);
   moveto(xpos+text_width*strlen(temp),ypos);
   pos=string_pos;
   }
 else{
   outtext(&temp[strlen(temp)-given_char]);
   moveto(xpos+text_width*given_char,ypos);
   pos=given_char;
   }
 setcolor(color.normaltext_color);
 str_char[1]=0;
 ptr=getmem(imagesize(0,0,(given_char+2)*text_width,13));
 end=false;
 do{
   cursor_on();
   //mouse_cursor_on();
   str_char[0]=readfunc();
   //mouse_cursor_off();
   cursor_off();
   if(!funckey){                    // What you type is Ascii
     if(str_char[0]>=32){           // What you type is String char
       if(first){
         strcpy(temp,"");
         pos=0;
         string_pos=0;
         setfillstyle(1,color.readtext_background_color);
         bar(xpos-2,ypos-3,xpos+2+text_width*(given_char+1),ypos+2+text_tall);
         moveto(xpos,ypos);
         first=false;
         }
       if(strlen(temp)<max_char){   // Accept this character
         if(pos<given_char){        // Cursor is in given bar
           getimage(xpos+text_width*pos,ypos-2
                   ,xpos+text_width*given_char,ypos+2+text_tall,ptr);
           putimage(xpos+text_width*(pos+1),ypos-2,ptr,COPY_PUT);
           bar(getx(),gety(),getx()+text_width-1,gety()+text_tall);
           pos++;
           outtext((char*)str_char);
           }
         else{// Cursor is to the end of given bar
           bar(xpos+text_width*(given_char),ypos-2
              ,xpos+text_width*(given_char+1)-1,ypos+text_tall+2);
           outcharxy(getx(),gety(),str_char[0]);
           getimage(xpos+text_width,ypos-2,xpos+text_width*(given_char+1)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                ,ypos+text_tall+2,ptr);
          putimage(xpos,ypos-2,ptr,COPY_PUT);
          bar(xpos+text_width*(given_char),ypos-2
             ,xpos+text_width*(given_char+1)-1,ypos+text_tall+2);
          if(string_pos<strlen(temp))
            outcharxy(getx(),gety(),temp[string_pos]);
          }
        for(i=max_char;i>=string_pos;i--)
          temp[i+1]=temp[i];
        temp[string_pos]=str_char[0];
        //outtext((char*)str_char);
        string_pos++;
        }
      }
     else if(str_char[0]==BACKSPACE_KEY){
       first=false;
       if(string_pos>0){
         if(pos>0){
           getimage(getx(),ypos-2,xpos+text_width*(given_char+1)-1
                  ,ypos+text_tall+2,ptr);
           putimage(getx()-text_width,ypos-2,ptr,COPY_PUT);
           bar(xpos+text_width*given_char,ypos-2,xpos+text_width*(given_char+1)-1
              ,ypos+text_tall+2);
           if(string_pos-pos+given_char<strlen(temp))
             outcharxy(xpos+text_width*given_char,ypos
                    ,temp[string_pos-pos+given_char+1]);
           moveto(getx()-text_width,gety());
           pos--;
           }
         string_pos--;
         for(i=string_pos;i<max_char;i++)
           temp[i]=temp[i+1];
         }
       }
     }
   else{
     first=false;                        // What you type is scan code
     switch(str_char[0]){
       case HOME_KEY  :bar(xpos-2,ypos-2,xpos+2+text_width*(given_char+1)
                         ,ypos+2+text_tall);
                       ch=temp[given_char+1];
                       temp[given_char+1]=0;
                       outtextxy(xpos,ypos,temp);
                       temp[given_char+1]=ch;
                       moveto(xpos,ypos);
                       string_pos=0;
                       pos=0;
                       break;
       case END_KEY   :if(strlen(temp)<=given_char){
                         pos=strlen(temp);
                         moveto(xpos+text_width*pos,ypos);
                         }
                       else if(strlen(temp)-string_pos<given_char-pos){
                         pos+=strlen(temp)-string_pos;
                         moveto(xpos+text_width*pos,ypos);
                         }
                       else{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    bar(xpos-2,ypos-2,xpos+2+text_width*(given_char+1)
                     ,ypos+2+text_tall);
                    outtextxy(xpos,ypos,&temp[strlen(temp)-given_char]);
                    moveto(xpos+text_width*given_char,ypos);
                    pos=given_char;
                    }
                  string_pos=strlen(temp);
                  break;
    case DELETE_KEY:if(string_pos<strlen(temp)){
                    getimage(getx()+text_width,ypos-2,xpos+
                            text_width*(given_char+1)-1,ypos+text_tall+2,ptr);
                    putimage(getx(),ypos-2,ptr,COPY_PUT);
                    bar(xpos+text_width*given_char,ypos-2,xpos+
                        text_width*(given_char+1)-1,ypos+text_tall+2);
                    for(i=string_pos;i<=max_char;i++)
                      temp[i]=temp[i+1];
                    if(string_pos-pos+given_char<strlen(temp))
                      outcharxy(xpos+text_width*given_char,ypos
                              ,temp[string_pos-pos+given_char]);
                    }
                  break;
    case LEFT_KEY  :if(string_pos>0){
                    string_pos--;
                    if(pos>0){
                     moveto(getx()-text_width,gety());
                     pos--;
                     }
                    else{
                     getimage(xpos,ypos-2,xpos+text_width*given_char-1
                              ,ypos+text_tall+2,ptr);
                     putimage(xpos+text_width,ypos-2,ptr,COPY_PUT);
                     bar(xpos,ypos-2,xpos+text_width-1,ypos+text_tall+2);
                     outcharxy(getx(),gety(),temp[string_pos]);
                     }
                    }
                  break;
    case RIGHT_KEY :if(string_pos<strlen(temp)){
                    string_pos++;
                    if(pos<given_char){
                     moveto(getx()+text_width,gety());
                     pos++;
                     }
                    else if(string_pos<max_char+1){
                     getimage(xpos+text_width,ypos-2,xpos+text_width
                             *(given_char+1),ypos+text_tall+2,ptr);
                     putimage(xpos,ypos-2,ptr,COPY_PUT);
                     bar(xpos+text_width*given_char,ypos-2
                        ,xpos+text_width*(given_char+1),ypos+text_tall+2);
                     if(string_pos<strlen(temp))
                       outcharxy(getx(),gety(),temp[string_pos]);
                     }
                    }
                  break;
    }
  }
  if(((str_char[0]==13)||(str_char[0]==27))&&(!funckey))
   end=true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        else if((edit)&&(((!funckey)&&(str_char[0]==9))||((funckey)&&((str_char[0]==UP_KEY)||
    (str_char[0]==15)||(str_char[0]==DOWN_KEY)))))
          end=true;
         }while(!end);
        if(!((ch==ESC_KEY)&&(!funckey)))
          strcpy(gettext,temp);
        farfree(ptr);
        return(str_char[0]);
        }

    void box(int x1,int y1,int x2,int y2,boolean type){
      int Save_color;
      struct fillsettingstype Save_fillstyle;

      Save_color=getcolor();
      getfillsettings(&Save_fillstyle);
      if(type){
        setcolor(color.bright);
        setfillstyle(1,color.button_normal);
        }
      else{
        setcolor(color.shadow);
        setfillstyle(1,color.button_pushed);
        }
      bar(x1+3,y1+3,x2-3,y2-3);
      line(x1+2,y1+2,x2-2,y1+2);
      line(x1+1,y1+1,x2-1,y1+1);
      line(x1,y1,x2,y1);
      line(x1+2,y1+2,x1+2,y2-2);
      line(x1+1,y1+1,x1+1,y2-1);
      line(x1,y1,x1,y2);
      if(type)
        setcolor(color.shadow);
      else
        setcolor(color.bright);
      line(x1+2,y2-2,x2-2,y2-2);
      line(x1+1,y2-1,x2-1,y2-1);
      line(x1,y2,x2,y2);
      line(x2-2,y1+2,x2-2,y2-2);
      line(x2-1,y1+1,x2-1,y2-1);
      line(x2,y1,x2,y2);
      setcolor(Save_color);
      setfillstyle(Save_fillstyle.pattern,Save_fillstyle.color);
      }

    void window_(int x1,int y1,int x2,int y2){

      int Save_color;
      struct fillsettingstype Save_fill;

      Save_color=getcolor();
      getfillsettings(&Save_fill);
      setfillstyle(1,color.shadow);
      bar(x1+4,y1+4,x2+4,y2+4);
      setfillstyle(1,color.border_color);
      bar(x1,y1,x2,y2);
      setfillstyle(1,color.frame_color);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น รวมทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    bar(x1+color.frame_width,y1+color.frame_width
       ,x2-color.frame_width,y2-color.frame_width);
    setcolor(0);
    rectangle(x1+color.frame_width,y1+color.frame_width
            ,x2-color.frame_width,y2-color.frame_width);
    rectangle(x1,y1,x2,y2);
    setcolor(0);
    line(x1+20,y1,x1+20,y1+color.frame_width);
    line(x1+20,y2,x1+20,y2-color.frame_width);
    line(x2-20,y1,x2-20,y1+color.frame_width);
    line(x2-20,y2,x2-20,y2-color.frame_width);

    line(x1+1,y1+20,x1+color.frame_width,y1+20);
    line(x2-1,y1+20,x2-color.frame_width,y1+20);
    line(x1+1,y2-20,x1+color.frame_width,y2-20);
    line(x2-1,y2-20,x2-color.frame_width,y2-20);
    setcolor(Save_color);
    setfillstyle(Save_fill.pattern,Save_fill.color);
    }

  void frame(int x1,int y1,int x2,int y2,boolean bar_ok){
    int Save_color;
    struct fillsettingstype Save_fill;

    Save_color=getcolor();
    getfillsettings(&Save_fill);
    setfillstyle(1,color.border_color);
    bar(x1,y1,x2,y2);
    setfillstyle(1,color.frame_color);
    bar(x1+color.frame_width,y1+color.frame_width
       ,x2-color.frame_width,y2-color.frame_width);
    setcolor(0);
    rectangle(x1+color.frame_width,y1+color.frame_width
            ,x2-color.frame_width,y2-color.frame_width);
    rectangle(x1,y1,x2,y2);
    box(x1+color.frame_width+2,y1+color.frame_width+2
       ,x1+color.frame_width+16,y1+color.frame_width+16,true);
    box(x2-color.frame_width-16,y1+color.frame_width+2
       ,x2-color.frame_width-2,y1+color.frame_width+16,true);
    setfillstyle(1,color.frame_title_color);
    bar(x1+color.frame_width+19,y1+color.frame_width+2
       ,x2-color.frame_width-19,y1+color.frame_width+16);

    setcolor(0);
    line(x1+20,y1,x1+20,y1+color.frame_width);
    line(x1+20,y2,x1+20,y2-color.frame_width);
    line(x2-20,y1,x2-20,y1+color.frame_width);
    line(x2-20,y2,x2-20,y2-color.frame_width);

    line(x1+1,y1+20,x1+color.frame_width,y1+20);
    line(x2-1,y1+20,x2-color.frame_width,y1+20);
    line(x1+1,y2-20,x1+color.frame_width,y2-20);
    line(x2-1,y2-20,x2-color.frame_width,y2-20);
    if(bar_ok){
      setfillstyle(1,color.bar_color);
      bar(x1+color.frame_width+3,y1+color.frame_width+18
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      ,x2-color.frame_width-3,y1+color.frame_width+33);

    setcolor(color.bright);
    line(x1+color.frame_width+2,y1+color.frame_width+18
      ,x2-color.frame_width-2,y1+color.frame_width+18);
    line(x1+color.frame_width+2,y1+color.frame_width+18
      ,x1+color.frame_width+2,y1+color.frame_width+34);

    setcolor(color.shadow);
    line(x1+color.frame_width+3,y1+color.frame_width+34
      ,x2-color.frame_width-2,y1+color.frame_width+34);
    line(x2-color.frame_width-2,y1+color.frame_width+17
      ,x2-color.frame_width-2,y1+color.frame_width+34);
    }
  setcolor(Save_color);
  setfillstyle(Save_fill.pattern,Save_fill.color);
  }

boolean prompt_execute(int x,int y,char *header,char *str){
  int x1,y1,x2,y2,x_mouse,y_mouse,save_color1,save_color2;
  void far *ptr;
  char ch=13;
  boolean execute=true;
  struct fillsettingstype fill_style;

  save_color1=getcolor();
  save_color2=color.normaltext_color;
  getfillsettings(&fill_style);
  if(strlen(str)*text_width+20<151){
    x1=x-85;
    x2=x+85;
    }
  else{
    x1=x-10-(int)(strlen(str)*text_width/2);
    x2=x+10+(int)(strlen(str)*text_width/2);
    }
  y1=y;
  y2=y+62;
  ptr=getmem(imagesize(x1,y1,x2+4,y2+4));
  getimage(x1,y1,x2+4,y2+4,ptr);
  setfillstyle(1,color.shadow);
  bar(x1+4,y1+4,x2+4,y2+4);
  frame(x1,y1,x2,y2,false);
  write_text(x1,x2,y1+10,header,color.title_text_color);
  setcolor(color.normaltext_color);
  outtextxy((x1+x2)/2-4*strlen(str),y1+25,str);
  color.normaltext_color=7;
  box(x-70,y+37,x-6,y+54,true);
  box(x+6,y+37,x+70,y+54,true);

  write_text(x-70,x-6,y+42,"OK!",color.active_text_color);
  write_text(x+6,x+70,y+42,"Cancel",color.normaltext_color);
  do{
    if(execute)
      write_text(x-70,x-6,y+42,"OK!",color.active_text_color);
    else
      write_text(x+6,x+70,y+42,"Cancel",color.active_text_color);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      mouse_cursor_on();
      do{
        }while(!kbhit()&&(!leftb_pressed()));
      mouse_cursor_off();
      write_text(x-70,x-6,y+42,"OK!",color.normaltext_color);
      write_text(x+6,x+70,y+42,"Cancel",color.normaltext_color);
      if(leftb_pressed()){
        mouse_position(&x_mouse,&y_mouse);
        if((y_mouse>y+37)&&(y_mouse<y+54)){
          funckey=false;
          ch=RETURN_KEY;
          if((x_mouse>x-70)&&(x_mouse<x-6))
            execute=true;
          else if((x_mouse>x+6)&&(x_mouse<x+70))
            execute=false;
          else funckey=true;
          }
        else{
          funckey=true;
          }
        }
      else{
        ch=readfunc();
        if(funckey)
          switch(ch){
            case LEFT_KEY  :execute=true; break;
            case RIGHT_KEY :execute=false;break;
            }
        }
      }while((funckey)||(ch!=RETURN_KEY));
     if(execute){
       box(x-70,y+37,x-6,y+54,false);
       write_text(x-70,x-6,y+42,"OK!",color.menu_hot_letter);
       if(!leftb_pressed())delay(200);
       else do{}while(leftb_pressed());
       }
     else{
       box(x+6,y+37,x+70,y+54,false);
       write_text(x+6,x+70,y+42,"Cancel",color.menu_hot_letter);
       if(!leftb_pressed())delay(200);
       else do{}while(leftb_pressed());
       }
     putimage(x1,y1,ptr,COPY_PUT);
     farfree(ptr);
     do{}while(leftb_pressed());
     color.normaltext_color=save_color2;
     setcolor(save_color1);
     setfillstyle(fill_style.pattern,fill_style.color);
     return(execute);
     }

void message(int x_pos,int y_pos,char *header,char *mess){
     void far *ptr;
     int x1,y1,x2,y2,save_color,x_mouse,y_mouse;
     struct fillsettingstype fill_info;
     boolean end;
     char ch=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    end=false;
    save_color=getcolor();
    getfillsettings(&fill_info);
    x1=x_pos;
    y1=y_pos;
    y2=y1+64;
    if(strlen(header)*text_width+34>strlen(mess)*text_width)
      x2=x_pos+strlen(header)*text_width+58;
    else
      x2=x_pos+strlen(mess)*text_width+24;
    ptr=getmem(imagesize(x1,y1,x2+4,y2+4));
    getimage(x1,y1,x2+4,y2+4,ptr);
    setfillstyle(1,color.shadow);
    bar(x1+4,y1+4,x2+4,y2+4);
    frame(x1,y1,x2,y2,false);
    write_text(x1,x2,y1+10,header,color.title_text_color);
    write_text(x1,x2,y1+25,mess,color.normaltext_color);
    box((x1+x2)/2-25,y1+38,(x1+x2)/2+25,y1+55,true);
    write_text(x1,x2,y1+43,"O.K.",color.active_text_color);
    mouse_cursor_on();
    do{
      if(kbhit()){
        ch=readfunc();
        if((ch==RETURN_KEY)&&(!funckey)) end=true;
        }
      if(leftb_pressed()){
        mouse_position(&x_mouse,&y_mouse);
        if((x_mouse>(x1+x2)/2-25)&&(x_mouse<(x1+x2)/2+25)
            &&(y_mouse>y1+38)&&(y_mouse<y1+55))
          end=true;
        }
      }while(!end);
    mouse_cursor_off();
    box((x1+x2)/2-25,y1+38,(x1+x2)/2+25,y1+55,false);
    write_text(x1,x2,y1+43,"O.K.",color.shadow);
    if(!ch)
      do{}while(leftb_pressed());
    else delay(200);
    box((x1+x2)/2-25,y1+38,(x1+x2)/2+25,y1+55,true);
    write_text(x1,x2,y1+43,"O.K.",color.active_text_color);
    delay(100);
    putimage(x1,y1,ptr,COPY_PUT);
    farfree(ptr);
    setcolor(save_color);
    setfillstyle(fill_info.pattern,fill_info.color);
    }

  void interrupt Blink(__CPPARGS){
    int save_color;

    blink_counter++;
    save_color=getcolor();
    if(blink_counter==1){
      setcolor(color.cursor_color);
      line(getx()+1,gety()+text_tall
          ,getx()+text_width-1,gety()+text_tall);
```

เอกสารนี้ใช้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      }
    else if(blink_counter==10){
      setcolor(color.readtext_background_color);
      line(getx()+1,gety()+text_tall
          ,getx()+text_width-1,gety()+text_tall);
      }
    else if(blink_counter==18)
      blink_counter=0;
    setcolor(save_color);
    Old_cursor_int();
    }

  void cursor_on(void){
    if(cursor_flag==false){
      Old_cursor_int=getvect(0x1C);
      setvect(0x1C,Blink);
      cursor_flag=true;
      }
    }

  void cursor_off(void){
    int save_color;

    if(cursor_flag==true){
      setvect(0x1C,Old_cursor_int);
      cursor_flag=false;
      save_color=getcolor();
      setcolor(color.readtext_background_color);
      line(getx()+1,gety()+text_tall
          ,getx()+text_width-1,gety()+text_tall);
      setcolor(save_color);
      blink_counter=0;
      }
    }

  void open_window(int x1,int y1,int x2,int y2,int num_win,char *txt){
    if((num_win<1)||(num_win>10)){
      message(300,200," ERROR "," Window ERROR ");
      }
    else{
      win_pos[num_win-1][0]=x1;
      win_pos[num_win-1][1]=y1;
      win_temp[num_win-1]=getmem(imagesize(x1,y1,x2+4,y2+4));
      getimage(x1,y1,x2+4,y2+4,win_temp[num_win-1]);
      setfillstyle(1,color.shadow);
      bar(x1+4,y1+4,x2+4,y2+4);
      frame(x1,y1,x2,y2,false);
      write_text(x1,x2,y1+10,txt,color.title_text_color);
      }
    }

  void release_window(int num_win){
    putimage(win_pos[num_win-1][0],win_pos[num_win-1][1],
      win_temp[num_win-1],COPY_PUT);
    farfree(win_temp[num_win-1]);
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
char _path[MAXPATH][13];
char  _cur_disk;
char _stack[MAXPATH][13];int _point;
char *_name,*_scroll;
int  _name_pos,_max_name=0,_path_pos,_max_path=0;
char _name_temp[80],_temp[80];

void create_all(char *wildcard){
  int i,j,c=0,done;
  struct find_t ffblk;
  struct ffblk n_ffblk;

  if(!strcmp(cur_path,"")){
    _cur_disk=getdisk();
    getcurdir(_cur_disk+1,cur_path);
    strcat(cur_path,"\\");
    }
  _point=0;
  _max_name=0;
  _max_path=0;
  //if(cur_path[0]==0)
  //  strcat(cur_path,"\\");
  if(cur_path[0]!='\\')
    for(i=0;i<strlen(cur_path);i++){
      if(cur_path[i]=='\\'){
        _stack[_point][c]=0;
        c=0;
       _point++;
        }
      else{
        _stack[_point][c++]=cur_path[i];
        }
      }
 if(cur_path[0]=='\\')
   sprintf(_temp,"%c:%s*.*",'A'+_cur_disk,cur_path);
 else
   sprintf(_temp,"%c:\\%s*.*",'A'+_cur_disk,cur_path);
 done=_dos_findfirst(_temp,FA_DIREC,&ffblk);
 while(!done){
   if((ffblk.attrib==FA_DIREC)&&(strcmp(ffblk.name,"."))&&
      (strcmp(ffblk.name,"..")))
     strcpy(_path[_max_path++],ffblk.name);
     done=_dos_findnext(&ffblk);
   }
 if(cur_path[0]=='\\')
   sprintf(_name_temp,"%c:%s%s",'A'+_cur_disk,cur_path,wildcard);
 else
   sprintf(_name_temp,"%c:\\%s%s",'A'+_cur_disk,cur_path,wildcard);
 done=findfirst(_name_temp,&n_ffblk,0);
 while(!done){
   strcpy((char*)(_name+13*_max_name++),n_ffblk.ff_name);
   done=findnext(&n_ffblk);
   }
 sprintf(_scroll,"%c:\\",'A'+_cur_disk);
 for(i=0;i<_point;i++){
   strcpy((char*)(_scroll+(i+1)*40),"  ");
   for(j=0;j<i;j++)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      strcat((char*)(_scroll+(i+1)*40),"  ");
    strcat((char*)(_scroll+(i+1)*40),_stack[i]);
    }
  strcpy(_temp,"  ");
  for(i=0;i<_point;i++)
   strcat(_temp,"  ");
  for(i=0;i<_max_path;i++){
    sprintf((char*)(_scroll+(_point+i+1)*40),"%s%s",_temp,_path[i]);
    }
  }

void new_screen(void){
  int i;

  setcolor(color.shadow);
  outtextxy(134,100+16,"Drive :  [A]  [B]  [C]  [D]");
  outtextxy(134,100+30,"File Name");
  outtextxy(100+183,100+30,"Directory");
  setcolor(color.normaltext_color);
  outtextxy(133,100+15,"Drive :  [A]  [B]  [C]  [D]");
  outtextxy(133,100+29,"File Name");
  outtextxy(282,100+29,"Directory");
  setfillstyle(1,color.frame_color);
  bar(282,144,526-color.frame_width-1,151);
  outtextxy(282,100+44,cur_path);
  setcolor(14);
  outcharxy(133+80+40*_cur_disk,115,'A'+_cur_disk);
  setcolor(color.normaltext_color);
  setfillstyle(1,1);
  setfillstyle(1,7);
  setcolor(0);
  bar(134,100+60,134+8*15+14,100+180);
  rectangle(134,100+60,100+168,100+180);
  line(134+8*15,100+60,100+154,100+180);
  bar(100+168+14,100+60,100+406,100+180);
  setcolor(0);
  rectangle(100+182,100+60,100+406,100+180);
  line(100+392,100+60,100+392,100+180);
  setcolor(15);
  for(i=0;i<8;i++){
    if(i<_max_name)
      outtextxy(100+46,100+68+14*i,(char*)(_name+13*i));
    }
  for(i=0;i<8;i++){
    if(i<=_max_path+_point){
      if(i==_point)
       setcolor(14);
      else
        setcolor(color.normaltext_color);
      outtextxy(100+190,100+68+14*i,(char*)(_scroll+40*i));
      }
    }
  setcolor(color.normaltext_color);
  setfillstyle(1,color.frame_color);
  bar(146+6*8,290,526-color.frame_width-1,297);
  outtextxy(146,100+190,"File :");
  box(240,306,304,326,true);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น รวมทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  write_text(240,304,312,"O.K.",7);
  box(336,306,400,326,true);
  write_text(336,400,312,"Cancle",7);
  }

void scroll_up(int screen,int n){
  void far *ptr;
  int i,x1,x2;

  ptr=getmem(imagesize(290-4,168-2,492-4,168+8));
  if(screen==2){
    x1=142;
    x2=146+8*12+4;
    }
  else{
    x1=290-4;
    x2=492-4;
    }
  for(i=0;i<7;i++){
    getimage(x1,168+14*(i+1)-2,x2,168+14*(i+1)+8,ptr);
    putimage(x1,168+14*i-2,ptr,COPY_PUT);
    }
  farfree(ptr);
  setfillstyle(1,7);
  bar(x1,168+14*7-2,x2,168+14*7+8);
  setcolor(color.normaltext_color);
  if(screen==2)
    outtextxy(100+46,100+68+14*7,(char*)(_name+13*n));
  else
    outtextxy(290,100+68+14*7,(char*)(_scroll+40*n));
  }

void scroll_dn(int screen,int n){
  int i,x1,x2;
  void far *ptr;

  ptr=getmem(imagesize(290-4,168-2,492-4,168+8));
  if(screen==2){
    x1=142;
    x2=146+8*12+4;
    }
  else{
    x1=290-4;
    x2=492-4;
    }
  for(i=7;i>0;i--){
    getimage(x1,168+14*(i-1)-2,x2,168+14*(i-1)+8,ptr);
    putimage(x1,168+14*i-2,ptr,COPY_PUT);
    }
  farfree(ptr);
  setfillstyle(1,7);
  bar(x1,168-2,x2,168+8);
  if(screen==2)
    outtextxy(100+46,100+68,(char*)(_name+13*n));
  else
    outtextxy(290,100+68,(char*)(_scroll+40*n));
  }
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
int load_save(char *file_name,int func){
  int  main_pos=1,misc_pos[6]={0,0,0,0,0,0};
  int name_point=0,dir_point;
  char temp[80],ch,fname[20];
  boolean end=false;
  void far *ptr;
  int i;
  boolean check;

  _name=getmem(1024*13);
  _scroll=getmem(1024*13);
  ptr=getmem(2048);
  create_all("*.MPJ");
  name_point=0;
  if(_point<7)
    dir_point=_point;
  else
    dir_point=7;
  misc_pos[3]=_point;
  misc_pos[0]=_cur_disk;
  frame(114,100-14,526,340,false);
  if(func==LOAD)
    write_text(114,526,110-14,"LOAD",15);
  else if(func==SAVE)
    write_text(114,526,110-14,"SAVE",15);
  new_screen();
  strcpy(fname,"*.MPJ");
  do{
    switch(main_pos){
      case 0:getimage(217+40*misc_pos[0]-16,115-2,217+40*misc_pos[0]+16,115+9,ptr);
             setfillstyle(1,color.menu_active_color);
             bar(217+40*misc_pos[0]-16,115-2,217+40*misc_pos[0]+16,115+9);
             sprintf(temp,"[%c]",'A'+misc_pos[0]);
             setcolor(15);
             write_text(217+40*misc_pos[0],217+40*misc_pos[0],115,temp,15);
             break;
      case 2:getimage(146-4,168+14*name_point-2,146+8*12+4,168+14*name_point+8,ptr);
             setfillstyle(1,color.menu_active_color);
             bar(146-4,168+14*name_point-2,146+8*12+4,168+14*name_point+8);
             setcolor(15);
             if(_max_name>0)
               outtextxy(146,168+14*name_point,(char*)(_name+13*misc_pos[2]));
             break;
      case 3:getimage(290-4,168+14*dir_point-2,492,168+14*dir_point+8,ptr);
             setfillstyle(1,color.menu_active_color);
             bar(290-4,168+14*dir_point-2,492-4,168+14*dir_point+8);
             setcolor(15);
             outtextxy(290,168+14*dir_point,(char*)(_scroll+40*(misc_pos[3])));
             break;
      case 4:getimage(240,306,304,326,ptr);
             write_text(240,304,312,"O.K.",12);
             break;
      case 5:getimage(336,306,400,326,ptr);
             write_text(336,400,312,"Cancle",12);
             break;
    }
```

เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if(main_pos==1)
  ch=readtext(148,100+44,fname,12,12,true);
else
  ch=readfunc();
switch(main_pos){
  case 0:putimage(217+40*misc_pos[0]-16,115-2,ptr,COPY_PUT);
         break;
  case 2:putimage(146-4,168+14*name_point-2,ptr,COPY_PUT);
         break;
  case 3:putimage(290-4,168+14*dir_point-2,ptr,COPY_PUT);
         break;
  case 4:putimage(240,306,ptr,COPY_PUT);
         break;
  case 5:putimage(336,306,ptr,COPY_PUT);
         break;
  }
if(funckey)
  switch(ch){
    case SHF_TAB_KEY:
                    if(main_pos>0)
                      main_pos--;
                    else
                     main_pos=5;
                    break;
    case LEFT_KEY  :if(main_pos==0){
                     if(misc_pos[0]>0)
                       misc_pos[0]--;
                     else
                       misc_pos[0]=3;
                     }
                    break;
    case RIGHT_KEY :if(main_pos==0){
                     if(misc_pos[0]<3)
                       misc_pos[0]++;
                     else
                       misc_pos[0]=0;
                     }
                   break;
    case UP_KEY    :switch(main_pos){
                      case 2:if(misc_pos[2]>0){
                               if(name_point>0)
                                 name_point--;
                               else
                                 scroll_dn(2,misc_pos[2]-1);
                               misc_pos[2]--;
                               }
                             break;
                      case 3:if(misc_pos[3]>0){
                               if(dir_point>0)
                                 dir_point--;
                               else
                                 scroll_dn(3,misc_pos[3]-1);
                               misc_pos[3]--;
                               }
                             break;
                      }
                    break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      case DOWN_KEY  :switch(main_pos){
                        case 2:if(misc_pos[2]<_max_name-1){
                                 if(name_point<7)
                                   name_point++;
                                 else
                                  scroll_up(2,misc_pos[2]+1);
                                 misc_pos[2]++;
                                 }
                               break;

                        case 3:if(misc_pos[3]<_max_path+_point){
                                 if(dir_point<7)
                                   dir_point++;
                                 else
                                   scroll_up(3,misc_pos[3]+1);
                                 misc_pos[3]++;
                                 }
                               break;
                      }
                     break;
    }
  else
    switch(ch){
      case TAB_KEY   :if(main_pos<5)
                       main_pos++;
                     else
                       main_pos=0;
                     break;
      case ESC_KEY   :end=true;
                     break;
      case RETURN_KEY:switch(main_pos){
                       case 0:if(_cur_disk!=misc_pos[0]){
                                _cur_disk=misc_pos[0];
                                getcurdir(_cur_disk+1,cur_path);
                                strcat(cur_path,"\\");
                                create_all(fname);
                                new_screen();
                                name_point=0;
                                dir_point=_point;
                                misc_pos[3]=_point;
                                misc_pos[2]=0;
                                }
                              break;
                       case 1:create_all(fname);
                              new_screen();
                              name_point=0;
                              misc_pos[2]=0;
                              check=true;
                              for(i=0;i<strlen(fname);i++)
                                if(fname[i]=='*'||fname[i]=='?')
                                  check=false;
                              if(check){
                                main_pos=4;
                                setcolor(color.normaltext_color);
                                setfillstyle(1,color.frame_color);
                                bar(146+6*8,290,526-color.frame_width-1,297);
                                setcolor(color.normaltext_color);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        if(cur_path[0]!='\\')
                          sprintf(temp,"%c:\\%s%s",'A'+_cur_disk,cur_path,fname
);
                        else
                          sprintf(temp,"%c:\\%s",'A'+_cur_disk,fname);
                        outtextxy(146+7*8,290,temp);
                        }
                      else
                        if(_max_name>0)
                          main_pos=2;
                        else
                          main_pos=3;
                      break;
              case 2:main_pos=4;
                      setcolor(color.normaltext_color);
                      setfillstyle(1,color.frame_color);
                      bar(146+6*8,290,526-color.frame_width-1,297);
                      setcolor(color.normaltext_color);
                      if(cur_path[0]!='\\')
                        sprintf(temp,"%c:\\%s%s",'A'+_cur_disk,cur_path,(char*)
(_name+13*misc_pos[2]));
                      else
                        sprintf(temp,"%c:\\%s",'A'+_cur_disk,(char*)(_name+13*m
isc_pos[2]));
                      outtextxy(146+7*8,290,temp);
                      break;
              case 3:if(misc_pos[3]>_point){
                        if(cur_path[0]=='\\')
                          strcpy(cur_path,"");
                        strcat(cur_path,_path[misc_pos[3]-_point-1]);
                        strcat(cur_path,"\\");
                        }
                      else{
                        strcpy(cur_path,"");
                        for(i=0;i<misc_pos[3];i++){
                          strcat(cur_path,_stack[i]);
                          strcat(cur_path,"\\");
                          }
                        }
                      create_all(fname);
                      new_screen();
                      dir_point=_point;
                      misc_pos[3]=_point;
                      misc_pos[2]=0;
                      name_point=0;
                      break;
              case 4:end=true;
                      box(240,306,304,326,false);
                      write_text(240,304,312,"O.K.",7);
                      delay(200);
                      box(240,306,304,326,true);
                      write_text(240,304,312,"O.K.",12);
                      delay(50);
                      break;
              case 5:end=true;
                      box(336,306,400,326,false);
                      write_text(336,400,312,"Cancle",7);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                delay(200);
                                box(336,306,400,326,true);
                                write_text(336,400,312,"Cancle",12);
                                delay(50);
                                break;
                              }
                            break;
          }
        }while(!end);
       farfree(ptr);
       farfree(_scroll);
       farfree(_name);
       if((main_pos==5)||((ch==ESC_KEY)&&(!funckey)))
         return(-1);
       else{
         strcpy(file_name,temp);
         return(0);
         }
       }

boolean read_time_arg(unsigned int *start,unsigned int *stop,unsigned int *dest){
  void far *ptr;
  char temp[7],ch;
  unsigned int time[3];
  int line=0,i;
  boolean ok=false,chk;

  time[0]=(*start+noteperbar)/noteperbar;
  time[1]=(*stop)/noteperbar;
  time[2]=1;
  ptr=getmem(imagesize(320-130,240-50,320+130+4,240+50+4));
  getimage(320-130,240-50,320+130+4,240+50+4,ptr);
  setfillstyle(1,color.shadow);
  bar(320-130+4,240-50+4,320+130+4,240+50+4);
  frame(320-130,240-50,320+130,240+50,false);
  setcolor(color.normaltext_color);
  outtextxy(320-130+10,240-25,   "Start  Time (Bar No.):");
  outtextxy(320-130+10,240-25+14,"Ending Time (Bar No.):");
  outtextxy(320-130+10,240-25+28,"Destination (Bar No.):");
  box(320-80,260,320-20,280,true);
  write_text(320-80,320-20,266,"O.K.",7);
  box(320+20,260,320+80,280,true);
  write_text(320+20,320+80,266,"Cancle",7);
  do{
    if((line>=0)&&(line<=2)){
      itoa(time[line],temp,10);
      ch=readtext(384,215+14*line,temp,5,5,true);
      chk=true;
      for(i=0;i<strlen(temp);i++){
        if((temp[i]>57)||(temp[i]<48))
          chk=false;
        }
      if(chk) time[line]=atoi(temp);
      if((chk)&&(!funckey)&&(ch==RETURN_KEY))line++;
      }
    else{
      if(line==3)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          write_text(320-80,320-20;266,"O.K.",12);
        else if(line==4)
          write_text(320+20,320+80,266,"Cancle",12);
        ch=readfunc();
        if((!funckey)&&(ch==RETURN_KEY)){
          if(line==3){
            box(320-80,260,320-20,280,false);
            write_text(320-80,320-20,266,"O.K.",12);
            }
          else if(line==4){
            box(320+20,260,320+80,280,true);
            write_text(320+20,320+80,266,"Cancle",7);
            }
          delay(200);
          ok=true;
          }
        if(line==3)
          write_text(320-80,320-20,266,"O.K.",7);
        else if(line==4)
          write_text(320+20,320+80,266,"Cancle",7);
        }
      if(funckey){
        switch(ch){
          case UP_KEY  :line--;break;
          case DOWN_KEY:line++;break;
          }
        if(line<0)line=4;
        if(line>4)line=0;
        }
      else
        if(ch==ESC_KEY){
          ok=true;
          line=4;
          }
      }while(!ok);
    *(unsigned int*)start=(unsigned int)(noteperbar*(time[0]-1));
    *(unsigned int*)stop= (unsigned int)(noteperbar*(time[1]));
    *(unsigned int*)dest= (unsigned int)(noteperbar*(time[2]-1));
    putimage(320-130,240-50,ptr,COPY_PUT);
    farfree(ptr);
    if((line==3)&&(ch==RETURN_KEY))return(true);
    else return(false);
    }

  void init_menu_command(void){
    int i;

    for(i=0;i<main_menu_no;i++){
      pull_menu[i].command_no=0;
      pull_menu[i].y_pos=40;
      }

    pull_menu[0].num_item=6; // File
    pull_menu[0].x_pos=20;
    pull_menu[0].menu_width=84;
    strcpy(pull_menu[0].command[0],"Load");     pull_menu[0].fastkey[0]=0;
    strcpy(pull_menu[0].command[1],"Save");     pull_menu[0].fastkey[1]=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
strcpy(pull_menu[0].command[2],"New");      pull_menu[0].fastkey[2]=0;
strcpy(pull_menu[0].command[3],"Comment");  pull_menu[0].fastkey[3]=0;
strcpy(pull_menu[0].command[4],"Os shell"); pull_menu[0].fastkey[4]=0;
strcpy(pull_menu[0].command[5],"Quit");     pull_menu[0].fastkey[5]=0;

pull_menu[1].num_item=9; // Track
pull_menu[1].x_pos=70;
pull_menu[1].menu_width=132;
strcpy(pull_menu[1].command[0],"Track Volume");  pull_menu[1].fastkey[0]=1;
strcpy(pull_menu[1].command[1],"Transpose");     pull_menu[1].fastkey[1]=5;
strcpy(pull_menu[1].command[2],"Clear");         pull_menu[1].fastkey[2]=0;
strcpy(pull_menu[1].command[3],"Merge Track");   pull_menu[1].fastkey[3]=0;
strcpy(pull_menu[1].command[4],"Detail");        pull_menu[1].fastkey[4]=0;
strcpy(pull_menu[1].command[5],"Tool");          pull_menu[1].fastkey[5]=0;
strcpy(pull_menu[1].command[6],"Assign Channel");pull_menu[1].fastkey[6]=0;
strcpy(pull_menu[1].command[7],"Voice Select");  pull_menu[1].fastkey[7]=0;
strcpy(pull_menu[1].command[8],"Present Track"); pull_menu[1].fastkey[8]=12;

pull_menu[2].num_item=4; // Record
pull_menu[2].x_pos=120;
pull_menu[2].menu_width=108;
strcpy(pull_menu[2].command[0],"Step");       pull_menu[2].fastkey[0]=0;
strcpy(pull_menu[2].command[1],"Realtime");   pull_menu[2].fastkey[1]=0;
strcpy(pull_menu[2].command[2],"MIDI Detail");pull_menu[2].fastkey[2]=0;
strcpy(pull_menu[2].command[3],"Score");      pull_menu[2].fastkey[3]=0;

pull_menu[3].num_item=0; // Playback
pull_menu[3].x_pos=200;
pull_menu[3].menu_width=108;

pull_menu[4].num_item=3; // Music
pull_menu[4].x_pos=270;
pull_menu[4].menu_width=92;
strcpy(pull_menu[4].command[0],"Tempo");      pull_menu[4].fastkey[0]=0;
strcpy(pull_menu[4].command[1],"Transpose");  pull_menu[4].fastkey[1]=5;
strcpy(pull_menu[4].command[2],"Note/Bar");   pull_menu[4].fastkey[2]=0;

pull_menu[5].num_item=7; // Option
pull_menu[5].x_pos=320;
pull_menu[5].menu_width=124;
strcpy(pull_menu[5].command[0],"Color");        pull_menu[5].fastkey[0]=0;
strcpy(pull_menu[5].command[1],"Directory");    pull_menu[5].fastkey[1]=0;
strcpy(pull_menu[5].command[2],"Auto Quantize");pull_menu[5].fastkey[2]=5;
strcpy(pull_menu[5].command[3],"Auto Rewind");  pull_menu[5].fastkey[3]=5;
strcpy(pull_menu[5].command[4],"VU Display");   pull_menu[5].fastkey[4]=0;
strcpy(pull_menu[5].command[5],"Clock");        pull_menu[5].fastkey[5]=4;
strcpy(pull_menu[5].command[6],"Save Option");  pull_menu[5].fastkey[6]=0;

pull_menu[6].num_item=4; // Setup
pull_menu[6].x_pos=370;
pull_menu[6].menu_width=124;
strcpy(pull_menu[6].command[0],"Voice Library");pull_menu[6].fastkey[0]=0;
strcpy(pull_menu[6].command[1],"Drum Library");pull_menu[6].fastkey[1]=0;
strcpy(pull_menu[6].command[2],"Drum Channel"); pull_menu[6].fastkey[2]=0;
strcpy(pull_menu[6].command[3],"Drum Track");   pull_menu[6].fastkey[3]=5;

pull_menu[7].num_item=0; // Drum Machine
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
pull_menu[7].x_pos=430;
pull_menu[7].menu_width=140;
strcpy(pull_menu[7].command[0],"Arrange Pattern");pull_menu[7].fastkey[0]=0;
strcpy(pull_menu[7].command[1],"Build Pattern");  pull_menu[7].fastkey[1]=0;
strcpy(pull_menu[7].command[2],"Save");           pull_menu[7].fastkey[2]=0;
strcpy(pull_menu[7].command[3],"Load");           pull_menu[7].fastkey[3]=0;
strcpy(pull_menu[7].command[4],"Play");           pull_menu[7].fastkey[4]=0;
strcpy(pull_menu[7].command[5],"New");            pull_menu[7].fastkey[5]=0;

pull_menu[8].num_item=0;
pull_menu[8].x_pos=340;
pull_menu[8].menu_width=124;

// Tool
popup_menu[0].num_item=7;
popup_menu[0].x_pos=100;
popup_menu[0].y_pos=112;
popup_menu[0].menu_width=124;
popup_menu[0].command_no=0;

strcpy(popup_menu[0].command[0],"Quantize");     popup_menu[0].fastkey[0]=0;
strcpy(popup_menu[0].command[1],"Kill Velocity");popup_menu[0].fastkey[1]=5;
strcpy(popup_menu[0].command[2],"Kill Whell");   popup_menu[0].fastkey[2]=5;
strcpy(popup_menu[0].command[3],"Kill Pressure");popup_menu[0].fastkey[3]=5;
strcpy(popup_menu[0].command[4],"Kill Program"); popup_menu[0].fastkey[4]=0;
strcpy(popup_menu[0].command[5],"Kill Control"); popup_menu[0].fastkey[5]=5;
strcpy(popup_menu[0].command[6],"Time Shift");   popup_menu[0].fastkey[6]=0;

popup_menu[1].num_item=4;
popup_menu[1].x_pos=500;
popup_menu[1].y_pos=86;
popup_menu[1].menu_width=76;
popup_menu[1].command_no=0;
strcpy(popup_menu[1].command[0],"Program"); popup_menu[1].fastkey[0]=0;
strcpy(popup_menu[1].command[1],"Realtime");popup_menu[1].fastkey[1]=0;
strcpy(popup_menu[1].command[2],"Setup");   popup_menu[1].fastkey[2]=0;
strcpy(popup_menu[1].command[3],"Delete");  popup_menu[1].fastkey[3]=0;

set_v.num_item=10;
set_v.x_pos=464;
set_v.y_pos=240;
set_v.menu_width=60;
set_v.command_no=0;
strcpy(set_v.command[0],"1  ppp");set_v.fastkey[0]=0;
strcpy(set_v.command[1],"2  pp"); set_v.fastkey[1]=0;
strcpy(set_v.command[2],"3  p");  set_v.fastkey[2]=0;
strcpy(set_v.command[3],"4  mp"); set_v.fastkey[3]=0;
strcpy(set_v.command[4],"5  m");  set_v.fastkey[4]=0;
strcpy(set_v.command[5],"6  mf"); set_v.fastkey[5]=0;
strcpy(set_v.command[6],"7  f");  set_v.fastkey[6]=0;
strcpy(set_v.command[7],"8  ff"); set_v.fastkey[7]=0;
strcpy(set_v.command[8],"9  fff");set_v.fastkey[8]=0;
strcpy(set_v.command[9],"Select");set_v.fastkey[9]=0;

step_menu.num_item=6;
step_menu.x_pos=160;
step_menu.y_pos=90;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    step_menu.menu_width=120;
    step_menu.command_no=0;
    strcpy(step_menu.command[0],"Load Track");   step_menu.fastkey[0]=0;
    strcpy(step_menu.command[1],"Save Track as");step_menu.fastkey[1]=0;
    strcpy(step_menu.command[2],"Goto Counter"); step_menu.fastkey[2]=0;
    strcpy(step_menu.command[3],"Goto Bar");     step_menu.fastkey[3]=5;
    strcpy(step_menu.command[4],"Tool");         step_menu.fastkey[4]=0;
    strcpy(step_menu.command[5],"Exit");         step_menu.fastkey[5]=0;

    quantize_menu.num_item=6;
    quantize_menu.x_pos=280;
    quantize_menu.y_pos=100;
    quantize_menu.menu_width=104;
    quantize_menu.command_no=0;
    strcpy(quantize_menu.command[0],"1  1   Note");quantize_menu.fastkey[0]=0;
    strcpy(quantize_menu.command[1],"2 1/2  Note");quantize_menu.fastkey[1]=0;
    strcpy(quantize_menu.command[2],"3 1/3  Note");quantize_menu.fastkey[2]=0;
    strcpy(quantize_menu.command[3],"4 1/4  Note");quantize_menu.fastkey[3]=0;
    strcpy(quantize_menu.command[4],"5 1/6  Note");quantize_menu.fastkey[4]=0;
    strcpy(quantize_menu.command[5],"6 1/8  Note");quantize_menu.fastkey[5]=0;
    }

int pull_menu_fx_m(pulldownmenu *active_menu){
  int x_mouse,y_mouse,i,y1,y2;
  int temp;

  temp=-1;
  mouse_position(&x_mouse,&y_mouse);
  for(i=0;i<active_menu->num_item;i++){
    y1=active_menu->y_pos+(text_tall+4)*i+3;
    y2=y1+11;
    if((y_mouse>=y1)&&(y_mouse<=y2)){
      temp=i;
      i=20;
      }
    }
  return(temp);
  }

void setmouse_pcursor(pulldownmenu *active_menu,int no){
  int y1;

  y1=active_menu->y_pos+(text_tall+4)*no+3;
  y1+=5;
  set_mouse_position(active_menu->x_pos+active_menu->menu_width,y1);
  }

unsigned char pull_down(pulldownmenu *active_menu){
  int i,test;
  void far *mainptr,*ptr;
  int save_color,x1,y1,x2,y2,x_mouse,y_mouse;
  boolean end;
  unsigned char ch;

  do{
    }while(leftb_pressed());
  x1=active_menu->x_pos;
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    y1=active_menu->y_pos;
    x2=x1+active_menu->menu_width;
    y2=(text_tall+4)*(active_menu->num_item)+y1+7;
    mainptr=getmem(imagesize(x1,y1,x2+4,y2+4));
    getimage(x1,y1,x2+4,y2+4,mainptr);
    save_color=getcolor();
    end=false;
    setfillstyle(1,color.shadow);
    bar(x1+4,y1+4,x2+4,y2+4);
    setfillstyle(1,color.bar_color);
    bar(x1,y1,x2,y2);
    setcolor(0);
    rectangle(x1,y1,x2,y2);
    for(i=0;i<active_menu->num_item;i++){
      write_menu_command(x1+text_width,y1+(text_tall+4)*i+5
        ,active_menu->command[i],active_menu->fastkey[i],false);
      }
    mouse_position(&x_mouse,&y_mouse);
    y_mouse=active_menu->y_pos+(text_tall+4)*active_menu->command_no+3;
    set_mouse_position(x_mouse,y_mouse);
    x1=active_menu->x_pos+text_width/2;
    x2=active_menu->x_pos+active_menu->menu_width-text_width/2;
    do{
      setmouse_pcursor(active_menu,active_menu->command_no);
      y1=active_menu->y_pos+(text_tall+4)*active_menu->command_no+3;
      y2=y1+11;
      ptr=getmem(imagesize(x1,y1,x2,y2));
      getimage(x1,y1,x2,y2,ptr);
      setfillstyle(1,color.menu_active_color);
      bar(x1,y1,x2,y2);
      i=active_menu->command_no;
      write_menu_command(active_menu->x_pos+text_width,active_menu->y_pos+(text_tall+4)*i+5
        ,active_menu->command[i],active_menu->fastkey[i],true);
      mouse_clicked=false;
      do{
        mouse_position(&x_mouse,&y_mouse);
        if(y_mouse<active_menu->y_pos+3)
          set_mouse_position(x_mouse,active_menu->y_pos+3);
        else if(y_mouse>active_menu->y_pos+(text_tall+4)*(active_menu->num_item-1)+14)
          set_mouse_position(x_mouse,active_menu->y_pos+(text_tall+4)*(active_menu->num_item
-1)+14);
        test=pull_menu_fx_m(active_menu);
        }while((test==-1)||((((!kbhit())&&(!leftb_pressed())&&(!rightb_pressed()))&&
          ((test==active_menu->command_no))));
      if(kbhit()){
        ch=readfunc();
        mouse_clicked=false;
        }
      else{
        mouse_clicked=true;
        }
      if((funckey)&&(!mouse_clicked)){
        switch(ch){
          case UP_KEY    :active_menu->command_no--;
                          break;
          case DOWN_KEY  :active_menu->command_no++;
                          break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }
      if(active_menu->command_no<0)
        active_menu->command_no=active_menu->num_item-1;
      if(active_menu->command_no==active_menu->num_item)
        active_menu->command_no=0;
      }
    else if(!mouse_clicked){
      for(i=0;i<active_menu->num_item;i++){
        if((ch==active_menu->command[i][active_menu->fastkey[i]])||
           ( ((ch>96)&&(ch<123))&&(ch-32==active_menu->command[i][active_menu->fastkey[i]]
) )){
          active_menu->command_no=i;
          ch=RETURN_KEY;
          funckey=false;
          putimage(x1,y1,ptr,COPY_PUT);
          farfree(ptr);
          setmouse_pcursor(active_menu,active_menu->command_no);
          y1=active_menu->y_pos+(text_tall+4)*active_menu->command_no+3;
          y2=y1+11;
          ptr=getmem(imagesize(x1,y1,x2,y2));
          getimage(x1,y1,x2,y2,ptr);
          setfillstyle(1,color.menu_active_color);
          bar(x1,y1,x2,y2);
          i=active_menu->command_no;
          write_menu_command(active_menu->x_pos+text_width,active_menu->y_pos+(text_tall+4
)*i+5
                             ,active_menu->command[i],active_menu->fastkey[i],true);
          mouse_clicked=false;
          }
        }
      }
    else if(mouse_clicked){
      if(leftb_pressed()){
        do{}while(leftb_pressed());
        mouse_clicked=false;
        ch=RETURN_KEY;
        funckey=false;
        end=true;
        }
      else if(rightb_pressed()){
        do{}while(rightb_pressed());
        mouse_clicked=false;
        ch=ESC_KEY;
        funckey=false;
        end=true;
        }
      else{
        active_menu->command_no=test;
        }
      }
    if(((!funckey)&&((ch==RETURN_KEY)||(ch==ESC_KEY)))
      ||((funckey)&&((ch==LEFT_KEY)||(ch==RIGHT_KEY)||(ch==ALT_X_KEY))))
      end=true;
    if(active_menu==&popup_menu[0]){
      if((funckey)||((ch!=RETURN_KEY)&&(ch!=ESC_KEY)))
        end=false;
      }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if((!funckey)&&(ch==RETURN_KEY)){

      if(!strcmp(active_menu->command[active_menu->command_no],"Tool")){
        ch=pull_down(&popup_menu[0]);
        if((ch==27)&&(!funckey))end=false;
        }
      else if(!strcmp(active_menu->command[active_menu->command_no],"Build Pattern")){
        ch=pull_down(&popup_menu[1]);
        if((ch==27)&&(!funckey))end=false;
        }
      }
    putimage(x1,y1,ptr,COPY_PUT);
    farfree(ptr);
    }while(!end);
  setcolor(save_color);
  x1=active_menu->x_pos;
  y1=active_menu->y_pos;
  putimage(x1,y1,mainptr,COPY_PUT);
  farfree(mainptr);
  mouse_clicked=false;
  return(ch);
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
extern int cardtype;

#define  LEFT_KEY      75
#define  RIGHT_KEY     77
#define  UP_KEY        72
#define  DOWN_KEY      80
#define  PAGEUP_KEY    73
#define  PAGEDOWN_KEY  81
#define  RETURN_KEY    13
#define  ESC_KEY       27
#define  BACKSPACE_KEY  8
#define  DELETE_KEY    83
#define  HOME_KEY      71
#define  END_KEY       79
#define  ALT_X_KEY     45
#define  CTRL_PGUP    132
#define  CRTL_PGDN    118
#define  TAB_KEY        9
#define  SHF_TAB_KEY   15
#define  INSERT_KEY    82

#define  Active_sense  0xFE
#define  Realtime      0xF8

// Channel Voice Message

#define  Note_off      0x80
#define  Note_on       0x90
#define  Key_Press     0xA0
#define  Ctrl_chng     0xB0
#define  Pgm_chng      0xC0
#define  Channel_press 0xD0
#define  Pitch_wheel   0xE0

#define  Exclusive     0xF0
#define  Endof_exc     0xF7

typedef  int boolean;
#define  true      1
#define  false     0

#define   DO            1
#define   ME            2

// Step Record Button Key

#define Shift_up  0
#define Shift_dn  1
#define Left_bar  2
#define Right_bar 3
#define Skip_back 4
#define Play      5
#define Rewind    6
#define Forward   7
#define Stop      8
#define Example   9
#define Mode     10
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
// Step Record Editing Mode

#define Put_note      0
#define Del_note      1
#define Bend          2
#define Whell         3
#define Pressure      4
#define Prog          5
#define Copy          6
#define Move          7
#define Cut           8
#define Chng_velocity 9
extern int edit_mode;


// Realtime Record and Playback Mode Assignment

#define RECORD        0
#define PLAYBACK      1
#define FASTPLAY      8
#define REWIND        2
#define FORWARD       3
#define SKIPBACK      4
#define SKIPFORWARD   5
#define STOP          6
#define PAUSE         7

#ifdef __cplusplus
        #define __CPPARGS ...
#else
        #define __CPPARGS
#endif

#define LOAD 0
#define SAVE 1

#define  main_menu_no 5

extern boolean mouse_installed,mouse_active;

typedef struct {
  int  num_item;
  int  command_no;
  int  x_pos;
  int  y_pos;
  int  menu_width;
  char command[10][16];
  int  fastkey[10];
  }pulldownmenu;

typedef struct {
  int  number[128];
  char name[128][16];
  int  max;
  }lib;
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
typedef struct{
  char frame_width,
       frame_title_color,
       border_color,
       bright,
       shadow,
       button_normal,
       button_pushed,
       screen_color,
       frame_color,
       bar_color,
       title_backgraound_color,
       title_text_color,
       menu_text_color,
       menu_hot_letter,
       active_menu_text_color,
       menu_active_color,
       normal_text_color,
       active_text_color,
       readtext_background_color,
       cursor_color;
 }color_struct;

typedef struct{
  char graphics_driver_path[40];
  char temporary_path[40];
  char lib_path[40];
  char program_path[40];
  char vcslib_name[9][11];
  boolean auto_quantize;
  boolean auto_rewind;
  boolean VU_installed;
  }option_struct;

extern color_struct color;
extern option_struct option;
extern boolean funckey;
extern int text_width;
extern int text_tall;
extern boolean mouse_installed,mouse_active;
extern char path[];
extern int noteperbar;
extern pulldownmenu pull_menu[main_menu_no],popup_menu[2],set_v,step_menu,quantize_menu;
extern int mouse_clicked;

void screen_saver(int exe);
void load_config(void);
void get_path(void);
unsigned char readfunc(void);
void outcharxy(int xpos,int ypos,char ch);
void write(char ch);
void write_text(int xpos_1,int xpos_2,int ypos,char *str,int col);
void write_menu_command(int xpos,int ypos,char *str,int hot_no,boolean active);
char readtext(int xpos,int ypos,char *gettext
              ,int max_char,int given_char,boolean edit);
void box(int x1,int y1,int x2,int y2,boolean type);
void window_(int x1,int y1,int x2,int y2);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น พร้อมทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void frame(int x1,int y1,int x2,int y2,boolean bar_ok);
boolean prompt_execute(int x,int y,char *header,char *str);
void message(int x_pos,int y_pos,char *header,char *mess);
void far *getmem(unsigned long size);
void quit(void);
void cursor_on(void);
void cursor_off(void);
void open_window(int x1,int y1,int x2,int y2,int num_win,char *txt);
void release_window(int num_win);
int load_save(char *file_name,int func,char *wildcard);
boolean read_time_arg(unsigned int *start,unsigned int *stop,unsigned int *dest);
void init_menu_command(void);
unsigned char pull_down(pulldownmenu *active_menu);
void load_lib(char *lib_name);
char lib_browser(int *number,char *name,int xpos,int ypos);
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#include <dos.h>
#include <stdio.h>
#include <stdlib.h>

typedef int boolean;

#define true  1
#define false 0

// Mouse Functions..

#define NOT_MOVED 0
#define RIGHT     1
#define LEFT      2
#define UP        3
#define DOWN      4

extern boolean mouse_installed,mouse_active;

void reset_mouse(void){
  union REGS r;

  mouse_active=true;
  r.x.ax=0; /* Reset the mouse */
  int86(0x33,&r,&r);
  if((int)(r.x.ax)!=-1){
    printf("Mouse Driver not Installed.\n");
    exit(1);
    }
  else
    mouse_installed=true;
  }

/* Turn of the mouse cursor */
void mouse_cursor_off(void){
  union REGS r;

  r.x.ax=2;/* Remove the cursor */
  int86(0x33,&r,&r);
  mouse_active=false;
  }

/* Turn on the mouse cursor */
void mouse_cursor_on(void){
  union REGS r;

  r.x.ax=1;/* Show the cursor */
  int86(0x33,&r,&r);
  mouse_active=false;
  }

/* Return true if right button is pressed , false otherwise */
boolean rightb_pressed(void){
  union REGS r;
  r.x.ax=3;/* get position and button status */
  int86(0x33,&r,&r);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(mouse_installed)
      return(r.x.bx&2);/* if it is pressed then BL=00000010 */
    else
      return(false);
    }

  /* Return true if left button is pressed , false otherwise */
  boolean leftb_pressed(void){
    union REGS r;

    r.x.ax=3;/* get position and button status */
    int86(0x33,&r,&r);
    if(mouse_installed)
      return(r.x.bx&1);/* if it is pressed then BL=00000001 */
    else
      return(false);
    }


  /* Set mouse cursor coordinates. */
  void set_mouse_position(int x,int y){
    union REGS r;

    r.x.ax=4;/* set position */
    r.x.cx=x;
    r.x.dx=y;
    int86(0x33,&r,&r);
    }

  /* Return mouse cursor coordinates. */
  void mouse_position(int *x,int *y){
    union REGS r;

    r.x.ax=3;/* get position and button status */
    int86(0x33,&r,&r);
    *x=r.x.cx;
    *y=r.x.dx;
    }

  /* Return the direction of motion */
  void mouse_direction(char *deltax,char *deltay){
    union REGS r;

    r.x.ax=11;/* get direction of motion */
    int86(0x33,&r,&r);

    if(r.x.cx>0)*deltax=RIGHT;
    else if((int)r.x.cx<0)*deltax=LEFT;
    else *deltax=NOT_MOVED;

    if(r.x.dx>0)*deltay=DOWN;
    else if((int)r.x.dx<0)*deltay=UP;
    else *deltay=NOT_MOVED;
    }

  /* Return true if mouse is moved */
  boolean moved(void){
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    char dx,dy;
    boolean click;

    mouse_direction(&dx,&dy);
    click=0;
    click|=(mouse_installed)&&(((dx!=0)||(dy!=0)));
    return(click);
    }

/* Return true if mouse button is clicked */
boolean clicked(void){
  union REGS r;'

  r.x.ax=3;/* get position and button status */
  int86(0x33,&r,&r);
  if(mouse_installed)
      return(r.x.bx&3);
  else
      return(false);
  }

// End of mouse functions..
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#include <graphics.h>
#include <conio.h>
#include <string.h>
#include <alloc.h>
#include <stdlib.h>
#include <stdio.h>
#include <dos.h>
#include "midi_d.h"
#include "mouse_d.h"

#define  maxx_allow 374
#define  mid_x      428

int start_y=80;
int start_x;
int xset=0;
int tempo_rate=120;
int pat_count=0;

extern lib  drum_lib;

char saved_file[80];
char mainmenu_command[main_menu_no][10]
    ={"File","Build","Arrange","Library","Quit"}
    ,comment_text[3][25];
int mainmenu_xpos[main_menu_no]
    ={30,78,134,206,278};
int main_menu_pos=0,popup_menu_no=-1;
boolean funckey;
color_struct color;
option_struct option;
int noteberbar;
int text_width=8;
int text_tall=8;
boolean mouse_installed,mouse_active;
lib  drum_lib;
char drum_ch=9;
char drum_track=9;

void main_screen(void);

typedef struct{
  int  bar_no;
  int  note_per_bar;
  int  misc_per_note;
  char name[40];
  char vol [32];
  char type[32];
  //char type_name[32][20];
  char perc[4][32][128];
  }pattern;

pattern *pat[256],edit_pat;
int play_pat[1024],times_pat[1024];
unsigned char *track,*track_node;

void quit(void);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้นำไปดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void midiout(int channel,unsigned char midi_data);

boolean chk_int(char *txt,int min,int max){
  boolean chk=true;
  int i;

  for(i=0;i<strlen(txt);i++)
    if((txt[i]<48)||(txt[i]>57))
      chk=false;
  if(chk){
    i=atoi(txt);
    if((i<min)||(i>max))
      chk=false;
    }
  return(chk);
  }

void lock_mouse(void){
  int x,y;

  mouse_position(&x,&y);
  do{}while(clicked());
  set_mouse_position(x,y);
  }

void init_edit_pat(void){
  int i,k,l;

  edit_pat.bar_no=1;
  edit_pat.note_per_bar=4;
  edit_pat.misc_per_note=4;
  edit_pat.name[0]=0;
  for(i=0;i<32;i++){
    edit_pat.vol[i]=0;
    edit_pat.type[i]=-1;
    //strcpy(edit_pat.type_name[i],"");
    }
  for(i=0;i<4;i++)
    for(k=0;k<32;k++)
      for(l=0;l<128;l++)
        edit_pat.perc[i][k][l]=0;
  }

char read_arg(int x,int y,char *header,char *message,char *ret_str,int max,int show){
  char ch,temp[80];

  frame(x,y,x+8*strlen(message)+8*max+38,y+48,false);
  write_text(x,x+8*strlen(message)+8*max+38,y+10,header,15);
  setcolor(color.normaltext_color);
  outtextxy(x+10,y+28,message);
  strcpy(temp,ret_str);
  ch=readtext(x+8*strlen(message)+18,y+28,temp,max,show,false);
  if((ch==RETURN_KEY)&&(!funckey))
    strcpy(ret_str,temp);
  return(ch);
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void save_pat(int pattern_no){
  char fname[80];
  FILE *fp;
  int i,j,k;

  sprintf(fname,"%spattern.%d",option.temporary_path,pattern_no);
  if((fp=fopen(fname,"wb"))==NULL){
    }
  else{// SAVE
    fprintf(fp,"Drum Machine ... Pattern File%c",26);
    fwrite(&edit_pat.bar_no,2,1,fp);
    fwrite(&edit_pat.note_per_bar,2,1,fp);
    fwrite(&edit_pat.misc_per_note,2,1,fp);
    fwrite(&edit_pat.type[0],32,1,fp);
    fwrite(edit_pat.name,40,1,fp);
    fwrite(edit_pat.vol,32,1,fp);
    for(i=0;i<edit_pat.bar_no;i++)
      for(j=0;j<32;j++)
        for(k=0;k<edit_pat.note_per_bar*edit_pat.misc_per_note;k++)
          fputc(edit_pat.perc[i][j][k],fp);
    fclose(fp);
    }
  }

void load_pat(int pattern_no){
  char fname[80];
  FILE *fp;
  int i,j,k;

  init_edit_pat();
  sprintf(fname,"%spattern.%d",option.temporary_path,pattern_no);
  if((fp=fopen(fname,"rb"))==NULL){
    }
  else{// Load
    for(j=0;j<30;j++)
      i=fgetc(fp);
    if((char)i==26){
      fread(&edit_pat.bar_no,2,1,fp);
      fread(&edit_pat.note_per_bar,2,1,fp);
      fread(&edit_pat.misc_per_note,2,1,fp);
      fread(&edit_pat.type[0],32,1,fp);
      fread(edit_pat.name,40,1,fp);
      fread(edit_pat.vol,32,1,fp);
      for(i=0;i<edit_pat.bar_no;i++)
        for(j=0;j<32;j++)
          for(k=0;k<edit_pat.note_per_bar*edit_pat.misc_per_note;k++)
            edit_pat.perc[i][j][k]=fgetc(fp);
      fclose(fp);
      }
    }
  }

void load_all_pattern(void){
  int i,j,k,m;
  FILE *fp;
  char fname[80];
```

เอกสารนี้มีลิขสิทธิ์ สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  pat_count=0;
  for(i=0;i<256;i++){
    sprintf(fname,"%spattern.%d",option.temporary_path,i);
    if((fp=fopen(fname,"rb"))!=NULL){
      pat[i]=getmem(sizeof(pattern));
      pat_count++;
      for(j=0;j<30;j++)
        m=fgetc(fp);
      if((char)m==26){
        fread(&pat[i]->bar_no,2,1,fp);
        fread(&pat[i]->note_per_bar,2,1,fp);
        fread(&pat[i]->misc_per_note,2,1,fp);
        fread(&pat[i]->type[0],32,1,fp);
        fread(&pat[i]->name,40,1,fp);
        fread(pat[i]->vol,32,1,fp);
        for(m=0;m<pat[i]->bar_no;m++)
          for(j=0;j<32;j++)
            for(k=0;k<pat[i]->note_per_bar*pat[i]->misc_per_note;k++)
              pat[i]->perc[m][j][k]=fgetc(fp);
        fclose(fp);
        }
      }
    }
  }

void destroy_all_pat(void){
  int i;

  for(i=255;i>=0;i--){
    if(pat[i])
      farfree(pat[i]);
    pat[i]=NULL;
    }
  }

void trans(void){
  FILE *fp;
  int pat_count=0;
  unsigned int timer_count=0;
  int i,j,k,l;
  unsigned char tick_count=0;
  unsigned int step,size;
  unsigned long int temp;
  boolean  find;
  char fname[80];

  track_node=getmem(65535u);
  track=track_node;
  *track++=0xFF;
  while(play_pat[pat_count]!=-1){
    load_pat(play_pat[pat_count]);
    step=96/edit_pat.misc_per_note;
    for(i=0;i<times_pat[pat_count];i++)
      for(j=0;j<edit_pat.bar_no;j++)
        for(k=0;k<edit_pat.note_per_bar*edit_pat.misc_per_note;k++){
          find=false;
          for(l=0;l<32;l++){
```

เอกสารนี้ใช้เพื่อการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น การอ้างอิงหรือเผยแพร่ต้องอ้างอิงถึงต้นฉบับของเอกสารทุกครั้งที่มีการนำไปใช้

```
            if(edit_pat.perc[j][l][k]!=0){
              if(!find){
                *(unsigned int*)track=timer_count;
                track+=2;
                *(unsigned char*)track++=tick_count;
                find=true;
                }
              *track++=0x90|drum_ch;
              *track++=edit_pat.type[l];
              *track++=(unsigned char)((edit_pat.vol[l]*127)/100);
              }
            }
          temp=(unsigned long int)timer_count*96+
            (unsigned long int)tick_count+(unsigned long int)(step-3);
          timer_count=(unsigned int)(temp/96);
          tick_count=(unsigned char)(temp%96);
          if(find){
            *track++=0xFF;
            *(unsigned int*)track=timer_count;
            track+=2;
            *(unsigned char*)track++=tick_count;
            }
          for(l=0;l<32;l++){
            if(edit_pat.perc[j][l][k]!=0){
              *track++=0x80|drum_ch;
              *track++=edit_pat.type[l];
              *track++=64;
              }
            }
          if(find)
            *track++=0xFF;
          temp=(unsigned long int)timer_count*96+
            (unsigned long int)tick_count+(unsigned long int)(3);
          timer_count=(unsigned int)(temp/96);
          tick_count=(unsigned char)(temp%96);
        }
      pat_count++;
      }
    *(unsigned int*)track=0xFFFFu;
    track=track_node;
    size=3;
    while(!((*(unsigned int*)track==0xFFFF)&&(*(track-1)==0xFF))){
      track++;
      size++;
      }
    sprintf(fname,"%sTRACK.%d",option.temporary_path,drum_track);
    fp=fopen(fname,"wb");
    fwrite(track_node,size,1,fp);
    fclose(fp);
    farfree(track_node);
    }

void init_graph(void){
    int gd=DETECT,gm,err_code;

    initgraph(&gd,&gm,option.graphics_driver_path);
    err_code=graphresult();
```

เอกสารนี้มีลิขสิทธิ์ สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(err_code!=grOk){
      printf("Graphics error: %s\n", grapherrormsg(err_code));
      printf("Press any key to halt:");
      getch();
      printf("\n\n%s\n",option.graphics_driver_path);
      exit(1); /* terminate with an error code */
      }
    text_width=textwidth(" ");
    text_tall=textheight(" ");
    }


  void osshell(void){
    char *comspec;

    comspec=getenv("COMSPEC");
    closegraph();
    printf("\n\nType EXIT to return to Drum Machine\n\n");
    system(comspec);
    init_graph();
    main_screen();
    }

  void load(char *name){
    FILE *fp;
    char fname[80];
    char command[128];
    int ch=-1,i,j,k,l;
    unsigned char chk1,chk2;
    int pat_num;

    strcpy(fname,name);
    if(strcmp(fname,"")==0){
      ch=load_save(fname,LOAD,"*.DRU");
      main_screen();
      }
    if(ch!=-1){
      // Load OK.
      strcpy(saved_file,fname);
      if((fp=fopen(fname,"rb"))!=NULL){
        for(i=0;i<34;i++){
          fgetc(fp);
          }
        chk1=fgetc(fp);
        chk2=fgetc(fp);
        if((chk1==26)&&(chk2==165)){
          // File Type is OK.
          sprintf(command,"IF EXIST %sPATTERN.$ DEL %sPATTERN.$",option.temporary_path,optio
  n.temporary_path);
          system(command);
          fread(play_pat,2048,1,fp);
          fread(times_pat,2048,1,fp);
          fread(&pat_count,2,1,fp);
          for(i=0;i<pat_count;i++){
            fread(&pat_num,2,1,fp);
            fread(&edit_pat.bar_no,2,1,fp);
            fread(&edit_pat.note_per_bar,2,1,fp);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            fread(&edit_pat.misc_per_note,2,1,fp);
            fread(&edit_pat.type[0],32,1,fp);
            fread(edit_pat.name,40,1,fp);
            fread(edit_pat.vol,32,1,fp);
            for(l=0;l<edit_pat.bar_no;l++)
              for(j=0;j<32;j++)
                for(k=0;k<edit_pat.note_per_bar*edit_pat.misc_per_note;k++)
                  edit_pat.perc[l][j][k]=fgetc(fp);
            save_pat(pat_num);
            }
          }
        fclose(fp);
        }
      }
    }

  void save(void){
    FILE *fp;
    int ch=-1,i,j,k,l;

    if(strcmp(saved_file,"")==0){
      ch=load_save(saved_file,SAVE,"*.DRU");
      main_screen();
      }
    else
      ch=1;
    if(ch!=-1){
      if((fp=fopen(saved_file,"wb"))!=NULL){
        load_all_pattern();
        fprintf(fp,"Drum Machine File.\r\n\tMIDI Project.%c%c",26,165);
        fwrite(play_pat,2048,1,fp);
        fwrite(times_pat,2048,1,fp);
        fwrite(&pat_count,2,1,fp);
        for(i=0;i<256;i++){
          if(pat[i]!=NULL){
            fwrite(&i,2,1,fp);
            fwrite(&pat[i]->bar_no,2,1,fp);
            fwrite(&pat[i]->note_per_bar,2,1,fp);
            fwrite(&pat[i]->misc_per_note,2,1,fp);
            fwrite(&pat[i]->type[0],32,1,fp);
            fwrite(&pat[i]->name,40,1,fp);
            fwrite(&pat[i]->vol,32,1,fp);
            for(l=0;l<pat[i]->bar_no;l++)
              for(j=0;j<32;j++)
                for(k=0;k<pat[i]->note_per_bar*pat[i]->misc_per_note;k++)
                  fputc(pat[i]->perc[l][j][k],fp);
            }
          }
        destroy_all_pat();
        fclose(fp);
        }
      }
    }

  void saveas(void){
    int ch=-1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    ch=load_save(saved_file,SAVE,"*.DRU");
    if(ch!=-1)
      save();
    main_screen();
    }

  void draw_edit_screen(int bar_num){
    int i,j,screen_max,step=16;

    frame(241,40,634,474,false);
    if((edit_pat.misc_per_note==1)||(edit_pat.misc_per_note==2))
      step=20;
    else if((edit_pat.misc_per_note==3)||(edit_pat.misc_per_note==4))
      step=16;
    else if(edit_pat.misc_per_note==6)
      step=14;
    else if(edit_pat.misc_per_note==8)
      step=10;
    screen_max=edit_pat.note_per_bar*edit_pat.misc_per_note;
    start_x=mid_x-step*screen_max/2+step;
    setfillstyle(1,0);
    for(j=0;j<28;j++)
      for(i=0;i<screen_max;i++){
        if(edit_pat.perc[bar_num][j][i]!=0)
          bar(start_x+i*step-1,start_y+j*14-1,start_x+i*step+1,start_y+j*14+1);
        else
          putpixel(start_x+i*step,start_y+j*14,0);
      }
    setcolor(15);
    setlinestyle(DASHED_LINE,0,0);
    for(i=0;i<=edit_pat.note_per_bar;i++)
      line(start_x-step/2+i*step*edit_pat.misc_per_note,start_y-10,
           start_x-step/2+i*step*edit_pat.misc_per_note,468);
    setlinestyle(0,0,0);
    }

  void perc_vol(int type,int level){
    struct fillsettingstype save_fill;
    int save_color;

    save_color=getcolor();
    getfillsettings(&save_fill);
    setfillstyle(1,7);
    bar(166+level/2-3,start_y+type*14-6,166+level/2+3,start_y+type*14+6);
    setcolor(15);
    line(166+level/2-3,start_y+type*14-6,166+level/2+3,start_y+type*14-6);
    line(166+level/2-3,start_y+type*14-6,166+level/2-3,start_y+type*14+6);
    setcolor(8);
    line(166+level/2-3,start_y+type*14+6,166+level/2+3,start_y+type*14+6);
    line(166+level/2+3,start_y+type*14-6,166+level/2+3,start_y+type*14+6);
    setcolor(save_color);
    setfillstyle(save_fill.pattern,save_fill.color);
    }

  void del_perc_vol(int type,int level){
    struct fillsettingstype save_fill;
    int save_color;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    save_color=getcolor();
    getfillsettings(&save_fill);
    setfillstyle(1,color.frame_color);
    bar(166+level/2-3,start_y+type*14-6,166+level/2+3,start_y+type*14+6);
    setcolor(0);
    line(166+level/2-3,start_y+type*14,166+level/2+3,start_y+type*14);
    setcolor(15);
    line(166+level/2-3,start_y+type*14+1,166+level/2+3,start_y+type*14+1);
    setcolor(save_color);
    setfillstyle(save_fill.pattern,save_fill.color);
    }

  void chk_vol(void){
    int type,x,y;
    boolean find=false;

    mouse_position(&x,&y);
    type=0;
    do{
      if((x>=166+edit_pat.vol[type]/2-3)&&(x<=166+edit_pat.vol[type]+3)&&
         (y>=start_y+type*14-6)&&(y<=start_y+type*14+6))
        find=true;
      if(!find)type++;
      }while((type<27)&&(!find));
    if(type!=27)
      do{
        mouse_position(&x,&y);
        if(x<166){
          set_mouse_position(166,y);
          x=166;
          }
        else if(x>216){
          set_mouse_position(216,y);
          x=216;
          }
        if(edit_pat.vol[type]!=(char)((x-166)/2)){
          del_perc_vol(type,edit_pat.vol[type]);
          edit_pat.vol[type]=(char)((x-166)/2);
          perc_vol(type,edit_pat.vol[type]);
          }
        }while(leftb_pressed());
    }

  void chk_type_name(void){
    int x,y;
    int type;
    int ret_type;
    boolean find=false;
    char ch;
    char temp[25];
    int save_col;
    mouse_position(&x,&y);
    do{}while(leftb_pressed());
    if((x>15)&&(x<130)){
      type=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      do{
        if((y>=start_y+type*14-4)&&(y<=start_y+type*14+4))
          find=true;
        if(!find)type++;
        }while((!find)&&(type<27));
      }
     if(find){
      save_col=color.frame_color;
      color.frame_color=7;
      ch=lib_browser(&ret_type,temp,150,80);
      color.frame_color=save_col;
      }
     if((!funckey)&&(ch==RETURN_KEY)){
      edit_pat.type[type]=(char)ret_type;
      setfillstyle(1,color.frame_color);
      bar(15,start_y+type*14-4,130,start_y+type*14+4);
      setcolor(color.normaltext_color);
      outtextxy(15,start_y+14*type-3,temp);
      }
    }

  void chk_pat_name(int pattern_no){
    char temp[80],ch;
    int x,y;

    mouse_position(&x,&y);
    if((x>241+color.frame_width+19)&&(x<634-color.frame_width-19)&&
       (y>40+color.frame_width+2)&&(y<40+color.frame_width+16)){
     do{}while(leftb_pressed());
     strcpy(temp,edit_pat.name);
     ch=readtext(297,50,temp,38,37,false);
     setfillstyle(1,color.frame_title_color);
     bar(241+color.frame_width+19,40+color.frame_width+2
        ,634-color.frame_width-19,40+color.frame_width+16);
     if((!funckey)&&(ch==RETURN_KEY))
       strcpy(edit_pat.name,temp);
     sprintf(temp,"%3d:%-39s",pattern_no,edit_pat.name);
     write_text(241,634,50,temp,15);
     }
    }

  int chk_menu(void){
    int i;
    int ret_val=0;
    int x,y;

    mouse_position(&x,&y);
    ret_val=-1;
    for(i=0;i<main_menu_no;i++){
      if((y>24)&&(y<26+text_tall+1)&&(x>mainmenu_xpos[i]-text_width)&&
         (x<mainmenu_xpos[i]+text_width*(strlen(mainmenu_command[i])+1)))
        ret_val=i;
      }
    return(ret_val);
    }

  void example(void){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    int i,j,k;

    do{
      for(i=0;(!kbhit())&&(i<edit_pat.bar_no);i++){
        for(j=0;(!kbhit())&&(j<edit_pat.note_per_bar*edit_pat.misc_per_note);j++){
          for(k=0;k<32;k++){
            if((edit_pat.type[k]!=-1)&&(edit_pat.perc[i][k][j]!=0)){
              midiout(0,0x99);
              midiout(0,(char)edit_pat.type[k]);
              midiout(0,(unsigned char)((edit_pat.vol[k]*127)/100));
              midiout(0,0x99);
              midiout(0,(char)edit_pat.type[k]);
              midiout(0,0);
              }
            }
          delay((60000u/tempo_rate)/edit_pat.misc_per_note);
          }
        }
      }while(!kbhit());
    readfunc();
    }

void build_pat(int *menu_pos,boolean *pull_active){
  int pattern_no;
  int bar_num=0,misc=0,type=0;
  int screen_max,step=16,x_mouse,y_mouse;
  void far *ptr;
  unsigned char *track;
  char ch,temp[80];
  boolean end=false;
  int i,j;

  track=getmem(2048);
  strcpy(temp,"");
  ch=read_arg(200,240,"Buile Pattern","Pattern No. :",temp,3,3);
  if((!funckey)&&(ch==RETURN_KEY))
    pattern_no=atoi(temp);
  load_pat(pattern_no);
    if((edit_pat.misc_per_note==1)||(edit_pat.misc_per_note==2))
      step=20;
    else if((edit_pat.misc_per_note==3)||(edit_pat.misc_per_note==4))
      step=16;
    else if(edit_pat.misc_per_note==6)
      step=14;
    else if(edit_pat.misc_per_note==8)
      step=10;
  screen_max=edit_pat.note_per_bar*edit_pat.misc_per_note;
  start_x=mid_x-step*screen_max/2+step;
  frame(5,40,141,474,false);
  setcolor(15);
  for(i=0;i<28;i++){
    if(edit_pat.type[i]>=0){
      j=0;
      while(drum_lib.number[j]!=edit_pat.type[i])j++;
      outtextxy(15,start_y+i*14-3,drum_lib.name[j]);
      }
    else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      outtextxy(15,start_y+i*14-3,"None.");
    }
  write_text(5,141,50,"Name",15);
  frame(141,40,241,474,false);
  for(i=0;i<28;i++){
    setcolor(0);
    line(191-30,start_y+i*14,191+30,start_y+i*14);
    setcolor(15);
    line(191-30,start_y+i*14+1,191+30,start_y+i*14+1);
    perc_vol(i,edit_pat.vol[i]);
    }
  write_text(141,241,50,"Volume",15);
  sprintf(temp,"%3d:%-39s",pattern_no,edit_pat.name);
  draw_edit_screen(0);
  write_text(241,634,50,temp,15);
  ptr=getmem(imagesize(0,0,14,14));
  do{
    getimage(start_x+misc*step-5,start_y+type*14-5,
             start_x+misc*step+5,start_y+type*14+5,ptr);
    setfillstyle(1,0);
    bar(start_x+misc*step-5,start_y+type*14-5,
             start_x+misc*step+5,start_y+type*14+5);
    if(edit_pat.perc[bar_num][type][misc]==0)
      putpixel(start_x+misc*step,start_y+type*14,15);
    else{
      setfillstyle(1,15);
      bar(start_x+misc*step-1,start_y+type*14-1,start_x+misc*step+1,start_y+type*14+1);
      }
    mouse_cursor_on();
    do{
      }while((!kbhit())&&(!leftb_pressed()));
    mouse_cursor_off();
    putimage(start_x+misc*step-5,start_y+type*14-5,ptr,COPY_PUT);
    if(kbhit())
      ch=readfunc();
    else{
      mouse_position(&x_mouse,&y_mouse);
      chk_vol();
      chk_type_name();
      chk_pat_name(pattern_no);
      i=chk_menu();
        if(i!=-1){
          *menu_pos=i;
          *pull_active=true;
          end=true;
          }
      do{}while(leftb_pressed());
      ch=0;
      if((x_mouse>=start_x-5)&&(x_mouse<=start_x+screen_max*step+5)&&
        (y_mouse>=start_y-5)&&(y_mouse<=start_y+27*14+5)){
        i=(x_mouse-start_x+step/2)/step;
        j=(y_mouse-start_y+7)/14;
        if((x_mouse>=start_x+step*i-5)&&(x_mouse<=start_x+step*i+5)&&
          (y_mouse>=start_y+14*j-5)&&(y_mouse<=start_y+14*j+5)){
          misc=i;
          type=j;
          ch=' ';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            funckey=false;
            }
          }
        }
      if(funckey){
        switch(ch){
          case UP_KEY     :if(type>0)type--;break;
          case DOWN_KEY   :if(type<27)type++;break;
          case LEFT_KEY   :if(misc>0)misc--;break;
          case RIGHT_KEY  :if(misc<screen_max-1)misc++;break;
          case ALT_X_KEY  :end=true;break;
          case SHF_TAB_KEY:if(bar_num>0){
                             bar_num--;
                             draw_edit_screen(bar_num);
                             }
                           break;
        }
       }
      else{
        switch(ch){
          case ' '       :edit_pat.perc[bar_num][type][misc]=~edit_pat.perc[bar_num][type][m
isc];
                          setfillstyle(1,color.frame_color);
                          bar(start_x+misc*step-1,start_y+type*14-1,start_x+misc*step+1,star
t_y+type*14+1);
                          if(edit_pat.perc[bar_num][type][misc]==0)
                           putpixel(start_x+misc*step,start_y+type*14,0);
                          else{
                           setfillstyle(1,0);
                           bar(start_x+misc*step-1,start_y+type*14-1,start_x+misc*step+1,st
art_y+type*14+1);
                           }
                          break;
          case ESC_KEY   :end=true;
                          break;
          case RETURN_KEY:example();
                          break;
          case TAB_KEY   :if(bar_num<edit_pat.bar_no-1){
                           bar_num++;
                           draw_edit_screen(bar_num);
                           }
                          break;
        }
      }
    }while(!end);
   farfree(ptr);
   save_pat(pattern_no);
   farfree(track);
   }

void draw_group(int group){
  switch(group){
    case 0:rectangle(110,140,110+15*8+4,140+28+4);break;
    case 1:rectangle(110,182,110+15*8+4,182+4*14+4);break;
    case 2:rectangle(245,140,245+16*8+4,140+4+84);break;
    }
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void setup_pat(void){
  int  set_n[3]={1,0,3};
  int  set_nmax[3]={1,3,5};
  int  group=0,pat;
  char pat_name[40],ch;
  boolean end=false;

  strcpy(pat_name,"");
  ch=read_arg(150,200,"Setup Pattern","Pattern Number : ",pat_name,3,3);
  if((ch==RETURN_KEY)&&(!funckey)){
    if(chk_int(pat_name,0,255)){
      pat=atoi(pat_name);
      load_pat(pat);
      strcpy(pat_name,edit_pat.name);
      set_n[0]=edit_pat.note_per_bar-3;
      set_n[1]=edit_pat.bar_no-1;
      switch(edit_pat.misc_per_note){
        case 1:set_n[2]=0;break;
        case 2:set_n[2]=1;break;
        case 3:set_n[2]=2;break;
        case 4:set_n[2]=3;break;
        case 6:set_n[2]=4;break;
        case 8:set_n[2]=5;break;
        }
      frame(100,100,540,400,true);
      setcolor(color.normaltext_color);
      outtextxy(115,145,"( ) 3 Note/Bar");
      outtextxy(115,159,"( ) 4 Note/Bar");
      outtextxy(115,187,"( ) 1 Bar");
      outtextxy(115,201,"( ) 2 Bar");
      outtextxy(115,215,"( ) 3 Bar");
      outtextxy(115,229,"( ) 4 Bar");
      outtextxy(250,145,"( ) 1 Misc/Note");
      outtextxy(250,159,"( ) 2 Misc/Note");
      outtextxy(250,173,"( ) 3 Misc/Note");
      outtextxy(250,187,"( ) 4 Misc/Note");
      outtextxy(250,201,"( ) 6 Misc/Note");
      outtextxy(250,215,"( ) 8 Misc/Note");
      outtextxy(115,263,"Name:");
      outtextxy(115,277,pat_name);
      outtextxy(115+8,145+14*set_n[0],"*");
      outtextxy(115+8,187+14*set_n[1],"*");
      outtextxy(250+8,145+14*set_n[2],"*");
      rectangle(110,140,110+15*8+4,140+28+4);
      do{
        if(group!=3)
          ch=readfunc();
        else
          ch=readtext(115,277,pat_name,39,39,true);
        if((funckey)&&(ch==SHF_TAB_KEY)){
          setcolor(color.frame_color);
          draw_group(group);
          if(group>0)group--;
          else group=5;
          setcolor(15);
          draw_group(group);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }
      else if((!funckey)&&(ch==TAB_KEY)){
        setcolor(color.frame_color);
        draw_group(group);
        if(group<5)group++;
        else group=0;
        setcolor(15);
        draw_group(group);
        }
      else{
        if(!funckey)switch(ch){
          case ESC_KEY   :end=true;break;
          }
        else switch(ch){
          case UP_KEY    :if(group<3){
                            setcolor(color.frame_color);
                            switch(group){
                              case 0:outtextxy(115+8,145+14*set_n[0],"*");break;
                              case 1:outtextxy(115+8,187+14*set_n[1],"*");break;
                              case 2:outtextxy(250+8,145+14*set_n[2],"*");break;
                              }
                            if(set_n[group]>0)
                              set_n[group]--;
                            else
                              set_n[group]=set_nmax[group];
                            setcolor(color.normaltext_color);
                            switch(group){
                              case 0:outtextxy(115+8,145+14*set_n[0],"*");break;
                              case 1:outtextxy(115+8,187+14*set_n[1],"*");break;
                              case 2:outtextxy(250+8,145+14*set_n[2],"*");break;
                              }
                            }
                          break;
          case DOWN_KEY  :if(group<3){
                            setcolor(color.frame_color);
                            switch(group){
                              case 0:outtextxy(115+8,145+14*set_n[0],"*");break;
                              case 1:outtextxy(115+8,187+14*set_n[1],"*");break;
                              case 2:outtextxy(250+8,145+14*set_n[2],"*");break;
                              }
                            if(set_n[group]<set_nmax[group])
                              set_n[group]++;
                            else
                              set_n[group]=0;
                            setcolor(color.normaltext_color);
                            switch(group){
                              case 0:outtextxy(115+8,145+14*set_n[0],"*");break;
                              case 1:outtextxy(115+8,187+14*set_n[1],"*");break;
                              case 2:outtextxy(250+8,145+14*set_n[2],"*");break;
                              }
                            }
                          break;
          }
        }
      }while(!end);
    }
  else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        message(150,200,"Error","  Invalid Number  ");
       }
      strcpy(edit_pat.name,pat_name);
      edit_pat.note_per_bar=set_n[0]+3;
      edit_pat.bar_no=set_n[1]+1;
      switch(set_n[2]){
        case 0:edit_pat.misc_per_note=1;break;
        case 1:edit_pat.misc_per_note=2;break;
        case 2:edit_pat.misc_per_note=3;break;
        case 3:edit_pat.misc_per_note=4;break;
        case 4:edit_pat.misc_per_note=6;break;
        case 5:edit_pat.misc_per_note=8;break;
        }
      save_pat(pat);
      main_screen();
      }

    void draw_pattern_no(int pat_no,int arr_index,int col,boolean edt){
      char temp[5];

      if(edt)
        setfillstyle(1,0);
      else
        setfillstyle(1,color.frame_color);
      bar(xset+63+4,90+14*arr_index-3,xset+63+4+32+4,90+14*arr_index+7+3);
      setcolor(col);
      sprintf(temp,"%3d.",pat_no);
      outtextxy(xset+63+4,90+14*arr_index,temp);
      }

    void draw_arr_no(int arr_no,int arr_index,int col,boolean edt){
      char temp[5];

      if(edt)
        setfillstyle(1,0);
      else
        setfillstyle(1,color.frame_color);
      bar(xset+15+4,90+14*arr_index-3,xset+15+4+32,90+14*arr_index+7+3);
      setcolor(col);
      sprintf(temp,"%3d.",arr_no);
      outtextxy(xset+15+4,90+14*arr_index,temp);
      }

    void draw_name(int p_pat,int arr_index,int col){
      setfillstyle(1,color.frame_color);
      bar(xset+115,90+14*arr_index,xset+435,90+14*arr_index+7);
      setcolor(col);
      if(pat[p_pat])
        if(strcmp(pat[p_pat]->name,"")!=0)
          write_text(xset+115,xset+435,90+14*arr_index,pat[p_pat]->name,col);
        else
          write_text(xset+115,xset+435,90+14*arr_index,"No Name.",col);
      else
        write_text(xset+115,xset+435,90+14*arr_index,"Pattern have not been built.",col);
      }

    void draw_times(int times,int arr_index,int col,boolean edt){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    char temp[50];

    if(edt)
      setfillstyle(1,0);
    else
      setfillstyle(1,color.frame_color);
    bar(xset+451-2,90+14*arr_index-3,xset+475+10,90+14*arr_index+7+3);
    setcolor(col);
    sprintf(temp,"%3d",times);
    outtextxy(xset+451,90+14*arr_index,temp);
    }

  void insert_arr(int pos){
    int i,j;

    i=0;
    while(play_pat[i+pos]!=-1)i++;
    for(j=pos+i+1;j>pos;j--){
      play_pat[j]=play_pat[j-1];
      times_pat[j]=times_pat[j-1];
      }
    play_pat[pos]=0;
    times_pat[pos]=0;
    }

  void del_arr(int pos){
    int i;

    if(play_pat[pos]!=-1){
      i=pos;
      do{
        play_pat[i]=play_pat[i+1];
        times_pat[i]=times_pat[i+1];
        i++;
        }while(play_pat[i]!=-1);
      }
    }

  void del_screen(int arr_index){
    int i;
    void far *ptr;

    ptr=getmem(imagesize(5,0,474,14));
    for(i=arr_index;i<26;i++){
      getimage(xset+69,90+14*(i+1),xset+474,90+14*(i+1)+10,ptr);
      putimage(xset+69,90+14*i,ptr,COPY_PUT);
      }
    farfree(ptr);
    }

  void insert_screen(int arr_index){
    int i;
    void far *ptr;

    ptr=getmem(imagesize(5,0,474,14));
    for(i=26;i>arr_index;i--){
      getimage(xset+69,90+14*(i-1),xset+474,90+14*(i-1)+10,ptr);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        putimage(xset+69,90+14*i,ptr,COPY_PUT);
        }
      farfree(ptr);
      }

    void draw_number(int first){
      int i;

      for(i=0;i<27;i++)
        draw_arr_no(i+first,i,15,false);
      }

    void save_arrange(void){
      char name[80];
      FILE *fp;

      sprintf(name,"%sDRUMMCHN.ARG",option.temporary_path);
      fp=fopen(name,"wb");
      if(fp){
        fprintf(fp,"Drum Machine Arrangement");
        fputc(26,fp);
        fputc(165,fp);
        fwrite(play_pat,2048,1,fp);
        fwrite(times_pat,2048,1,fp);
        fclose(fp);;
        }
      }

    void load_arrange(void){
      char name[80];
      FILE *fp;
      int i,chk;

      sprintf(name,"%sDRUMMCHN.ARG",option.temporary_path);
      fp=fopen(name,"rb");
      if(fp){
        for(i=0;i<26;i++){
          chk=fgetc(fp);
          }
        if(chk==165){
          fread(play_pat,2048,1,fp);
          fread(times_pat,2048,1,fp);
          }
        fclose(fp);;
        }
      }

    void arrange_pattern(void){
      int i;
      int first_arr=0;
      int mode=0,arr_index=0,arr_no=0;
      char temp[80],ch,ch1;
      boolean end=false;

      load_all_pattern();
      frame(xset+5,40,xset+494,474,true);
      i=0;
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    while((play_pat[i]!=-1)&&(i<27)){
      draw_pattern_no(play_pat[i],i,15,false);
      draw_name(play_pat[i],i,15);
      draw_times(times_pat[i],i,15,false);
      i++;
      }
    draw_number(first_arr);
    do{
       if(play_pat[arr_no]==-1)
         mode=0;
       draw_arr_no(arr_no,arr_index,14,false);
       if(play_pat[arr_no]!=-1){
        draw_pattern_no(play_pat[arr_no],arr_index,14,false);
        draw_name(play_pat[arr_no],arr_index,14);
        draw_times(times_pat[arr_no],arr_index,14,false);
        }
       else{
         setfillstyle(1,color.frame_color);
         bar(xset+15+4,90+14*arr_index-3,xset+475+10,90+14*arr_index+7+3);
         write_text(xset+115,xset+435,90+14*arr_index,"End of Arrangement",color.normaltext_
color);
        }
      switch(mode){
       case 0:draw_arr_no(arr_no,arr_index,14,true);
              break;
       case 1:draw_pattern_no(play_pat[arr_no],arr_index,14,true);
              break;
       case 2:draw_times(times_pat[arr_no],arr_index,14,true);
              break;
       }
      ch=readfunc();
      draw_arr_no(arr_no,arr_index,15,false);
      if(play_pat[arr_no]!=-1){
       draw_pattern_no(play_pat[arr_no],arr_index,15,false);
       draw_name(play_pat[arr_no],arr_index,15);
       draw_times(times_pat[arr_no],arr_index,15,false);
       }
      if(funckey){
       switch(ch){
        case LEFT_KEY   :if(mode>0)mode--;break;
        case RIGHT_KEY  :if(mode<2)mode++;break;
        case UP_KEY     :if(arr_no>0){
                           arr_no--;
                           if(arr_index>0)
                             arr_index--;
                           else{
                             insert_screen(0);
                             first_arr--;
                             draw_number(first_arr);
                             }
                           }
                         break;
        case DOWN_KEY   :if(play_pat[arr_no]!=-1){
                           arr_no++;
                           if(arr_index<26)
                             arr_index++;
                           else{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        del_screen(0);
                        first_arr++;
                        draw_number(first_arr);
                        }
                      }
                    break;
        case ALT_X_KEY  :end=true;break;
        case DELETE_KEY :del_screen(arr_index);
                         del_arr(arr_no);
                         break;
        case INSERT_KEY :insert_screen(arr_index);
                         insert_arr(arr_no);
                         break;
        }
      }
     else{
       switch(ch){
         case ESC_KEY    :end=true;break;
         case RETURN_KEY :switch(mode){
                           case 1:sprintf(temp,"%d",play_pat[arr_no]);
                                  ch1=readtext(xset+69,90+14*arr_index,temp,
                                    3,3,false);
                                  if((!funckey)&&(ch1==RETURN_KEY))
                                   if(chk_int(temp,0,255))
                                    play_pat[arr_no]=atoi(temp);
                                  break;
                           case 2:sprintf(temp,"%d",times_pat[arr_no]);
                                  ch1=readtext(xset+451,90+14*arr_index,temp,
                                    3,3,false);
                                  if((!funckey)&&(ch1==RETURN_KEY))
                                   if(chk_int(temp,0,256))
                                    times_pat[arr_no]=atoi(temp);
                                  break;
                          }
                         break;
       }
      }
    }while(!end);
   destroy_all_pat();
   }

int main_menu_fx_m(void){
  int x_mouse,y_mouse,i;
  int temp;

  temp=-1;
  mouse_position(&x_mouse,&y_mouse);
  for(i=0;i<main_menu_no;i++){
    if((x_mouse>mainmenu_xpos[i]-8)&&(x_mouse<mainmenu_xpos[i]+8*strlen(mainmenu_command[i
])+8)){
      temp=i;
      i=main_menu_no;
    }
   }
  return(temp);
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void main_screen(void){
  int i;

  i=color.frame_color;
  color.frame_color=color.screen_color;
  frame(0,0,639,479,true);
  color.frame_color=i;
  write_text(0,639,10,"<---  Drum Machine  --->",color.title_text_color);
  for(i=0;i<main_menu_no;i++){
    write_menu_command(mainmenu_xpos[i],26,mainmenu_command[i],0,false);
    }
  }

void setmouse_cursor(int no){
  int x,y;

  x=mainmenu_xpos[no]-text_width+4*strlen(mainmenu_command[no]);
  y=34;
  set_mouse_position(x,y);
  }

void quit(void){
  trans();
  closegraph();
  save_arrange();
  exit(0);
  }

void menu(void){
  int main_menu_pos=0,i,x1,y1,x2,y2,x_mouse,y_mouse,test;
  boolean pull_down_active=false;
  unsigned char ch;
  void far *ptr;

  do{
    x1=mainmenu_xpos[main_menu_pos]-text_width;
    y1=24;
    x2=mainmenu_xpos[main_menu_pos]+text_width
      *(strlen(mainmenu_command[main_menu_pos])+1);
    y2=26+text_tall+1;
    ptr=getmem(imagesize(x1,y1,x2,y2));
    getimage(x1,y1,x2,y2,ptr);
    setfillstyle(1,color.menu_active_color);
    bar(x1,y1,x2,y2);
    write_menu_command(mainmenu_xpos[main_menu_pos],26
                    ,mainmenu_command[main_menu_pos],0,true);
    setfillstyle(1,0);
    if(!pull_down_active){
      do{
        setmouse_cursor(main_menu_pos);
        mouse_clicked=false;
        do{
          mouse_position(&x_mouse,&y_mouse);
          if(x_mouse<22) set_mouse_position(22,y_mouse);
          else if(x_mouse>616) set_mouse_position(616,y_mouse);
          test=main_menu_fx_m();
          }while((test==-1)||((!kbhit())&&(!leftb_pressed())&&(!rightb_pressed()))&&
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            ((test==main_menu_pos))));
        if(kbhit())
          mouse_clicked=false;
        else
          mouse_clicked=true;
        }while((!kbhit())&&(!mouse_clicked));
      if(!mouse_clicked)
        ch=readfunc();
      }
    else{
      if(pull_menu[main_menu_pos].num_item!=0){
        ch=pull_down(&pull_menu[main_menu_pos]);
        }
      else{
        setmouse_cursor(main_menu_pos);
        do{
          mouse_clicked=false;
          do{
            mouse_position(&x_mouse,&y_mouse);
            if(x_mouse<22) set_mouse_position(22,y_mouse);
            else if(x_mouse>616) set_mouse_position(616,y_mouse);
            test=main_menu_fx_m();
            }while((test==-1)||((!kbhit())&&(!leftb_pressed())&&(!rightb_pressed()))&&
               ((test==main_menu_pos))));
          if(kbhit())
            mouse_clicked=false;
          else
            mouse_clicked=true;
          }while((!kbhit())&&(!mouse_clicked));
        if(!mouse_clicked)
          ch=readfunc();
        }
      }
    putimage(x1,y1,ptr,COPY_PUT);
    if(mouse_clicked){ // mouse is moved or button is pressed
      if(leftb_pressed()){// select menu command and execute
        mouse_clicked=false;
        ch=RETURN_KEY;
        funckey=false;
        }
      else if(rightb_pressed()){ // cancel menu command not execute
        mouse_clicked=false;
        ch=ESC_KEY;
        funckey=false;
        }
      else{
        main_menu_pos=test;  // test = main_menu_fx_m()
        }
      }
    if((funckey)&&(!mouse_clicked)){
      switch(ch){
        case RIGHT_KEY :main_menu_pos++;
                        break;
        case LEFT_KEY  :main_menu_pos--;
                        break;
        }
      if(main_menu_pos<0)main_menu_pos=main_menu_no-1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(main_menu_pos==main_menu_no)main_menu_pos=0;
    }
   else{
    switch(ch){
     case ESC_KEY   :pull_down_active=false;
     default        :if((ch>96)&&(ch<123)&&(!mouse_clicked))
                       ch-=32;
                     for(i=0;i<main_menu_no;i++){
                       if(ch==mainmenu_command[i][0]){
                         main_menu_pos=i;
                         i=main_menu_no;
                         pull_down_active=true;
                         }
                       }
                     break;
    }
   }
  putimage(x1,y1,ptr,COPY_PUT);
  farfree(ptr);
  if((!funckey)&&(ch==RETURN_KEY)){
   do{}while(leftb_pressed());
   if((pull_down_active)&&(pull_menu[main_menu_pos].num_item!=0)){
    //Execute menu command
    pull_down_active=false;
    mouse_clicked=false;
    ch=ESC_KEY;
    switch(main_menu_pos){
     // From Pull down menu
     case 0:switch(pull_menu[main_menu_pos].command_no){
              case 0:load("");break;
              case 1:save();break;
              case 4:saveas();break;
              }
            break;
     case 1:switch(pull_menu[main_menu_pos].command_no){
              case 0:build_pat(&main_menu_pos,&pull_down_active);
                     main_screen();
                     break;
              case 1:setup_pat();break;
              }
            break;
     case 2:switch(pull_menu[main_menu_pos].command_no){
              case 1:arrange_pattern();
                     main_screen();
                     break;
              }
            break;
     case 4:switch(pull_menu[main_menu_pos].command_no){
              case 0:osshell();break;//os shell
              case 1:quit();break;
              }
            break;
    }
   }
  else{
   if(pull_menu[main_menu_pos].num_item==0){
    //Execute main menu command
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            switch(main_menu_pos){
              // There is'n pull down menu for this main menu
              }
            pull_down_active=false;
            }
          else{// First time you press Return key
            pull_down_active=true;
            }
          }
        }
      }while((!funckey)||(ch!=ALT_X_KEY));
    }

void main(int param_count,char *arg[]){
  int i;

  if(param_count>1)
    drum_ch=atoi(arg[1]);
  if(param_count>2)
    drum_track=atoi(arg[2]);
  if(param_count>3){
    strcpy(saved_file,arg[3]);
    load(saved_file);
    }
  load_config();
  load_lib("PERCSTD.LIB");
  strcpy(saved_file,"");
  init_menu_command();
  init_graph();
  reset_mouse();
  for(i=0;i<1024;i++){
    play_pat[i]=-1;
    times_pat[i]=0;
    }
  for(i=0;i<256;i++)
    pat[i]=NULL;
  play_pat[0]=-1;
  for(i=0;i<256;i++){
    pat[i]=NULL;
    }
  init_edit_pat();
  load_arrange();
  main_screen();
  menu();
  quit();
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <graphics.h>
#include <string.h>
#include <alloc.h>
#include <dos.h>
#include <dir.h>
#include "midi_d.h"
#include "mouse_d.h"

void far interrupt ( *Old_cursor_int)(void);

int blink_counter=0;
boolean cursor_flag=false;
void far *win_temp[10];int win_pos[10][2];
char cur_path[80]="",path[40];
pulldownmenu pull_menu[main_menu_no],popup_menu[2],set_v,step_menu,quantize_menu;
int mouse_clicked=false;

extern color_struct color;
extern lib  drum_lib;

void far *getmem(unsigned long size){
  char far *ptr;

  //if((ptr=(char far *)farcalloc(size,sizeof(char)))==NULL){
  if((ptr=(char far*)farmalloc(size))==NULL){
    closegraph();
    printf("Not enough memory !\n");
    printf("Program Abort\n");
    quit();
    }
  return(ptr);
  }

void midiout(int channel,unsigned char midi_data){
  char buffer_status;
  unsigned data_port,control_port;

  switch(channel){
    case 0:data_port=0x2B0;
           control_port=0x2B1;
           break;
    case 1:data_port=0x2A0;
           control_port=0x2A1;
           break;
    }
  do{
    outportb(control_port,0);
    buffer_status=inportb(control_port)&0x04;
    }while(buffer_status==0);
  outportb(data_port,midi_data);
  }

unsigned char readfunc(void){
  unsigned char temp;
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    temp=getch();
    if(temp==0x00){
      temp=getch();
      funckey=true;
      }
    else
      funckey=false;
    return(temp);
    }

  void get_path(void){
    strcpy(path,"X:\\");
    path[0]='A'+getdisk();
    getcurdir(0,path+3);
    if(path[strlen(path)-1]!=0x5C)
      strcat(path,"\\");
    }

  void load_config(void){
    int i;
    FILE *config;

    get_path();
    if((config=fopen("MIDIPROJ.CFG","rb"))==NULL){
      color.frame_width=4;
      color.frame_title_color=0;
      color.border_color=7;
      color.bright=7;
      color.shadow=8;
      color.button_normal=15;
      color.button_pushed=7;
      color.screen_color=11;
      color.frame_color=2;
      color.bar_color=7;
      color.title_backgraound_color=2;
      color.title_text_color=15;
      color.menu_text_color=15;
      color.menu_hot_letter=14;
      color.menu_active_color=1;
      color.active_menu_text_color=7;
      color.normaltext_color=15;
      color.active_text_color=12;
      color.readtext_background_color=1;
      color.cursor_color=15;

      strcpy(option.graphics_driver_path,path);
      strcat(option.graphics_driver_path,"DRIVER\\");
      sprintf(option.temporary_path,"%s%s",path,"TEMP\\");
      strcpy(option.lib_path,path);
      strcpy(option.program_path,path);
      for(i=0;i<9;i++)
        strcpy(option.vcslib_name[i],"");
      strcpy(option.vcslib_name[0],"GM.VCS");
      strcpy(option.vcslib_name[1],"SP.VCS");
      strcpy(option.vcslib_name[2],"K1.VCS");
      strcpy(option.vcslib_name[3],"KN650.VCS");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      strcpy(option.vcslib_name[4],"DRUM.VCS");
      //strcpy(option.vcslib_name[8],"PERCSTD.LIB");
      option.auto_quantize=false;
      option.auto_rewind=true;
      option.VU_installed=true;
      }
    else{
      fread(&color,sizeof(color_struct),1,config);
      fread(&option,sizeof(option_struct),1,config);
      fclose(config);
      }
    }

  void outcharxy(int xpos,int ypos,char ch){
    char temp[2];

    temp[1]=0;
    temp[0]=ch;
    outtextxy(xpos,ypos,temp);
    }

  void write(char ch){
    char temp[2];
    int save_color;

    save_color=getcolor();
    setcolor(color.shadow);
    temp[1]=0;
    temp[0]=ch;
    outtextxy(getx()+1,gety()+1,temp);
    setcolor(color.normaltext_color);
    outtext(temp);
    setcolor(save_color);
    }

  void write_text(int xpos_1,int xpos_2,int ypos,char *str,int col){
    int save_color,xpos;

    xpos=(xpos_1+xpos_2)/2-(text_width/2)*strlen(str);
    save_color=getcolor();
    setcolor(color.shadow);
    outtextxy(xpos+1,ypos+1,str);
    setcolor(col);
    outtextxy(xpos,ypos,str);
    setcolor(save_color);
    }

  void write_menu_command(int xpos,int ypos,char *str,int hot_no,boolean active){
    int save_color;

    save_color=getcolor();
    setcolor(color.shadow);
    outtextxy(xpos+1,ypos+1,str);
    if(active)
      setcolor(color.active_menu_text_color);
    else
      setcolor(color.menu_text_color);
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งนี้ต้องอ้างอิงถึงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  outtextxy(xpos,ypos,str);
  if(!active){
    setcolor(color.menu_hot_letter);
    outcharxy(xpos+text_width*hot_no,ypos,*(str+hot_no));
    }
  setcolor(save_color);
  }

char readtext(int xpos,int ypos,char *gettext
           ,int max_char,int given_char,boolean edit){
  int  string_pos,pos,i;
  char temp[100],ch;
  unsigned char str_char[2];
  void far *ptr;
  boolean end=false;
  boolean first=true;

  if(strlen(gettext)>max_char){
    closegraph();
    printf("Error in module Readtext\n");
    exit(0);
    }
  strcpy(temp,gettext);
  setfillstyle(1,color.readtext_background_color);
  bar(xpos-2,ypos-3,xpos+2+text_width*(given_char+1),ypos+2+text_tall);
  moveto(xpos,ypos);
  setcolor(color.normaltext_color);
  string_pos=strlen(temp);
  if(strlen(temp)<=given_char){
    outtext(temp);
    moveto(xpos+text_width*strlen(temp),ypos);
    pos=string_pos;
    }
  else{
    outtext(&temp[strlen(temp)-given_char]);
    moveto(xpos+text_width*given_char,ypos);
    pos=given_char;
    }
  setcolor(color.normaltext_color);
  str_char[1]=0;
  ptr=getmem(imagesize(0,0,(given_char+2)*text_width,13));
  end=false;
  do{
    cursor_on();
    //mouse_cursor_on();
    str_char[0]=readfunc();
    //mouse_cursor_off();
    cursor_off();
    if(!funckey){                    // What you type is Ascii
      if(str_char[0]>=32){           // What you type is String char
        if(first){
          strcpy(temp,"");
          pos=0;
          string_pos=0;
          setfillstyle(1,color.readtext_background_color);
          bar(xpos-2,ypos-3,xpos+2+text_width*(given_char+1),ypos+2+text_tall);
          moveto(xpos,ypos);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        first=false;
        }
      if(strlen(temp)<max_char){   // Accept this character
        if(pos<given_char){        // Cursor is in given bar
          getimage(xpos+text_width*pos,ypos-2
                 ,xpos+text_width*given_char,ypos+2+text_tall,ptr);
          putimage(xpos+text_width*(pos+1),ypos-2,ptr,COPY_PUT);
          bar(getx(),gety(),getx()+text_width-1,gety()+text_tall);
          pos++;
          outtext((char*)str_char);
          }
        else{// Cursor is to the end of given bar
          bar(xpos+text_width*(given_char),ypos-2
             ,xpos+text_width*(given_char+1)-1,ypos+text_tall+2);
          outcharxy(getx(),gety(),str_char[0]);
          getimage(xpos+text_width,ypos-2,xpos+text_width*(given_char+1)
                 ,ypos+text_tall+2,ptr);
          putimage(xpos,ypos-2,ptr,COPY_PUT);
          bar(xpos+text_width*(given_char),ypos-2
             ,xpos+text_width*(given_char+1)-1,ypos+text_tall+2);
          if(string_pos<strlen(temp))
            outcharxy(getx(),gety(),temp[string_pos]);
          }
        for(i=max_char;i>=string_pos;i--)
          temp[i+1]=temp[i];
        temp[string_pos]=str_char[0];
        //outtext((char*)str_char);
        string_pos++;
        }
      }
    else if(str_char[0]==BACKSPACE_KEY){
      first=false;
      if(string_pos>0){
        if(pos>0){
          getimage(getx(),ypos-2,xpos+text_width*(given_char+1)-1
                 ,ypos+text_tall+2,ptr);
          putimage(getx()-text_width,ypos-2,ptr,COPY_PUT);
          bar(xpos+text_width*given_char,ypos-2,xpos+text_width*(given_char+1)-1
             ,ypos+text_tall+2);
          if(string_pos-pos+given_char<strlen(temp))
            outcharxy(xpos+text_width*given_char,ypos
                    ,temp[string_pos-pos+given_char+1]);
          moveto(getx()-text_width,gety());
          pos--;
          }
        string_pos--;
        for(i=string_pos;i<max_char;i++)
          temp[i]=temp[i+1];
        }
      }
    }
  else{
    first=false;                   // What you type is scan code
    switch(str_char[0]){
      case HOME_KEY  :bar(xpos-2,ypos-2,xpos+2+text_width*(given_char+1)
                        ,ypos+2+text_tall);
                      ch=temp[given_char+1];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น รวมทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                temp[given_char+1]=0;
                outtextxy(xpos,ypos,temp);
                temp[given_char+1]=ch;
                moveto(xpos,ypos);
                string_pos=0;
                pos=0;
                break;
case END_KEY    :if(strlen(temp)<=given_char){
                  pos=strlen(temp);
                  moveto(xpos+text_width*pos,ypos);
                  }
                 else if(strlen(temp)-string_pos<given_char-pos){
                  pos+=strlen(temp)-string_pos;
                  moveto(xpos+text_width*pos,ypos);
                  }
                 else{
                  bar(xpos-2,ypos-2,xpos+2+text_width*(given_char+1)
                   ,ypos+2+text_tall);
                  outtextxy(xpos,ypos,&temp[strlen(temp)-given_char]);
                  moveto(xpos+text_width*given_char,ypos);
                  pos=given_char;
                  }
                 string_pos=strlen(temp);
                 break;
case DELETE_KEY:if(string_pos<strlen(temp)){
                  getimage(getx()+text_width,ypos-2,xpos+
                           text_width*(given_char+1)-1,ypos+text_tall+2,ptr);
                  putimage(getx(),ypos-2,ptr,COPY_PUT);
                  bar(xpos+text_width*given_char,ypos-2,xpos+
                    text_width*(given_char+1)-1,ypos+text_tall+2);
                  for(i=string_pos;i<=max_char;i++)
                   temp[i]=temp[i+1];
                  if(string_pos-pos+given_char<strlen(temp))
                   outcharxy(xpos+text_width*given_char,ypos
                             ,temp[string_pos-pos+given_char]);
                  }
                 break;
case LEFT_KEY  :if(string_pos>0){
                  string_pos--;
                  if(pos>0){
                    moveto(getx()-text_width,gety());
                    pos--;
                    }
                  else{
                    getimage(xpos,ypos-2,xpos+text_width*given_char-1
                            ,ypos+text_tall+2,ptr);
                    putimage(xpos+text_width,ypos-2,ptr,COPY_PUT);
                    bar(xpos,ypos-2,xpos+text_width-1,ypos+text_tall+2);
                    outcharxy(getx(),gety(),temp[string_pos]);
                    }
                  }
                 break;
case RIGHT_KEY :if(string_pos<strlen(temp)){
                  string_pos++;
                  if(pos<given_char){
                    moveto(getx()+text_width,gety());
                    pos++;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        }
                      else if(string_pos<max_char+1){
                        getimage(xpos+text_width,ypos-2,xpos+text_width
                                *(given_char+1),ypos+text_tall+2,ptr);
                        putimage(xpos,ypos-2,ptr,COPY_PUT);
                        bar(xpos+text_width*given_char,ypos-2
                           ,xpos+text_width*(given_char+1),ypos+text_tall+2);
                        if(string_pos<strlen(temp))
                          outcharxy(getx(),gety(),temp[string_pos]);
                        }
                      }
                    break;
          }
        }
        if(((str_char[0]==13)||(str_char[0]==27))&&(!funckey))
          end=true;
        else if((edit)&&(((!funckey)&&(str_char[0]==9))||((funckey)&&((str_char[0]==UP_KEY)||
(str_char[0]==15)||(str_char[0]==DOWN_KEY)))))
          end=true;
      }while(!end);
     if(!((ch==ESC_KEY)&&(!funckey)))
       strcpy(gettext,temp);
     farfree(ptr);
     return(str_char[0]);
     }

void box(int x1,int y1,int x2,int y2,boolean type){
  int Save_color;
  struct fillsettingstype Save_fillstyle;

  Save_color=getcolor();
  getfillsettings(&Save_fillstyle);
  if(type){
    setcolor(color.bright);
    setfillstyle(1,color.button_normal);
    }
  else{
    setcolor(color.shadow);
    setfillstyle(1,color.button_pushed);
    }
  bar(x1+3,y1+3,x2-3,y2-3);
  line(x1+2,y1+2,x2-2,y1+2);
  line(x1+1,y1+1,x2-1,y1+1);
  line(x1,y1,x2,y1);
  line(x1+2,y1+2,x1+2,y2-2);
  line(x1+1,y1+1,x1+1,y2-1);
  line(x1,y1,x1,y2);
  if(type)
    setcolor(color.shadow);
  else
    setcolor(color.bright);
  line(x1+2,y2-2,x2-2,y2-2);
  line(x1+1,y2-1,x2-1,y2-1);
  line(x1,y2,x2,y2);
  line(x2-2,y1+2,x2-2,y2-2);
  line(x2-1,y1+1,x2-1,y2-1);
  line(x2,y1,x2,y2);
```

เอกสารนี้จัดทำขึ้นสำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  setcolor(Save_color);
  setfillstyle(Save_fillstyle.pattern,Save_fillstyle.color);
  }

void window_(int x1,int y1,int x2,int y2){

  int Save_color;
  struct fillsettingstype Save_fill;

  Save_color=getcolor();
  getfillsettings(&Save_fill);
  setfillstyle(1,color.shadow);
  bar(x1+4,y1+4,x2+4,y2+4);
  setfillstyle(1,color.border_color);
  bar(x1,y1,x2,y2);
  setfillstyle(1,color.frame_color);
  bar(x1+color.frame_width,y1+color.frame_width
     ,x2-color.frame_width,y2-color.frame_width);
  setcolor(0);
  rectangle(x1+color.frame_width,y1+color.frame_width
          ,x2-color.frame_width,y2-color.frame_width);
  rectangle(x1,y1,x2,y2);
  setcolor(0);
  line(x1+20,y1,x1+20,y1+color.frame_width);
  line(x1+20,y2,x1+20,y2-color.frame_width);
  line(x2-20,y1,x2-20,y1+color.frame_width);
  line(x2-20,y2,x2-20,y2-color.frame_width);

  line(x1+1,y1+20,x1+color.frame_width,y1+20);
  line(x2-1,y1+20,x2-color.frame_width,y1+20);
  line(x1+1,y2-20,x1+color.frame_width,y2-20);
  line(x2-1,y2-20,x2-color.frame_width,y2-20);
  setcolor(Save_color);
  setfillstyle(Save_fill.pattern,Save_fill.color);
  }

void frame(int x1,int y1,int x2,int y2,boolean bar_ok){
  int Save_color;
  struct fillsettingstype Save_fill;

  Save_color=getcolor();
  getfillsettings(&Save_fill);
  setfillstyle(1,color.border_color);
  bar(x1,y1,x2,y2);
  setfillstyle(1,color.frame_color);
  bar(x1+color.frame_width,y1+color.frame_width
     ,x2-color.frame_width,y2-color.frame_width);
  setcolor(0);
  rectangle(x1+color.frame_width,y1+color.frame_width
          ,x2-color.frame_width,y2-color.frame_width);
  rectangle(x1,y1,x2,y2);
  box(x1+color.frame_width+2,y1+color.frame_width+2
     ,x1+color.frame_width+16,y1+color.frame_width+16,true);
  box(x2-color.frame_width-16,y1+color.frame_width+2
     ,x2-color.frame_width-2,y1+color.frame_width+16,true);
  setfillstyle(1,color.frame_title_color);
  bar(x1+color.frame_width+19,y1+color.frame_width+2
```

เอกสารนี้ ... ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อนุญาตให้ใช้เพื่อการศึกษาเท่านั้น ต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      ,x2-color.frame_width-19,y1+color.frame_width+16);

    setcolor(0);
    line(x1+20,y1,x1+20,y1+color.frame_width);
    line(x1+20,y2,x1+20,y2-color.frame_width);
    line(x2-20,y1,x2-20,y1+color.frame_width);
    line(x2-20,y2,x2-20,y2-color.frame_width);

    line(x1+1,y1+20,x1+color.frame_width,y1+20);
    line(x2-1,y1+20,x2-color.frame_width,y1+20);
    line(x1+1,y2-20,x1+color.frame_width,y2-20);
    line(x2-1,y2-20,x2-color.frame_width,y2-20);

    if(bar_ok){
      setfillstyle(1,color.bar_color);
      bar(x1+color.frame_width+3,y1+color.frame_width+18
        ,x2-color.frame_width-3,y1+color.frame_width+33);

      setcolor(color.bright);
      line(x1+color.frame_width+2,y1+color.frame_width+18
        ,x2-color.frame_width-2,y1+color.frame_width+18);
      line(x1+color.frame_width+2,y1+color.frame_width+18
        ,x1+color.frame_width+2,y1+color.frame_width+34);

      setcolor(color.shadow);
      line(x1+color.frame_width+3,y1+color.frame_width+34
        ,x2-color.frame_width-2,y1+color.frame_width+34);
      line(x2-color.frame_width-2,y1+color.frame_width+17
        ,x2-color.frame_width-2,y1+color.frame_width+34);
      }
    setcolor(Save_color);
    setfillstyle(Save_fill.pattern,Save_fill.color);
    }

boolean prompt_execute(int x,int y,char *header,char *str){
  int x1,y1,x2,y2,x_mouse,y_mouse,save_color1,save_color2;
  void far *ptr;
  char ch=13;
  boolean execute=true;
  struct fillsettingstype fill_style;

  save_color1=getcolor();
  save_color2=color.normaltext_color;
  getfillsettings(&fill_style);
  if(strlen(str)*text_width+20<151){
    x1=x-85;
    x2=x+85;
    }
  else{
    x1=x-10-(int)(strlen(str)*text_width/2);
    x2=x+10+(int)(strlen(str)*text_width/2);
    }
  y1=y;
  y2=y+62;
  ptr=getmem(imagesize(x1,y1,x2+4,y2+4));
  getimage(x1,y1,x2+4,y2+4,ptr);
  setfillstyle(1,color.shadow);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    bar(x1+4,y1+4,x2+4,y2+4);
    frame(x1,y1,x2,y2,false);
    write_text(x1,x2,y1+10,header,color.title_text_color);
    setcolor(color.normaltext_color);
    outtextxy((x1+x2)/2-4*strlen(str),y1+25,str);
    color.normaltext_color=7;
    box(x-70,y+37,x-6,y+54,true);
    box(x+6,y+37,x+70,y+54,true);

    write_text(x-70,x-6,y+42,"OK!",color.active_text_color);
    write_text(x+6,x+70,y+42,"Cancel",color.normaltext_color);
    do{
      if(execute)
        write_text(x-70,x-6,y+42,"OK!",color.active_text_color);
      else
        write_text(x+6,x+70,y+42,"Cancel",color.active_text_color);
      mouse_cursor_on();
      do{
        }while(!kbhit()&&(!leftb_pressed()));
      mouse_cursor_off();
      write_text(x-70,x-6,y+42,"OK!",color.normaltext_color);
      write_text(x+6,x+70,y+42,"Cancel",color.normaltext_color);
      if(leftb_pressed()){
        mouse_position(&x_mouse,&y_mouse);
        if((y_mouse>y+37)&&(y_mouse<y+54)){
          funckey=false;
          ch=RETURN_KEY;
          if((x_mouse>x-70)&&(x_mouse<x-6))
            execute=true;
          else if((x_mouse>x+6)&&(x_mouse<x+70))
            execute=false;
          else funckey=true;
          }
        else{
          funckey=true;
          }
        }
      else{
        ch=readfunc();
        if(funckey)
          switch(ch){
            case LEFT_KEY  :execute=true; break;
            case RIGHT_KEY :execute=false;break;
            }
        }
      }while((funckey)||(ch!=RETURN_KEY));
    if(execute){
      box(x-70,y+37,x-6,y+54,false);
      write_text(x-70,x-6,y+42,"OK!",color.menu_hot_letter);
      if(!leftb_pressed())delay(200);
      else do{}while(leftb_pressed());
      }
    else{
      box(x+6,y+37,x+70,y+54,false);
      write_text(x+6,x+70,y+42,"Cancel",color.menu_hot_letter);
      if(!leftb_pressed())delay(200);
      else do{}while(leftb_pressed());
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }
   putimage(x1,y1,ptr,COPY_PUT);
   farfree(ptr);
   do{}while(leftb_pressed());
   color.normaltext_color=save_color2;
   setcolor(save_color1);
   setfillstyle(fill_style.pattern,fill_style.color);
   return(execute);
   }

void message(int x_pos,int y_pos,char *header,char *mess){
   void far *ptr;
   int x1,y1,x2,y2,save_color,x_mouse,y_mouse;
   struct fillsettingstype fill_info;
   boolean end;
   char ch=0;

   end=false;
   save_color=getcolor();
   getfillsettings(&fill_info);
   x1=x_pos;
   y1=y_pos;
   y2=y1+64;
   if(strlen(header)*text_width+34>strlen(mess)*text_width)
     x2=x_pos+strlen(header)*text_width+58;
   else
     x2=x_pos+strlen(mess)*text_width+24;
   ptr=getmem(imagesize(x1,y1,x2+4,y2+4));
   getimage(x1,y1,x2+4,y2+4,ptr);
   setfillstyle(1,color.shadow);
   bar(x1+4,y1+4,x2+4,y2+4);
   frame(x1,y1,x2,y2,false);
   write_text(x1,x2,y1+10,header,color.title_text_color);
   write_text(x1,x2,y1+25,mess,color.normaltext_color);
   box((x1+x2)/2-25,y1+38,(x1+x2)/2+25,y1+55,true);
   write_text(x1,x2,y1+43,"O.K.",color.active_text_color);
   mouse_cursor_on();
   do{
     if(kbhit()){
       ch=readfunc();
       if((ch==RETURN_KEY)&&(!funckey)) end=true;
       }
     if(leftb_pressed()){
       mouse_position(&x_mouse,&y_mouse);
       if((x_mouse>(x1+x2)/2-25)&&(x_mouse<(x1+x2)/2+25)
           &&(y_mouse>y1+38)&&(y_mouse<y1+55))
         end=true;
       }
     }while(!end);
   mouse_cursor_off();
   box((x1+x2)/2-25,y1+38,(x1+x2)/2+25,y1+55,false);
   write_text(x1,x2,y1+43,"O.K.",color.shadow);
   if(!ch)
     do{}while(leftb_pressed());
   else delay(200);
   box((x1+x2)/2-25,y1+38,(x1+x2)/2+25,y1+55,true);
   write_text(x1,x2,y1+43,"O.K.",color.active_text_color);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ทำซ้ำ อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    delay(100);
    putimage(x1,y1,ptr,COPY_PUT);
    farfree(ptr);
    setcolor(save_color);
    setfillstyle(fill_info.pattern,fill_info.color);
    }

  void interrupt Blink(__CPPARGS){
    int save_color;

    blink_counter++;
    save_color=getcolor();
    if(blink_counter==1){
      setcolor(color.cursor_color);
      line(getx()+1,gety()+text_tall
          ,getx()+text_width-1,gety()+text_tall);
      }
    else if(blink_counter==10){
      setcolor(color.readtext_background_color);
      line(getx()+1,gety()+text_tall
          ,getx()+text_width-1,gety()+text_tall);
      }
    else if(blink_counter==18)
      blink_counter=0;
    setcolor(save_color);
    Old_cursor_int();
    }

  void cursor_on(void){
    if(cursor_flag==false){
      Old_cursor_int=getvect(0x1C);
      setvect(0x1C,Blink);
      cursor_flag=true;
      }
    }

  void cursor_off(void){
    int save_color;

    if(cursor_flag==true){
      setvect(0x1C,Old_cursor_int);
      cursor_flag=false;
      save_color=getcolor();
      setcolor(color.readtext_background_color);
      line(getx()+1,gety()+text_tall
          ,getx()+text_width-1,gety()+text_tall);
      setcolor(save_color);
      blink_counter=0;
      }
    }

  void open_window(int x1,int y1,int x2,int y2,int num_win,char *txt){
    if((num_win<1)||(num_win>10)){
      message(300,200," ERROR "," Window ERROR ");
      }
    else{
      win_pos[num_win-1][0]=x1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    win_pos[num_win-1][1]=y1;
    win_temp[num_win-1]=getmem(imagesize(x1,y1,x2+4,y2+4));
    getimage(x1,y1,x2+4,y2+4,win_temp[num_win-1]);
    setfillstyle(1,color.shadow);
    bar(x1+4,y1+4,x2+4,y2+4);
    frame(x1,y1,x2,y2,false);
    write_text(x1,x2,y1+10,txt,color.title_text_color);
    }
  }

void release_window(int num_win){
  putimage(win_pos[num_win-1][0],win_pos[num_win-1][1],
    win_temp[num_win-1],COPY_PUT);
  farfree(win_temp[num_win-1]);
  }

char _path[MAXPATH][13];
char  _cur_disk;
char _stack[MAXPATH][13];int _point;
char *_name,*_scroll;
int  _name_pos, _max_name=0,_path_pos,_max_path=0;
char _name_temp[80],_temp[80];

void create_all(char *wildcard){
  int i,j,c=0,done;
  struct find_t ffblk;
  struct ffblk n_ffblk;

  if(!strcmp(cur_path,"")){
    _cur_disk=getdisk();
    getcurdir(_cur_disk+1,cur_path);
    strcat(cur_path,"\\");
    }
  _point=0;
  _max_name=0;
  _max_path=0;
  //if(cur_path[0]==0)
  //  strcat(cur_path,"\\");
  if(cur_path[0]!='\\')
    for(i=0;i<strlen(cur_path);i++){
      if(cur_path[i]=='\\'){
        _stack[_point][c]=0;
        c=0;
       _point++;
        }
      else{
        _stack[_point][c++]=cur_path[i];
        }
      }
  if(cur_path[0]=='\\')
    sprintf(_temp,"%c:%s*.*",'A'+_cur_disk,cur_path);
  else
    sprintf(_temp,"%c:\\%s*.*",'A'+_cur_disk,cur_path);
  done=_dos_findfirst(_temp,FA_DIREC,&ffblk);
  while(!done){
    if((ffblk.attrib==FA_DIREC)&&(strcmp(ffblk.name,"."))&&
       (strcmp(ffblk.name,".."))
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      strcpy(_path[_max_path++],ffblk.name);
      done=_dos_findnext(&ffblk);
     }
    if(cur_path[0]=='\\')
     sprintf(_name_temp,"%c:%s%s",'A'+_cur_disk,cur_path,wildcard);
    else
     sprintf(_name_temp,"%c:\\%s%s",'A'+_cur_disk,cur_path,wildcard);
    done=findfirst(_name_temp,&n_ffblk,0);
    while(!done){
     strcpy((char*)(_name+13*_max_name++),n_ffblk.ff_name);
     done=findnext(&n_ffblk);
     }
    sprintf(_scroll,"%c:\\",'A'+_cur_disk);
    for(i=0;i<_point;i++){
     strcpy((char*)(_scroll+(i+1)*40),"  ");
     for(j=0;j<i;j++)
      strcat((char*)(_scroll+(i+1)*40),"  ");
     strcat((char*)(_scroll+(i+1)*40),_stack[i]);
     }
    strcpy(_temp,"  ");
    for(i=0;i<_point;i++)
     strcat(_temp,"  ");
    for(i=0;i<_max_path;i++){
     sprintf((char*)(_scroll+(_point+i+1)*40),"%s%s",_temp,_path[i]);
     }
   }

  void new_screen(void){
   int i;

   setcolor(color.shadow);
   outtextxy(134,100+16,"Drive :  [A]  [B]  [C]  [D]");
   outtextxy(134,100+30,"File Name");
   outtextxy(100+183,100+30,"Directory");
   setcolor(color.normaltext_color);
   outtextxy(133,100+15,"Drive :  [A]  [B]  [C]  [D]");
   outtextxy(133,100+29,"File Name");
   outtextxy(282,100+29,"Directory");
   setfillstyle(1,color.frame_color);
   bar(282,144,526-color.frame_width-1,151);
   outtextxy(282,100+44,cur_path);
   setcolor(14);
   outcharxy(133+80+40*_cur_disk,115,'A'+_cur_disk);
   setcolor(color.normaltext_color);
   setfillstyle(1,1);
   setfillstyle(1,7);
   setcolor(0);
   bar(134,100+60,134+8*15+14,100+180);
   rectangle(134,100+60,100+168,100+180);
   line(134+8*15,100+60,100+154,100+180);
   bar(100+168+14,100+60,100+406,100+180);
   setcolor(0);
   rectangle(100+182,100+60,100+406,100+180);
   line(100+392,100+60,100+392,100+180);
   setcolor(15);
   for(i=0;i<8;i++){
    if(i<_max_name)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        outtextxy(100+46,100+68+14*i,(char*)(_name+13*i));
      }
    for(i=0;i<8;i++){
      if(i<=_max_path+_point){
        if(i==_point)
         setcolor(14);
        else
          setcolor(color.normaltext_color);
        outtextxy(100+190,100+68+14*i,(char*)(_scroll+40*i));
        }
      }
    setcolor(color.normaltext_color);
    setfillstyle(1,color.frame_color);
    bar(146+6*8,290,526-color.frame_width-1,297);
    outtextxy(146,100+190,"File :");
    box(240,306,304,326,true);
    write_text(240,304,312,"O.K.",7);
    box(336,306,400,326,true);
    write_text(336,400,312,"Cancle",7);
    }

  void scroll_up(int screen,int n){
    void far *ptr;
    int i,x1,x2;

    ptr=getmem(imagesize(290-4,168-2,492-4,168+8));
    if(screen==2){
      x1=142;
      x2=146+8*12+4;
      }
    else{
      x1=290-4;
      x2=492-4;
      }
    for(i=0;i<7;i++){
      getimage(x1,168+14*(i+1)-2,x2,168+14*(i+1)+8,ptr);
      putimage(x1,168+14*i-2,ptr,COPY_PUT);
      }
    farfree(ptr);
    setfillstyle(1,7);
    bar(x1,168+14*7-2,x2,168+14*7+8);
    setcolor(color.normaltext_color);
    if(screen==2)
      outtextxy(100+46,100+68+14*7,(char*)(_name+13*n));
    else
      outtextxy(290,100+68+14*7,(char*)(_scroll+40*n));
    }

  void scroll_dn(int screen,int n){
    int i,x1,x2;
    void far *ptr;

    ptr=getmem(imagesize(290-4,168-2,492-4,168+8));
    if(screen==2){
      x1=142;
      x2=146+8*12+4;
      }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    else{
      x1=290-4;
      x2=492-4;
      }
    for(i=7;i>0;i--){
      getimage(x1,168+14*(i-1)-2,x2,168+14*(i-1)+8,ptr);
      putimage(x1,168+14*i-2,ptr,COPY_PUT);
      }
    farfree(ptr);
    setfillstyle(1,7);
    bar(x1,168-2,x2,168+8);
    if(screen==2)
      outtextxy(100+46,100+68,(char*)(_name+13*n));
    else
      outtextxy(290,100+68,(char*)(_scroll+40*n));
    }

int load_save(char *file_name,int func,char *wildcard){
  int  main_pos=1,misc_pos[6]={0,0,0,0,0,0};
  int name_point=0,dir_point;
  char temp[80],ch,fname[20];
  boolean end=false;
  void far *ptr;
  int i;
  boolean check;

  _name=getmem(1024*13);
  _scroll=getmem(1024*13);
  ptr=getmem(2048);
  create_all(wildcard);
  name_point=0;
  if(_point<7)
    dir_point=_point;
  else
    dir_point=7;
  misc_pos[3]=_point;
  misc_pos[0]=_cur_disk;
  frame(114,100-14,526,340,false);
  if(func==LOAD)
    write_text(114,526,110-14,"LOAD",15);
  else if(func==SAVE)
    write_text(114,526,110-14,"SAVE",15);
  new_screen();
  strcpy(fname,wildcard);
  do{
    switch(main_pos){
      case 0:getimage(217+40*misc_pos[0]-16,115-2,217+40*misc_pos[0]+16,115+9,ptr);
             setfillstyle(1,color.menu_active_color);
             bar(217+40*misc_pos[0]-16,115-2,217+40*misc_pos[0]+16,115+9);
             sprintf(temp,"[%c]",'A'+misc_pos[0]);
             setcolor(15);
             write_text(217+40*misc_pos[0],217+40*misc_pos[0],115,temp,15);
             break;
      case 2:getimage(146-4,168+14*name_point-2,146+8*12+4,168+14*name_point+8,ptr);
             setfillstyle(1,color.menu_active_color);
             bar(146-4,168+14*name_point-2,146+8*12+4,168+14*name_point+8);
             setcolor(15);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        if(_max_name>0)
          outtextxy(146,168+14*name_point,(char*)(_name+13*misc_pos[2]));
        break;
   case 3:getimage(290-4,168+14*dir_point-2,492,168+14*dir_point+8,ptr);
        setfillstyle(1,color.menu_active_color);
        bar(290-4,168+14*dir_point-2,492-4,168+14*dir_point+8);
        setcolor(15);
        outtextxy(290,168+14*dir_point,(char*)(_scroll+40*(misc_pos[3])));
        break;
   case 4:getimage(240,306,304,326,ptr);
        write_text(240,304,312,"O.K.",12);
        break;
   case 5:getimage(336,306,400,326,ptr);
        write_text(336,400,312,"Cancle",12);
        break;
 }
if(main_pos==1)
  ch=readtext(148,100+44,fname,12,12,true);
else
  ch=readfunc();
switch(main_pos){
  case 0:putimage(217+40*misc_pos[0]-16,115-2,ptr,COPY_PUT);
        break;
  case 2:putimage(146-4,168+14*name_point-2,ptr,COPY_PUT);
        break;
  case 3:putimage(290-4,168+14*dir_point-2,ptr,COPY_PUT);
        break;
  case 4:putimage(240,306,ptr,COPY_PUT);
        break;
  case 5:putimage(336,306,ptr,COPY_PUT);
        break;
  }
if(funckey)
  switch(ch){
    case SHF_TAB_KEY:
                  if(main_pos>0)
                    main_pos--;
                  else
                    main_pos=5;
                  break;
    case LEFT_KEY  :if(main_pos==0){
                    if(misc_pos[0]>0)
                      misc_pos[0]--;
                    else
                      misc_pos[0]=3;
                    }
                  break;
    case RIGHT_KEY :if(main_pos==0){
                    if(misc_pos[0]<3)
                      misc_pos[0]++;
                    else
                      misc_pos[0]=0;
                    }
                  break;
    case UP_KEY    :switch(main_pos){
                    case 2:if(misc_pos[2]>0){
                             if(name_point>0)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                name_point--;
                              else
                                scroll_dn(2,misc_pos[2]-1);
                              misc_pos[2]--;
                              }
                            break;
                    case 3:if(misc_pos[3]>0){
                              if(dir_point>0)
                                dir_point--;
                              else
                                scroll_dn(3,misc_pos[3]-1);
                              misc_pos[3]--;
                              }
                            break;
                    }
                  break;
  case DOWN_KEY  :switch(main_pos){
                    case 2:if(misc_pos[2]<_max_name-1){
                              if(name_point<7)
                                name_point++;
                              else
                                scroll_up(2,misc_pos[2]+1);
                              misc_pos[2]++;
                              }
                            break;

                    case 3:if(misc_pos[3]<_max_path+_point){
                              if(dir_point<7)
                                dir_point++;
                              else
                                scroll_up(3,misc_pos[3]+1);
                              misc_pos[3]++;
                              }
                            break;
                    }
                  break;
  }
else
  switch(ch){
    case TAB_KEY   :if(main_pos<5)
                      main_pos++;
                    else
                      main_pos=0;
                    break;
    case ESC_KEY   :end=true;
                    break;
    case RETURN_KEY:switch(main_pos){
                      case 0:if(_cur_disk!=misc_pos[0]){
                               _cur_disk=misc_pos[0];
                               getcurdir(_cur_disk+1,cur_path);
                               strcat(cur_path,"\\");
                               create_all(fname);
                               new_screen();
                               name_point=0;
                               dir_point=_point;
                               misc_pos[3]=_point;
                               misc_pos[2]=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                }
                break;
        case 1:create_all(fname);
                new_screen();
                name_point=0;
                misc_pos[2]=0;
                check=true;
                for(i=0;i<strlen(fname);i++)
                  if(fname[i]=='*'||fname[i]=='?')
                    check=false;
                if(check){
                  main_pos=4;
                  setcolor(color.normaltext_color);
                  setfillstyle(1,color.frame_color);
                  bar(146+6*8,290,526-color.frame_width-1,297);
                  setcolor(color.normaltext_color);
                  if(cur_path[0]!='\\')
                    sprintf(temp,"%c:\\%s%s",'A'+_cur_disk,cur_path,fname
);
                  else
                    sprintf(temp,"%c:\\%s",'A'+_cur_disk,fname);
                  outtextxy(146+7*8,290,temp);
                  }
                else
                  if(_max_name>0)
                    main_pos=2;
                  else
                    main_pos=3;
                break;
        case 2:main_pos=4;
                setcolor(color.normaltext_color);
                setfillstyle(1,color.frame_color);
                bar(146+6*8,290,526-color.frame_width-1,297);
                setcolor(color.normaltext_color);
                if(cur_path[0]!='\\')
                  sprintf(temp,"%c:\\%s%s",'A'+_cur_disk,cur_path,(char*)
(_name+13*misc_pos[2]));
                else
                  sprintf(temp,"%c:\\%s",'A'+_cur_disk,(char*)(_name+13*m
isc_pos[2]));
                outtextxy(146+7*8,290,temp);
                break;
        case 3:if(misc_pos[3]>_point){
                if(cur_path[0]=='\\')
                  strcpy(cur_path,"");
                strcat(cur_path,_path[misc_pos[3]-_point-1]);
                strcat(cur_path,"\\");
                }
               else{
                strcpy(cur_path,"");
                for(i=0;i<misc_pos[3];i++){
                  strcat(cur_path,_stack[i]);
                  strcat(cur_path,"\\");
                  }
                }
               create_all(fname);
               new_screen();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        dir_point=_point;
                        misc_pos[3]=_point;
                        misc_pos[2]=0;
                        name_point=0;
                        break;
                case 4:end=true;
                        box(240,306,304,326,false);
                        write_text(240,304,312,"O.K.",7);
                        delay(200);
                        box(240,306,304,326,true);
                        write_text(240,304,312,"O.K.",12);
                        delay(50);
                        break;
                case 5:end=true;
                        box(336,306,400,326,false);
                        write_text(336,400,312,"Cancle",7);
                        delay(200);
                        box(336,306,400,326,true);
                        write_text(336,400,312,"Cancle",12);
                        delay(50);
                        break;
                }
              break;
        }
      }while(!end);
    farfree(ptr);
    farfree(_scroll);
    farfree(_name);
    if((main_pos==5)||((ch==ESC_KEY)&&(!funckey)))
      return(-1);
    else{
      strcpy(file_name,temp);
      return(0);
      }
    }

void init_menu_command(void){
  int i;

  for(i=0;i<main_menu_no;i++){
    pull_menu[i].command_no=0;
    pull_menu[i].y_pos=40;
    }

  pull_menu[0].num_item=5; // File
  pull_menu[0].x_pos=20;
  pull_menu[0].menu_width=112;
  strcpy(pull_menu[0].command[0],"Load");       pull_menu[0].fastkey[0]=0;
  strcpy(pull_menu[0].command[1],"Save");       pull_menu[0].fastkey[1]=0;
  strcpy(pull_menu[0].command[2],"Delete");     pull_menu[0].fastkey[2]=0;
  strcpy(pull_menu[0].command[3],"Change Name");pull_menu[0].fastkey[3]=0;
  strcpy(pull_menu[0].command[4],"Save As");    pull_menu[0].fastkey[4]=5;
  pull_menu[1].num_item=5; // Pattern
  pull_menu[1].x_pos=70;
  pull_menu[1].menu_width=132;
  strcpy(pull_menu[1].command[0],"Pattern");        pull_menu[1].fastkey[0]=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    strcpy(pull_menu[1].command[1],"Setup  Pattern"); pull_menu[1].fastkey[1]=2;
    strcpy(pull_menu[1].command[2],"Delete Pattern"); pull_menu[1].fastkey[2]=0;
    strcpy(pull_menu[1].command[3],"Load   Pattern"); pull_menu[1].fastkey[3]=0;
    strcpy(pull_menu[1].command[4],"Save   Pattern"); pull_menu[1].fastkey[4]=0;

    pull_menu[2].num_item=3; // Arrange Pattern
    pull_menu[2].x_pos=120;
    pull_menu[2].menu_width=140;
    strcpy(pull_menu[2].command[0],"Standard Rhythm");pull_menu[2].fastkey[0]=0;
    strcpy(pull_menu[2].command[1],"Edit");          pull_menu[2].fastkey[1]=0;
    strcpy(pull_menu[2].command[2],"New");           pull_menu[2].fastkey[2]=0;

    pull_menu[3].num_item=3; // Playback
    pull_menu[3].x_pos=200;
    pull_menu[3].menu_width=60;
    strcpy(pull_menu[3].command[0],"Load");    pull_menu[3].fastkey[0]=0;
    strcpy(pull_menu[3].command[1],"Place");   pull_menu[3].fastkey[1]=0;
    strcpy(pull_menu[3].command[2],"Edit");    pull_menu[3].fastkey[2]=0;

    pull_menu[4].num_item=2; // Playback
    pull_menu[4].x_pos=240;
    pull_menu[4].menu_width=108;
    strcpy(pull_menu[4].command[0],"Os Shell");       pull_menu[4].fastkey[0]=0;
    strcpy(pull_menu[4].command[1],"Quit Program");   pull_menu[4].fastkey[1]=0;
    }

int pull_menu_fx_m(pulldownmenu *active_menu){
  int x_mouse,y_mouse,i,y1,y2;
  int temp;

  temp=-1;
  mouse_position(&x_mouse,&y_mouse);
  for(i=0;i<active_menu->num_item;i++){
    y1=active_menu->y_pos+(text_tall+4)*i+3;
    y2=y1+11;
    if((y_mouse>=y1)&&(y_mouse<=y2)){
      temp=i;
      i=20;
      }
    }
  return(temp);
  }

void setmouse_pcursor(pulldownmenu *active_menu,int no){
  int y1;

  y1=active_menu->y_pos+(text_tall+4)*no+3;
  y1+=5;
  set_mouse_position(active_menu->x_pos+active_menu->menu_width,y1);
  }

unsigned char pull_down(pulldownmenu *active_menu){
  int i,test;
  void far *mainptr,*ptr;
  int save_color,x1,y1,x2,y2,x_mouse,y_mouse;
  boolean end;
  unsigned char ch;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    do{
      }while(leftb_pressed());
    x1=active_menu->x_pos;
    y1=active_menu->y_pos;
    x2=x1+active_menu->menu_width;
    y2=(text_tall+4)*(active_menu->num_item)+y1+7;
    mainptr=getmem(imagesize(x1,y1,x2+4,y2+4));
    getimage(x1,y1,x2+4,y2+4,mainptr);
    save_color=getcolor();
    end=false;
    setfillstyle(1,color.shadow);
    bar(x1+4,y1+4,x2+4,y2+4);
    setfillstyle(1,color.bar_color);
    bar(x1,y1,x2,y2);
    setcolor(0);
    rectangle(x1,y1,x2,y2);
    for(i=0;i<active_menu->num_item;i++){
      write_menu_command(x1+text_width,y1+(text_tall+4)*i+5
        ,active_menu->command[i],active_menu->fastkey[i],false);
      }
    mouse_position(&x_mouse,&y_mouse);
    y_mouse=active_menu->y_pos+(text_tall+4)*active_menu->command_no+3;
    set_mouse_position(x_mouse,y_mouse);
    x1=active_menu->x_pos+text_width/2;
    x2=active_menu->x_pos+active_menu->menu_width-text_width/2;
    do{
      setmouse_pcursor(active_menu,active_menu->command_no);
      y1=active_menu->y_pos+(text_tall+4)*active_menu->command_no+3;
      y2=y1+11;
      ptr=getmem(imagesize(x1,y1,x2,y2));
      getimage(x1,y1,x2,y2,ptr);
      setfillstyle(1,color.menu_active_color);
      bar(x1,y1,x2,y2);
      i=active_menu->command_no;
      write_menu_command(active_menu->x_pos+text_width,active_menu->y_pos+(text_tall+4)*i+5
        ,active_menu->command[i],active_menu->fastkey[i],true);
      mouse_clicked=false;
      do{
        mouse_position(&x_mouse,&y_mouse);
        if(y_mouse<active_menu->y_pos+3)
          set_mouse_position(x_mouse,active_menu->y_pos+3);
        else if(y_mouse>active_menu->y_pos+(text_tall+4)*(active_menu->num_item-1)+14)
          set_mouse_position(x_mouse,active_menu->y_pos+(text_tall+4)*(active_menu->num_item
-1)+14);
        test=pull_menu_fx_m(active_menu);
        }while((test==-1)||((!kbhit())&&(!leftb_pressed())&&(!rightb_pressed()))&&
          ((test==active_menu->command_no)));
      if(kbhit()){
        ch=readfunc();
        mouse_clicked=false;
        }
      else{
        mouse_clicked=true;
        }
      if((funckey)&&(!mouse_clicked)){
        switch(ch){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        case UP_KEY    :active_menu->command_no--;
                        break;
        case DOWN_KEY  :active_menu->command_no++;
                        break;
        }
      if(active_menu->command_no<0)
        active_menu->command_no=active_menu->num_item-1;
      if(active_menu->command_no==active_menu->num_item)
        active_menu->command_no=0;
      }
    else if(!mouse_clicked){
      for(i=0;i<active_menu->num_item;i++){
        if((ch==active_menu->command[i][active_menu->fastkey[i]])||
           ( ((ch>96)&&(ch<123))&&(ch-32==active_menu->command[i][active_menu->fastkey[i]]
) )){
          active_menu->command_no=i;
          ch=RETURN_KEY;
          funckey=false;
          putimage(x1,y1,ptr,COPY_PUT);
          farfree(ptr);
          setmouse_pcursor(active_menu,active_menu->command_no);
          y1=active_menu->y_pos+(text_tall+4)*active_menu->command_no+3;
          y2=y1+11;
          ptr=getmem(imagesize(x1,y1,x2,y2));
          getimage(x1,y1,x2,y2,ptr);
          setfillstyle(1,color.menu_active_color);
          bar(x1,y1,x2,y2);
          i=active_menu->command_no;
          write_menu_command(active_menu->x_pos+text_width,active_menu->y_pos+(text_tall+4
)*i+5
                      ,active_menu->command[i],active_menu->fastkey[i],true);
          mouse_clicked=false;
          }
        }
      }
    else if(mouse_clicked){
      if(leftb_pressed()){
        do{}while(leftb_pressed());
        mouse_clicked=false;
        ch=RETURN_KEY;
        funckey=false;
        end=true;
        }
      else if(rightb_pressed()){
        do{}while(rightb_pressed());
        mouse_clicked=false;
        ch=ESC_KEY;
        funckey=false;
        end=true;
        }
      else{
        active_menu->command_no=test;
        }
      }
    if(((!funckey)&&((ch==RETURN_KEY)||(ch==ESC_KEY)))
      ||((funckey)&&((ch==LEFT_KEY)||(ch==RIGHT_KEY)||(ch==ALT_X_KEY))))
      end=true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(active_menu==&popup_menu[0]){
      if((funckey)||((ch!=RETURN_KEY)&&(ch!=ESC_KEY)))
        end=false;
      }
    if((!funckey)&&(ch==RETURN_KEY)){

      if(!strcmp(active_menu->command[active_menu->command_no],"Tool")){
        ch=pull_down(&popup_menu[0]);
        if((ch==27)&&(!funckey))end=false;
        }
      else if(!strcmp(active_menu->command[active_menu->command_no],"Build Pattern")){
        ch=pull_down(&popup_menu[1]);
        if((ch==27)&&(!funckey))end=false;
        }
      }
    putimage(x1,y1,ptr,COPY_PUT);
    farfree(ptr);
    }while(!end);
  setcolor(save_color);
  x1=active_menu->x_pos;
  y1=active_menu->y_pos;
  putimage(x1,y1,mainptr,COPY_PUT);
  farfree(mainptr);
  mouse_clicked=false;
  return(ch);
  }

char lib_browser(int *number,char *name,int xpos,int ypos){
  int save_color,i,return_no,no,x_mouse,y_mouse,x1,y1;
  struct fillsettingstype save_fill;
  void far *ptr,*ptr1,*ptr2;
  boolean end,khit;
  char ch;

  end=false;
  save_color=getcolor();
  getfillsettings(&save_fill);
  return_no=0;
  no=0;
  ptr=getmem(imagesize(xpos,ypos,xpos+150,ypos+224+40));
  ptr1=getmem(imagesize(0,0,150,15));
  getimage(xpos,ypos,xpos+150,ypos+224+40,ptr);
  frame(xpos,ypos,xpos+150,ypos+224+40,false);
  write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_color);
  setcolor(color.menu_text_color);
  for(i=0;i<16;i++){
    outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28,drum_lib.name[i]);
    }
  setcolor(color.active_menu_text_color);
  setfillstyle(1,1);
  do{
    khit=false;
    x_mouse=xpos+30;
    y_mouse=ypos+14*no+color.frame_width+28+7;
    set_mouse_position(x_mouse,y_mouse);
    x1=xpos+color.frame_width+4;
    y1=ypos+color.frame_width+25+no*14;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    getimage(x1,y1,x1+16*text_width,y1+13,ptr1);
    bar(x1,y1,x1+16*text_width,y1+13);
    outtextxy(xpos+color.frame_width+8,ypos+14*no+color.frame_width+28,drum_lib.name[retur
n_no]);
    do{
      do{
        mouse_position(&x_mouse,&y_mouse);
        }while((!kbhit())&&(!clicked())&&(y_mouse>=ypos+14*no+color.frame_width+28)&&(y_mo
use<=ypos+14*no+color.frame_width+28+13));
      khit=true;
      if(!kbhit()){
        if(y_mouse<ypos+14*no+color.frame_width+21){
          ch=UP_KEY;
          funckey=true;
          }
        else if(y_mouse>ypos+14*no+color.frame_width+21){
          ch=DOWN_KEY;
          funckey=true;
          }
        }
      }while((!kbhit())&&(!khit)&&(!clicked()));
    if(leftb_pressed()){
      end=true;
      ch=RETURN_KEY;
      funckey=false;
      }
    else if(rightb_pressed()){
      end=true;
      ch=ESC_KEY;
      funckey=false;
      }
    else if(kbhit()){
      ch=readfunc();
      }
    putimage(x1,y1,ptr1,COPY_PUT);
    if(funckey)switch((unsigned char)ch){
      case UP_KEY   :if(return_no>0){
                       return_no--;
                       if(no>0)no--;
                       else{// scroll screen
                         y1=ypos+color.frame_width+25;
                         ptr2=getmem(imagesize(x1,y1,x1+16*text_width,y1+15*14));
                         getimage(x1,y1,x1+16*text_width,y1+15*14,ptr2);
                         putimage(x1,y1+14,ptr2,COPY_PUT);
                         setcolor(color.menu_text_color);
                         setfillstyle(1,color.frame_color);
                         bar(x1,y1,x1+16*text_width,y1+14);
                         outtextxy(xpos+color.frame_width+8,ypos+color.frame_width+28,drum
_lib.name[return_no]);
                         setfillstyle(1,1);
                         setcolor(color.active_menu_text_color);
                         farfree(ptr2);
                         }
                       }
                     break;
      case DOWN_KEY :if((return_no<drum_lib.max-1)&&
                     (drum_lib.number[return_no+1]>=0)){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        return_no++;
                        if(no<15) no++;
                        else{//scroll screen
                          y1=ypos+color.frame_width+25+14;
                          ptr2=getmem(imagesize(x1,y1,x1+16*text_width,y1+15*14));
                          getimage(x1,y1,x1+16*text_width,y1+15*14,ptr2);
                          putimage(x1,y1-14,ptr2,COPY_PUT);
                          setcolor(color.menu_text_color);
                          setfillstyle(1,color.frame_color);
                          bar(x1,y1+14*14,x1+16*text_width,y1+15*14);
                          outtextxy(xpos+color.frame_width+8,ypos+14*no+color.frame_width+2
8,drum_lib.name[return_no]);
                          setfillstyle(1,1);
                          setcolor(color.active_menu_text_color);
                          farfree(ptr2);
                          }
                        }
                      break;
      case PAGEUP_KEY:
                      if((return_no<16)&&(return_no==no)){
                       return_no=0;
                       no=0;
                       }
                      else{
                       if(return_no>=16)
                         return_no-=16;
                       else{
                         no=0;
                         return_no=0;
                         }
                       frame(xpos,ypos,xpos+150,ypos+224+40,false);
                       write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_co
lor);
                       setcolor(color.menu_text_color);
                       if(return_no-no<16){
                         no=0;
                         }
                       for(i=0;i<16;i++){
                         outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28
,drum_lib.name[return_no-no+i]);
                         }
                       setcolor(color.active_menu_text_color);
                       }
                      break;
     case PAGEDOWN_KEY:
                     if(return_no+16<drum_lib.max){
                       return_no+=16;
                       frame(xpos,ypos,xpos+150,ypos+224+40,false);
                       write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_co
lor);
                       setcolor(color.menu_text_color);
                       if(return_no+no+16>drum_lib.max)
                         no=16-(drum_lib.max-return_no);
                       for(i=0;i<16;i++)
                         outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28
,drum_lib.name[return_no-no+i]);
                       setcolor(color.active_menu_text_color);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    }
                  else{
                    no+=drum_lib.max-return_no-1;
                    return_no=drum_lib.max-1;
                    }
                  break;
      case CTRL_PGUP:no=0;
                  return_no=0;
                  frame(xpos,ypos,xpos+150,ypos+224+40,false);
                  write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_colo
r);
                  setcolor(color.menu_text_color);
                  for(i=0;i<16;i++)
                    outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28,d
rum_lib.name[return_no-no+i]);
                  setcolor(color.active_menu_text_color);
                  break;
      case CRTL_PGDN:return_no=drum_lib.max-1;
                  if(return_no<15)
                    no=return_no;
                  else
                    no=15;
                  frame(xpos,ypos,xpos+150,ypos+224+40,false);
                  write_text(xpos,xpos+150,ypos+10,"Voice Select",color.title_text_colo
r);
                  setcolor(color.menu_text_color);
                  for(i=0;i<16;i++)
                    outtextxy(xpos+color.frame_width+8,ypos+14*i+color.frame_width+28,d
rum_lib.name[return_no-no+i]);
                  setcolor(color.active_menu_text_color);
                  break;
     }
   else{
     if((ch==RETURN_KEY)||(ch==ESC_KEY)) end=true;
     else if(ch==' '){//Test Sound
       midiout(0,0x99);
       midiout(0,(char)drum_lib.number[return_no]);
       midiout(0,95);
       midiout(0,0x99);
       midiout(0,(char)drum_lib.number[return_no]);
       midiout(0,0);
       }
     }
   }while(!end);
 setcolor(save_color);
 setfillstyle(save_fill.pattern,save_fill.color);
 if((!funckey)&&(ch==RETURN_KEY)){
   *number=drum_lib.number[return_no];
   strcpy(name,drum_lib.name[return_no]);
   }
 else if(ch==ESC_KEY) *number=-1;
 putimage(xpos,ypos,ptr,COPY_PUT);
 farfree(ptr1);
 farfree(ptr);
 setcolor(save_color);
 setfillstyle(save_fill.pattern,save_fill.color);
 return(ch);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}

void load_lib(char *lib_name){
  int c,cont_count;
  char filename[40],temp[40],ch;
  FILE *flib;

  sprintf(filename,"%s%s",option.lib_path,lib_name);
  if((flib=fopen(filename,"rb"))==NULL){
    message(240,200,"Error","   Error Loading Library File   ");
    }
  else{
    drum_lib.max=0;
    c=0;
    do{
      ch=fgetc(flib);
      if((ch!=13)&&(ch!=10))
        temp[c++]=ch;
      }while((ch!=13)&&(ch!=10));
    temp[c]=0;
    do{ch=fgetc(flib);}while((ch==13)||(ch==10));
    if(!strcmp(temp,"Voice Library")){//This File is Voice Library File
      c=0;
      do{
        ch=fgetc(flib);
        if((ch==13)||(ch==10)) c++;
        }while(c<2);
      temp[0]=fgetc(flib);
      temp[1]=fgetc(flib);
      temp[2]=fgetc(flib);
      temp[3]=0;
      fgetc(flib);
      fgetc(flib);
      drum_lib.max=atoi(temp);
      for(cont_count=0;cont_count<drum_lib.max;cont_count++){
        drum_lib.number[cont_count]=0;
        if(!feof(flib)){
          for(c=0;c<3;c++){
            ch=fgetc(flib);
            switch(c){
              case 0:drum_lib.number[cont_count]+=((ch-48)*100);break;
              case 1:drum_lib.number[cont_count]+=((ch-48)*10);break;
              case 2:drum_lib.number[cont_count]+=(ch-48);break;
              }
            }
          c=0;
          ch=fgetc(flib);
          do{
            ch=fgetc(flib);
            if((ch!=13)&&(ch!=10))
              drum_lib.name[cont_count][c++]=ch;
            }while((ch!=13)&&(ch!=10));
          drum_lib.name[cont_count][c]=0;
          ch=fgetc(flib);
          }
        }
      }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    fclose(flib);
    }
}
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Voice Library
KAWAI K11 GM BANK
128
000 GrPiano
001 BrPiano
002 ElGrand
003 HnkyTonk
004 Elpiano1
005 Elpiano2
006 Hrpschrd
007 Clavi
008 Celesta
009 Glocken
010 MusicBox
011 Vibes
012 Marimba
013 Xylophon
014 Tubulbel
015 Dulcimer
016 DrawOrgn
017 PercOrgn
018 RockOrgn
019 ChrcOrgn
020 ReedOrgn
021 Acordion
022 Harmnica
023 TangoAcd
024 NylonGtr
025 SteelGtr
026 Jazz Gtr
027 CleanGtr
028 Mute Gtr
029 Ovrdrive
030 Distortd
031 Harmnics
032 WoodBass
033 FngrBass
034 PickBass
035 Fretless
036 SlapBas1
037 SlapBas2
038 SynBass1
039 SynBass2
040 Violin
041 Viola
042 Cello
043 Contra
044 TremStrg
045 Pizzicto
046 Harp
047 Timpani
048 StrgEns1
049 StrgEns2
050 SynStrg1
051 SynStrg2
052 AahChoir
053 OohChoir

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

054 SynChoir
055 Orch Hit
056 Trumpet
057 Trombone
058 Tuba
059 MuteTrmp
060 FrenchHr
061 BrasSect
062 SynBras1
063 SynBras2
064 SprnoSax
065 Alto Sax
066 TenorSax
067 Bari Sax
068 Oboe
069 EnglHorn
070 Bassoon
071 Clarinet
072 Piccolo
073 Flute
074 Recorder
075 PanFlute
076 Bottle
077 Shakhach
078 Whistle
079 Ocarina
080 SquareLd
081 Saw Ld
082 CaliopLd
083 Chiff Ld
084 CharanLd
085 Voice Ld
086 Fifth Ld
087 Bass &Ld
088 NewAgePd
089 Warm Pd
090 PolySyPd
091 Choir Pd
092 Bowed Pd
093 Metal Pd
094 Halo Pd
095 Sweep Pd
096 Rain
097 SoundTrk
098 Crystal
099 Atmosphr
100 Bright
101 Goblin
102 Echoes
103 SciFi
104 Sitar
105 Banjo
106 Shamisen
107 Koto
108 Kalimba
109 Bagpipe
110 Fiddle

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

111 Shanai
112 TnklBell
113 Agogo
114 Stl Drum
115 WoodBlok
116 TaikoDrm
117 MelodTom
118 SynthTpm
119 RevCymbl
120 FretNoiz
121 BrthNoiz
122 Seashore
123 Brdtweet
124 Telphone
125 Helicptr
126 Applause
127 Gunshot



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Voice Library
K11 SP Bank
128
000 GrandPf1
001 BrightPf
002 GrandPf2
003 E.Pf1
004 E.Gr Pf1
005 E.Pf2
006 E.Gr Pf2
007 HnkyTonk
008 RockOrgn
009 DrawOrgn
010 PercOrgn
011 ElecOrgn
012 PipOrgn1
013 PipOrgn2
014 PipOrgn3
015 Acordion
016 Harpsi 1
017 Harpsi 2
018 Harpsi 3
019 Clavi 1
020 Clavi 2
021 Clavi 3
022 Celesta1
023 Celesta2
024 SynBras1
025 SynBras2
026 Ovdrve6t
027 Dist 6t
028 SynBs1
029 SynBs2
030 SynBs3
031 SynBs4
032 NewAgePd
033 Warm Pd
034 Choir Pd
035 Bowed Pd
036 SoundTrk
037 Atmosphr
038 SynWarm
039 SynVoice
040 EchoBell
041 Rain
042 SynWind
043 Echoes
044 SynSolo
045 ReedOrgn
046 SynBell
047 Square
048 String1
049 String2
050 SynStrgs
051 Pizzicto
052 Violin
053 Viola

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

054 Cello1
055 Cello2
056 Con Bass
057 Harp 1
058 Harp 2
059 NylonGtr
060 SteelGtr
061 E.Gt 1
062 E.Gt 2
063 Sitar
064 Wood Bs1
065 Wood Bs2
066 PickedBs
067 FingerBs
068 SlapBs 1
069 SlapBs 2
070 Fretles1
071 Fretles2
072 Flute 1
073 Flute 2
074 Piccolo1
075 Piccolo2
076 Recorder
077 PanFlute
078 S. Sax
079 A. Sax
080 T. Sax
081 B. Sax
082 Cl 1
083 Cl 2
084 Oboe
085 SawBrass
086 Bassoon
087 Harmnica
088 Trunpet1
089 Trumpet2
090 Tb 1
091 Tb 2
092 FreHorn1
093 FreHorn2
094 Tuba
095 BrsEns 1
096 BrsEns 2
097 Vibe 1
098 Vibe 2
099 Mallet
100 WindBell
101 Glocken
102 TubulBel
103 Xylophon
104 Maimba
105 Koto
106 Shamisen
107 Shaku8
108 Whistle1
109 Whistle2
110 Bottle

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

111 Chiff
112 Timpani
113 Melo Tom
114 DeepSnar
115 SynDrum1
116 SynDrum2
117 Taiko
118 TaikoRim
119 Cymbal
120 Castanet
121 Triangle
122 Orch Hit
123 NewSyn1
124 NewSyn2
125 NewSyn3
126 NewSyn4
127 NewSyn5



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Voice Library
K1 Internal Voice Library
064
000 String 1
001 String 2
002 D-50Brass1
003 Brass K1
004 Dx-7Piano1
005 Dx-7Piano2
006 Jazz Organ
007 Rock Organ
008 String 3
009 Orchestra
010 OB-Brass
011 KORG'Brass
012 Flute
013 Bell 1
014 Analog Synth
015 ORCTA HIT!
016 Ch.Piano!
017 Chorus
018 Kimono
019 Pan Flute
020 Ch. Piano2
021 Marimba 2
022 ChorusVibe
023 OrientBell
024 TenorSax
025 BlueMonica
026 Clarinet
027 Trumpet
028 XFadeOrgan
029 Hammond!
030 Mornin'Brass
031 TheNewSax
032 Chorus 2
033 Bum'sTheme
034 Felix Solo
035 FrenchHorn
036 Brite EP
037 Jazz Harp
038 Dora
039 Cannon&Gun
040 Cellos
041 Lord Organ
042 Voice Ahh
043 Oboe
044 Visitors
045 K1 Goes LA
046 Marimba
047 Timpani
048 Harpsicord
049 Stratocas
050 Sitar
051 DElayXlo
052 SteelDrums
053 HardMallet

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

054 Glocken
055 SuperJet1
056 Synth Ens
057 Harmonica
058 KrystalVyb
059 JazzPianno
060 Phantom
061 Fend Bass
062 DX-7 Bass
063 Pull Bass



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Voice Library
Technics KN650
100
000 PAINO1
001 PAINO2
002 PAINO OCT
003 E GRAND
004 HONKY TONK1
005 E PAINO1
006 E PAINO2
007 HARPSICHORD
008 CLAVI
009 GLOCKEN1
010 VIBRAPHONE1
011 MARIMBA
012 XYLOPHONE
013 CHIMS
014 STEEL DRUM
015 CLASSICAL GTR1
016 FOLK GTR
017 JAZZ GTR1
018 SOLID GTR1
019 SOLID GTR2
020 MUTE GTR1
021 ROCK GTR1
022 ROCK GTR2
023 HAWAIIAN GTR
024 BANJO
025 HARP
026 MUSIC BOX
027 KALIMBA 1
028 SITAR
029 KOTO
030 VIOLIN
031 CELLO
032 STRINGS
033 STRINGS OCT
034 PIZZICATO STR
035 SYNTH STRINGS1
036 VOCAL OOH
037 VOCAL DOO1
038 SYNTH VOCAL
039 TIMPANI
040 ORCHESTRA HIT1
041 PIPE ORGAN
042 THEATRE ORGAN1
043 JAZZ ORGAN1
044 JAZZ ORGAN2
045 POP ORGAN1
046 ROCK ORGAN1
047 ACCORDIAN
048 HARMONICA
049 PICCOLO
050 FLUTE
051 RECORDER
052 PAN FLUTE
053 SHAKUHACHI

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

054 WHISTLE1
055 BRASS
056 BRASS OCT
057 TRUMPET
058 MUTE TURMPET
059 TROMBONE1
060 FRENCH HORN1
061 SYNTH BRASS1
062 SYNTH BRS ENS1
063 SOPRANO SAX
064 ALTO SAX1
065 TENOR SAX1
066 BARIONE SAX
067 CLARINET
068 OBOE
069 BASSOON
070 SYNTH PERC1
071 SYNTH PERC2
072 SYNTH PERC3
073 SYNTH PERC4
074 SYNTH PERC5
075 SYNTH LEAD1
076 SYNTH LEAD2
077 SYNTH LEAD3
078 SYNTH LEAD4
079 SYNTH PAD1
080 SYNTH PAD2
081 SYNTH PAD3
082 SYNTH PAD4
083 SYNTH PAD5
084 SYNTH EFFECT 1
085 ACOUSTICS BASS1
086 BOWED BASS1
087 E BASS FINGER 1
088 FRETLESS BASS
089 EBASS PICKED 1
090 MUTE BASS
091 CHOPPER BASS 1
092 TUBA
093 SYNTH TUBA
094 SYNTH CHOPPER 1
095 JAZZ KIT
096 ROCK KIT 1
097 ROCK KIT 2
098 DANCE KIT
099 SOUL KIT

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Voice Library
K11 GM Standard
128
036 Bass Drum 1
037 Side Stick
038 AC Snare 1
039 Hand Clap
040 AC Snare 2
041 Low F Tom
042 Close HH
043 High F Tom
044 Pedal HH
045 Low Tom
046 Open HH
047 Low-Mid-Tom
048 Hi-Mid-Tom
049 TopCym1
050 High Tom
051 SidCym1
052 ChinaCym
053 RideBell
054 TamBourine
055 SplashCym
056 CowBell
057 TopCym1
058 VibraSlap
059 SidCym2
060 Hi Bongo
061 Lo Bongo
062 Mute Hi Conga
063 Open Hi Conga
064 Lo Conga
065 Hi Timbale
066 Lo Timbale
067 High Agogo
068 Lo Agogo
069 Cabasa
070 Maracas
071 Short Whistle
072 Long Whistle
073 Short Guiro
074 Long Guiro
075 Clabs
076 Hi Wood Block
077 LO Wood Block
078 Mute Cuica
079 Open Cuica
080 Mute Triangle
081 Open Triangle
082 Shaker
083 Jing
084 Belltree
085 Castanets
086 MuteSurdo
087 OpenSurdo
088 Elec.BD
089 Elec.SD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

090 Elec.Lo Tom2
091 Elec.Lo Tom1
092 Elec.mid Tom2
093 Elec.mid Tom1
094 Elec.hi Tom2
095 Elec.hi Tom1
096 ReverseCym
000 BOB BD
001 BOB Rim
002 BOB SD
003 BOB LoTom2
004 BOB CloseHH.
005 BOB LoTom1
006 BOB MidTom2
007 BOB OpenHH.
008 BOB MidTom1
009 BOB HiTom2
010 BOB Cym
011 BOB HiTom1
012 BOB CowBell
013 BOB HiConga
014 BOB MidConga
015 BOB LowConga
016 BOB Maracas
017 BOB Clabs
018 MONDO Kick
019 Gate SN
020 PowerTomLow2
021 PowerTomLow1
022 PowerMidTom2
023 PowerMidTom1
024 PowerHiTom2
025 PowerHiTom1
026 MUTE
027 HighQ
028 Slap
029 Scrach Push
030 Scrach Pull
031 Sticks
032 Square Click
033 MetoronomeClick
034 Metoronome Bell
035 AC Bass Drum 2
097 Brush Tap
098 Brush Slap
099 Brush
100 Jazz BD
101 Concert BD2
102 Concert BD1
103 Concert SD
104 Timpani F
105 Timpani F#
106 Timpani G
107 Timpani G#
108 Timpani A
109 Timpani A#
110 Timpani B

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

111 Timpani c
112 Timpani c#
113 Timpani d
114 Timpani d#
115 Timpani e
116 Timpani f2
117 Concert Cym2
118 Concert Cym1
119 Applause
120 Room Tom Low2
121 Room Tom Low1
122 Room Tom Mid2
123 Room Tom Mid1
124 Room Tom Hi2
125 Room Tom Hi1
126 EFF Clap
127 Echo Gras



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้