ภาควิชาครุศาสตร์วิศวกรรม

คณะครุศาสตร์อุตสาหกรรม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อ แหล่งจ่ายไฟฟ้าสำรอง

Uninterruptible Power Supply

ชื่อนักศึกษา

1. นายกำธร	ธงทอง	รหัสประจำตัว	42035329
2. นางสาวนารีรัตน์	แย้มมา	รหัสประจำตัว	42035337
3. นายภูวไนย	สูนย์สาทร	รหัสประจำตัว	42035343
4. นายณรงค์	บรรณเลิศ	รหัสประจำตัว	42035345

หลักสูตร ครุศาสตร์อุตสาหกรรมบัณฑิต **สาขาวิชา** อิเล็กทรอนิกส์และคอมพิวเตอร์

อาจารย์ที่ปรึกษา อาจารย์สุรพงษ์ สิริพงศ์ดี

อาจารย์ที่ปรึกษาร่วม ดร.สุรสิทธิ์ ราตรี

คณะกรรมการสอบปริญญานิพนธ์		ลายมือชื่อ
1. อาจารย์สุรพงษ์	สิริพงศ์ดี	.....
2. ดร.สุรสิทธิ์	ราตรี	.....
3. อาจารย์สุชิน	อาจหาญ	.....
4. อาจารย์อำพล	ทองระอา	.....
5. อาจารย์สุระชัย	พิมพ์สาลี	.....

วัน/เดือน/ปีที่สอบ วันเสาร์ที่ 9 ธันวาคม พ.ศ. 2543 เวลา 11.30 น.

สถานที่สอบ ห้อง ค.311 คณะครุศาสตร์อุตสาหกรรม สจล.

ภาควิชารับรองแล้ว

ลงนาม.....

(ผศ.วิสุทธิ์ อธิพรธรรม)

หัวหน้าภาควิชาครุศาสตร์วิศวกรรม

วันที่.....เดือน.....พ.ศ.....

ปริญญานิพนธ์

แหล่งจ่ายไฟฟ้าสำรอง

UNINTERRUPTIBLE POWER SUPPLY

นายกำธร	ธงทอง
นางสาวนารีรัตน์	แย้มมา
นายภูวไนย	สูนย์สาทร
นายณรงค์	บรรณเลิศ

ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรครุศาสตร์อุตสาหกรรมบัณฑิต
สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์
ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2543

ปริญญานิพนธ์

เรื่อง แหล่งจ่ายไฟฟ้าสำรอง

Uninterrupt Power Supply

วัตถุประสงค์

1. เพื่อศึกษาถึงระบบการทำงานของเครื่องจ่ายไฟฟ้าสำรอง
2. เพื่อศึกษาการใช้งานไมโครคอนโทรลเลอร์ในการควบคุมการทำงาน
3. เพื่อศึกษาการปรับปรุงทางฮาร์ดแวร์โดยพัฒนาทางด้านซอฟต์แวร์
4. เพื่อออกแบบวงจรเครื่องจ่ายไฟฟ้าสำรอง
5. เพื่อทดสอบการทำงานของเครื่องจ่ายไฟฟ้าสำรอง
6. เพื่อสร้างเครื่องจ่ายไฟฟ้าสำรองต้นแบบ

ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถเข้าใจถึงระบบการทำงานของเครื่องจ่ายไฟฟ้าสำรองได้
2. สามารถใช้งานไมโครคอนโทรลเลอร์ในการควบคุมการทำงานได้
3. สามารถปรับปรุงทางฮาร์ดแวร์โดยพัฒนาทางด้านซอฟต์แวร์ได้
4. สามารถออกแบบวงจรเครื่องจ่ายไฟฟ้าสำรองได้
5. สามารถทดสอบการทำงานของเครื่องจ่ายไฟฟ้าสำรองได้
6. สามารถสร้างเครื่องจ่ายไฟฟ้าสำรองต้นแบบได้

ชื่อหัวข้อ	แหล่งจ่ายไฟฟ้าสำรอง
นักศึกษา	นายกำธร ธงทอง
	นางสาวนารีรัตน์ แย้มมา
	นายภูวไนย สูนย์สาทร
	นายณรงค์ บรรณเลิศ
อาจารย์ที่ปรึกษา	อาจารย์สุรพงษ์ สิริพงศ์ดี
ที่ปรึกษาร่วม	ดร. สุรสิทธิ์ ราตรี
หลักสูตร	ครุศาสตร์อุตสาหกรรมบัณฑิต
สาขาวิชา	อิเล็กทรอนิกส์และคอมพิวเตอร์
ปีการศึกษา	2543

บทคัดย่อ

แหล่งจ่ายไฟฟ้าสำรองหรือยูพีเอส (Uninterruptible Power Supply; UPS) คืออุปกรณ์ที่ออกแบบมาเพื่อเตรียมจ่ายพลังงานไฟฟ้าสำรองให้กับภาระได้อย่างต่อเนื่อง ถึงแม้ว่าไฟฟ้าจากการไฟฟ้าดับ หรือเกิดขัดข้องอันเนื่องมาจากสาเหตุใดก็ตาม จะเห็นได้ว่ายูพีเอสเป็นอุปกรณ์เพิ่มประสิทธิภาพให้กับระบบไฟฟ้า แต่ยูพีเอสก็มีขีดจำกัดในการจ่ายพลังงานไฟฟ้า ณ ระดับหนึ่งเท่านั้น โดยแหล่งจ่ายไฟฟ้าสำรองในโครงงานนี้เป็นแหล่งจ่ายไฟฟ้าชนิด 1 เฟส ขนาดพิกัด 1000 VA และสามารถจ่ายไฟฟ้าสำรองได้นานประมาณ 30 นาทีที่พิกัดกำลังไฟฟ้า โดยการควบคุมระบบการทำงานทั้งหมดจะอาศัยไมโครคอนโทรลเลอร์ ซึ่งเป็นตัวสร้างสัญญาณรูปคลื่นสี่เหลี่ยมที่มีความถี่ 50 Hz เป็นสัญญาณสวิตชิ่งให้กับวงจรอินเวอร์เตอร์ซึ่งเป็นชนิด (Push Pull Inverter) ซึ่งทำงานแบบ Line interactive UPS (ยูพีเอสที่มี Stabilize) และมีการแสดงผล (Display) บนคอมพิวเตอร์ในรูปแบบกราฟฟิก สำหรับในส่วนท้ายของปริญญานิพนธ์ฉบับนี้เป็นผลการทดลองที่ได้จากเครื่องต้นแบบที่สร้างขึ้น

Thesis Title	Un - interruptible Power Supply	
Students	Mr. Gumthon	Thongtong
	Miss Nareerut	Yamma
	Mr. Phuvanai	Soonsatorn
	Mr. Narong	Bannalerd
Advisor	Mr. Surapong	Siripongdee
Co-Advisor	Dr. Surasit	Ratree
Education Level	Bachelor of Science in Industrial Education	
Program in	Electronics and Computer	
Academic	2000	

ABSTRACT

Uninterruptible Power Supply or UPS is designed to supply reserved electrical energy for Un - interruptible load. Inspire of the failing of system or no power supply from source, load is able to work continuously. So UPS is a equipment which improve efficiency to load but it is limited to supply electrical energy which is appropriate with a lower power system. The UPS in this project is single phase power supply, 1000VA rated. It can supply 30 minutes at rated power. It is controlled by 8-bit Microcontroller, which generates 50 Hz square wave as switching signal for inverter circuit. This type of inverter is Push – Pull inverter, which work in Line – interactive mode. Graphic display is shown on PC monitor. Lastly this thesis has experiment of UPS processing is in at the finish.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี เนื่องจากความร่วมมือของสมาชิกภายในกลุ่มทุกท่าน ผู้จัดทำขอกราบขอบพระคุณอาจารย์สุรพงษ์ สิริพงศ์ดี อาจารย์ที่ปรึกษาปริญญานิพนธ์ รวมถึงคณาจารย์ประจำภาควิชาครุศาสตร์วิศวกรรมทุกท่าน ที่กรุณาให้คำแนะนำ แนวความคิด ความรู้ต่างๆ แนวทางในการแก้ไขปัญหา ในการจัดทำปริญญานิพนธ์ และเพื่อนๆ ทุกท่านที่ให้ความอนุเคราะห์ในการช่วยเหลือด้านต่างๆ สุดท้ายที่ควรระลึกถึงอย่างยิ่ง บิดา และมารดา ที่เป็นผู้ให้ความสนับสนุนทางด้านการศึกษา เงินทุน และให้กำลังใจด้วยดีตลอดมาตั้งแต่อดีตจนถึงปัจจุบัน จนทำให้ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มาของปริญญานิพนธ์

ในปัจจุบันความเจริญก้าวหน้าทางเทคโนโลยีในด้านต่าง ๆ ได้นำเครื่องมืออิเล็กทรอนิกส์ที่ใช้เทคโนโลยีระดับสูง เช่น คอมพิวเตอร์ เครื่องควบคุมการผลิต และอื่น ๆ เพิ่มขึ้นอย่างมากมาย ซึ่งเครื่องมือเหล่านี้ ล้วนแต่ต้องการแหล่งจ่ายพลังงานที่มีคุณภาพดี มีเสถียรภาพแรงดันไฟฟ้าที่คงที่ และมีความต่อเนื่องเพื่อป้องกันผลเสียที่จะเกิดขึ้นจากการผิดพลาดของระบบการจ่ายพลังงานไฟฟ้า อันเนื่องมาจากสาเหตุ เช่นไฟตก ไฟเกิน ไฟดับ เป็นต้นหรือสาเหตุใด ๆ ก็ตาม

จึงมีการใช้เครื่องจ่ายไฟฟ้าสำรองกันอย่างแพร่หลายในปัจจุบัน เพื่อให้สามารถเลือกและใช้งานเครื่องจ่ายไฟฟ้าสำรองได้อย่างถูกต้องและมีประสิทธิภาพสูงสุด เราจึงควรที่จะทราบถึงคุณสมบัติ โครงสร้าง ลักษณะ การทำงาน ของเครื่องจ่ายไฟฟ้าสำรองและแนวทางในการแก้ไขปัญหาที่จะเกิดขึ้นกับการใช้งานเครื่องจ่ายไฟฟ้าสำรองได้ กลุ่มกระผมจึงมีแนวคิดในการสร้างเครื่องจ่ายไฟฟ้าสำรองขึ้น

1.2 ขีดความสามารถของโครงงาน

1) เป็นแหล่งจ่ายไฟฟ้าสำรองให้กับคอมพิวเตอร์และอุปกรณ์ไฟฟ้าอื่นๆได้
2) เป็นแหล่งจ่ายไฟฟ้าสำรองที่ใช้ไมโครคอนโทรลเลอร์ในการควบคุม
3) เป็นแหล่งจ่ายไฟฟ้าสำรองที่ใช้ซอฟแวร์ในการปรับปรุงการทำงาน
4) เป็นแหล่งจ่ายไฟฟ้าสำรองที่มีระบบป้องกันโหลดเกิน
5) เป็นแหล่งจ่ายไฟฟ้าสำรองที่มีการแสดงผลการทำงานผ่านทางคอมพิวเตอร์

1.3 เนื้อหาโดยสังเขป

เนื้อหาภายในปริญญานิพนธ์ฉบับนี้ได้มีการแบ่งออกเป็นบทต่างๆ เพื่อความสะดวกต่อการศึกษาและทำความเข้าใจ ในแต่ละบทประกอบด้วยเนื้อหาที่สำคัญดังนี้

บทที่ 1 ได้กล่าวถึงที่มาของโครงงานนี้ ขีดความสามารถ และเนื้อหาพอสังเขป

บทที่ 2 ทฤษฎีและหลักการที่ใช้ในการสร้างและการออกแบบของแหล่งจ่าย ไฟฟ้าสำรอง โดยจะกล่าวถึงหลักการทำงานของส่วนต่างๆ ได้แก่ วงจรอินเวอร์เตอร์ วงจรอัดประจุแบตเตอรี่ วงจรสวิตซ์สับเปลี่ยน วงจรควบคุมไมโครคอนโทรลเลอร์ และการแสดงผลทางคอมพิวเตอร์

บทที่ 3 กล่าวถึงการออกแบบ การสร้างและการทำงาน ซึ่งจะกล่าวถึงการออกแบบวงจรอินเวอร์เตอร์ซึ่งใช้การออกแบบอินเวอร์เตอร์ชนิดพุช-พูล การออกแบบวงจรอัดประจุแบตเตอรี่ การออกแบบวงจรสวิตซ์สับเปลี่ยน การออกแบบวงจรรักษาระดับแรงดัน และ การออกแบบการแสดงผลทางคอมพิวเตอร์

บทที่ 4 กล่าวถึงการทดลองและผลการทดลอง การทำงานของแหล่งจ่ายไฟฟ้าสำรองในส่วนต่างๆ ได้แก่ การทดลองและผลการทดลองในส่วนฮาร์ดแวร์ การทดลองและผลการทดลองในส่วนของการควบคุมไมโครคอนโทรลเลอร์ และ การทดลองและผลการทดลองในส่วนของการแสดงผลทางคอมพิวเตอร์

บทที่ 5 เป็นบทสรุป ปัญหาในการทำงานในการสร้างโครงงานชิ้นนี้ และ แนวทางการแก้ไขในการพัฒนาต่อไปในอนาคต

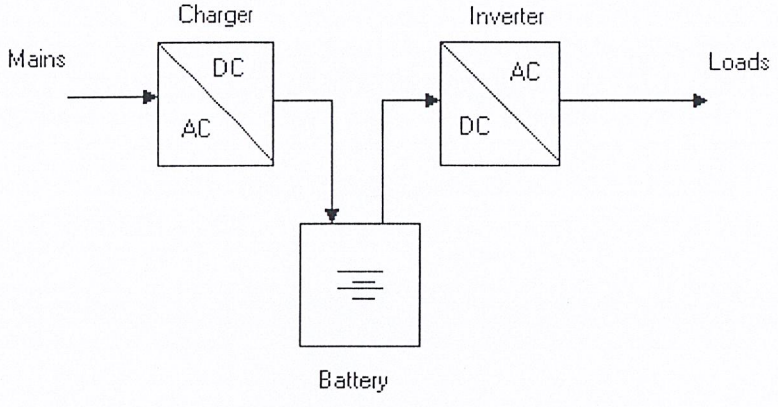
และในภาคผนวก ก จะเป็นการแสดงรายละเอียดของตัวอุปกรณ์ที่ใช้ในการสร้าง ภาคผนวก ข เป็นการแสดงโค๊ดโปรแกรมในส่วนของไมโครคอนโทรลเลอร์ และ ภาคผนวก ค จะเป็นการแสดงโค๊ดโปรแกรมในส่วนของการแสดงผลทางคอมพิวเตอร์

บทที่ 2

ทฤษฎีและหลักการ

2.1 กล่าวนำ

ยูพีเอส (UPS) ย่อมาจาก Uninterruptible Power Supply ซึ่งนิยมเรียกเป็นภาษาไทยว่า "แหล่งจ่ายไฟต่อเนื่อง" เป็นเครื่องมือชนิดหนึ่งที่นำไฟฟ้ากระแสสลับมาเปลี่ยนเป็นกระแสตรงพร้อมๆกับนำไปเก็บไว้ในแบตเตอรี่และแปลงไฟกลับจากไฟฟ้ากระแสตรงเป็นกระแสสลับอีกครั้งเพื่อจ่ายให้กับโหลด (Load) และโหลดยังสามารถทำงานได้ตลอดอย่างเป็นปกติดังแสดงในรูปที่ 2.1



รูป 2.1 โครงสร้างและหลักการทำงานโดยรวมของยูพีเอส

จากที่กล่าวมาจึงสามารถกำหนดลักษณะของยูพีเอสได้ดังต่อไปนี้

1) ยูพีเอส ต้องสามารถแก้ไขปัญหามลภาวะในระบบไฟฟ้าได้ ไฟฟ้าขาเข้า (Input) และขาออก (Output) ของยูพีเอสจะแยกอิสระออกจากกันทำให้ไฟฟ้าที่ออกจากยูพีเอสเป็นไฟฟ้าที่บริสุทธิ์ปราศจากมลภาวะทางไฟฟ้า เมื่อเปรียบกรรมวิธีทางไฟฟ้าของยูพีเอสแล้วจะคล้ายๆ กับกรรมวิธีในการกลั่นน้ำที่สะอาดบริสุทธิ์ ไฟฟ้าที่ได้จากยูพีเอส จึงเป็นไฟฟ้าที่สะอาดบริสุทธิ์เหมาะที่จะนำไปจ่ายให้กับคอมพิวเตอร์และอุปกรณ์ไฟฟ้า

2) ในกรณีที่ไฟฟ้าขาเข้าขาดหายไป ไฟฟ้าขาออกต้องไม่หายไปเลย แม้เพียงเสี้ยววินาทีจึงสามารถแก้ไขปัญหาไฟตก ไฟเกิน ไฟกระพริบ ไฟดับได้ในเวลาที่แน่นอนตามคุณสมบัติของยูพีเอสแต่ละตัว

ยูพีเอส จะต้องมีลักษณะการทำงานตามที่กล่าวมา มีประชาชนจำนวนมากยังเข้าใจว่าเครื่องมือที่จ่ายพลังงานไฟฟ้าให้เมื่อไฟฟ้าดับเรียกว่ายูพีเอส (UPS) ซึ่งยังไม่ถูกต้องนัก เพราะยังมีเครื่องมืออีกชนิดหนึ่งเรียกว่า Standby Power Supply จะมีลักษณะการทำงานแตกต่างกับยูพีเอสโดยชัดเจน

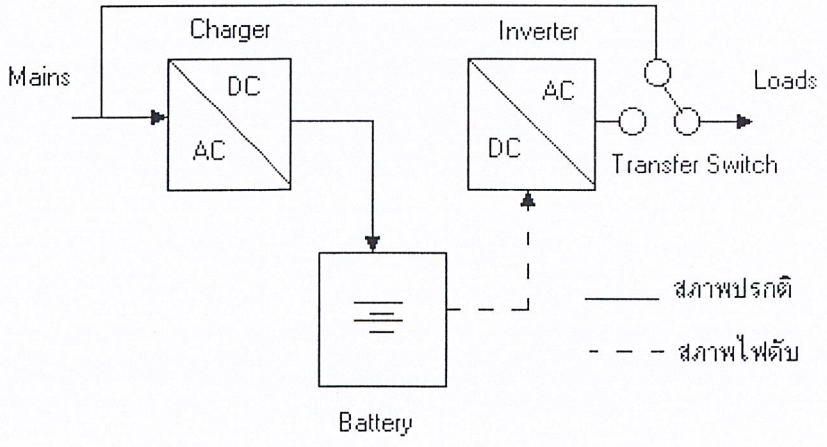
2.2 ชนิดของ UPS

จะแบ่งออกเป็น 3 ชนิด คือ

1) ชนิด Off-line UPS

เป็นยูพีเอสที่ออกแบบให้สามารถป้องกันปัญหาไฟดับได้เพียงอย่างเดียวเท่านั้น ทำให้ราคาถูกที่สุดในชนิดของยูพีเอสทั้งหมด แต่ไม่ค่อยจะเหมาะกับการใช้งานในบ้านเราเท่าไรนัก เนื่องจากปัญหาจากไฟฟ้าที่เกิดขึ้นส่วนใหญ่นั้นเป็นปัญหาไฟตก ไฟกระชาก และสัญญาณรบกวนมาก เมื่อเทียบกับอัตราการเกิดไฟดับดังรูปที่ 2.2

หลักการทำงาน

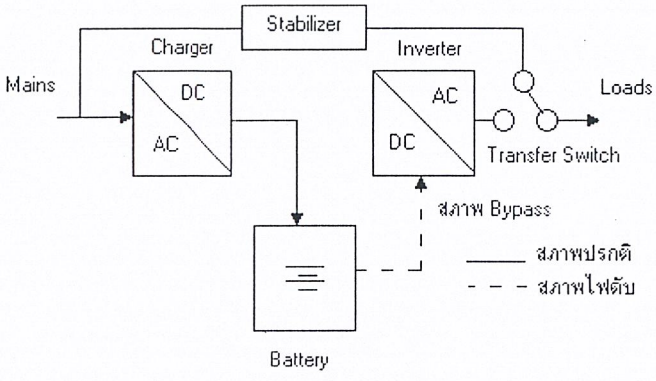


รูปที่ 2.2 แผนผังการทำงาน การทำงานของ off-line UPS

2) ชนิด Line-interactive UPS (ยูพีเอสที่มี Stabilizer อยู่ในตัว)

เป็นยูพีเอสที่นำลักษณะต่างๆที่มีอยู่ในยูพีเอสแบบที่หนึ่งมาปรับปรุงโดยเพิ่มระบบป้องกันแรงดันไฟฟ้าสูงหรือต่ำ หรือเท่าที่เรียกกันว่า Stabilizer เข้าไป ทำให้ยูพีเอสไม่ต้องจ่ายไฟสำรองจากแบตเตอรี่ทุกครั้งที่ไฟตก หรือไฟเกินจำนวนไม่มากนัก มีหลักการทำงานดังรูปที่ 2.3

หลักการทำงาน

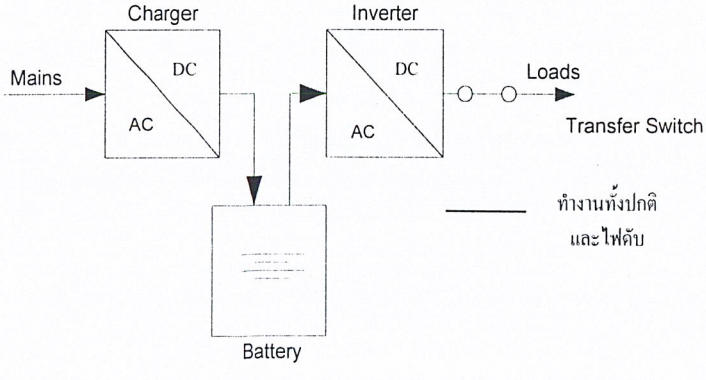


รูปที่ 2.3 แผนผังการทำงาน การทำงานของ Line-interactive UPS

3) Online UPS หรือ Double Conversion UPS

เป็นยูพีเอสที่มีประสิทธิภาพดีที่สุดที่ได้รับการพัฒนาในปัจจุบันและสามารถป้องกันปัญหาทางไฟฟ้าได้เกือบทุกกรณี ไม่ว่าจะเป็นไฟดับ ไฟตก ไฟเกิน หรือสัญญาณรบกวนทำให้ราคาของยูพีเอสแบบนี้มีราคาสูงกว่ายูพีเอสชนิดอื่นๆ ดังรูปที่ 2.4

หลักการทำงาน



รูปที่ 2.4 แผนผังการทำงาน การทำงานของ Double Conversion UPS

ตารางที่ 2.1 การเปรียบเทียบข้อดี-ข้อเสียของ UPS ชนิดต่างๆ

ชนิด UPS	ข้อดี	ข้อเสีย
1) Off-Line UPS	1) ราคาถูก 2) รองรับการทำงานในสภาวะที่ระดับแรงดันไฟฟ้าที่มีการ เปลี่ยนแปลงสูงได้ดี 3) อายุการใช้งานของตัวแบตเตอรี่และตัวยูพีเอสยาวนาน	1) ไม่สามารถป้องกันเรื่องของแรงดันไฟฟ้าที่ไม่คงที่ และการเกิดสัญญาณรบกวนได้ 2) ถ้าเกิดไฟต่ำมาก (ต่ำกว่า 190 โวลต์) ยูพีเอสจะจ่ายไฟจากแบตเตอรี่ให้กับเครื่องใช้ไฟฟ้าทันทีทั้งที่ไฟยังไม่ดับทั้งนี้เนื่องจากค่าที่ตั้งไว้นั้นต่ำกว่าช่วงที่กำหนดทำให้ยูพีเอส เข้าใจว่าไฟดับ จึงจ่ายไฟออกมา ทำให้มักพบปัญหาประจุไฟฟ้าที่อยู่ในแบตเตอรี่หมดไปทั้งที่ใช้งานไปได้เพียงเล็กน้อย
2) Line-interactive UPS	1) ราคาไม่แตกต่างจากแบบแรกมากนัก คุณภาพที่ได้ใกล้เคียงกับแบบ Double Conversion UPS 2) รองรับการทำงาน ณ สภาวะที่ระดับแรงดันไฟฟ้ามีการเปลี่ยนแปลงสูงได้ดี 3) อายุการใช้งานของตัวแบตเตอรี่และตัวยูพีเอสยาวนาน	1) ไม่สามารถป้องกันการรบกวน ทางไฟฟ้าบางอย่าง เช่น สัญญาณขนาดเล็กหรือแรงดันไฟฟ้าตก/เกิน เพียงเล็กน้อยสามารถผ่านเข้าไปในเครื่องคอมพิวเตอร์ได้ แต่ไม่มีผลต่อเครื่องใช้ไฟฟ้าโดยทั่วไป 2) ไม่เหมาะกับเครื่องใช้ไฟฟ้าที่ไวต่อคุณภาพของกระแสไฟฟ้ามากๆเช่น เครื่องจักรที่ใช้ในโรงงานอุตสาหกรรม เป็นต้น
3) Online UPS หรือ Double Conversion UPS	1) การใช้ยูพีเอสแบบนี้สามารถจะช่วยทำให้เครื่องคอมพิวเตอร์ได้รับกระแสสลับที่มีคุณภาพสูง และมีความเที่ยงตรงของระดับ แรงดัน และป้องกันสัญญาณรบกวนได้ทุกชนิด 2) กรณีที่ไฟดับหรือขาดช่วงยูพีเอสจะจ่ายไฟสำรองเก็บไว้ในแบตเตอรี่ออกมา แล้วแปลงไฟฟ้ากระแสตรงให้ไปเป็นกระแสสลับได้ทันที	1) มีราคาแพง 2) อายุการใช้งานของแบตเตอรี่ สั้นกว่าชนิดอื่น 3) มีกำลังสูญเสียภายในตัวสูง

2.3 วงจรอินเวอร์เตอร์

สำหรับในโครงงานนี้วงจรอินเวอร์เตอร์ที่เลือกใช้งานจะใช้แบบ พุช-พูล อินเวอร์เตอร์ (Push-Pull Inverter) ซึ่งคุณลักษณะที่เหมาะสมของการเลือกใช้แบบนี้ก็เพื่อ

1) สามารถจ่ายกำลังไฟฟ้าได้สูง 500 วัตต์ขึ้นไป ดังนั้นค่าพิกัดของแหล่งจ่ายไฟฟ้าสำรองในโครงงานนี้จึงอยู่ในเกณฑ์ที่พิจารณา

2) แรงดันที่ใช้ในการสวิตซ์ชิ่งของอุปกรณ์มีค่าต่ำ

3)ราคาถูกที่สุดเมื่อเปรียบเทียบกับอินเวอร์เตอร์ชนิดอื่นๆ เนื่องจากว่าใช้อุปกรณ์จำนวนน้อยในการสร้างวงจรอินเวอร์เตอร์

4) ไม่คำนึงถึงผลของคุณสมบัติของรูปคลื่นเท่าไรนัก เนื่องจากว่าจุดประสงค์ของโครงงานนี้ เพื่อสามารถจ่ายกำลังไฟฟ้าสำรองแทนระบบได้เท่านั้น เมื่อเกิดผิดปกติขึ้นกับระบบ แต่สิ่งที่พิเศษกว่าคือการควบคุมที่ง่ายด้วยไมโครคอนโทรลเลอร์ และมีการแสดงผลด้วยคอมพิวเตอร์

5) เนื่องจากรูปคลื่นสัญญาณในการสวิตซ์ชิ่งวงจรอินเวอร์เตอร์ จะสวิตซ์ชิ่งที่ความถี่ 50 Hz ดังนี้อุปกรณ์สวิตซ์ชิ่งจึงไม่จำเป็นต้องตอบสนองใช้เวลาที่รวดเร็ว (Slow dynamic Response)

6) เนื่องจากสัญญาณที่ใช้ในการขับเป็นรูปคลื่นสัญญาณสี่เหลี่ยม มีความถี่ 50Hz ดังนั้นการสร้างสัญญาณขับนี้ โดยใช้ไมโครคอนโทรลเลอร์จะทำได้ง่าย

7) การออกแบบวงจรได้ง่าย เนื่องจากว่าใช้งานที่ความถี่ต่ำ จึงมักไม่ค่อยมีปัญหาเรื่องของสไปค์ (Spike) การใช้หม้อแปลงความถี่สูง หรือผลจากเส้นแรงแม่เหล็กมีค่าไม่เท่ากัน

8) การออกแบบหม้อแปลงไม่ต้องใช้แกนเฟอร์ไรท์ ซึ่งจะช่วยประหยัดได้มาก

ดังนั้นจากคุณลักษณะข้างต้นจึงได้ทำการออกแบบวงจรอินเวอร์เตอร์ที่ใช้ในโครงงานนี้ให้มีเงื่อนไขเป็นไปตามคุณลักษณะดังกล่าว โดยแสดงเป็นหัวข้อได้ดังนี้

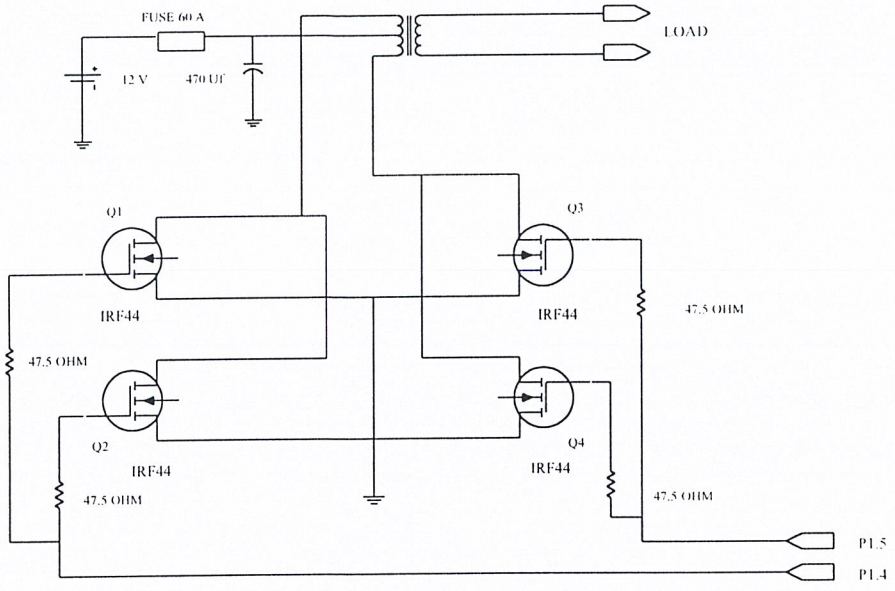
1) วงจรอินเวอร์เตอร์ใช้อุปกรณ์สวิตซ์ชิ่ง คือ มอสเฟตจำนวน 4 ตัว แบ่งเป็นบวกและลบ โดยทำงานด้านละ 2 ตัว

2) ใช้รูปคลื่นสัญญาณสี่เหลี่ยม (Square wave signal) มีความถี่ที่ 50 Hz จำนวน 2 สัญญาณกลับเฟสกัน 180 องศาไฟฟ้า ขับวงจรอินเวอร์เตอร์ซึ่งสร้างจากไมโครคอนโทรลเลอร์

3) แรงดันไฟฟ้าตรงที่ใช้ขับเกทมอสเฟตมีค่า 12 โวลต์

4) หม้อแปลงถูกออกแบบใช้งานที่ความถี่ 50Hz

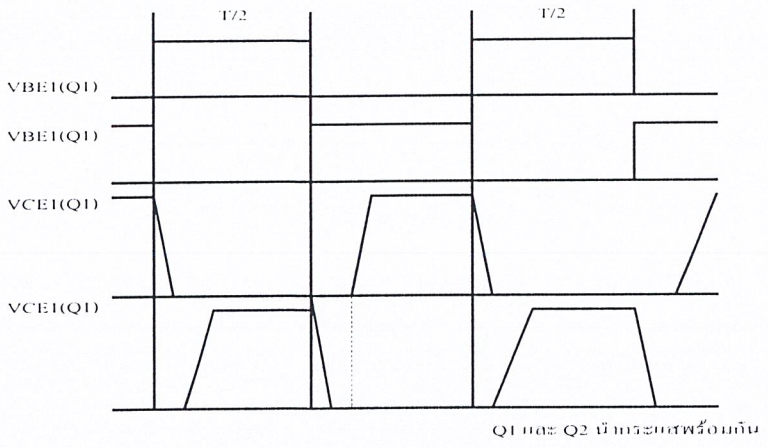
ดังนั้นในการออกแบบจะอาศัยคุณลักษณะและเงื่อนไขดังกล่าวนี้ในการออกแบบวงจรอินเวอร์เตอร์ โดยต่อไปจะกล่าวถึงทฤษฎีการใช้งานไปจนถึงการออกแบบวงจรที่ใช้ในโครงงานนี้ วงจรพื้นฐานการทำงานของวงจร พุช-พูล อินเวอร์เตอร์ (Basic of Push-Pull Inverter) ดังรูปที่ 2.5



รูปที่ 2.5 วงจรพื้นฐานของวงจรพุช-พูลอินเวอร์เตอร์

หลักการทำงานของวงจรพุช-พูล อินเวอร์เตอร์ โดยอธิบายได้คือเพาเวอร์มอสเฟต Q1และ Q2 จะสลับการทำงานโดยผลัดกันนำกระแสในแต่ละครึ่งของคาบเวลา T ขณะที่ Q1 นำกระแสจะมีกระแสไพรมารี่ (I_{P1})ไหลผ่านขดไพรมารี่ (N_{P1}) เนื่องมาจากมีสัญญาณมาขับเกทและในจังหวะนี้จะไม่มีสัญญาณขับที่เกทของ Q2 ดังนั้น Q2จะไม่ทำงาน แรงดันที่คร่อมเดรน – ซอร์ส (V_{DS}) จะมีค่าเป็น $2V_{in}$ ซึ่งเป็นผลรวมของแรงดันของขด NP1 และ NP2 รวมกัน เนื่องจากถูกเหนี่ยวเกิดขึ้นมีทิศทางเดียวกัน ดังนั้นจากเหตุผลนี้ทั้ง Q1 และ Q2 จะต้องสามารถทนแรงดันอย่างต่ำได้เป็น 2 เท่าของแรงดันอินพุต โดยในสภาวะขณะนี้หม้อแปลงไฟฟ้าทำงานในครึ่งคาบ เหนี่ยวนำแรงดันขึ้นในเวลาต่อมาสัญญาณขับเกทจะขับที่ Q2 แทนและ Q1 สัญญาณขับจะเป็นศูนย์ ซึ่งจะมีความถี่ของสัญญาณเท่ากันแต่มีความต่างเฟสกัน 180 องศาไฟฟ้า กระแสไพรมารี่ (I_{P2}) จะไหลผ่านขดไพรมารี่ (N_{P2}) ไหลครบวงจรโดยผ่าน Q2 ซึ่งหม้อแปลงก็จะเหนี่ยวนำแรงดันขึ้นโดยมีทิศทางดอทจะอยู่ด้านล่างซึ่งแรงดันที่ปรากฏที่ด้านทุติยภูมิจะมีลักษณะเป็นกระแสสลับ โดยเกิดจากการสลับการทำงานของอุปกรณ์สวิตซ์ชิ่งนี้เอง สำหรับสัญญาณขับจะเป็นรูปคลื่นสี่เหลี่ยม (Square Wave) กลับเฟสกันและทำการขับเพียงครึ่งคาบเท่านั้น ซึ่งในความเป็นจริงแล้วหากสัญญาณขับพร้อมกันหรือมีการทำงานที่ต่างกันบางประการ

ซึ่งจะเป็นส่วนที่ทำให้วงจรอินเวอร์เตอร์มีการทำงานผิดพลาดขึ้น ดังนั้นเพื่อความปลอดภัยจึงต้องมีเวลาเผื่อสำหรับการสวิตซ์ชิ่งของอุปกรณ์ด้วย ซึ่งจะแสดงได้ดังรูปที่ 2.6



รูปที่ 2.6 การหน่วงของช่วงเวลานำกระแสของอุปกรณ์สวิตซ์ชิ่งเนื่องจากประจุสะสม

โดยจะขออธิบายโดยการพิจารณา ซึ่งจะเป็นเพาเวอร์ทรานซิสเตอร์แทน ซึ่งจะมีหลักการในการพิจารณาแบบเดียวกัน กล่าวคือการกำหนดค่าเวลาเผื่อ (t_D;dead time) ไว้ เพื่อไม่ให้เพาเวอร์ทรานซิสเตอร์มีช่วงการนำกระแสมากเกินไป เมื่อสัญญาณขับมีความกว้างพัลส์ใกล้ 180 องศาไฟฟ้า ซึ่งจะมีเหตุผลหลักสองประการคือ

1) ถ้าไม่มีการจำกัดช่วงเวลานำกระแสเอาไว้ เมื่อเกิดกรณีโหลดดึงกระแสมากขึ้น วงจรควบคุมจะสั่งงานให้เพาเวอร์ทรานซิสเตอร์มีค่าช่วงเวลานำกระแสเพิ่มขึ้น ทำให้ช่วงเวลาการนำกระแสของอีกตัวหนึ่งลดลง และเกิดการไม่สมมาตรฟลักซ์ (Flux imbalance) ขึ้นในแกนเหล็กของหม้อแปลง ซึ่งจะทำให้แกนอิ่มตัวอย่างรวดเร็วและอุปกรณ์สวิตซ์ชิ่งเกิดพังเสียหายขึ้นได้

โดยสาเหตุที่สำคัญที่ทำให้เกิดการไม่สมมาตรฟลักซ์ของแกนเหล็กนั้น เนื่องมาจากการมีช่วงเวลาก่อนหยุดนำกระแส (turn off time) ไม่เท่ากันอย่างแท้จริงเนื่องจากข้อจำกัดในการผลิต ถ้าหากว่าอุปกรณ์ตัวใดใช้ในการหยุดนำกระแสมากกว่า จะทำให้ค่ากระแสสูงสุดของมันในช่วงที่นำกระแสมีค่ามากกว่าของอีกตัวหนึ่ง ลักษณะเช่นนี้จะทำให้ฟลักซ์แม่เหล็กที่เกิดขึ้นในหม้แปลงมีลักษณะไม่สมมาตร และจะมีการเลื่อนของฟลักซ์ได้หรือที่เรียกว่า “Flux walking” เกิดขึ้น โดยค่าฟลักซ์สูงสุดจะวิ่งเข้าหาจุดอิ่มตัวอย่างรวดเร็ว ถ้าแกนเกิดการอิ่มตัวของฟลักซ์แม่เหล็กก็จะเป็นอันตรายต่อตัวเพาเวอร์มอสเฟตได้

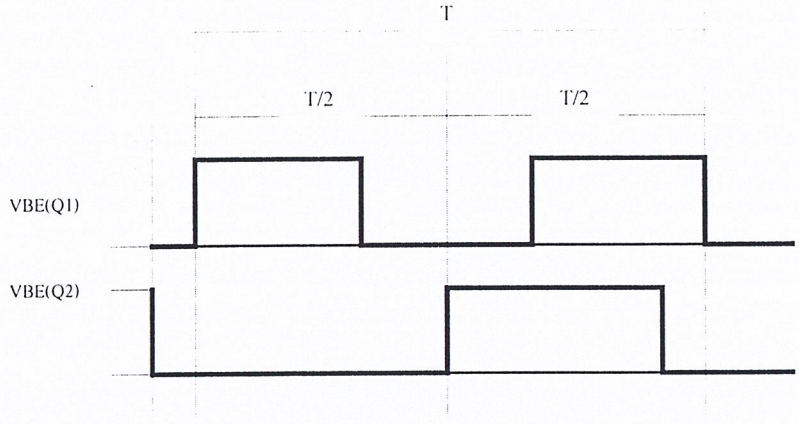
2) ถึงแม้จะกำหนดให้วงจรควบคุมสั่งงานให้ทำงานครึ่งหนึ่งของคาบเท่ากันแล้วก็ตาม ก็ยังไม่มีความปลอดภัยเพียงพอเพราะการตอบสนองของอุปกรณ์อาจเป็นไปได้ช้า เนื่องจากเกิดประจุสะสมขึ้นในขณะที่นำกระแส เมื่อหยุดจ่ายสัญญาณขับให้กับเพาเวอร์ทรานซิสเตอร์ที่ เวลา T – T/2 ประจุสะสมจะทำให้นำกระแสไปช่วงระยะเวลาหนึ่ง ในขณะเดียวกันอีกตัวหนึ่งเริ่มรับสัญญาณขับ

ที่เวลา T =T/2 เช่นเดียวกัน ทำให้เพาเวอร์ทรานซิสเตอร์ทั้งสองตัวเกิดนำกระแสพร้อมกัน ในกรณีนี้แรงดันตกคร่อมทรานซิสเตอร์ทั้งสองตัวขณะนำกระแสจะเท่ากับค่าแรงดันของอินพุตและทำให้เกิดการพังเสียหายอย่างรวดเร็ว

การกำหนดค่าเวลาเผื่อ t_D สำหรับช่วงเวลานำกระแสให้กับเพาเวอร์ทรานซิสเตอร์ Q_1 และ Q_2 ดังรูปที่ 2.7 จะช่วยป้องกันปัญหาที่กล่าวมาแล้วได้ ค่าเวลาเผื่อ t_D โดยทั่วไปจะกำหนดไว้ประมาณ 20% ของครึ่งคาบเวลา ดังนั้นช่วงเวลานำกระแสสูงสุดของเพาเวอร์ทรานซิสเตอร์ Q_1และ Q_2 จึงไม่ควรมีค่าเกิน

$$t_{On(max)} = 0.8(T/2)$$
$$= 0.4T \qquad (2.1)$$

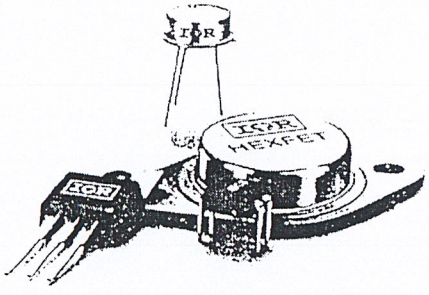
โดยจะแสดงได้ดังรูปที่ 2.7



รูปที่ 2.7 การกำหนดค่าเผื่อ t_D ให้กับเพาเวอร์ทรานซิสเตอร์

2.3.1 มอสเฟตกำลัง (Power MOSFET)

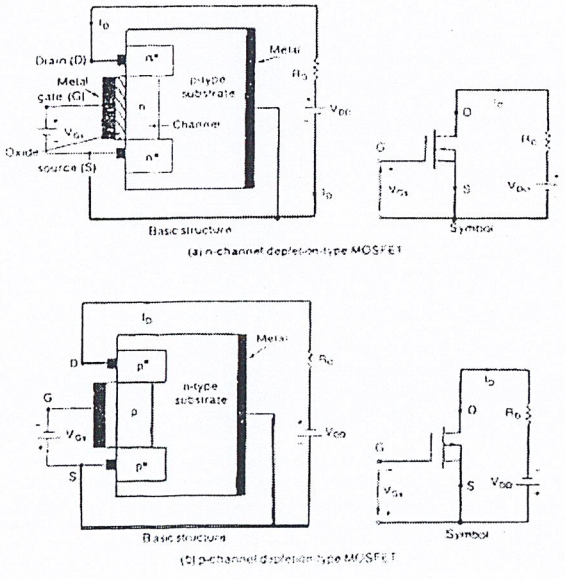
มอสเฟตเป็นอุปกรณ์ประเภทใช้แรงดันในการควบคุม กล่าวคือต้องใช้แรงดัน V_{GS} ควบคุมในการไหลของกระแสเดรน (I_D)



รูปที่ 2.8 มอสเฟตประเภทใช้แรงดันควบคุม

2.3.2 ประเภทของมอสเฟต

มอสเฟตเป็นอุปกรณ์ที่มี 3 ขั้ว คือ ขั้วซอร์ส (Source;S) เดรน (drain;D) และเกท (Gate;G) แบ่งออกเป็น 2 ประเภทคือมอสเฟตแบบดีพลีชัน (Depletion MOSFET) เรียกสั้นๆว่า ดีมอสเฟต และมอสเฟตแบบเอนฮานซ์เมนท์ (Enhancement MOSFET) หรือเรียกว่า อีมอสเฟต แต่ละประเภทยังแบ่งออกเป็น 2 แบบคือ แชนเนล N (N-Channel) และ แชนเนล P(P-Channel) ซึ่งแสดงได้ดังรูปที่ 2.9

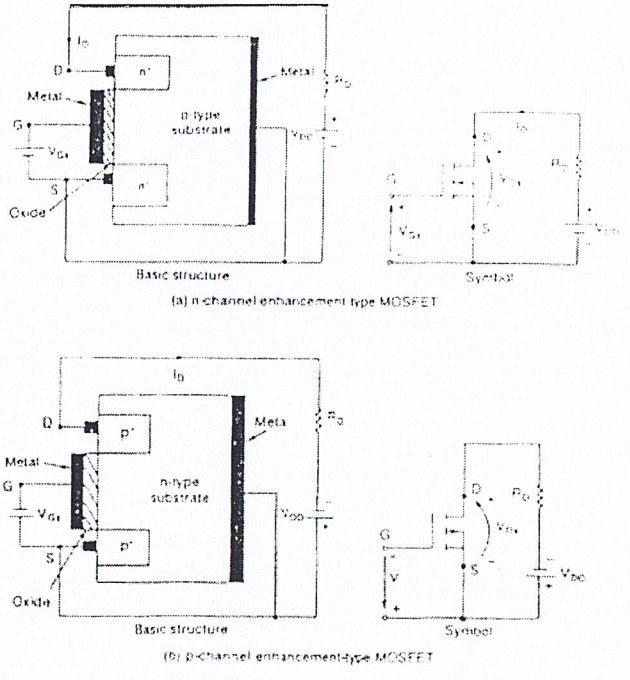


รูปที่ 2.9 โครงสร้างของมอสเฟตแบบ N และ P ชนิดดีพลีชั่น

โครงสร้างของดีมอสเฟตแบบแชนเนล N ในรูปที่ a ประกอบด้วยสารกึ่งตัวนำชนิด P ซึ่งสร้างขึ้นจากซิลิคอน และเรียกว่าแผ่นผนึกฐาน P(P-Type Substrate) โดยขั้ว D และขั้ว S ต่อกับบริเวณสารกึ่งตัวนำ N ทั้งสองส่วนนี้ต่อกับวัสดุผิวนอกที่เป็นโลหะ (Metal) โดยมีซิลิคอนไดออกไซด์ (SiO_2) กั้นระหว่างแชนเนล N กับขั้ว G [ซิลิคอนไดออกไซด์เป็นฉนวนประเภทไดอิเล็กทริก (Dielectric)] ถ้าแรงดันเกท-ซอร์ส (V_{GS}) มีค่าเป็นลบ อิเล็กตรอนบางส่วนในแชนแนล N สร้างบริเวณปลอดพาหะ(Depletion Region) ภายในขั้วออกไซด์ ทำให้แชนแนลแคบลง ความต้านทานจากขั้วเดรนไปซอร์ส (R_{DS}) เริ่มมีค่าสูง เมื่อ V_{GS} มีค่าเป็นลบเพียงพอ แชนแนลจะหายไป R_{DS} จึงมีค่าสูงมาก และไม่มีกระแสไหลจากขั้วเดรนไปซอร์สนั่นคือ I_{DS} = ศูนย์ ระดับ V_{GS} นี้เรียกว่า แรงดันพินช์ออฟ (pinch-off Voltage;Vp)

ในทางตรงข้าม ถ้า V_{GS} เป็นบวก แชนแนลจะกว้างขึ้นเป็นผลให้ I_{DS} มีค่าเพิ่มขึ้น นั่นเนื่องจาก R_{DS} มีค่าลดลง

หลักการที่กล่าวมานี้ ใช้ได้กับดีมอสเฟตแบบแชนเนล P (ดังรูป B) โดยขั้วของ V_{DS},I_{DS} และ V_{GS} เปลี่ยนเป็นขั้วตรงข้ามกับกรณีดีมอสเฟตแบบแชนเนล N



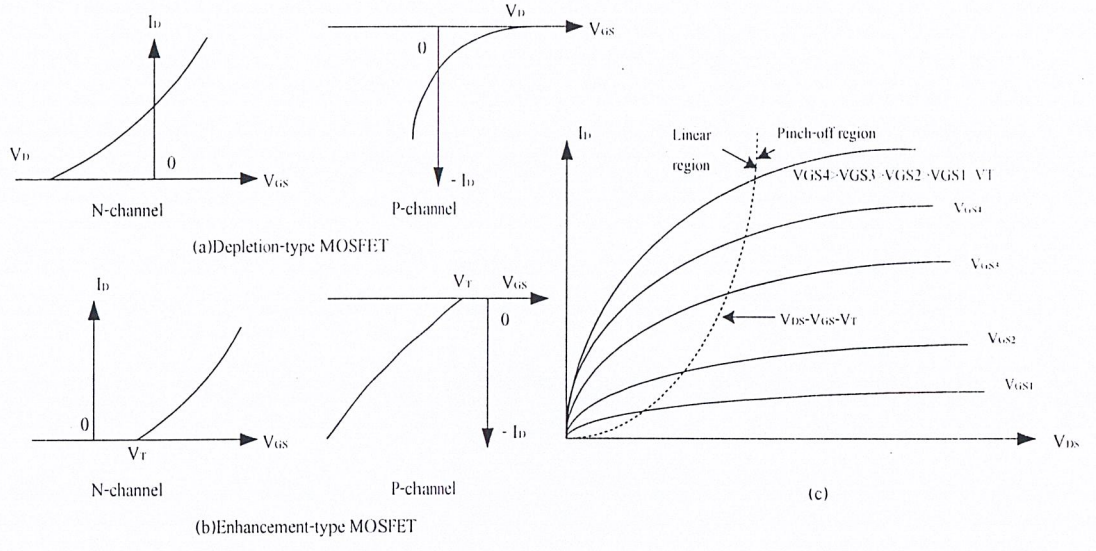
รูปที่ 2.10 โครงสร้างของมอสเฟตแบบ N และ P ชนิดเอนฮานท์เมนต์

อีมอสเฟตแบบแชนแนล N นั้นไม่มีแชนแนลดังรูปที่ a ถ้า V_{GS} เป็นบวก แรงดันเหนี่ยวนำจะดึงดูดอิเล็กตรอนจากแผ่นผนึกฐาน P และรวบรวมไว้ที่ผิวภายในขั้นออกไซด์ ถ้า V_{GS} มากกว่าหรือเท่ากับค่าแรงดันเธรชโฮลด์ (Threshold voltage; VT) จึงมีกระแสไหลจากเดรนไปซอร์ส (I_{DS})

หลักการที่กล่าวมานี้ ใช้ได้กับอีมอสเฟตแบบแชนแนล P โดยขั้วของ V_{DS},I_{DS} และ V_{GS} เปลี่ยนเป็นขั้วตรงข้ามกับกรณี อีมอสเฟตแบบแชนแนล N

2.3.3 คุณลักษณะของมอสเฟต (MOSFET Characteristic)

ได้กล่าวในตอนต้นแล้วว่า มอสเฟตเป็นอุปกรณ์ประเภทใช้แรงดันในการควบคุม และมีอิมพิแดนซ์อินพุตสูงมาก โดยขั้วเกทจะดึงกระแส่รั่วไหลต่ำมาก ซึ่งมีค่าโดยประมาณนาโนแอมป์ (Nanoampare) และเราทราบว่า ทรานส์คอนดัคแตนซ์ (Transconductance; g_m) ซึ่งเป็นอัตราส่วนของ I_D ต่อแรงดันเกท (V_{GS})เป็นตัวกำหนดคุณลักษณะการถ่ายโอนของมอสเฟตดังรูปที่ 2.11



รูปที่ 2.11 คุณลักษณะในการถ่ายโอนของอีมอสเฟตแบบแชนแนล N และ P และคุณลักษณะเอาต์พุตของอีมอสเฟตแบบแชนแนล N

โดยที่คุณลักษณะเอาต์พุตของอีมอสเฟตแบบแชนแนล N จะมีพื้นที่บริเวณการทำงาน 3 บริเวณ คือ

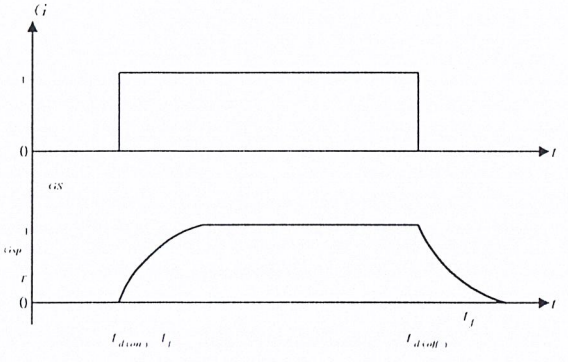
1. บริเวณคัทออฟ (cutoff Region) เกิดขึ้นเมื่อ $V_{GS}<V_T$
2. บริเวณพินช์ออฟหรือบริเวณอิ่มตัว (Pinch-off or Saturation Region) เกิดขึ้นเมื่อ $V_{DS}>V_{GS}-V_T$
3. บริเวณที่เป็นเชิงเส้น (Linear Region) เกิดขึ้นเมื่อ $V_{DS}>V_{GS}-V_T$

ในบริเวณที่เป็นเชิงเส้น Id จะเปลี่ยนแปลงตามสัดส่วนของ V_{DS} และเนื่องจาก I_D มีค่าสูง ขณะ V_{DS} มีค่าต่ำ อีมอสเฟตจึงใช้บริเวณที่เป็นเชิงเส้นนี้สำหรับพิจารณาการใช้งานสวิทชิ่ง

สำหรับอีมอสเฟต แรงดันเกท (หรือแรงดันอินพุต) อาจเป็นบวกหรือลบก็ได้ แต่อีมอสเฟตตอบสนองกับแรงดันบวกเพียงอย่างเดียว ตามปกติมอสเฟตกำลังมักเป็นประเภทอีมอสเฟต จึงขอกล่าวถึงเฉพาะคุณลักษณะของอีมอสเฟตเท่านั้น

2.3.4 คุณลักษณะในการสวิตช์ (Switching Characteristic)

ผลกระทบที่เกิดจากคาปาซิเตอร์บริเวณรอยต่อของสารกึ่งตัวนำในอีมอสเฟต ทำให้เขียนรูปคลื่นในการสวิตช์เป็นดังรูปที่ 2.12

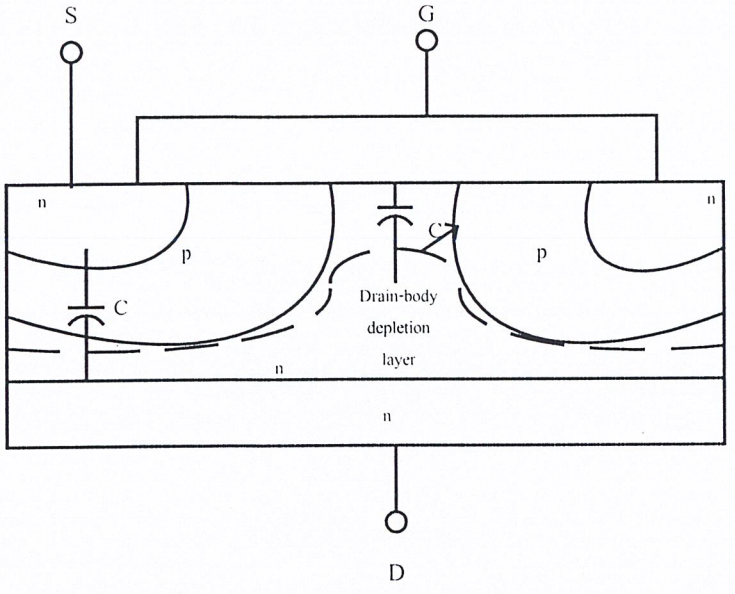


รูปที่ 2.12 รูปคลื่นแรงดันในการสวิตช์ของแรงดันเกท

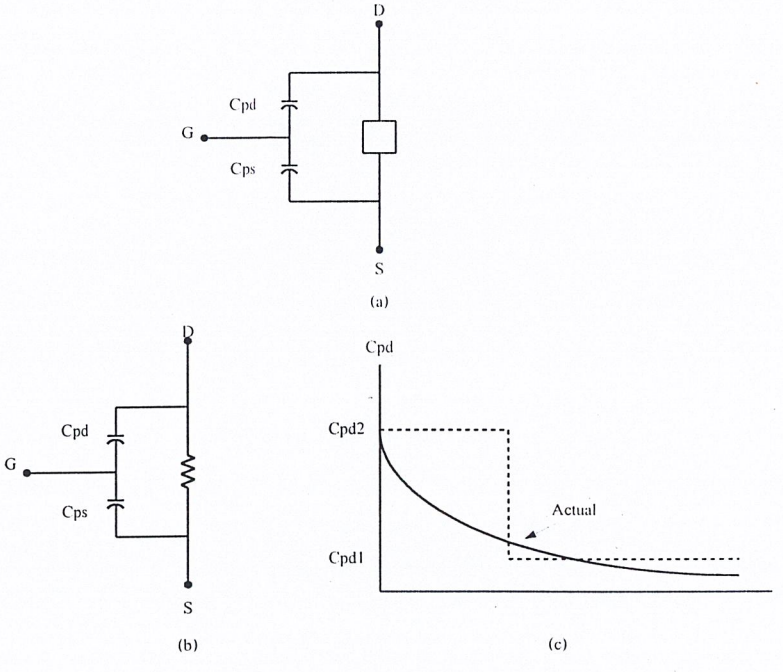
เมื่อป้อนแรงดันเกทเพิ่มจากศูนย์ถึง V_1 เวลาที่ใช้ในการอัดประจุคาปาซิเตอร์ (ภายในตัวของอีมอสเฟต) ให้มีค่าถึงระดับแรงดันเธรสโฮลด์ (V_T) ซึ่งเรียกว่า ช่วงเวลาดีเลย์ในการเปิด (turn - on delay Time ;$t_{d(ON)}$) และช่วงเวลาขาขึ้น (t_r) เป็นเวลาอัดประจุจากระดับ V_Tจนกระทั่งมีค่าถึงระดับแรงดันเกทเต็มที่ (Full gate voltage ; V_{GSP}) ซึ่งใช้ในการขับมอสเฟตไปสู่บริเวณเชิงเส้น โดยดูรูป 2.14C ประกอบ

เวลาที่คาปาซิเตอร์ในการคายประจุจาก v_1 ไปจนถึงบริเวณพินช์ออฟ เรียกว่าช่วงเวลาดีเลย์ในการปิด (turn –off delay time ;$t_{d(OFF)}$) และช่วงเวลาขาลง (t_f) เป็นเวลาที่คาปาซิเตอร์ใช้ในการคายประจุจากบริเวณพินช์ออฟจนถึง V_T (ถ้า $V_{GS}<V_T$ มอสเฟตจะปิด)

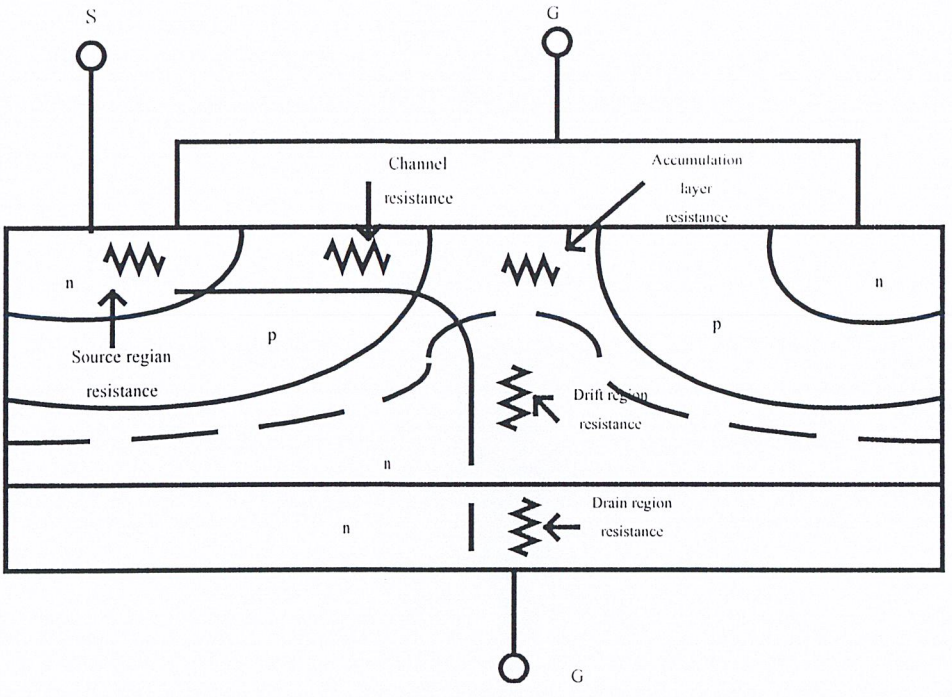
โดยจะแสดงค่าคาปาซิเตอร์ ณ จุดต่างๆ และวงจรโมเดลภายในตัวมอสเฟต ที่มีผลต่อการทำงานสวิตซ์ชิ่ง ได้ดังรูปที่ 2.13 และ รูปที่ 2.14



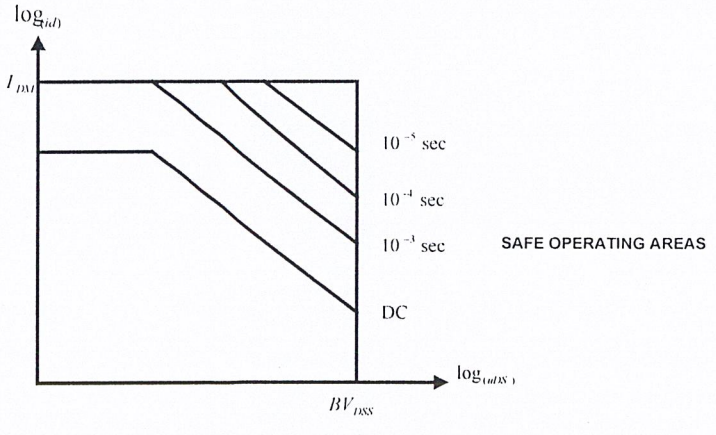
รูปที่ 2.13 พื้นที่หน้าตัดของมอสเฟตชนิด N โดยจะประมาณพื้นที่ที่เกิดค่าปาซิเตอร์



รูปที่ 2.14 วงจรโมเดลของมอสเฟตโดยแสดงพารามิเตอร์ต่างๆที่เกี่ยวข้อง



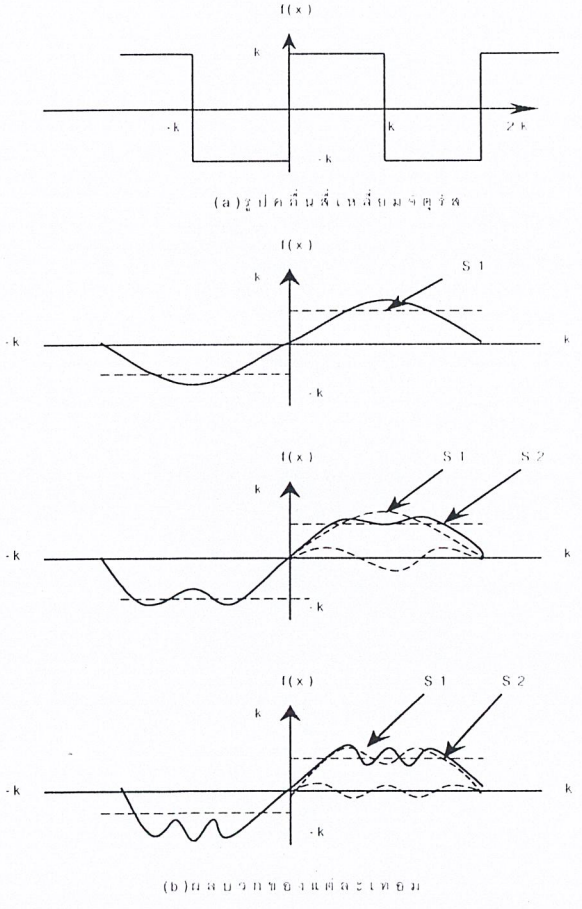
รูปที่ 2.15 ส่วนประกอบที่เป็นผลรวมค่าความต้าน R_{DS} ขณะมอสเฟตนำกระแส



รูปที่ 2.16 พื้นที่การใช้งานที่ปลอดภัย (Save Operation Area; SOA) ของมอสเฟต

2.3.5 คุณลักษณะของสัญญาณขับ (Drive Signal Characteristic)

ดังที่กล่าวมาแล้วว่าสัญญาณที่ใช้ขับมอสเฟตจะเป็นรูปคลื่นสัญญาณสี่เหลี่ยม โดยมีค่าดิวตี้ไซเคิล (k) เท่ากัน ดังนั้นค่าแรงดันที่ได้จากวงจรอินเวอร์เตอร์ และเข้าสู่หม้อแปลงเพื่อทำการแปลงแรงดันให้สูงขึ้น ดังนั้นสัญญาณที่เอาต์พุตหรือภาระจะมีสัญญาณเป็นรูปคลื่นสี่เหลี่ยมเหมือนสัญญาณขับ ซึ่งแรงดันที่ใช้งานสำหรับภาระนั้นจะมีค่าเป็นแรงดันส่วนหลัก (Fundamental Voltage) ของสัญญาณสี่เหลี่ยมนี้ โดยที่แรงดันส่วนหลักเมื่อรวมกันกับแรงดันฮาร์โมนิกส์ทุกๆ ฮาร์โมนิกส์จะรวมกันมีค่าเป็นสัญญาณรูปคลื่นสี่เหลี่ยมนี้ ซึ่งแสดงดังรูปที่ 2.17 โดยจะต้องทำการฟูเรียร์สัญญาณรูปคลื่นสี่เหลี่ยมนี้เพื่อต้องการทราบค่าแรงดันส่วนหลัก โดยจะแสดงได้คือ



รูปที่ 2.17 รูปคลื่นแบบสี่เหลี่ยมที่ได้จากการรวมสัญญาณความถี่หลักและฮาร์โมนิกส์ต่างๆ

จาก

$$f(x) = a_0 + \sum_{n=1}^{\infty} (a \cos nx + b_n \sin nx) \qquad (2.2)$$

เมื่อพิจารณาจากรูป พื้นที่ใต้กราฟในช่วง -π ถึง π เป็นศูนย์ ดังนั้น $a_0 = 0$ จึงไม่จำเป็นต้องอินทิเกรต ดังนั้น

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos nx\,dx \tag{2.3}$$

$$a_n = \frac{1}{\pi}\left[\int_{-\pi}^{0}(-k)\cos nx\,dx + \int k\cos nx\,dx\right] \tag{2.4}$$

$$= \frac{1}{\pi}\left[-\frac{k}{n}\sin(nx)\Big|_{-\pi}^{0} + \frac{k}{n}\sin(nx)\Big|_{0}^{\pi}\right] \tag{2.5}$$

$$= 0$$

เพราะว่า Sin nx = 0 ที่ x = 0,π, 2π สำหรับทุกๆค่าของ n = 1,2,.... ดังนั้น

$$b_n = \frac{1}{\pi}\left[\int_{-\pi}^{\pi} f(x)\sin nx\,dx\right. \tag{2.6}$$

$$b_n = \frac{1}{\pi}\left[\int_{-\pi}^{0}(-k)\sin nx\,dx + \int_{0}^{\pi} k\sin nx\,dx\right] \tag{2.7}$$

$$= \frac{1}{\pi}\left[\frac{k}{n}\cos nx\Big|_{-\pi}^{0} - \frac{k}{n}\cos(nx)\Big|_{0}^{\pi}\right] \tag{2.8}$$

$$= \frac{k}{n\pi}[\cos 0 - \cos(-n\pi) - \cos n\pi + \cos 0] \tag{2.9}$$

เนื่องจาก COS (-A) =COS A และ COS 0 = 1 ดังนั้น

$$b_n = \frac{k}{n\pi}(1 - \cos n\pi - \cos n\pi + 1) \tag{2.10}$$

$$= \frac{k}{n\pi}(2 - 2\cos n\pi) = \frac{2k}{n\pi}(1 - \cos n\pi) \tag{2.11}$$

ซึ่ง COS π = -1, COS 2π = 1, COS 3π = -1 ฯลฯ ดังนั้น

$$\cos nx = \begin{cases} -1, n = 1,3,5,7,... \\ 1, n = 2,4,6,8,.... \end{cases} \tag{2.12}$$

และ

$$(1-\cos nx) = \begin{cases} 2, n = 1,3,5,7,... \\ 0, n = 2,4,6,8,... \end{cases} \quad (2.13)$$

ดังนั้น

$$b_1 = \frac{4k}{\pi}, b_2 = 0, b_3 = \frac{4k}{3\pi}, b_4 = 0, b_5 = \frac{4k}{5\pi}, ..., \quad (2.14)$$

เพราะฉะนั้น จาก(1) จะได้ว่า

$$f(x) = \sum_{n=1}^{\alpha} b_n \sin nx \quad \text{เมื่อ n เป็นเฉพาะเลขคี่} \quad (2.15)$$

$$f(x) = \frac{4k}{\pi}(\sin x + \frac{1}{3}\sin 3x + \frac{1}{5}\sin 5x + ...) \quad (2.16)$$

ผลบวกของเทอมดังแสดงในรูปที่ 2.13 (b)คือ

$$S_1 = \frac{4k}{\pi}\sin x, S_2 = \frac{4k}{\pi}(\sin x + \frac{1}{3}\sin 3x), \text{ฯลฯ} \quad (2.17)$$

$$(t) = \frac{4v_d}{\pi}\sin \omega t \quad (2.18)$$

ซึ่งจากการฟูเรียร์สัญญาณรูปคลื่นสี่เหลี่ยมแล้ว เมื่อคิดเฉพาะพจน์แรกที่เป็นค่าส่วนหลัก (Fundamental) โดยเทียบ S_1=V (t), k=V_d และ X =ωt จะได้ดังสมการ

ดังนั้นจากสมการนี้ จะใช้ในการกำหนดค่าการกำเนิดสัญญาณรูปคลื่นสี่เหลี่ยมที่สร้างจากไมโครคอนโทรลเลอร์ ซึ่งจะได้กล่าวต่อไป

2.3.6 การกำเนิดรูปคลื่นสัญญาณขับ (Drive Signal)

เนื่องจากในการกำเนิดสัญญาณพัลส์ที่ใช้ในการขับ ให้กับมอสเฟตจะใช้วงจรหน่วงเวลา โดยที่วงจรหน่วงเวลาที่ใช้ในโครงการนี้สามารถที่จะเปลี่ยนเวลาในการหน่วงได้ตามสภาวะของแรงดันเอาต์พุตของโหลด ซึ่งเราสามารถที่จะอธิบายทฤษฎีของวงจรหน่วงเวลาได้ดังนี้

โปรแกรมหน่วงเวลาแบบ 2 LOOP ซึ่งในโครงงานนี้จะใช้โปรแกรมแบบนี้เป็นพื้นฐานในการออกแบบซึ่งมีรูปแบบดังนี้

```
                        MACHINE CYCLE
DELAY: MOV R1, #X1          (1)
DEL2: MOV R0, #X2           (1)
DEL1: DJNZ R0, DEL1         (2)
DJNZ R1, DEL2               (2)
RET
```

จากโปรแกรมเราสามารถที่จะเขียนสูตรในการคำนวณ จำนวน Machine Cycle ได้ดังนี้

$$\text{จำนวน Machine Cycle} = 3+R1\{3+R0(2)\} \quad \text{us} \tag{2.19}$$

สูตรนี้จะใช้กับไมโครคอนโทรลเลอร์ที่ใช้สัญญาณ Clock เท่ากับ 2 MHz ดังนั้นเราสามารถที่จะนำสูตรนี้ไปใช้ได้เลย เมื่อไมโครคอนโทรลเลอร์ MCS-51 ทำงานที่ความเร็ว 12 MHz แต่ถ้าใช้งานที่ความถี่ 11.059 MHz เราจะใช้สูตรดังข้างล่าง

$$\text{จำนวน Machine Cycle} =[3+R1\ \{3+R0\ (2)\}]\ x1.085\ \text{us} \tag{2.20}$$

ซึ่งในโครงงานนี้ ไมโครคอนโทรลเลอร์จะใช้สัญญาณ Clock ที่ 11.059 MHz ดังนั้นเราจึงใช้สมการที่ 2.8 เป็นสมการคำนวณในโปรแกรมหน่วงเวลา ที่จะใช้สร้างพัลส์ไปขับให้กับมอสเฟตต่อไป

2.4 ภาคอัดประจุแบตเตอรี่ (Battery Charger)

ภาคอัดประจุแบตเตอรี่ เป็นอุปกรณ์ทำหน้าที่อัดประจุให้กับแบตเตอรี่ โดยอุปกรณ์นี้จะเปลี่ยนไฟฟ้ากระแสสลับเป็นไฟฟ้ากระแสตรง สำหรับยูพีเอสระบบ Line-interactive UPS (ยูพีเอสที่มี Stabilizer อยู่ในตัว)

2.4.1 ความรู้พื้นฐานเกี่ยวกับแบตเตอรี่

แบตเตอรี่ที่ใช้ในโครงงานนี้ เป็นแบตเตอรี่แบบตะกั่ว-กรด ซึ่งจะขอกล่าวชนิดและคุณสมบัติกว้างๆ โดยแบ่งได้เป็น 2 ชนิด ดังนี้

1) Non spill lead-acid Battery แบ่งเป็น 2 ชนิดได้แก่

1.1) standard battery แบตเตอรี่ชนิดนี้เมื่อไม่ได้ใช้งานจะมี self-discharge ที่ขึ้นอยู่กับอุณหภูมิ สิ่งแวดล้อม จึงต้องมีการอัดประจุใหม่ทุกๆ4-6สัปดาห์เพื่อชดเชยค่า self-discharge ซึ่งแบตเตอรี่นี้เหมาะสำหรับการใช้งานในช่วงสั้นๆ

1.2)permanent battery แบตเตอรี่ชนิดนี้สามารถเก็บในที่อุณหภูมิ 20 องศาเซลเซียสได้เป็นเวลานานหลายๆเดือน โดยไม่ต้องมีการดูแลรักษาและสามารถอัดประจุให้กับแบตเตอรี่ทุกๆ10 เดือนและมี self-discharge ต่ำ เนื่องจากใช้ lead alloy ชนิดพิเศษ เหมาะสำหรับการใช้ในงานพิเศษที่ต้องการ อายุการใช้งานสูงกว่าแบบ standard battery

2) Recombining sealed lead-acid battery

ปกติแล้วเมื่อแบตเตอรี่อัดประจุจนถึงระดับ Overcharge กรดซัลฟูลิกที่เป็นอิเล็คโตรไลท์จะเกิดปฏิกิริยาเป็นก๊าซ H_2 และ O_2 ทำให้ความเข้มข้นของอิเล็คโตรไลท์เพิ่มขึ้น จึงเป็นเหตุให้ต้องมีการเติมน้ำกลั่นมิฉะนั้นแบตเตอรี่จะเสื่อมคุณภาพได้ การเติมน้ำกลั่นเป็นการจำกัดวงในการใช้งานแบตเตอรี่ ฉะนั้นจึงมีการพัฒนาแบตเตอรี่ใช้งานโดยไม่ต้องเติมน้ำกลั่นซึ่งแบ่งได้ 2 ชนิด คือ

2.1) แบตเตอรี่แบบ Non-recombining หรือ Partially Recombining แบตเตอรี่ชนิดนี้ไม่ต้องบำรุงรักษาใดๆและไม่ต้องอัดประจุให้ถึงระดับ overcharge เพื่อให้แบตเตอรี่เต็ม ในกรณีที่อุณหภูมิใช้งานสูงกว่าอุณหภูมิแวดล้อมมากๆ แบตเตอรี่จะเกิดก๊าซทำให้ความดันสูงขึ้น ซึ่งแบตเตอรี่จะมีวาล์วที่คอยเปิดให้ก๊าซออกไปในทันทีและวาล์วจะปิดอัตโนมัติเมื่อความดันก๊าซต่ำลง ด้วยเหตุนี้จึงใช้ได้กับทุกสภาวะแวดล้อม แบตเตอรี่แบบนี้จะปิดผนึกไว้ ซึ่งภายในแบตเตอรี่จะไม่เกิดก๊าซอีกทั้งมีความต้านทานภายในต่ำและจ่ายกระแสได้สูง ภายใต้การอัดประจุแบบรักษาระดับแรงดันคงที่จะอัดประจุด้วยกระแสอัดประจุสูงๆในตอนเริ่มต้น และในขั้นที่ 2 เมื่อแรงดันย้อนกลับของแบตเตอรี่เริ่มสูงขึ้นและต้องควบคุมกระแสในการอัดประจุ เมื่อแบตเตอรี่เต็มจะต้องรักษาระดับแรงดันแบตเตอรี่โดยลดกระแสให้เหลือน้อยๆเพื่อไม่ให้เกิด overcharge และป้องกัน electrolyte loss partially recombining แบตเตอรี่แบบตะกั่ว-กรดที่มีขนาดความจุเดียวกันสามารถต่อกันแบบอนุกรมหรือ ขนานก็ได้ ถ้าประจุแบตเตอรี่แบบอนุกรมจะใช้แรงดันสูงแลกกระแสน้อยๆส่วน ถ้าอัดประจุแบตเตอรี่แบบขนานใช้แรงดันต่ำและกระแสสูงๆแบตเตอรี่แบบนี้สามารถอัดประจุและจ่ายประจุ ได้ที่อุณหภูมิตั้งแต่ -20 ถึง 50 องศาเซลเซียส ซึ่งดีกว่าแบบนิเกิล-แคดเมียม

2.2) แบตเตอรี่แบบ Fully Recombining โครงสร้างคล้ายๆกับแบบ Standard Cylindrical Nicad เหมาะสำหรับการใช้งาน standby มีการรวมตัวของก๊าซ O_2 ที่ดี ก๊าซเกิดขึ้นคือ H_2 และ O_2 จะรวมตัวกันเป็นน้ำ ทำให้ปริมาตรของอิเล็คโตรไลท์ไม่ลดลง อายุการใช้งาน ขึ้นอยู่กับลักษณะการใช้งานและสภาพแวดล้อม มีความต้านทานภายในต่ำ อัตราการ discharge สูงการอัดประจุส่วนใหญ่เป็นแบบ constant voltage, constant current, pulse charging และอื่นๆ

ขบวนการปฏิกิริยาทางไฟฟ้าเคมี

$$PbO_2 + 2H_2SO_4 + Pb \underset{CHARGE}{\overset{DISCHARGE}{\rightleftharpoons}} PbSO_4 + 2H_2O + PbSO_4 \qquad (2.21)$$

2.4.2 พารามิเตอร์ในการอัดประจุ มีดังนี้

1) ปรับค่าแรงดันที่อัดประจุซึ่งต้องมีการชดเชยทางอุณหภูมิ โดยการตรวจจับอุณหภูมิของสภาวะแวดล้อม

2) เวลาที่อัดประจุเสร็จ เวลาที่แบตเตอรี่ถูกอัดประจุเต็มขึ้นอยู่กับสภาวะการดิสชาร์จและลักษณะการอัดประจุ รวมทั้งอุณหภูมิด้วย

3) อุณหภูมิในขณะการอัดประจุ ควรทำการอัดประจุที่อุณหภูมิในช่วง 0 ถึง 40 องศาเซลเซียส ซึ่งอุณหภูมิในการอัดประจุมีประสิทธิภาพดีคือในช่วง 5 ถึง 30 องศาเซลเซียส โดยที่อุณหภูมิต่ำกว่าหรือสูงกว่าที่กำหนด อาจทำให้การอัดประจุได้ไม่มากหรือร้อนเกินไป

4) การชาร์จประจุเกิน การอัดประจุเพิ่มขึ้นอีกหลังจากแบตเตอรี่อัดประจุเต็มแล้ว เรียกว่าโอเวอร์ชาร์จ (Overcharge) ซึ่งเกิดขึ้นอย่างต่อเนื่องมีผลทำให้อายุการใช้งานของแบตเตอรี่สั้นลง

2.4.3 วิธีการอัดประจุแบตเตอรี่ แบ่งได้ 4 ชนิดคือ

1) การอัดประจุแบบกระแสคงที่ (Constant Current Charging)

วิธีนี้มีข้อดีตรงที่ว่าไม่จำเป็นต้องมีการชดเชยทางอุณหภูมิเหมือนกับการชาร์จแบบแรงดันคงที่ แต่ก็มีข้อเสียคือต้องระวังเรื่องเวลาในการอัดประจุ โดยเฉพาะอย่างยิ่งในการอัดประจุด้วยกระแสสูงๆซึ่งแบตเตอรี่จะเต็มภายในเวลาอันสั้นๆเท่านั้น การอัดประจุด้วยอัตราสูงมีผลทำให้แรงดันที่แบตเตอรี่สูงเกินไป น้ำจะเกิดการแตกตัวและมีความร้อนเกิดขึ้น การอัดประจุแบบนี้อาจใช้ในการอัดประจุใหม่ให้กับแบตเตอรี่ที่เก็บไว้นานแล้ว ซึ่งสามารถอัดประจุแบตเตอรี่ได้ทีเดียวหลายๆลูก

2) การอัดประจุแบบแรงดันคงที่ (Constant Voltage Charging)

โดยมีความต้องการให้แบตเตอรี่อยู่ในสภาวะประจุเต็ม (Fully Charge) โดยต้องระวังเรื่องอัตราการอัดประจุสูงเกินไป ซึ่งการอัดประจุแบบนี้ทำให้แรงดันเอาต์พุตมีเสถียรภาพพร้อมทั้งมีความจุกระแสสูง ซึ่งในช่วงที่แรงดันแบตเตอรี่มีค่าต่ำๆความต้านทานของแบตเตอรี่มีค่าน้อยมาก ฉะนั้นกระแสที่อัดประจุจึงมีค่าสูงมาก เป็นผลให้เครื่องอัดประจุต้องมีขนาดที่ใหญ่และราคาแพง อีกทั้งยังทำให้เกิดความร้อนภายในแบตเตอรี่ ซึ่งปกติเครื่องอัดประจุแบบนี้จะมีอุปกรณ์ช่วยในการจำกัดกระแสเริ่มต้น โดยใช้ constant current regulation ซึ่งในช่วงสุดท้ายของการอัดประจุกระแสจะถูกลดลงอย่างอัตโนมัติ ซึ่งไม่ทำให้เกิดการสลายตัว

3) การอัดประจุแบบลดค่ากระแส (Tapered Current Charging)

เป็นวิธีการอัดประจุแบตเตอรี่ที่ง่ายไม่ซับซ้อน ราคาถูก วงจรการอัดประจุประกอบด้วยหม้อแปลงไฟฟ้ากำลัง วงจรเร็คติไฟล์ และค่าความต้านทานที่เหมาะสมสำหรับจำกัดกระแส แต่วิธีนี้ไม่เหมาะสำหรับในการอัดประจุแบตเตอรี่แบบตะกั่ว –กรด ปิดผนึก เนื่องจากกระแสที่ใช้อัดประจุจะกระเพื่อมตามแรงดันไฟฟ้าของระบบ ซึ่งทำให้แรงดันแบตเตอรี่เปลี่ยนแปลงไปด้วย แต่ก็สามารถลดผลอันนี้ได้โดยการใช้หม้อแปลงไฟฟ้ากำลังที่มีแรงดันด้านทุติยภูมิสูงกว่าแรงดันแบตเตอรี่ และเลือกค่าความต้านทานที่เหมาะสมในการกำจัดกระแส โดยที่การอัดประจุแบบนี้สามารถใช้แทนแบบการอัดแบบกระแสคงที่ได้ ไม่เพียงเฉพาะการอัดประจุแบตเตอรี่หลายๆลูกในครั้งเดียวเท่านั้น ยังสามารถใช้ในระบบ trickle charge ได้อีกด้วย

4) การอัดประจุแบบรวม (Combination Charging: two-step)

วิธีนี้บางทีเรียกว่า " Two-rate" หรือ " Two-step" โดยเกิดจากการรวมกัน 2 แบบคือ การอัดประจุแบบกระแสคงที่และแรงดันคงที่เข้าด้วยกัน โดยในขั้นแรกจะมีวิธีการอัดประจุเร็ว และขั้นที่สองจะอัดประจุด้วยกระแสค่าต่ำๆการสวิตช์จากขั้นแรกไปยังขั้นที่สองอาจทำได้หลายวิธีเช่น ตรวจสอบแรงดันที่แบตเตอรี่หรือควบคุมเวลาในการอัดประจุ หรือตรวจสอบกระแสในการอัดประจุเป็นต้น

2.4.4 การอัดประจุแบตเตอรี่ชนิด ตะกั่ว – กรด โดยทั่วไป

ในการอัดประจุแบตเตอรี่ชนิดนี้ จะมีการจำกัดกระแสที่ใช้อัดประจุและแรงดันต้องพยายามให้คงที่มากที่สุด ปริมาณกระแสที่ใช้อัดประจุโดยปกติจะจำกัดอยู่ที่ 0.2 C สำหรับ Lead – antimony Battery และ0.5 C สำหรับ Lead –calcium Battery เมื่อ C คือความจุแบตเตอรี่ (หน่วย Ampere –hour (Ah)) ถ้าหากการอัดประจุโดยใช้กระแสมากเกินไป จะทำให้อุณหภูมิในแบตเตอรี่สูงและทำให้อายุการใช้งานของแบตเตอรี่ลดลง และถ้าหากอัดประจุที่กระแสน้อยเกินไปจะทำให้มี

Lead –sulfate เกาะหลงเหลืออยู่ในแผ่นเพลท (plate) ของแบตเตอรี่ ซึ่งจะทำให้การจ่าย ไม่ได้เท่ากับพิกัดของแบตเตอรี่ และ ถ้าต้องการให้แบตเตอรี่มีอายุการใช้งานยาวนาน แรงดันพีคทูพีค (peak to peak ripple) ซึ่งเป็นแรงดันเอาต์พุตของอุปกรณ์อัดประจุแบตเตอรี่ ควรจะมีค่าไม่เกิน 0.5% ของแรงดันดีซี พึงระลึกเสมอว่าการอัดประจุแบตเตอรี่ที่ค่าพิกัดของแบตเตอรี่ จะสามารถยืดอายุการใช้งานของแบตเตอรี่ได้ยาวนานขึ้น

2.4.5 การประยุกต์ใช้งาน

ในการประยุกต์ใช้งานของ Sealed lead –acid Battery แบ่งออกเป็น 2 แบบคือ

1) Cyclic Operation

การทำงานเป็นรอบๆต้องการเวลาในการอัดประจุสั้น และต้องป้องกันการอัดประจุ และดิสชาร์จเกิน สิ่งสำคัญในการอัดประจุด้วยเทคนิคการอัดประจุด้วยแรงดันคงที่คือ การรักษาแรงดันเอาต์พุตไว้ที่ระดับสุดท้ายของการอัดประจุและจำกัดกระแสให้ต่ำกว่าค่าที่ออกแบบไว้สูงสุด ถ้าแบตเตอรี่ถูกอัดประจุในช่วงอุณหภูมิสภาวะแวดล้อมกว้างๆจะต้องมีการชดเชยอุณหภูมิด้วย มิฉะนั้นการอัดประจุอาจสูงไปหรือต่ำไปในที่อุณหภูมิสูงและอุณหภูมิต่ำ

2) Standby Backup Charging

ใช้งานในประเภทสำรองเอาไว้ใช้ในยามต้องการ ปกติจะเก็บในสภาวะแบตเตอรี่เต็มตลอดเวลา และจะจ่ายพลังงานไปให้ภาระเมื่อระบบผิดปกติ ซึ่งการทำงานแบบนี้จะแบ่งเป็น 2 โหมด คือ

2.1) Trickle Charge

โดยปกติระบบของการไฟฟ้าฯทำการจ่ายพลังงานให้ภาระหรืออุปกรณ์ในขณะที่แบตเตอรี่ไม่ได้ถูกต่อกับภาระ ถ้าหากเกิดความผิดปกติขึ้นที่ระบบนั้น วงจรแบตเตอรี่จะถูกเชื่อมเข้ากับภาระเพื่อจ่ายพลังงาน โดยจะต้องพิจารณาการชดเชยสำหรับ Self Discharge โดยการอัดประจุแบตเตอรี่ที่กระแสคงที่ค่าต่ำๆเพื่อให้แบตเตอรี่อยู่ในสภาวะประจุเต็มตลอดเวลา ในกรณี Deep Discharge จะต้องอัดประจุแบตเตอรี่เป็นเวลานาน ฉะนั้นวิธีอัดประจุแบบ two-rate และแบบแรงดันคงที่ จะเหมาะสมมากในการอัดประจุ เนื่องจากมีโหมดการอัดประจุเริ่มต้นเร็ว แต่แบบ two-rate charger จะเหมาะสมกว่าเพราะไม่จำเป็นต้องชดเชยทางอุณหภูมิ

2.2) Float Charge

โดยในแบบนี้ ทั้งภาระและแบตเตอรี่ต่อกันแบบขนานกับวงจรเร็คติไฟล์ ระบบนี้จะใช้ได้เฉพาะการอัดประจุแบบแรงดันคงที่เท่านั้น ซึ่งแรงดันที่อัดประจุต้องมีเสถียรภาพ การกระเพื่อมอย่างมากของแรงดันที่อัดประจุ มีผลให้แบตเตอรี่ดิสชาร์จเพียงเล็กน้อย ซึ่งการอัดประจุแบบแรงดัน

คงที่ จะต้องออกแบบให้จ่ายได้ที่ภาระมีค่าสูงสุด มิฉะนั้นจะทำให้อายุการใช้งานของแบตเตอรี่สั้นลง โดยตามปกติอายุการใช้งานของแบตเตอรี่แบบ Float จะสั้นกว่าการอัดประจุแบบ Trickle

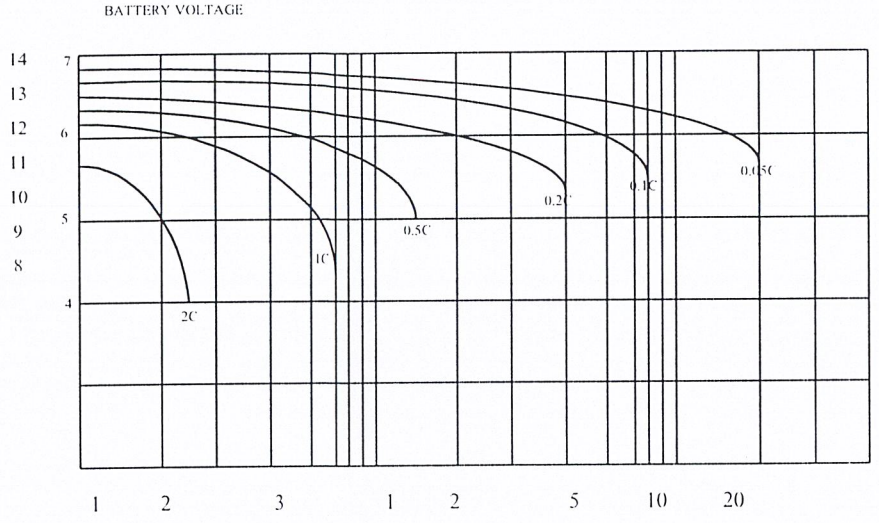
2.4.6 คุณลักษณะที่สำคัญของแบตเตอรี่ชนิดตะกั่ว –กรด

โดยปกติแบตเตอรี่ชนิด Lead-Acid ที่ออกแบบให้ใช้ได้กับ UPS ต้องมีความจุสูง อายุการใช้งานที่ยาวนานถึง 80% ของช่วงเปิดวงจร แรงดันที่อัดประจุแบตเตอรี่จะเท่ากับ 2.1 เซลและจะลดลงขณะแบตเตอรี่คายประจุหรืออุณหภูมิที่เพิ่มขึ้น โดยที่แรงดันของแบตเตอรี่และแรงดันที่ป้อนให้กับอินเวอร์เตอร์ เทียบกับย่านการทำงานที่เปลี่ยนแปลงไป โดยจะแสดงไว้ดังในตารางที่ 2.2 ถ้าหากว่ากำลังเอาต์พุตเพิ่มขึ้น จำนวนเซลที่ต่ออนุกรมกันจะต้องเพิ่มขึ้นเพื่อป้องกันกระแสเกิน สำหรับรูปกราฟแสดงการคายประจุ (Discharge) ของแบตเตอรี่ชนิดตะกั่ว-กรด ซึ่งเป็นการคายประจุที่ค่าความจุต่างๆของแบตเตอรี่ แสดงดังรูปที่ 2.18

ตารางที่2.2 ค่าต่างๆของแบตเตอรี่ชนิดตะกั่ว –กรด

	12 V	24 V	48 V	72 V	125 V	250 V
Inverter input Voltage range	11 - 15	22 30	42 - 56	63 – 85	105 - 140	210-280
Number of Lead Acid cells	6	12	24	36	60	120
Typical Float voltage / Cell	2.25 – 2.3	2.25 – 2.3	2.2 – 2.25	2.2 – 2.25	2.2 – 2.25	2.2 – 2.25
Typical battery Float Voltage / Cell	2.4	2.4	2.33	2.33	2.33	2.33
Typical equalize Voltage / Cell	14.4	28.8	53 - 54	79 - 81	132 – 135	264 – 270
End-of-discharge Cell Voltage	18.3	1.83	1.75	1.75	1.75	1.75

ส่วนกราฟการคายประจุแสดงได้ดังรูป



รูปที่ 2.18 การคายประจุของแบตเตอรี่ที่ค่าความจุต่างๆ

จากกราฟอธิบายได้ว่า ในระดับแรงดันแบตเตอรี่ที่ค่าต่างๆ การคายประจุจะใช้เวลาไม่เท่ากัน ซึ่งขึ้นอยู่กับความจุที่ใช้งานในขณะนั้น เช่น เมื่อพิจารณาที่ค่า 0.05 C จะเห็นได้ว่าจะสามารถจ่ายกระแสดิสชาร์จได้นานถึง 20 ชั่วโมง แต่ถ้าค่าความจุใน ขณะนั้นมีค่าสูงเวลาที่คายประจุจะต่ำคือ
มีค่าไม่เกิน 0.3 ชั่วโมง

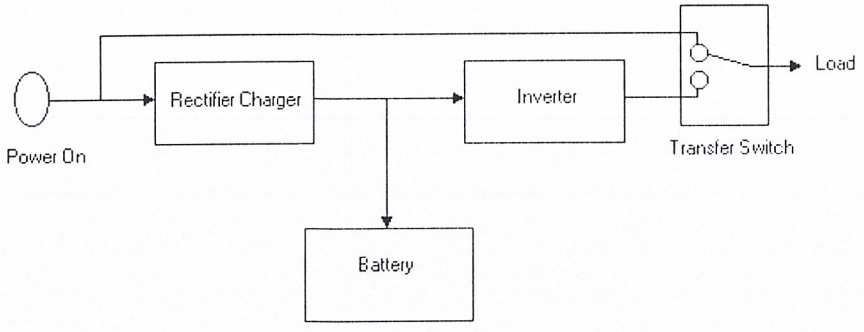
2.5 สวิตช์สับเปลี่ยน (Transfer Switch)

สวิตช์สับเปลี่ยน (Transfer Switch) นับได้ว่าเป็นอุปกรณ์ที่มีความสำคัญอย่างมากทีเดียวในแหล่งจ่ายไฟฟ้าต่อเนื่อง โดยจะทำงานใน 2 สภาวะ คือ

1) ในสภาวะปกติ สวิตช์สับเปลี่ยนจะสับไปที่ระบบ เพื่อรับกำลังไฟฟ้าจากระบบ โดยรีเลย์จะถูกควบคุมการทำงานจากไมโครคอนโทรลเลอร์ ซึ่งในสภาวะนี้ไมโครคอนโทรลเลอร์ตรวจสอบแล้วว่าระบบมีสภาวะปกติ มีสัญญาณแรงดันไฟฟ้าพร้อมที่จะจ่ายให้กับภาระ ดังนั้นรีเลย์จึงถูกสับเข้ากับระบบเพื่อเป็นทางผ่านของพลังงานไฟฟ้าของระบบไปสู่ภาระ

2)ในสภาวะผิดปกติ สวิตช์สับเปลี่ยนจะสับไปที่อินเวอร์เตอร์ เพื่อรับกำลังไฟฟ้าจากอินเวอร์เตอร์ซึ่งในสภาวะนี้ไมโครคอนโทรลเลอร์ตรวจสอบแล้วพบว่าระบบมีสภาวะผิดปกติเนื่องมาจากสาเหตุ ระบบไม่จ่ายกำลังไฟฟ้า (Brownout) แรงดันระบบมีค่าต่ำเกินไป (Under Voltage) ซึ่งในสภาวะนี้จะไม่มีสัญญาณแรงดันไฟฟ้าพร้อมที่จะจ่ายให้กับภาระ ดังนั้นรีเลย์จึงถูกสับเข้ากับ

อินเวอร์เตอร์เพื่อจ่ายพลังงานไฟฟ้าไปสู่ภาระแทนระบบ ซึ่งจะแสดงผังการทำงานโดยรวมดังรูปที่ 2.19



รูปที่ 2.19 ผังการทำงานของแหล่งจ่ายไฟฟ้าสำรองซึ่งแสดงให้เห็นส่วนสวิตช์สับเปลี่ยน

2.5.1 เงื่อนไขการทำงานของสวิตช์สับเปลี่ยนมีดังนี้

1) เมื่อมีการเปลี่ยนโหมดการทำงานของสวิตช์สับเปลี่ยนที่แรงดันศูนย์ เนื่องจากแรงดันไฟฟ้ากระแสสลับที่ได้จากอินเวอร์เตอร์มีค่าคงที่ แต่แรงดันของระบบมีการเปลี่ยนแปลงอยู่ในช่วง 198-242 โวลต์ ซึ่งหากไม่มีการสับเปลี่ยนที่แรงดันศูนย์แล้ว อาจทำให้อินเวอร์เตอร์ได้รับความเสียหายได้ กล่าวคือถ้าแรงดันของระบบมีค่าไม่เท่ากบแรงดันของอินเวอร์เตอร์ในช่วงที่มีการสับเปลี่ยน จะทำให้มีกระแสทรานเซียนส์เกิดขึ้น ดังนั้นเพื่อป้องกันในส่วนนี้จึงต้องมีการสับเปลี่ยนที่แรงดันศูนย์ ซึ่งจะดูได้ในรูปที่ 2.20

2) แรงดันไฟฟ้าผิดปกติ เนื่องจากไฟฟ้าดับ (Brownout) เป็นลักษณะเมื่อในสภาวะปกติจะมีแรงดันไฟฟ้าจากระบบให้กับภาระ แต่ช่างระยะเวลาหนึ่งหลังจากนั้นได้มีกาขาดหายไปของแรงดันระบบ ทำให้ภาระไม่ได้รับแรงดันไฟฟ้าจากระบบ ซึ่งแสดงได้ดังรูปที่ 2.21

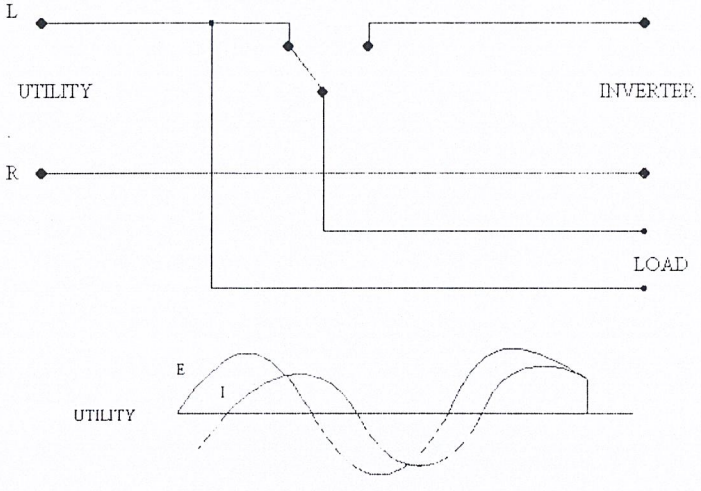
3) แรงดันไฟตกและแรงดันไฟเกินของระบบ (Under Voltage and Over Voltage) คือสภาพที่แรงดันไฟฟ้าสลับมีค่าต่ำกว่าและสูงกว่า 10% ของแรงดันไฟฟ้าปกติ หรือมีค่า 198 โวลต์ และ 242 โวลต์ ตามลำดับ จากค่าแรงดันปกติ 220 โวลต์ ซึ่งค่าแรงดันทั้งสองนี้ จะเป็นเหตุให้เกิดความเสียหายเกิดขึ้นกับภาระได้

4) กระแสไฟฟ้ามีค่าเกินพิกัด (Over Current) คือเมื่อขณะที่มีการเพิ่มภาระ ยูพีเอสจะทำการจ่ายกระแสให้กับภาระมากขึ้นจนอาจเกินพิกัดของยูพีเอส โดยแบ่งลักษณะของการเกิดกระแสไฟฟ้าเกินได้ 3 ประเภท คือ

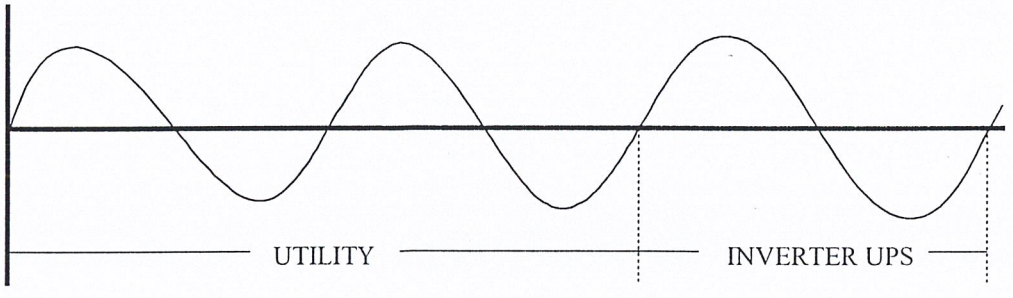
4.1 กระแสลัดวงจร จะเกิดขึ้นจากการลัดวงจรของเอาต์พุต ค่าของกระแสที่จัดเป็นกระแสลัดวงจรคือ มีค่ามากกว่า 200% ของค่ากระแสที่พิกัด

4.2 กระแสภาระเกิน เป็นค่าของกระแสที่มากกว่าพิกัดของยูพีเอส แต่ยังน้อยกว่ากระแสลัดวงจรซึ่งกระแสทั้งสองตัวนี้ได้มีการป้องกันไว้อีกส่วนคือ ฟิวส์ ที่ต่ออนุกรมไว้ด้านเอาต์พุต

4.3 กระแสอันรัช (Inrush Current) เป็นลักษณะของกระแสที่มีค่าสูงในเวลาสั้นๆและเกิดขึ้นไม่กี่ลูกคลื่น มักจะเกิดขึ้นในขณะเปิด –ปิดภาระ ซึ่งจะมีค่าประมาณ 150%ของกระแสพิกัด แม้จะเกิดในช่วงเวลาสั้นๆก็อาจทำอันตรายให้กับยูพีเอสได้ แต่ได้มีการป้องกันไว้อีกลักษณะคือ ใช้วาร์ริสเตอร์คร่อมแหล่งจ่ายทางด้านอินพุต ซึ่งจะคอยจำกัดกระแสอินรัชนี้อยู่แล้ว



รูปที่ 2.20 กระแสทรานเซียนส์ที่เกิดขึ้นเนื่องจากการเปลี่ยนโหมดการสับเปลี่ยน



รูปที่ 2.21 แรงดันไฟฟ้าของระบบเกิดการขาดหายไปเนื่องจากความผิดพลาดของระบบ

2.5.2 ลำดับการทำงานของสวิตช์สับเปลี่ยน

ที่กล่าวมาทั้งหมดอาจกล่าวได้ว่า สวิตช์สับเปลี่ยนมีหน้าที่หลักคือเลือกแหล่งจ่ายไฟฟ้ากระแสสลับที่ดีที่สุดให้กับภาระ ทั้งนี้อาจจะเป็นจากระบบหรืออินเวอร์เตอร์ เพื่อป้องกันความเสียหายที่จะเกิดกับภาระ ทั้งยังป้องกันอันตรายที่จะเกิดกับอินเวอร์เตอร์ด้วย ซึ่งในการป้องกันความเสียหายที่จะเกิดขึ้นนั้น ซึ่งในโครงงานนี้จะมีการพิจารณาการออกแบบสวิตช์สับเปลี่ยน โดยจะแสดงให้เห็นเป็นลำดับในลักษณะโฟลชาร์ต ดังนี้

1) แรงดันไฟฟ้าผิดปกติ เนื่องจากไฟฟ้าดับ (Brownout)

2) แรงดันไฟตกและแรงดันไฟเกินของระบบและภาระ(Under Voltage and Over Voltage)

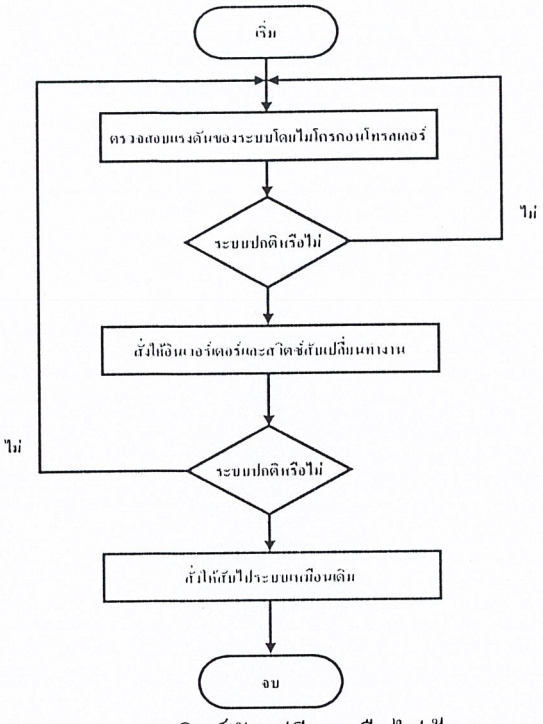
3) กระแสไฟฟ้ามีค่าเกินพิกัด (Over Current)

4) กระแสไฟฟ้าลัดวงจร (Short Circuit Current)

จากทั้ง 4 กรณีจะขอกล่าวขบวนการทำงาน ดังนี้

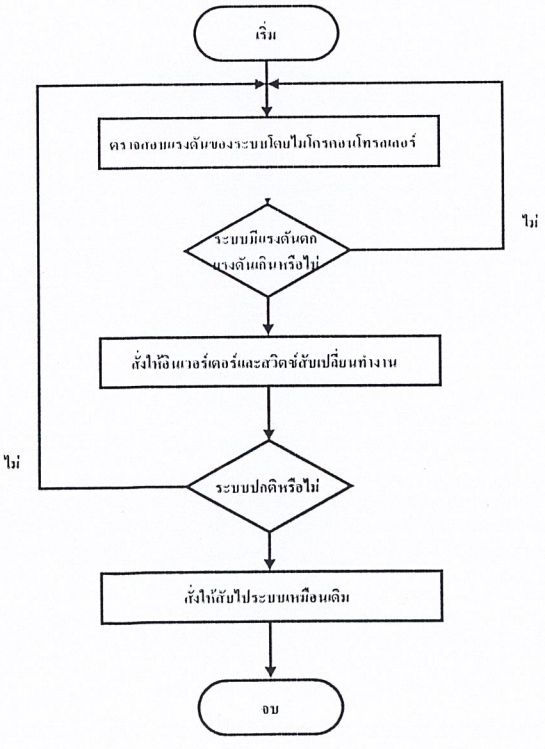
1) เมื่อแรงดันระบบเกิดผิดปกติขึ้น (Brownout)

โดยแรงดันได้มีการขาดหายไปทันทีทันใด ไมโครคอนโทรลเลอร์จะทำการตรวจสอบสภาวะนี้ และเมื่อตรวจสอบแล้วพบว่าแรงดันอินพุตเป็นศูนย์ ไมโครคอนโทรลเลอร์จะเข้าสู่การทำงานในโหมดสภาวะผิดปกติ โดยจะทำการสร้างเฟสสัญญาณพัลส์รูปคลื่นสี่เหลี่ยม ที่มีความถี่ 50 Hz จำนวน 2 สัญญาณ ซึ่งมีความต่างเฟสกัน 180 องศาไฟฟ้า ไปขับให้กับวงจรอินเวอร์เตอร์ ในขณะเดียวกันก็ส่งสัญญาณทริกไปที่สวิตช์สับเปลี่ยนเพื่อให้เปลี่ยนสถานะการทำงานมาทำในสภาวะผิดปกติดังนั้นในขณะนี้ภาระจะได้รับกำลังไฟฟ้าจากอินเวอร์เตอร์แทน ส่วนไมโครคอนโทรลเลอร์ก็จะทำการ วนรอบตรวจแรงดันของระบบตลอดเวลาถ้าหากว่าระบบมีแรงดันไฟฟ้าอยู่ในสภาวะปกติแล้วไมโครคอนโทรลเลอร์ทำการตรวจสอบถ้าหากพบว่าอยู่ในสภาพปกติแล้ว ก็จะเข้าสู่การทำงานในโหมดสภาวะปกติคือ ไม่ส่งสัญญาณรูปคลื่นสี่เหลี่ยมให้กับวงจรอินเวอร์เตอร์ และส่งสัญญาณอีกสถานะให้กับสวิตช์สับเปลี่ยนเพื่อให้ทำงานในโหมดสภาวะปกติ แต่ถ้าหากว่าแรงดันของระบบยังผิดปกติอยู่ ไมโครคอนโทรลเลอร์ก็จะยังทำงานในโหมดผิดปกติต่อไปจนกว่าถึงเวลาที่กำหนดไว้ ไมโครคอนโทรลเลอร์ก็จะหยุดทำงานโดยตัวมันเอง ซึ่งแสดงได้ดังรูปที่ 2.22



รูปที่ 2.22 ลำดับการทำงานของสวิตช์สับเปลี่ยน เมื่อไฟฟ้าของระบบดับ

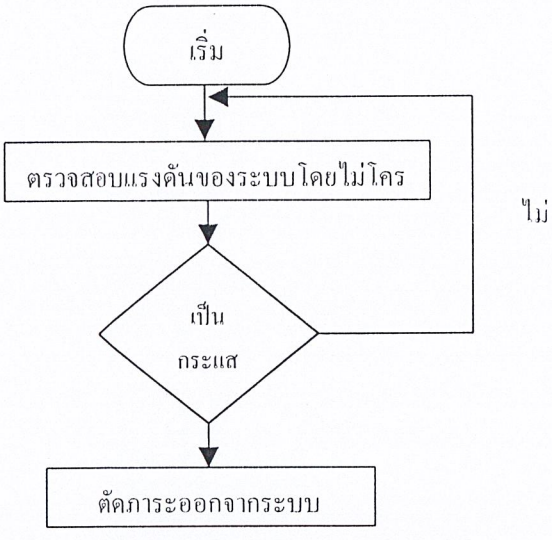
2) เมื่อระบบเกิดแรงดันไฟตก –เกิน (Over and Under Voltage)



รูปที่ 2.23 ลำดับการทำงานของสวิตช์สับเปลี่ยน เมื่อไฟฟ้าของระบบต่ำหรือเกินกว่าปกติ

เมื่อระบบเกิดแรงดันไฟตก-เกิน ทางด้านภาระจะมีแรงดันต่ำหรือสูงกว่าปกติ ในวงจรตรวจสอบโดยใช้หม้อแปลงเป็นตัวตรวจสอบสถานะและนำสัญญาณเข้าสู่วงจรตรวจสอบ ดังนั้นเมื่อไมโครคอนโทรลเลอร์ตรวจสอบแล้วพบว่ามีความผิดปกติเกิดขึ้น แรงดันที่ภาระมีสภาพไม่เหมาะสม ซึ่งจะเป็นอันตรายต่อภาระอย่างมากทั้งสองสภาวะ ดังนั้นไมโครคอนโทรลเลอร์จะเข้าสู่โหมดการทำงานในสภาวะผิดปกติเช่นเดียวกับในกรณีแรกที่กล่าวมาแล้ว และจะทำการตรวจสอบแรงดันเอาต์พุตตลอดเวลา ถ้าหากแรงดันของระบบเสถียรภาพ มีค่าอยู่ในระดับที่ไมโครคอนโทรลเลอร์ยอมรับได้ ก็จะกลับเข้าสู่โหมดการทำงานในสภาวะปกติเช่นเดิมต่อไป

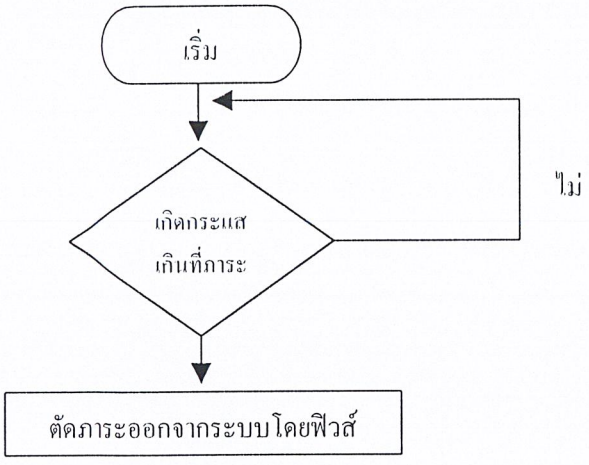
3) กระแสไฟฟ้ามีค่าเกินพิกัด (Over Current)



รูปที่ 2.24 ลำดับการทำงานของสวิตช์สับเปลี่ยน เมื่อกระแสไฟฟ้าของภาระเกินกว่าพิกัด

เมื่อเกิดกระแสไฟฟ้าเกินพิกัด แสดงว่าทางด้านภาระมีการใช้งานเกินพิกัด ซึ่งจะเป็นอันตรายอย่างยิ่งหากให้เหตุการณ์เช่นนี้เกิดขึ้นต่อเนื่องไป ดังนั้นจะมีการป้องกันโดยไมโครคอนโทรลเลอร์ได้รับสัญญาณตรวจสอบมาจากหม้อแปลงกระแส (Current Transformer) ที่ต่ออนุกรมอยู่กับภาระซึ่งหากเกิดกระแสภาระเกินพิกัดจริง ไมโครคอนโทรลเลอร์จะทำการตัดภาระออกจากระบบทันที โดยไม่มีการหน่วงเวลาเพื่อป้องกันอันตรายที่จะเกิดขึ้นตามมา ซึ่งจะมีความไวในการตรวจสอบและตัดภาระที่เร็วมาก ดังนั้นภาระที่ใช้งานจึงต้องมั่นใจว่าไม่ควรมีค่าเกินพิกัด เมื่อภาระถูกตัดออกจากระบบแล้ว จำเป็นต้องตรวจสอบภาระให้มีค่าเหมาะสมก่อนที่จะทำการใช้งานครั้งใหม่ต่อไป

4) กระแสไฟฟ้าลัดวงจร (Short Circuit Current)



รูปที่ 2.25 ลำดับการทำงานเมื่อกระแสไฟฟ้าลัดวงจรที่ภาระ

เมื่อเกิดกระแสไฟฟ้าลัดวงจร แสดงว่ามีการลัดวงจรทางด้านภาระ ซึ่งในสภาวะนี้จะเป็นอันตรายต่อยูพีเอสมากที่สุด ดังนั้นเพื่อความแน่นอนและปลอดภัยต่อระบบจึงใช้อุปกรณ์ป้องกันจากการลัดวงจรคือ ฟิวส์จำกัดกระแส โดยต่ออนุกรมกับภาระ ซึ่งถ้าหากมีการลัดวงจรด้านภาระ ฟิวส์ก็จะตัดภาระออกก่อนทันที เป็นการป้องกันก่อนที่จะเกิดความเสียหายต่อยูพีเอส

2.6 ไมโครคอนโทรลเลอร์ (Microcontroller)

ไมโครคอนโทรลเลอร์ตระกูล MCS-51 มีอยู่ด้วยกันหลายเบอร์แต่โครงสร้างพื้นฐานจะมีลักษณะคล้ายกัน ซึ่งในโครงงานนี้จะใช้เบอร์ AT 89C52

2.6.1 คุณสมบัติของไมโครคอนโทรลเลอร์ตระกูล MCS-51

1) ต้องการแหล่งจ่ายไฟเพียง 5 โวลต์ เพียงชุดเดียว

2) มีหน่วยความจำสำหรับเก็บโปรแกรมควบคุมการทำงานอยู่ภายในชิพจำนวน8กิโลไบท์ สำหรับเบอร์ 8051 และเบอร์ AT 89C52 โดยเฉพาะเบอร์ AT 89C52 นี้เป็นเบอร์ที่พัฒนาขึ้นมาใหม่ของบริษัท INTEL ซึ่งมีข้อเด่นกว่าเบอร์อื่นๆคือหน่วยความจำโปรแกรมสามารถที่จะลบหรือเขียนโปรแกรมลงไปใหม่ได้ถึง 1000 ครั้ง ข้อมูลหรือโปรแกรมสามารถที่อยู่ได้ ยาวนานถึง 10 ปี ส่วนเบอร์ 8031,8032 ไม่มีหน่วยความจำส่วนนี้อยู่

3) หน่วยความจำสำหรับเก็บข้อมูล (Data memory) ขนาด128 กิโลไบท์สำหรับ 8031,8032 และ AT89C52 ส่วนเบอร์ 8052 มีถึง 256 กิโลไบท์

4) สามารถใช้หน่วยความจำสำหรับโปรแกรม และ ข้อมูลที่อยู่ภายนอกชิพได้อย่างละ 64

กิโลไบท์ แยกจากกัน

5) คำสั่งส่วนใหญ่ใช้เวลาทำงานเพียง 1 ไมโครวินาที เมื่อทำงานที่ความถี่ 12 MHz และ 11.059 ไมโครวินาที สำหรับ 11.059 MHz

6) มีพอร์ทสามารถรับและส่งข้อมูลได้ทั้ง 2 ทิศทาง จำนวน 4 พอร์ท ๆละ 8 บิท หรือสามารถใช้งานเป็นพอร์ทขนาด 1 บิทแยกจากกัน ทำให้เสมือนมีพอร์ทขนาด 1 บิท แยกจากกัน ทำให้เสมือนมีพอร์ทขนาด 1 บิตไว้ใช้งานรวมทั้งสิ้น 32 พอร์ท

7) รับอินเตอร์รัพได้ 6 แหล่ง 5 เวคเตอร์

8) มีพอร์ทที่สามารถที่จะรับส่งข้อมูลอนุกรม (UART) 2 พอร์ท ทั้งรับและส่งในเวลาเดียวกันได้ (Full Duplex) เลือกรูปแบบการส่งข้อมูลได้ 4 โหมด

9) มี Timer/counter ขนาด 16 บิต 2 ชุด สามารถทำงานได้ 4 โหมด

10) มีคำสั่งคูณและหารเลขขนาด 8 บิตในตัวเอง

11) หน่วยความจำสำหรับเก็บข้อมูลภายในบางส่วนจะสามารถเข้าถึงข้อมูลได้ทั้งระดับบิทและระดับไบท์เพื่อให้การออกแบบโปรแกรมและควบคุมระบบทำได้ง่ายขึ้น

2.6.2 การใช้งานพอร์ตอนุกรม

1) การสื่อสารข้อมูลแบบอนุกรม

การสื่อสารข้อมูลแบบอนุกรมเป็นการรับหรือส่งข้อมูลในลักษณะกลุ่มของบิตคราวละหนึ่งบิตเรียงลำดับเรื่อยไปจนสิ้นสุด การสื่อสารแบบนี้จะมีข้อแตกต่างจากการสื่อสารแบบขนานเป็นอย่างมากเนื่องจากข้อมูลมีการโอนย้ายมาพร้อมกันจึงมีความจำเป็นต้องใช้ จำนวนเส้นสัญญาณจะมากขึ้นตามจำนวนบิตของข้อมูลด้วยในขณะที่การสื่อสารแบบอนุกรมนั้นต้องการเส้นสัญญาณเพียงสองหรือสามเส้นเท่านั้น ดังนั้นการสื่อสารแบบขนานจึงไม่เหมาะสมในการสื่อสารกับอุปกรณ์ภายนอกเป็นระยะทางไกลๆเพราะจะทำให้สิ้นเปลืองค่าใช้จ่ายมากขึ้น

2) ความเร็วในการสื่อสารข้อมูลแบบอนุกรม

เนื่องจากการสื่อสารแบบอนุกรมเป็นการรับและส่งข้อมูลในลักษณะกลุ่มของบิตข้อมูล (Bit Stream) ดังนั้นจึงต้องให้ความสนใจในการพิจารณาถึงเรื่องอัตราความเร็วในการรับและส่งบิตเหล่านี้เป็นอันดับแรก โดยทั่วไปมักจะระบุกันในหน่วยของจำนวนบิตของข้อมูลภายในหนึ่งหน่วยวินาที เรียกว่าอัตราบอร์ด ตามค่ามาตรฐานเหล่านี้ ได้แก่ 4,800,9600,19200 บอร์ด ข้อมูลทั้ง 8 บิตนี้หากว่าถูกส่งออกมาด้วยอัตรา 9600 บอร์ดจะใช้เวลาในการส่งข้อมูลหนึ่งบิตมีค่าเท่ากับ 1/9600 หรือ 104ไมโครวินาที (us) และ เวลาในการส่งข้อมูลทั้ง 8 บิตจะมีค่าเท่ากับ (8x104) หรือ 832 ไมโครวินาที

3) รูปแบบของการส่งข้อมูลอนุกรม

การสื่อสารอนุกรมแบบอะซิงโครนัสจะใช้การแปลงข้อมูลแบบขนานให้เป็นอนุกรมแล้วเพิ่มเติมบิตบางอย่างรวมไปกับการส่งข้อมูลจริงซึ่งได้แก่

1) บิตเริ่มต้น (Start Bit)

บิตเริ่มต้นมีหน้าที่สำหรับการบ่งบอกให้ทราบถึงตำแหน่งจุดเริ่มต้นก่อนบิตข้อมูล ตามปกติแล้วค่าของบิตเริ่มต้นจะเป็นระดับของลอจิกต่ำ

2) บิตแสดงภาวะความเป็นเลขคู่หรือเลขคี่ (Parity Bit)

บิตนี้มีหน้าที่เพื่อการตรวจสอบความถูกต้องของข้อมูลโดยทั่วไปมักเรียกว่า บิตพาริตี้ และจะนำไปต่อท้ายบิตข้อมูล ค่าของบิตนี้ขึ้นอยู่กับจำนวนค่าของบิตที่เป็น 1 ซึ่งจะเป็นได้สองลักษณะคือ พาริตี้คู่ (Even Parity) หรือพาริตี้คี่ (Odd Parity) ตัวอย่างเช่นระบบที่ติดต่อกันโดยระบุว่าจะใช้พาริตี้คี่ทางด้านส่งจะนำข้อมูลที่จะส่งมาพิจารณาหาจำนวนของบิตที่มีค่าเป็น 1 เป็นเลขจำนวนคู่อยู่แล้ว ค่าของบิตพาริตี้จะมีค่าเป็น 0 แต่หากว่าจำนวนของบิตที่มีค่าเป็น 1 เป็นเลขจำนวนคี่ ค่าของบิตพาริตี้ก็จะมีค่าเป็น 1 การพิจารณาทางด้านรับเป็นการตรวจสอบจำนวนบิตที่มีค่าเป็น 1 ของข้อมูลที่ได้รับมาทั้งหมดรวมทั้งบิตพาริตี้ ถ้ามีค่าเป็นเลขจำนวนคู่แสดงว่าข้อมูลที่รับเข้ามานี้ถูกต้อง แต่หากไม่เป็นเลขจำนวนคู่แสดงว่าเกิดการผิดพลาดของข้อมูลขึ้นเป็นต้น

3) บิตสุดท้าย (Stop Bit)

บิตสุดท้ายเป็นบิตที่เพิ่มขึ้นเพื่อระบุถึงขอบเขตการสิ้นสุดของกลุ่มบิตข้อมูลบิตสุดท้ายนี้สามารถโปรแกรมบิตได้คือ 1 บิต,1 ½ บิต หรือ 2 บิต ดังนั้นกรณีของการส่งข้อมูล 8 บิต หากข้อมูลถูกส่งออกไปด้วยอัตราเร็ว 2400 บอร์ด เวลาโดยรวมในการส่งข้อมูลหนึ่งไบต์จะมีค่าเวลาเป็น (12x104)us หรือ 1.248ms

4) การส่งข้อมูลอนุกรมของ MCS-51

พอร์ตอนุกรมของ MCS-51 มีโครงสร้างการทำงานในแบบที่เรียกว่า ฟูลดูเพล็กซ์ (Full Duplex) จะเป็นการรับและการส่งข้อมูลแบบอนุกรมได้ในเวลาเดียวกัน โดยทางด้านวงจรของตัวส่ง (Transmitter) ประกอบด้วยข้อมูลออกไปยังพอร์ตอนุกรมทางขาสัญญาณ TXD (พอร์ต P3.1) ส่วนวงจรทางด้านตัวรับ (Receiver) ประกอบด้วย SBUF เช่นเดียวกันสัญญาณข้อมูลอนุกรมที่รับเข้ามาทางขาสัญญาณ RXD (P3.0)

พอร์ตอนุกรมของ MCS-51 สามารถทำงานโปรแกรมได้หลายโหมดด้วยกันโดยเลือกที่บิต SM0 และ SM1 ซึ่งอยู่ในรีจิสเตอร์ควบคุม SCON การทำงานทั้ง 4 โหมดของพอร์ตอนุกรมมีดังนี้

โหมด 0: ใช้รับส่งข้อมูล 8 บิต โดยการส่งจะเลื่อนออกทีละบิต โดยส่งบิต D0 ออกไปก่อนทางขา RXD และไม่มีการส่ง Start Bit แต่จะส่ง Shift clock ทางขา TXD ความเร็ว 1.5 เท่าของ CPU CLOCK

โหมด 1: ใช้สำหรับการเชื่อมต่ออนุกรมแบบ UART (Universal Asynchronous Receiver/Transmitter) โดยส่งแบบ 10 บิต ข้อมูล 8 บิต 1 Start Bit และ 1 Stop Bit และสามารถที่จะเปลี่ยนแปลงอัตราความเร็วในการส่งข้อมูลได้ โดยขึ้นอยู่กับบิต SMOD ใน PCON และ อัตราโอเวอร์โฟลว์ของ Timer1

โหมด 2: ใช้สำหรับการเชื่อมต่ออนุกรมแบบ UART โดยการใช้กลุ่มข้อมูลแบบ 11 บิตและกำหนดอัตราความเร็วในการส่งข้อมูลเท่ากับ 1/32 และ 1/64 ของ CPU CLOCK โดยโปรแกรมที่บิต SMOD ใน PCON

โหมด 3: ใช้สำหรับการเชื่อมต่ออนุกรมแบบ UART โดยการใช้กลุ่มข้อมูลแบบ 11 บิตและสามารถเปลี่ยนแปลงอัตราความเร็วในการส่งข้อมูลได้ โดยควบคุมที่บิต SMOD และ อัตราโอเวอร์โฟลว์ของ Timer1 นอกจากนี้โหมด 2 และ 3 ยังมีการดำเนินการอีกแบบหนึ่ง โดยสามารถนำมาใช้ประโยชน์ในการสื่อสารข้อมูล แบบที่มีไมโครโพรเซสเซอร์หลายตัวทำงานร่วมกันได้ซึ่งมีชื่อเรียกว่า Multi Processor Mode

Multiprocessor Mode: ในโหมดนี้เราจะใช้ไมโครโพรเซสเซอร์ 1 ตัว สำหรับเป็น Master และอีก 0-256 ตัวเป็น Slave รีจิสเตอร์ที่ใช้ควบคุมการรับส่งข้อมูลอนุกรมดังรายละเอียดดังรูปที่ 2.26

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

รูปที่ 2.26 รายละเอียดใน SCON

SM0, SM1 บิตเลือกโหมดการทำงาน

ตารางที่ 2.3 การเลือกโหมดการทำงาน

SM1	SM2	โหมด	การทำงาน
0	0	0	ทำงานเป็น Shift register อัตราเร็วในการรับหรือส่งข้อมูลเท่ากับ 1/12 ของความถี่ออสซิลเลเตอร์
0	1	1	8 bit UART อัตราเร็วในการรับหรือส่งข้อมูลกำหนดเองได้
1	0	2	9 bit UART อัตราเร็วในการรับหรือส่งข้อมูล = 1/32 หรือ 1/64 ของความถี่ออสซิลเลเตอร์ขึ้นกับบิต SMOD ใน PCON ดูตารางที่ 5.2
1	1	3	9 bit UART อัตราเร็วในการรับและส่งข้อมูลกำหนดเองได้

SM2 บิตเลือกการทำงานแบบ

1: เลือก Multiprocessor Mode ใช้ได้กับโหมด 2,3

0: เลือก Single Processor Mode ใช้ได้กับทุกโหมด

(เมื่อเลือกการทำงานรับข้อมูลแบบ Multi Processor Mode แล้วเมื่อข้อมูลบิตที่ 9 ที่รับได้มีค่าเป็น 1 RI จะเซ็ททันที)

REN บิตที่ควบคุมให้รับหรือไม่รับข้อมูล

1: ให้รับข้อมูลได้

0: ห้ามรับข้อมูล

TB 8 (Transmitter bit D8) ข้อมูลบิตที่ 9 ที่จะส่งออกไปในโหมด2,3 ให้ใส่ในบิต TB8

RB 8 (Receiver bit D8) ข้อมูลบิตที่ 9 ที่รับเข้ามาจะมาเก็บในบิตนี้ ข้อมูลบิตที่ 9 ก็คือค่าใน TB8 ทางด้านส่งนั่นเอง (ข้อมูลบิตที่ 9 คือค่าใน TB8 และ RB8 นั่นเอง)

TI บิต TI จะเป็น 1 เมื่อสิ้นสุดการส่งข้อมูล 1 ไบต์

RI บิต RI จะเป็น 1 เมื่อรับข้อมูลเสร็จ 1 ไบต์ (บิต RI, TI ผู้เขียนโปรแกรมจะต้องเคลียร์เอง)

5) การอินเตอร์รัพต์ของพอร์ตสื่อสารอนุกรม

เนื่องจากการส่งหรือการรับข้อมูลแบบอนุกรมจะส่งทีละไบต์ MCS-51 จึงได้กำหนดให้บิตหรือแฟล็กสถานะที่จัดรวมอยู่ในรีจิสเตอร์ SCON เช่น แฟล็ก TI ซึ่งจะมีค่าเป็น 1 เมื่อข้อมูลได้ทำการส่งออกไปภายนอกเสร็จสิ้นแล้ว และแฟล็ก RI จะมีค่าเป็น 1 เพื่อให้รู้ว่าได้รับข้อมูลผ่านเข้ามาทางพอร์ตอนุกรม เสร็จแล้วเมื่อแฟล็ก RI,TI นี้มีค่าเป็น 1 จะมีผลทำให้เกิดการอินเตอร์รัพต์ขึ้น

ดังนั้นภายในโปรแกรมรับหรือส่งข้อมูลจะต้องทำการตรวจสอบจากสภาวะของแฟล็กเหล่านี้เองว่าเป็นการรับข้อมูลหรือส่งข้อมูล

6) กระบวนการรับและส่งข้อมูลอนุกรมของ MCS-51

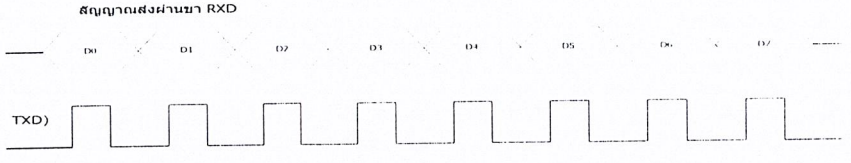
การส่งข้อมูลออกทางพอร์ตอนุกรมของ MCS-51 จะเริ่มต้น ขึ้นภายหลังเมื่อมีการเขียนข้อมูลลงใน SBUF ข้อมูลนี้จะถูกเลื่อนทีละบิต และส่งสัญญาณออกไปภายนอกโดยอัตโนมัติเมื่อข้อมูลเหล่านี้ได้ส่งออกไปครบถ้วนแล้วจะทำให้ค่าของแฟล็ก TI ให้เป็น 1 เพื่อแจ้งว่าขณะนี้ SBUF ว่าง และพร้อมที่จะส่งข้อมูลไบต์ต่อไปแล้ว ในกรณีที่ใช้เขียนข้อมูลใหม่ลงในรีจิสเตอร์ SBUF โดยไม่รอให้แฟล็ก TI มีค่าเป็น 1 ก่อนจะมีผลทำให้ข้อมูลที่ส่งออกไปมีค่าผิดพลาดได้

สำหรับการรับข้อมูลจากพอร์ตอนุกรมจะต้องเริ่มต้นโดยกำหนดเซ็ทค่าดังนี้ REN (Receiver Enable) ให้มีค่าเป็น 1 ก่อน หลังจากนั้นเมื่อมีข้อมูลภายนอกถูกส่งเข้ามายัง MCS-51 ทีละบิตจนครบ และ เมื่อบิตสุดท้ายถูกเลื่อนเข้ามาเรียบร้อยแล้วข้อมูลนั้นจะถูกย้ายมาเก็บไว้ยัง

รีจิสเตอร์ SBUF และ แฟล็ก RI ก็จะมีค่าเป็น 1 (ถูกเซ็ท) หลังจากนั้นก็จะเกิดการอินเตอร์รัพต์ขึ้น

7) พอร์ตสื่อสารอนุกรมโหมด 0

การทำงานของพอร์ตอนุกรม (โหมด 0) เป็นการรับและส่งข้อมูลอนุกรมจำนวน 8 บิต โดยใช้เพียงขาสัญญาณ RXD เท่านั้น (ขานี้ใช้งาน 2 หน้าที่ใช้ส่งและรับข้อมูล) ส่วนขาสัญญาณ TXD จะนำไปใช้เพื่อเป็นขาสัญญาณนาฬิกาในการให้จังหวะการเลื่อนข้อมูลกับวงจรเลื่อนบิตภายนอก สำหรับอัตราเร็วจะถูกกำหนดคงที่ไว้ที่ค่า 1/12 ของความถี่ออสซิลเลเตอร์ การทำงานของพอร์ตสื่อสารข้อมูลในโหมด 0 จะไม่มีบิตเริ่มต้นของข้อมูล(Start bit)และบิตสิ้นสุดของข้อมูล (Stop bit) เพราะจังหวะการรับและส่งข้อมูลจะถูกกำหนดจากสัญญาณ Shift clock แล้ว



รูปที่ 2.27 ข้อมูลที่รับและส่งในการทำงานของพอร์ตสื่อสารอนุกรมในโหมด 0

8) พอร์ตสื่อสารอนุกรม โหมด 1

การทำงานในโหมด 1 เป็นการสื่อสารข้อมูลอนุกรมจำนวน 10 บิต ประกอบด้วยบิตเริ่มต้น 1 บิต บิตข้อมูลจำนวน 8 บิต และบิตสุดท้ายอีก 1 บิต โดยข้อมูลจะถูกส่งออกทาง TXD และรับเข้ามาทางขาสัญญาณ RXD ในส่วนของข้อมูล 8 บิตที่ได้รับหรือทำการส่งออกจะเป็นบิตต่ำเป็นอันดับแรก ส่วนทางฝ่ายรับค่าของ Stop bit จะส่งเข้ามาจัดเก็บไว้ใน RB8 ภายในรีจิสเตอร์ SCON สำหรับอัตราความเร็วในการส่งข้อมูลของโหมด 1 นั้นสามารถกำหนดเลือกได้จาก Timer 1 ซึ่งในโครงงานนี้จะเลือกใช้โหมดนี้ในการรับและส่งข้อมูลเพราะว่าในโหมดนี้สามารถที่จะปรับอัตราเร็วในการส่งได้ซึ่งในโครงงานนี้จะส่งที่ความเร็ว 9600 ผังเวลาและการทำงาน

ดังได้กล่าวแล้วว่า การส่งข้อมูลอนุกรมในโหมด 1 สามารถที่จะเปลี่ยนแปลงความเร็วได้โดยการใช้ Timer 1 โดยโปรแกรม Timer 1 ทำงานในโหมด 2 8-bit automatic reload

$$\text{ความถี่อัตราบอด} = \frac{(2)^{\text{SNOD}}}{(32)} \text{x อัตราโอเวอร์โฟลว์ของ Timer1}$$

$$\text{หรือความถี่อัตราบอด} = \frac{(2)^{\text{SMOD}}}{(32)} \text{X} \frac{\text{fosc}}{12\text{x}(256 - \text{TH1})}$$

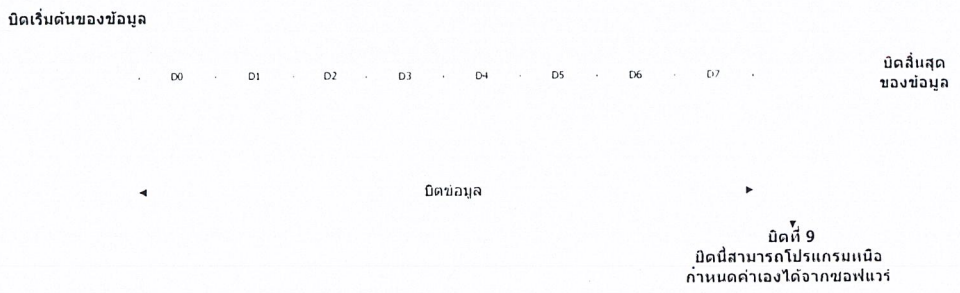
โดยที่ SMOD เป็นค่าของบิตภายในรีจิสเตอร์ PCON (มีค่าเป็น 0 หรือ 1) ค่าภายในรีจิสเตอร์ TH1 ซึ่งใช้เป็นค่าสำหรับโหลดซ้ำ

ตารางที่ 2.4 ค่าที่ต้องโหลดเข้าไปใน TH1

BAUD RATE	FOSC	SMOD			
			CT	MODE	Reload Value
(MODE 0)Max :1 MHz	12 MHz	X	X	X	X
(MODE 2)Max :375KHz	12 MHz	1	X	X	X
(MODE 2)Min : 18705 KHz	12 MHz	0	X	X	X
(MODE 1,3) :62.5 KHz	11.059 MHz	1	0	2	FFH
192.2 KHz	11.059 MHz	1	0	2	FDH
9.6 KHz	11.059 MHz	0	0	2	FDH
4.8 KHz	11.059 MHz	0	0	2	FAH
2.4 KHz	11.059 MHz	0	0	2	F4H
1.2 KHz	11.059 MHz	0	0	2	E8H
137.5 KHz	11.059 MHz	0	0	2	1DH
110 KHz	6 MHz	0	0	2	72H
110 KHz	6 MHz	0	0	1	FEEBH

9) พอร์ตสื่อสารอนุกรมโหมด 2

การทำงานแบบที่ สาม หรือการทำงานโหมด 2 จะเป็นการรับและส่งข้อมูลครั้งละ 11 บิต ข้อมูลจะถูกส่งออกภายนอกผ่านทางขา TXD และรับเข้ามาทางขา RXD ข้อมูลที่รับและส่งทั้ง 11 บิตประกอบด้วยบิตเริ่มต้นของข้อมูล 1 บิต (มีค่าเป็น 0 เสมอ) บิตข้อมูล 8 บิต (รับหรือส่งบิตต่ำสุดก่อน) ตามด้วยบิตที่ 9 (ต่อจากบิตข้อมูลบิตสุดท้าย) ซึ่งเป็นบิตที่สามารถกำหนดให้มีค่าเป็นศูนย์หรือหนึ่งได้ (Programmable 9^{th} data bit) และบิตสุดท้ายคือบิตสิ้นสุดของข้อมูล (มีค่าเป็นหนึ่งเสมอ) ดังนั้นจำนวนบิตที่รับส่งทั้งหมด 11 บิต จะประกอบด้วยบิตต่างๆดังในรูปที่ 2.28



รูปที่ 2.28 ข้อมูลรับและส่งในการทำงานของพอร์ตสื่อสารอนุกรม โหมด 2และ 3

ในขณะทำการส่งข้อมูล บิตที่ 9 จะได้จากค่าในบิต TB8 ของรีจิสเตอร์ใช้งานเฉพาะ SCON บิตนี้สามารถที่จะกำหนดให้มีค่าเป็น 0 หรือ 1 อย่างไรก็ได้ ส่วนใหญ่ในการใช้งานจริงมักจะใช้บิตนี้ในการตรวจสอบความถูกต้องของข้อมูลที่รับหรือส่ง (parity bit) โดยจะนำบิต P(Parity) ในรีจิสเตอร์ PSW ไปไว้ในบิต TB8 ส่วนใน ขณะรับข้อมูลบิตที่ 9 จะไปปรากฏอยู่ในบิต RB8 ของรีจิสเตอร์ SCON โดยไม่สนใจบิตสิ้นสุดของข้อมูล ค่าอัตราเร็วในการรับหรือส่งข้อมูลนี้จะถูกกำหนดไว้ที่ 1/32 หรือ 1/64 ของความถี่ออสซิลเลเตอร์ที่ใช้

10) พอร์ตสื่อสารอนุกรมโหมด 3

การทำงานของพอร์ตสื่อสารข้อมูลแบบอนุกรมแบบสุดท้าย คือการทำงานในโหมด 3 ในการทำงานโหมดนี้ข้อมูลจำนวน 11 บิตถูกส่งผ่านขา TXD และถูกรับเข้ามาทางขา RXD ข้อมูลทั้ง 11 บิตประกอบไปด้วยบิตเริ่มต้นของข้อมูล 1 บิต (เป็น 0 เสมอ) บิตข้อมูล 8 บิต (รับและส่งบิตต่ำสุดก่อน)ตามด้วยบิตที่ 9 ซึ่งเป็นบิตที่สามารถกำหนดค่าได้เหมือนในโหมด 2 (Programmable 9th bit) และบิตสุดท้ายคือบิตสิ้นสุดของข้อมูล (เป็น 1 เสมอ) อัตราเร็วในการรับหรือส่งข้อมูลสามารถที่จะเปลี่ยนแปลงได้ ดังนั้นจะเห็นว่ารูปแบบการรับส่งข้อมูลในโหมด 3 จะเหมือนกับโหมด 2 ทุกประการ แต่ในโหมดนี้สามารถกำหนดค่าอัตราเร็วในการรับส่งข้อมูลได้ตามความต้องการของผู้ใช้

การทำงานของพอร์ตสื่อสารแบบอนุกรมทั้ง 4 โหมดที่กล่าวมาแล้ว การส่งข้อมูลจะเริ่มทันทีเมื่อมีคำสั่งใดๆที่ใช้รีจิสเตอร์ใช้งานเฉพาะ (SBUF) เป็นรีจิสเตอร์ปลายทาง (Destination register) เช่น MOV SBUF, A

ส่วนในการรับข้อมูลจะเริ่มต้นโดยมีเงื่อนไขดังนี้

1) ในโหมด 0 จะเริ่มเมื่อค่าในบิต RI=0 และบิต REN=1

2) ในโหมดอื่นๆการรับข้อมูลเริ่มเมื่อ MCS-51 ได้รับบิตเริ่มต้นของข้อมูลเข้ามา โดยที่บิต REN ในขณะนั้นต้องมีค่า 1

2.7 แสดงผลทางคอมพิวเตอร์ (Display on Computer)

2.7.1 การนำเข้าสู่วิชชวลเบสิก (Introduction to visual Basic Program)

วิชชวลเบสิก (Visual Basic) มีสภาพแวดล้อมสำหรับการพัฒนาโปรแกรมบนวินโดวส์ (Window) ประกอบไปด้วยเครื่องมือต่างๆ ครบถ้วน ไม่ว่าจะเป็นส่วนของการออกแบบ ยูสเซอร์อินเตอร์เฟซ (User Interface) ส่วนออกแบบเมนู (Menu Designer) ส่วนช่วยเหลือการเขียนโปรแกรม (Help for program writing) และการดีบัก (Debugger) เพื่อการตรวจหาข้อผิดพลาดในโปรแกรม องค์ประกอบเหล่านี้นับว่าเอื้ออำนวยต่อการเขียนโปรแกรมมาก

ในด้านตัวภาษาวิชชวลเบสิก ได้นำไวยากรณ์ของเบสิก A และ GW-BASIC มาใช้โดยสนับสนุนความสามารถเดิมทั้งหมด นอกจากนี้ยังได้เพิ่มโปรแกรมแบบมีโครงสร้างของ QuickBasic ซึ่งคล้ายกับในภาษาที่มีโครงสร้าง เช่นภาษาปาสคาล (Pascal Program) หรือ ภาษาซี (C Program) เข้าไปด้วย

ด้วยความสามารถของตัวภาษาวิชชวลเบสิก สามารถนำไปสร้างแอปพลิเคชั่นได้หลายประเภท ไม่ว่าจะเป็นโปรแกรมวาดภาพ การคำนวณทางการเงิน หรือ แม้แต่โปรแกรมการรับค่าอินพุทเพื่อแสดงผลบนโปรแกรมวิชชวลเบสิก

2.7.2 เริ่มต้นในการเขียนโปรแกรมวิชชวลเบสิก (Beginning to Visual Programming)

โปรแกรมวิชชวลเบสิกจะใช้หลักของภาพและการมองเห็น โดยเริ่มจากการออกแบบวินโดวส์ย่อยหรือที่เรียกว่าฟอร์ม (Form) ในฟอร์มจะประกอบด้วยสิ่งต่างๆที่จะใช้งานหรือที่เรียกว่าเป็นออบเจ็กต์ (Object) เช่น ข้อความ ช่องรับข้อความ สคอลบาร์ (Scoll Bar) หรือปุ่ม (Button) เมื่อกำหนดสิ่งเหล่านี้ครบตามความต้องการ แล้วจึงระบุว่าองค์ประกอบแต่ละอย่างทำงานอย่างไร โดยการเขียนโปรแกรมย่อย (Private Sub) เข้าไปควบคุมออปเจ็กต์เหล่านี้ ซึ่งที่เป็นเช่นนี้เพราะว่าการทำงานในวินโดวส์จะเป็นแบบที่เรียกว่า อีเวนท์-ไดรเวนท์ (Event – Driven) คือ ขึ้นกับเหตุการณ์ (Event) เนื่องจากวินโดวส์จะต้องจัดการกับทุกแอพพลิเคชั่นที่ทำงานในขณะนั้นทั้งหมดไปพร้อมๆกัน เพราะการทำงานเป็นแบบระบบรันงานได้หลายๆงานในเวลาเดียวกัน (Multitasking) จึงต้องใช้รูปแบบการโปรแกรมในลักษณะ Event--Driven ดังกล่าว ซึ่งออบเจ็กต์แต่ละตัวก็จะมีเหตุการณ์เกิดขึ้นได้หลายอย่าง ถ้าสนใจเฉพาะเหตุการณ์ใดก็เขียนโปรแกรมสั่งงานให้ทำงานตามเฉพาะเหตุการณ์นั้นๆ เช่น ปุ่มควบคุมถ้าต้องการให้ทำงานเมื่อคลิกหรือดับเบิลคลิกก็จะต้องระบุว่าหากมีการคลิกที่ปุ่มควบคุมนี้โปรแกรมจะต้องทำงานอย่างไร หรือถ้ามีการดับเบิลคลิกจะต้องทำอย่างไร ส่วนเหตุการณ์อื่นๆที่ไม่ได้ระบุไว้ก็จะไม่มีผลต่อออบเจ็กต์นั้น

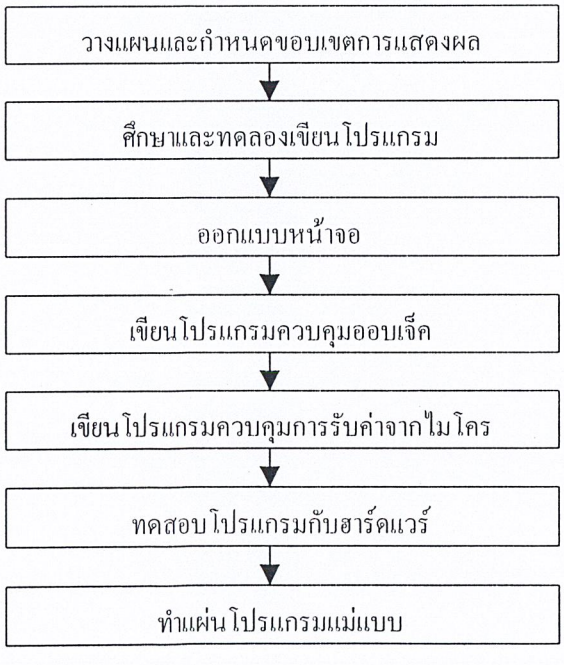
ซึ่งจะเห็นได้ว่า จะมีลักษณะเป็นธรรมชาติมาก คือกำหนดรูปแบบของจอภาพ หรือ ส่วนติดต่อกับผู้ใช้ (User Interface) และระบุว่าถ้าเกิดเหตุการณ์อย่างนี้กับสิ่งนี้จะต้องทำอย่างไร ซึ่งโปรแกรมที่เขียนก็คือส่วนที่จะบอกว่าเหตุการณ์จะเป็นอย่างไรต่อไป

นอกจากออปเจ็กต์จะมีการตอบสนองต่อเหตุการณ์ต่างๆที่กำหนดแล้ว ทุกออปเจ็กต์จะมีลักษณะหรือคุณสมบัติ (Property) ของตัวเอง เช่น ช่องรับข้อความ (Text Box) จะมีชื่อ ข้อความในนั้น ความกว้าง ความสูง สี โดยสามารถอ้างถึงหรือเปลี่ยนคุณสมบัติเหล่านี้ได้ขณะที่โปรแกรมทำงานอยู่ หรือขณะที่เขียนโปรแกรมก็ได้ โดยคุณสมบัติของทุกหน้าจอภาพดูได้ที่ภาคผนวก

ในการที่จะกระทำสิ่งใดสิ่งหนึ่งกับออบเจ็กต์นั้น จะมีสิ่งที่เรียกว่า เมทท็อด (Method) ซึ่งเปรียบเสมือนเป็นกระบวนการทำงานของออปเจ็กต์ ซึ่งออบเจ็กต์แต่ละแบบก็มี Method ที่แตกต่างกันออกไป เช่นถ้าต้องการจะแสดงข้อความหรือเลเบล (Label) ก็จะมีขบวนการหรือ Method ชื่อ Caption ของเลเบล เพื่อทำงานนี้โดยสั่งว่า Label Caption - ข้อความที่จะแสดง ซึ่งสรุปแล้ว Method ก็คล้ายคำสั่งที่ใช้ได้กับออบเจ็กต์แต่ละชนิดนั้นเอง ซึ่งทั้งหมดนี้เป็นแนวทางในการเขียนโปรแกรมวิชชวลเบสิกในโครงงานนี้ คือ

1) ออกแบบจอภาพหรือส่วนติดต่อกับผู้ใช้ (User Interface)

2) เขียนโปรแกรมควบคุมแต่ละออปเจ็กต์ให้ทำงานสอดคล้องกันกับโปรแกรมควบคุมการติดต่อรับค่าจากไมโครคอนโทรลเลอร์



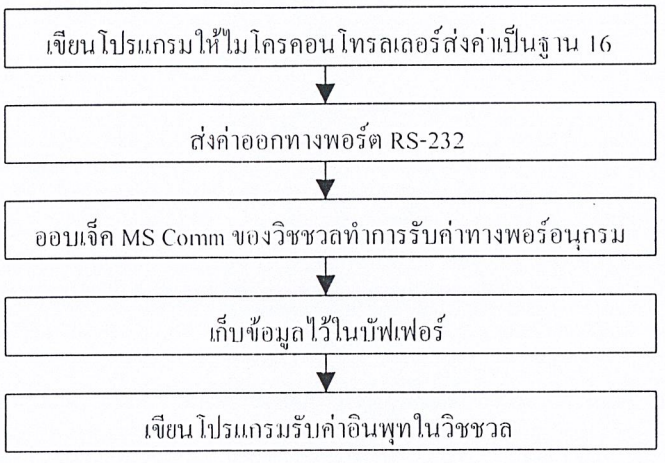
รูปที่ 2.29 ลำดับในการเขียนโปรแกรมวิชชวลเบสิกในโครงงานนี้

ลำดับขั้นตอนการออกแบบควบคุมการติดต่อรับค่าอินพุทจากไมโครคอนโทรลเลอร์

(Input from Microcontroller received by Programming)

สำหรับโปรแกรมควบคุมการติดต่อรับค่าจากไมโครคอนโทรลเลอร์เพื่อแสดงผล พร้อมโปรแกรมทั้งหมดได้แสดงไว้ในบทต่อไป ส่วนในหัวข้อนี้จะเป็นการกล่าวถึงโปรแกรมสำคัญที่ใช้ในการเขียนรับค่านี้ และลำดับขั้นตอนการส่งและรับค่า

โดยขั้นตอนสุดท้ายในการเขียนทั้งหมดจะเริ่มตั้งแต่กระบวนการรับ – ส่งค่าอินพุท จากไมโครคอนโทรลเลอร์ สุดท้ายเป็นการรับค่าเพื่อมาแสดงผล ซึ่งจะแสดงได้ผังการทำงานดังนี้



รูปที่ 2.30 กระบวนการรับ – ส่งค่าอินพุท

ขั้นตอนที่1 ในขั้นตอนนี้สำหรับการเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์ บนคอมพิวเตอร์ จะเขียนเป็นภาษาแอสเซมบลี แล้วทำการโหลดลงอีพร็อมเพื่อเก็บโปรแกรม และกระทำตามกระบวนการการเขียนโปรแกรม ส่วนข้อมูลที่จะส่งจะเป็นเลขฐาน 16 ดังรูปที่ 2.31

```
MOV A,#07FH
MOV DPTR,#8800 H
MOVX @DPTR,A
```

รูปที่ 2.31 การเขียนภาษาเครื่องด้วยภาษาแอสเซมบลี และข้อมูลเป็นเลขฐาน 16

ขั้นตอนที่ 2 เมื่อทำการเขียนโปรแกรมส่งข้อมูลทั้งหมดเรียบร้อยแล้ว การส่งข้อมูลถ้าเป็นแบบขนานจะส่งออกทางพอร์ตขนาน ส่วนถ้าส่งออกทางพอร์ตอนุกรมจะส่งผ่านทาง RS 232 ซึ่งอยู่บนบอร์ดไมโครคอมพิวเตอร์ ดังนั้น ข้อมูลจะถูกทยอยส่งออกอย่างเป็นระเบียบทีละนิด ซึ่งในการส่งข้อมูลหลังจากนี้ จะเป็นรหัสแอสกี้ (ASCII Code) โดยที่จะถูกปรับเปลี่ยนอัตโนมัติจากเลขฐาน 16 กลายเป็นรหัสแอสกี้ ซึ่งจะแสดงได้ดังรูปที่ 2.32

```
" @#$#$@^&&*&ER$*())(_+9^&^$$$^&***"
```

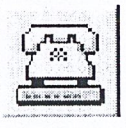
รูปที่ 2.32 รหัสแอสกี้

ขั้นตอนที่ 3 เมื่อข้อมูลทั้งหมดที่ส่งมาจากพอร์ต RS-232 คอมพิวเตอร์ทางฝ่ายผู้ใช้ จะทำการรับข้อมูลทางพอร์ตอนุกรม ซึ่งอาจจะเป็นพอร์ตคอม 1 หรือ 2 ขึ้นอยู่กับการใช้งาน โดยในโครงงานนี้จะใช้พอร์ตคอม 2 ในการรับข้อมูล

ขั้นตอนที่ 4 หลังจากนั้นจะนำข้อมูลทั้งหมดเก็บเข้าไว้ในบัฟเฟอร์ (Buffer) ซึ่งจะเร็วหรือช้าจะขึ้นอยู่กับอัตราความเร็วในการส่งจากไมโครคอนโทรลเลอร์ (Board Rate) โดยคอมพิวเตอร์จะจัดการให้อย่างอัตโนมัติ เช่นความเร็วที่ 1200,2400,9600,19200

ขั้นตอนที่ 5 ในขั้นตอนนี้ ผู้ใช้จำเป็นต้องเขียนโปรแกรมไปรับข้อมูลจากบัฟเฟอร์ นั่นคือโปรแกรมวิชชวลเบสิก คำสั่งสื่อสาร และออบเจ็กต์เพื่อการสื่อสาร ในการที่จะนำข้อมูลที่ส่งมานั้นมาทำการแสดงผล ซึ่งจะแบ่งออกเป็น 3 ส่วนหลัก คือ

1) ออบเจ็กต์ MSComm ซึ่งจะแสดงได้ดังรูปที่ 2.33



รูปที่ 2.33 ออปเจ็กต์ของพอร์ตสื่อสารบนวิชชวลเบสิก

ซึ่งจะต้องศึกษาในส่วนของออปเจ็กต์ ในส่วนของการขอความช่วยเหลือ (Help)

2) โปรแกรมวิชชวลเบสิก โดยยกตัวอย่างดังรูปที่ 2.34

```
EE2.Label45.Caption = "OK"
EE3.Gauge2.Value = A
EE3.Label22.Caption = A
EE3.Label25.Caption = " Input  Normal "
```

รูปที่ 2.34 ตัวอย่างโปรแกรมบนวิชชวลเบสิก

3) โปรแกรมติดต่อสื่อสาร ซึ่งก็คือโปรแกรมวิชชวลนั่นเองแต่จำเป็นต้องศึกษาเพิ่มเติมเพื่อใช้ในการติดต่อสื่อสารข้อมูล ดังรูปที่ 2.35

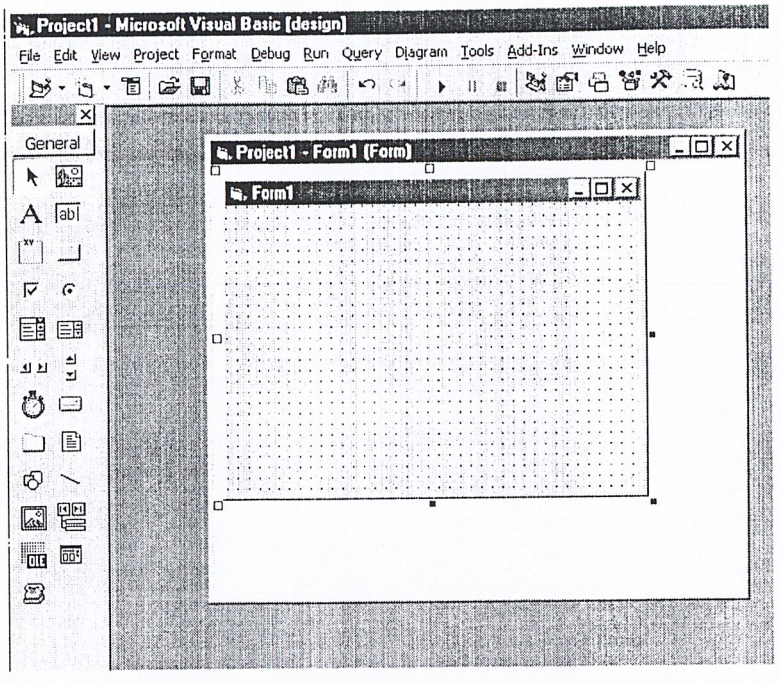
```
Const port = 2
MSComm1.CommPort = port
Open Port
MSComm1.PortOpen = true
```

รูปที่ 2.35 ตัวอย่างของโปรแกรมสื่อสารกับพอร์ตอนุกรม

2.7.3 การออกแบบจอภาพและเขียนโปรแกรม

1) การวาดภาพด้วยออบเจ็กต์ต่างๆ

เมื่อทำการเรียกวิชชวลเบสิกขึ้นมา ทูลบ็อกซ์ (Tool Box) ปรากฏบนจอภาพด้านซ้ายมือ ทูลบ็อกซ์นี้จะบรรจุรูปภาพซึ่งเป็นตัวแทนออปเจ็กต์ต่างๆที่จะวาดลงบนฟอร์มได้ดังรูปที่ 2.36



รูปที่ 2.36 ทูลบ็อกซ์ของวิชชวลเบสิก

การวาดออบเจ็กต์ลงบนฟอร์มมีขั้นตอนดังต่อไปนี้เสมอ คือ

1) ต้องคลิกออบเจ็กต์ที่ต้องการวาดจากทูลบ็อกซ์

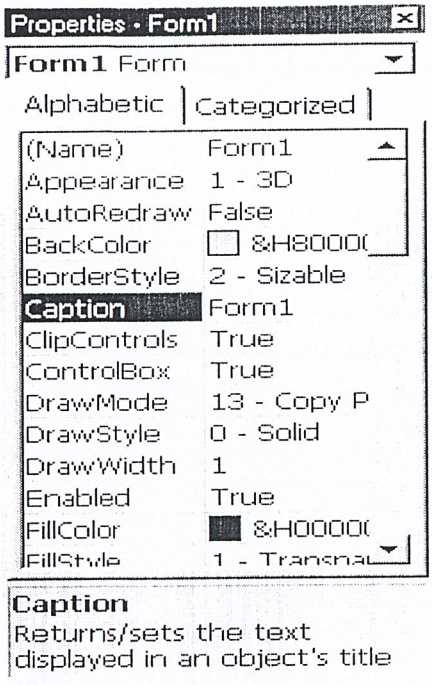
2) เลื่อนเมาส์บนฟอร์มไปยังตำแหน่งที่ต้องการจะวาดออบเจ็กต์

3) กดปุ่มซ้ายค้างไว้แล้วลาก จะปรากฏออบเจ็กต์ที่ต้องการบนฟอร์ม

2) การกำหนดคุณสมบัติ (Property) ให้กับยูสเซอร์อินเตอร์เฟส

การวาดยูสเซอร์อินเตอร์เฟสเป็นขั้นตอนแรกสำหรับการเขียนโปรแกรมด้วยวิชชวลเบสิก ส่วนขั้นตอนที่สองก็คือ การกำหนดคุณลักษณะต่างๆให้กับแต่ละออบเจ็กต์บนยูสเซอร์อินเตอร์เฟส

วิชชวลเบสิกจะตั้งค่าคุณสมบัติโดยปริยายให้กับออบเจ็กต์ทุกตัว อย่างอัตโนมัติ แต่อย่างไรก็ตาม ถ้าไม่ต้องการใช้ค่าโดยปริยายเหล่านี้ ก็สามารถปรับปรุงยูสเซอร์อินเตอร์เฟสได้ด้วย การเปลี่ยนแปลงค่าคุณสมบัติของออปเจ็กต์ใหม่ ซึ่งจะแสดงหน้าต่างคุณสมบัติค่าพร็อพเพอตี้ต่างๆ ดังรูปที่ 2.37

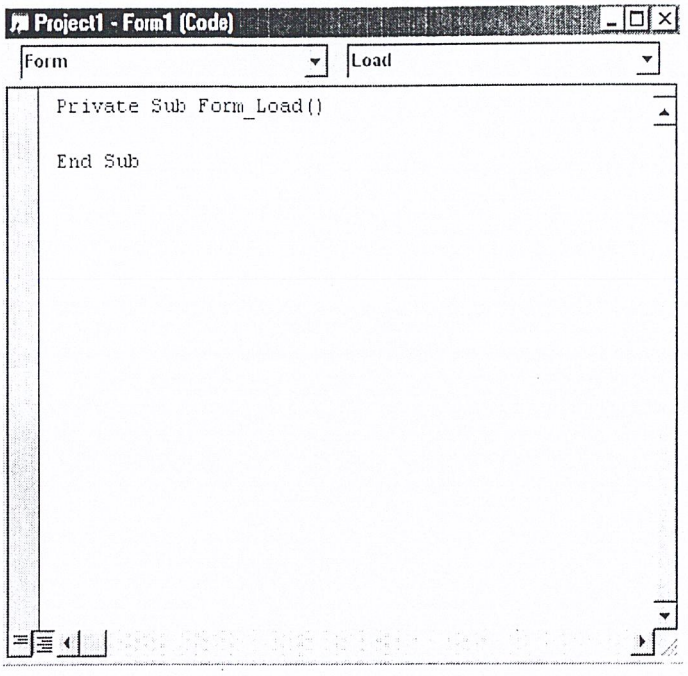


รูปที่ 2.37 หน้าต่างคุณสมบัติแสดงค่าพร็อพเพอตี้ต่างๆ

สำหรับการเขียนโปรแกรมวิชชวลเบสิกนั้น เพื่อให้คอมพิวเตอร์ทำงานต้องใช้คำสั่งที่มันสามารถเข้าใจได้ ซึ่งถ้ามีหลายๆโปรแกรมวิชชวลเบสิกจะเรียกว่าโพรซีเยอร์ (Procedure) ซึ่งจะได้กล่าวต่อไป

3) การเขียนโปรแกรมย่อย (procedure) ในวิชชวลเบสิก

สิ่งแรกที่ต้องทำก่อนคือ วาดออบเจ็กต์ลงมาบนฟอร์ม ต่อจากนั้นอาจจะเปลี่ยนแปลงคุณสมบัติของแต่ละออบเจ็กต์ เพื่อให้มีชื่อแตกต่างกัน และ จำได้ง่าย ถ้าไม่ทำการตั้งชื่อใหม่ออปเจ็กต์เหล่านั้นก็จะใช้ชื่อโดยปริยายที่วิชชวลเบสิกตั้งให้ เช่น ออปชั่น 1 (Option 1) หรือเท็กซ์ 3 (text 3) เป็นต้น จากนั้นก็เริ่มเปิดหน้าต่างที่ใช้เขียนโปรแกรมย่อยขึ้นมาดังรูปที่ 2.38 (โดยคลิกบนออบเจ็กต์ที่ต้องการ แล้วกด F7 หรือดับเบิลคลิกบนออบเจ็กต์) ก็เริ่มเขียนโปรแกรมได้แล้ว



รูปที่ 2.38 โค้ดหน้าต่างที่จะเขียนโปรแกรม

ซึ่งจะสังเกตุได้ว่า ในบรรทัดแรกของโปรแกรมย่อย (Procedure) เริ่มด้วย Private Sub ซึ่งแสดงว่าโปรแกรมย่อยนี้ เป็นของออปเจ็กต์ที่ระบุเท่านั้นคือ Form ต่อจากนั้นในระหว่างสองบรรทัดนี้จะต้องทำการเขียนโปรแกรมขึ้นมาเพื่อให้คอมพิวเตอร์รับรู้ บรรทัดสุดท้ายของโพรซีเยอร์ทุกๆตัว จะต้องปิดด้วย End Sub เสมอ โดยจะแสดงว่าสิ้นสุดการทำทั้งหมดของโปรแกรมย่อยนี้แล้ว ซึ่งในหนึ่งหน้าต่างจอภาพใดๆที่ต้องการสร้างนั้นจะมีหลายออปเจ็กต์ ดังนั้นจะมีหลายโพรซีเยอร์ โดยจะต้องเขียนประกอบกันรวมกันเป็นหนึ่งโปรแกรม ซึ่งหลังจากเขียนโพรซีเยอร์เสร็จ คอมพิวเตอร์จะทำงานตามคำสั่งทีละบรรทัดจากบนลงล่าง และจะจบการทำงานเมื่อสิ้นสุดบรรทัดสุดท้าย คือเมื่อพบคำสั่ง End Sub นั่นเอง

ในวิชชวลเบสิก คำสั่งจะทำงานเมื่อมีเหตุการณ์จำเพาะเกิดขึ้นเท่านั้น อย่างเช่น ทุกๆครั้งที่คลิกเมาส์บนออบเจ็กต์ ชุดคำสั่งชุดเดียวกันของเหตุการณ์นี้จะถูกดำเนินการ และ โปรแกรมจะสิ้นสุดเมื่อออบเจ็กต์เจาะจงว่าต้องจบการทำงาน

2.7.4 ข้อเปรียบเทียบภาษาวิชชวลเบสิกกับภาษาอื่น

1) การสร้างยูสเซอร์อินเตอร์เฟสโดยการวาด

ในภาษาทั่วๆไป เช่น ภาษาซีพลัสพลัสแต่จะเรียกย่อๆว่า ภาษาซี (C++Language) และปาสคาล (Pascal Language) จะต้องทำการเขียนโปรแกรม (Program) เพื่อผล 3 สิ่งคือ

บทที่ 3

การออกแบบและการสร้าง

3.1 การออกแบบวงจรพูช – พูล อินเวอร์เตอร์

สำหรับในโครงงานนี้วงจรอินเวอร์เตอร์ที่เลือกใช้งานจะใช้แบบ พุช-พูล อินเวอร์เตอร์ (Push-Pull Inverter) ซึ่งคุณลักษณะที่เหมาะสมของการเลือกใช้แบบนี้ก็เพื่อ

1) สามารถจ่ายกำลังไฟฟ้าได้สูง 500 วัตต์ขึ้นไป ดังนั้นค่าพิกัดของแหล่งจ่ายไฟฟ้าสำรองในโครงงานนี้จึงอยู่ในเกณฑ์ที่พิจารณา

2) แรงดันที่ใช้ในการสวิตชิ่งของอุปกรณ์มีค่าต่ำ

3) ราคาถูกที่สุด เมื่อเปรียบเทียบกับอินเวอร์เตอร์ชนิดอื่นๆ เนื่องจากว่าใช้อุปกรณ์จำนวนน้อยในการสร้างวงจรอินเวอร์เตอร์

4) ไม่คำนึงถึงผลของคุณสมบัติของรูปคลื่นเท่าไรนัก เนื่องจากว่าจุดประสงค์ของโครงงานเพื่อสามารถจ่ายกำลังไฟฟ้าสำรองแทนระบบได้เท่านั้น เมื่อเกิดผิดปกติขึ้นกับระบบ แต่สิ่งที่พิเศษกว่าคือการควบคุมที่ง่ายด้วยไมโครคอนโทรลเลอร์ และมีการแสดงผลด้วยคอมพิวเตอร์

5) เนื่องจากรูปคลื่นสัญญาณในการสวิตชิ่งวงจรอินเวอร์เตอร์จะสวิตชิ่งที่ความถี่ 50 Hz ดังนี้อุปกรณ์สวิตชิ่งจึงไม่จำเป็นต้องตอบสนองใช้เวลาที่รวดเร็ว (Slow dynamic Response)

6) เนื่องจากสัญญาณที่ใช้ในการขับเป็นรูปคลื่นสัญญาณสี่เหลี่ยม มีความถี่ 50Hz ดังนั้นการสร้างสัญญาณขับนี้ โดยใช้ไมโครคอนโทรลเลอร์จะทำได้ง่าย

7) การออกแบบวงจรได้ง่าย เนื่องจากว่าใช้งานที่ความถี่ต่ำ จึงมักไม่ค่อยมีปัญหาเรื่องของสไปค์ (Spike) การใช้หม้อแปลงความถี่สูง หรือผลจากเส้นแรงแม่เหล็กมีค่าไม่เท่ากัน

8) การออกแบบหม้อแปลงไม่ต้องใช้แกนเฟอร์ไรท์ ซึ่งจะช่วยประหยัดได้มาก

ดังนั้นจากคุณลักษณะข้างต้นจึงได้ทำการออกแบบวงจรอินเวอร์เตอร์ที่ใช้ในโครงงานนี้ให้มีเงื่อนไขเป็นไปตามคุณลักษณะดังกล่าว โดยแสดงเป็นหัวข้อได้ดังนี้

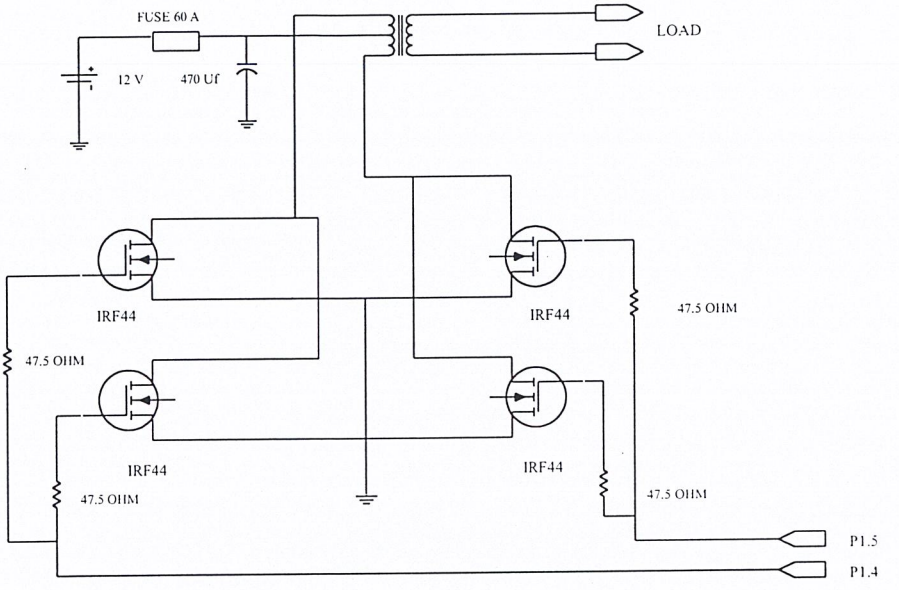
1) วงจรอินเวอร์เตอร์ใช้อุปกรณ์สวิตชิ่ง คือ ฟอสเฟตจำนวน 4 ตัวแบ่งเป็นกิ่งละ 2 ตัว

2) ใช้รูปคลื่นสัญญาณสี่เหลี่ยม (Square wave signal) มีความถี่ที่ 50 Hz จำนวน 2 สัญญาณกลับเฟสกัน 180 องศาไฟฟ้า ขับวงจรอินเวอร์เตอร์ซึ่งสร้างจากไมโครคอนโทรลเลอร์

3) แรงดันไฟฟ้าตรงที่ใช้ขับเกทมอสเฟตมีค่า 12 โวลท์

4) หม้อแปลงถูกออกแบบใช้งานที่ความถี่ 50 Hz

ดังนั้น ในการออกแบบจะอาศัยคุณลักษณะและเงื่อนไขดังกล่าวนี้ในการออกแบบวงจรอินเวอร์เตอร์ โดยต่อไปจะกล่าวถึงทฤษฎีการใช้งานไปจนถึงการออกแบบวงจรที่ใช้ในโครงงานนี้ วงจรพื้นฐานการทำงานของวงจร พุช-พูล อินเวอร์เตอร์ (Basic of Push-Pull Inverter)



รูปที่ 3.1 วงจรพื้นฐานของวงจรพุช-พูลอินเวอร์เตอร์

3.2 การออกแบบวงจรอัดประจุแบตเตอรี่

ภาคอัดประจุแบตเตอรี่ เป็นอุปกรณ์ทำหน้าที่อัดประจุให้กับแบตเตอรี่ โดยอุปกรณ์นี้จะเปลี่ยนไฟฟ้ากระแสสลับเป็นไฟฟ้ากระแสตรง สำหรับยูพีเอสระบบ Line-interactive UPS (ยูพีเอสที่มี Stabilizer อยู่ในตัว)

ในโครงงานแหล่งจ่ายไฟฟ้าสำรองนี้ แบตเตอรี่ที่เลือกใช้นั้นจะเป็นชนิดตะกั่ว – กรด ขนาด 46 แอมแปร์ – ชั่วโมง โดยที่แบตเตอรี่นี้ไม่ต้องมีการเติมน้ำกลั่นหรือบำรุงรักษาแบตเตอรี่ใดๆ อีกทั้งแบตเตอรี่นี้ไม่มีก๊าซเกิดขึ้น ซึ่งเป็นผลดีในการช่วยแก้ปัญหาอายุการใช้งานแหล่งจ่ายไฟฟ้าสำรองสั้น เนื่องจากการเกิดก๊าซในการอัดประจุมากเกินไป ส่วนการอัดประจุแบตเตอรี่จะเป็นแบบแรงดันคงที่ (Constant Voltage Charging) เนื่องจากแบตเตอรี่ในแหล่งจ่ายไฟสำรองจะทำงานแบบแบ็คอัพ (Backup) หรือ Standby ซึ่งแบตเตอรี่เป็นแหล่งพลังงานสำรองที่ต้องนำมาใช้ในสภาวะเกิดความผิดปกติของระบบการไฟฟ้าฯ ฉะนั้นแบตเตอรี่จะต้องถูกอัดเมื่อมีแรงดันต่ำกว่าที่กำหนดไว้ โดยในช่วงที่แบตเตอรี่เต็มแล้วจะต้องอัดประจุให้กับแบตเตอรี่ด้วยอัตราการอัดประจุ

ต่ำๆเพื่อชดเชยค่าแรงดันเนื่องจากความต้านทานภายในแบตเตอรี่โดยแรงดันที่จ่ายให้แบตเตอรี่เพื่ออัดประจุ จะเป็นแบบ Float Charge ที่ประมาณ 12 โวลท์

ในการอัดประจุแบตเตอรี่ชนิดนี้ จะมีการจำกัดกระแสที่ใช้อัดประจุและแรงดันต้องพยายามให้คงที่มากที่สุด ปริมาณกระแสที่ใช้อัดประจุโดยปกติจะจำกัดอยู่ที่ 0.2 C สำหรับ Lead – antimony Battery และ0.5 C สำหรับ Lead – calcium Battery เมื่อ C คือความจุแบตเตอรี่ (หน่วย Ampere –hour (Ah) ถ้าหากการอัดประจุโดยใช้กระแสมากเกินไป จะทำให้อุณหภูมิในแบตเตอรี่สูงและทำให้อายุการใช้งานของแบตเตอรี่ลดลง และถ้าหากอัดประจุที่กระแสน้อยเกินไปจะทำให้มี Lead –sulfate เกาะหลงเหลืออยู่ในแผ่นเพลท (plate) ของแบตเตอรี่ ซึ่งจะทำให้การจ่าย ไม่ได้เท่ากับพิกัดของแบตเตอรี่ และถ้าต้องการให้แบตเตอรี่มีอายุการใช้งานยาวนาน แรงดันพีคทูพีค (peak to peak ripple) ซึ่งเป็นแรงดันเอาท์พุทของอุปกรณ์อัดประจุแบตเตอรี่ ควรจะมีค่าไม่เกิน 0.5% ของแรงดันดีซี พึงระลึกเสมอว่าการอัดประจุแบตเตอรี่ที่ค่าพิกัดของแบตเตอรี่ จะสามารถยืดอายุการใช้งานของแบตเตอรี่ได้ยาวนานขึ้น

การออกแบบเพื่อหาขนาดของแบตเตอรี่

ก่อนอื่นต้องทราบพารามิเตอร์ของแหล่งจ่ายไฟฟ้าต่อเนื่องในโครงงานนี้ก่อน เพื่อที่จะนำค่าต่างๆมาคำนวณหาขนาดแบตเตอรี่ ซึ่งกำหนดค่าต่างๆที่สำคัญได้ดังนี้

1) กำลังไฟฟ้าพิกัดของแหล่งจ่ายไฟฟ้าสำรอง 1000 VA ที่ภาระมีค่า PF = 1

2) เวลาสำรองจ่ายภาระได้ระยะเวลา 30 นาทีที่กำลังไฟฟ้าและเพาเวอร์แฟคเตอร์ที่พิกัด

3) ประสิทธิภาพของอินเวอร์เตอร์เท่ากับ 0.9

4) แรงดันปกติของแบตเตอรี่ 12 โวลท์

ในการคำนวณหาขนาดแบตเตอรี่ จะกำหนดให้

VA = โวลท์แอมป์แปร์พิกัดของภาระ

PF = ค่าเพาเวอร์แฟคเตอร์ของภาระ

η = ประสิทธิภาพของอินเวอร์เตอร์ (efficiency of Inverter)

V_b = แรงดันปกติของแบตเตอรี่

ดังนั้นคำนวณกำลังไฟฟ้าที่ส่งจากแบตเตอรี่ คือ

กำลังไฟฟ้าจากแบตเตอรี่ = VA × PF × Efficiency of Inverter

$$= 1000 \times 1 \times \frac{1}{0.9}$$

$$= 1111.11 \text{ W}$$

กระแสจากแบตเตอรี่ $= \dfrac{W}{V_{Battery}}$

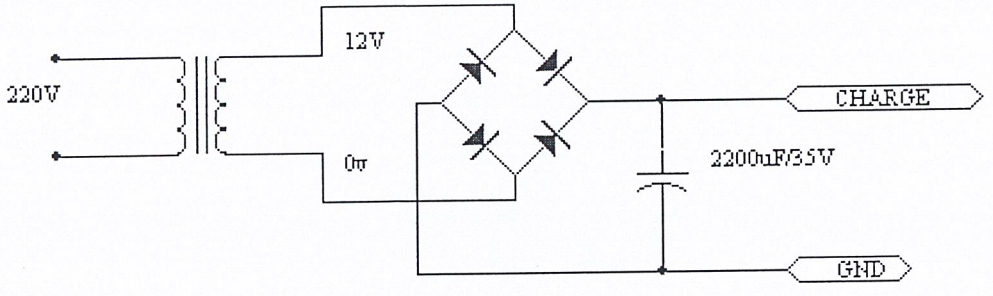
$= \dfrac{1111.11}{12} = 92.59 \text{ A}$

ค่าแอมแปร์ – ชั่วโมง (Ah) $= W \times \left(\dfrac{t}{60}\right) \times \left(\dfrac{1}{V_{Battery}}\right)$

$= \dfrac{1111.11 \times 30 \times 1}{60 \times 12}$

$= 46.29 \text{ Ah}$

ดังนั้นขนาดของแอมป์ – ชั่วโมงจะใช้ขนาด 46 Ah ที่พิกัดของแหล่งจ่ายไฟฟ้าสำรองนี้ วงจรอัดประจุแบตเตอรี่ที่ใช้ในโครงงานออกแบบดังนี้



รูปที่ 3.2 วงจรอัดประจุแบตเตอรี่

สรุปแบตเตอรี่ที่ใช้ในโครงงาน

1) ชนิด Sealed Lead – Acid Battery ขนาด 46 Ah

2) การใช้งานแบบ Standby สำรองเวลาได้ประมาณ 30 นาทีที่พิกัด 1000 VA

3) การทำงานเป็นแบบ Float Charge

4) เครื่องอัดประจุแบตเตอรี่เป็นแบบแรงดันคงที่ (Constant Voltage Charging)

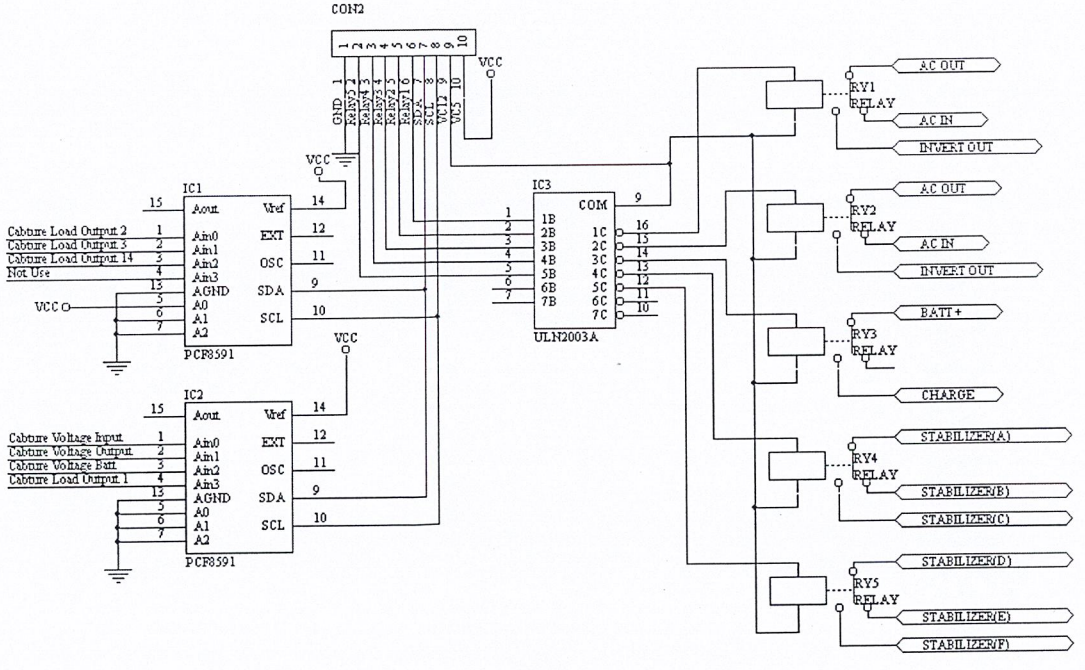
5) การอัดประจุจะใช้กระแสเท่ากับ 15 A

3.3 การออกแบบวงจรสวิตซ์สับเปลี่ยนและวงจรรักษาระดับแรงดัน

วงจรสวิตช์สับเปลี่ยน (Transfer Switch) นับได้ว่าเป็นอุปกรณ์ที่มีความสำคัญอย่างมากทีเดียวในแหล่งจ่ายไฟฟ้าต่อเนื่อง โดยจะทำงานใน 2 สภาวะ คือ

1) ในสภาวะปกติ สวิตช์สับเปลี่ยนจะสับไปที่ระบบ เพื่อรับกำลังไฟฟ้าจากระบบ โดยรีเลย์จะถูกควบคุมการทำงานจากไมโครคอนโทรลเลอร์ ซึ่งในสภาวะนี้ไมโครคอนโทรลเลอร์ตรวจสอบแล้วว่าระบบมีสภาวะปกติ มีสัญญาณแรงดันไฟฟ้าพร้อมที่จะจ่ายให้กับภาระ ดังนั้นรีเลย์จึงถูกสับเข้ากับระบบเพื่อเป็นทางผ่านของพลังงานไฟฟ้าของระบบไปสู่ภาระ

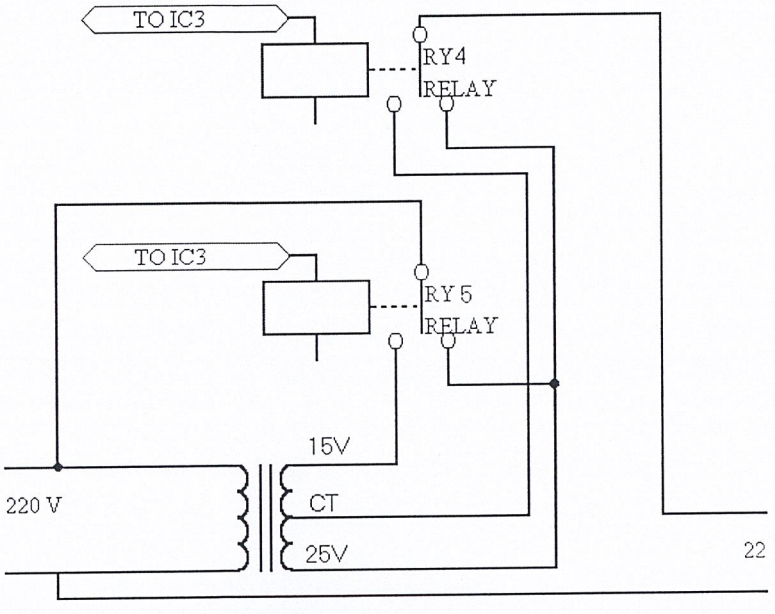
2) ในสภาวะผิดปกติ สวิตช์สับเปลี่ยนจะสับไปที่อินเวอร์เตอร์ เพื่อรับกำลังไฟฟ้าจากอินเวอร์เตอร์ซึ่งในสภาวะนี้ไมโครคอนโทรลเลอร์ตรวจสอบแล้วพบว่าระบบมีสภาวะผิดปกติเนื่องมาจากสาเหตุ ระบบไม่จ่ายกำลังไฟฟ้า (Brownout) แรงดันระบบมีค่าต่ำเกินไป (Under Voltage) ซึ่งในสภาวะนี้จะไม่มีสัญญาณแรงดันไฟฟ้าพร้อมที่จะจ่ายให้กับภาระ ดังนั้นรีเลย์จึงถูกสับเข้ากับอินเวอร์เตอร์เพื่อจ่ายพลังงานไฟฟ้าไปสู่ภาระแทนระบบการออกแบบวงจรสวิตช์สับเปลี่ยน



รูปที่ 3.3 วงจรสวิตช์สับเปลี่ยน

วงจรรักษาระดับแรงดัน (Stabilizer) เป็นวงจรที่มีความสำคัญอีกวงจรหนึ่งซึ่งเป็นวงจรที่ทำการรักษาระดับแรงดันไฟฟ้าไว้ในกรณีที่ระดับแรงดับไฟฟ้าเกิดการผิดพลาด เช่นไฟฟ้าตกหรือไฟฟ้าเกิน วงจรรักษาระดับแรงดันจะทำการตรวจเช็คแรงดันอินพุตอยู่ตลอดเวลาเมื่อมีการเปลี่ยนแปลงของแรงดับเกินกว่าที่กำหนดไว้วงจรรักษาระดับแรงดันจะทำงานทันที

การออกแบบวงจรรักษาระดับแรงดัน

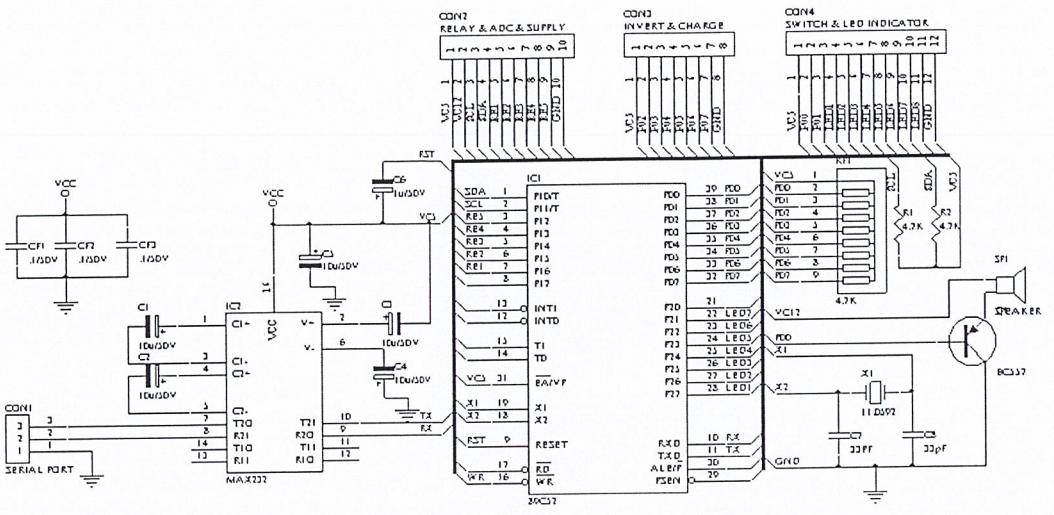


รูปที่ 3.4 วงจรรักษาระดับแรงดัน

3.4 การออกแบบวงจรควบคุมไมโครคอนโทรลเลอร์

ส่วนนี้ถือว่าเป็นส่วนที่สำคัญที่สุดของแหล่งจ่ายไฟฟ้าสำรอง โดยจะมีทั้งการรับอินพุตประมวลผล และส่งสัญญาณไปควบคุมอุปกรณ์ในส่วนต่างๆและส่งสัญญาณเอาท์พุตออกพอร์ตเอาท์พุต RS 232 เพื่อที่เป็นสัญญาณอินพุตให้กับอุปกรณ์ MSComm ในโปรแกรมวิชวลเบสิก (Visual Basic) เพื่อแสดงผลออกทางคอมพิวเตอร์โดยใช้ไมโครคอนโทรลเลอร์ตระกูล MCS – 51 และมีไอซี ADC เป็นตัวแปลงสัญญาณแอนาล็อกเป็นสัญญาณดิจิตอล ซึ่งจะทำการรับสัญญาณแอนาล็อกจากวงจรตรวจจับแล้วแปลงสัญญาณให้กับไมโครคอนโทรลเลอร์ เพื่อประมวลผลแล้วทำการควบคุมระบบต่อไป

การออกแบบวงจรควบคุมด้วยไมโครคอนโทรลเลอร์ดังแสดงดังรูป



รูปที่ 3.5 วงจรควบคุมไมโครคอนโทรลเลอร์

3.5 การออกแบบการแสดงผลทางคอมพิวเตอร์

การออกแบบการแสดงผลทางคอมพิวเตอร์มีขั้นตอนดังนี้

ขั้นตอนที่ 1 ในขั้นตอนนี้สำหรับการเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์ บนคอมพิวเตอร์ จะเขียนเป็นภาษาแอสเซมบลี แล้วทำการโหลดลงในหน่วยความจำของ MCS-51 เพื่อเก็บโปรแกรม และกระทำตามกระบวนการการเขียนโปรแกรม ส่วนข้อมูลที่จะส่งจะเป็นเลขฐาน 16 ดังรูปที่ 3.6

เช่น และ

MOV A,#07FH	07FH = 127
MOV DPTR,#8800 H	0FFH = 225
MOVX @DPTR,A	031H = 49

รูปที่ 3.6 การเขียนภาษาเครื่องด้วยภาษาแอสเซมบลี และข้อมูลเป็นเลขฐาน 16

ขั้นตอนที่ 2 เมื่อทำการเขียนโปรแกรมส่งข้อมูลทั้งหมดเรียบร้อยแล้ว การส่งข้อมูลถ้าเป็นแบบขนานจะส่งออกทางพอร์ตขนาน ส่วนถ้าส่งออกทางพอร์ตอนุกรมจะส่งผ่านทาง RS 232 ซึ่งอยู่บนบอร์ดไมโครคอมพิวเตอร์ ดังนั้นข้อมูลจะถูกทยอยส่งออกอย่างเป็นระเบียบทีละนิด ซึ่งในการส่งข้อมูลหลังจากนี้ จะเป็นรหัสแอสกี้ (ASCII Code) โดยที่จะถูกปรับเปลี่ยนอัตโนมัติจากเลขฐาน 16 กลายเป็นรหัสแอสกี้ ซึ่งจะแสดงได้ดังรูปที่ 3.7

```
" @#$#$@^&&*&ER$*())(_+9^&^$$$^&***"
```

รูปที่ 3.7 รหัสแอสกี้

ขั้นตอนที่ 3 เมื่อข้อมูลทั้งหมดที่ส่งมาจากพอร์ต RS-232 คอมพิวเตอร์ทางฝ่ายผู้ใช้ จะทำการรับข้อมูลทางพอร์ตอนุกรม ซึ่งอาจจะเป็นพอร์ตคอม 1 หรือ 2 ขึ้นอยู่กับการใช้งาน โดยในโครงงานนี้จะใช้พอร์ตคอม 2 ในการรับข้อมูล

ขั้นตอนที่ 4 หลังจากนั้นจะนำข้อมูลทั้งหมดเก็บเข้าไว้ในบัฟเฟอร์ (Buffer) ซึ่งจะเร็วหรือช้าจะขึ้นอยู่กับอัตราความเร็วในการส่งจากไมโครคอนโทรลเลอร์ (Board Rate) โดยคอมพิวเตอร์จะจัดการให้อย่างอัตโนมัติ เช่นความเร็วที่ 1200,2400,9600,19200

ขั้นตอนที่ 5 ในขั้นตอนนี้ ผู้ใช้จำเป็นต้องเขียนโปรแกรมไปรับข้อมูลจากบัฟเฟอร์ นั่นคือโปรแกรมวิชชวลเบสิก คำสั่งสื่อสาร และออบเจ็กต์เพื่อการสื่อสาร ในการที่จะนำข้อมูลที่ส่งมานั้นมาทำการแสดงผล ซึ่งจะแบ่งออกเป็น 3 ส่วนหลัก คือ

1) ออบเจ็กต์ MSComm ซึ่งจะแสดงได้ดังรูปที่ 3.8

รูปที่ 3.8 ออบเจ็กต์ของพอร์ตสื่อสารบนวิชชวลเบสิก

ซึ่งจะต้องศึกษาในส่วนของออปเจ็กต์ ในส่วนของการขอความช่วยเหลือ (Help)

2) โปรแกรมวิชชวลเบสิก โดยยกตัวอย่างดังรูปที่ 3.9

```
EE2.Label45.Caption = "OK"
EE3.Gauge2.Value = A
EE3.Label22.Caption = A
EE3.Label25.Caption = " Input Normal "
```

รูปที่ 3.9 ตัวอย่างโปรแกรมบนวิชชวลเบสิก

3) โปรแกรมติดต่อสื่อสาร ซึ่งก็คือโปรแกรมวิชชวลนั่นเองแต่จำเป็นต้องศึกษาเพิ่มเติมเพื่อใช้ในการติดต่อสื่อสารข้อมูล ดังรูปที่ 3.10

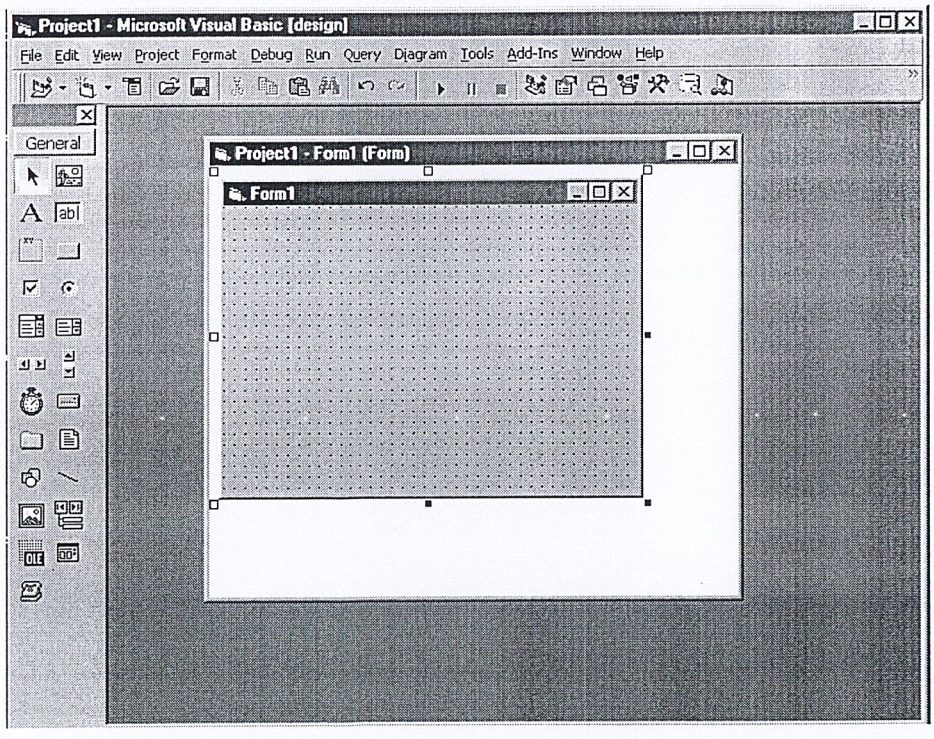
```
Const port = 2
MSComm1.CommPort = port
Open Port
MSComm1.PortOpen = true
```

รูปที่ 3.10 ตัวอย่างของโปรแกรมสื่อสารกับพอร์ตอนุกรม

การออกแบบจอภาพและเขียนโปรแกรม

1) การวาดภาพด้วยออบเจ็กต์ต่างๆ

เมื่อทำการเรียกวิชชวลเบสิกขึ้นมา ทูลบ็อกซ์ (Tool Box) ปรากฏบนจอภาพด้านซ้ายมือ ทูลบ็อกซ์นี้จะบรรจุรูปภาพซึ่งเป็นตัวแทนออปเจ็กต์ต่างๆที่จะวาดลงบนฟอร์มได้ดังรูปที่ 3.11



รูปที่ 3.11 ทูลบ็อกซ์ของวิชชวลเบสิก

การวาดออบเจ็กต์ลงบนฟอร์มมีขั้นตอนดังต่อไปนี้เสมอ คือ

1) ต้องคลิกออบเจ็กต์ที่ต้องการวาดจากทูลบ็อกซ์

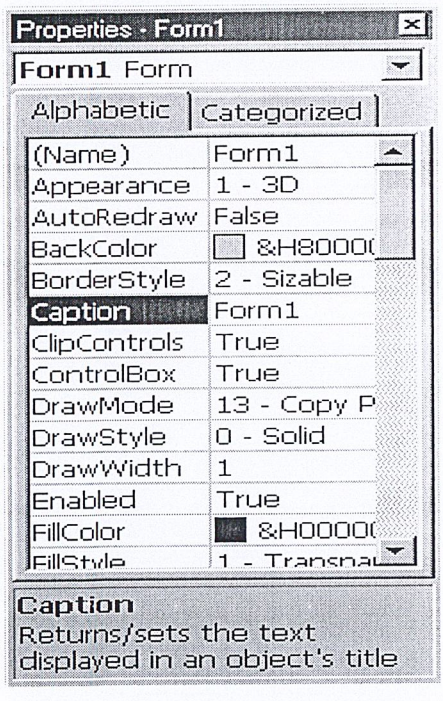
2) เลื่อนเมาส์บนฟอร์มไปยังตำแหน่งที่ต้องการจะวาดออบเจ็กต์

3) กดปุ่มซ้ายค้างไว้แล้วลากจะปรากฏออบเจ็กต์ที่ต้องการบนฟอร์ม

2) การกำหนดคุณสมบัติ (Property) ให้กับยูสเซอร์อินเตอร์เฟส

การวาดยูสเซอร์อินเตอร์เฟสเป็นขั้นตอนแรกสำหรับการเขียนโปรแกรมด้วยวิชชวลเบสิก ส่วนขั้นตอนที่สองก็คือ การกำหนดคุณลักษณะต่างๆให้กับแต่ละออบเจ็กต์บนยูสเซอร์อินเตอร์เฟส

วิชชวลเบสิกจะตั้งค่าคุณสมบัติโดยปริยายให้กับออบเจ็กต์ทุกตัว อย่างอัตโนมัติ แต่อย่างไรก็ตาม ถ้าไม่ต้องการใช้ค่าโดยปริยายเหล่านี้ ก็สามารถปรับปรุงยูสเซอร์อินเตอร์เฟสได้ด้วย การเปลี่ยนแปลงค่าคุณสมบัติของออปเจ็กต์ใหม่ ซึ่งจะแสดงหน้าต่างคุณสมบัติค่าพร็อพเพอตี้ต่างๆดังนี้

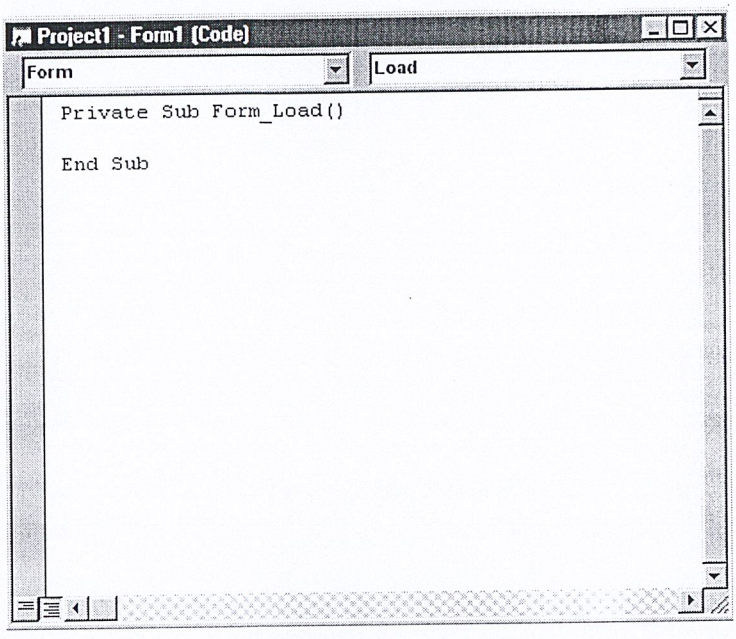


รูปที่ 3.12 หน้าต่างคุณสมบัติแสดงค่าพร็อพเพอตี้ต่างๆ

สำหรับการเขียนโปรแกรมวิชชวลเบสิกนั้น เพื่อให้คอมพิวเตอร์ทำงานต้องใช้คำสั่งที่มันสามารถเข้าใจได้ ซึ่งถ้ามีหลายๆโปรแกรมวิชชวลเบสิกจะเรียกว่าโพรซีเยอร์ (Procedure) ซึ่งจะได้กล่าวต่อไป

3) การเขียนโปรแกรมย่อย (Procedure) ในวิชชวลเบสิก

สิ่งแรกที่ต้องทำก่อนคือ วาดออบเจ็กต์ลงมาบนฟอร์ม ต่อจากนั้นอาจจะเปลี่ยนแปลงคุณสมบัติของแต่ละออบเจ็กต์ เพื่อให้มีชื่อแตกต่างกันและจำได้ง่าย ถ้าไม่ทำการตั้งชื่อใหม่ออปเจ็กต์เหล่านั้นก็จะใช้ชื่อโดยปริยายที่วิชชวลเบสิกตั้งให้ เช่น ออปชั่น 1 (Option 1) หรือเท็กซ์ 3 (text 3) เป็นต้น จากนั้นก็เริ่มเปิดหน้าต่างที่ใช้เขียนโปรแกรมย่อยขึ้นมาดังรูปที่ 3.11 (โดยคลิกบนออบเจ็กต์ที่ต้องการ แล้วกด F7 หรือดับเบิลคลิกบนออบเจ็กต์) ก็เริ่มเขียนโปรแกรมได้แล้ว

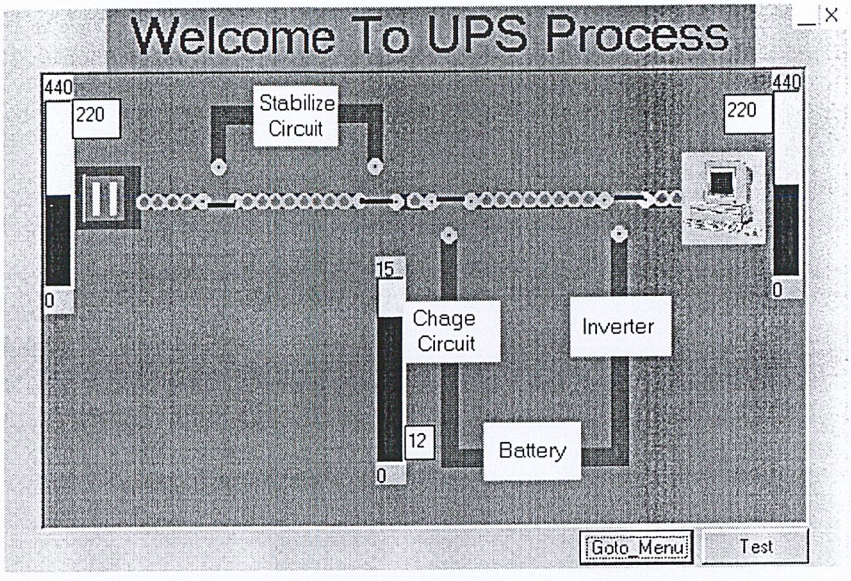


รูปที่ 3.13 หน้าต่างโค้ดที่จะเขียนโปรแกรม

ซึ่งจะสังเกตได้ว่า ในบรรทัดแรกของโปรแกรมย่อย (Procedure) เริ่มด้วย Private Sub ซึ่งแสดงว่าโปรแกรมย่อยนี้ เป็นของออปเจ็กต์ที่ระบุเท่านั้นคือ Form ต่อจากนั้นในระหว่างสองบรรทัดนี้จะต้องทำการเขียนโปรแกรมขึ้นมาเพื่อให้คอมพิวเตอร์รับรู้ บรรทัดสุดท้ายของโพรซีเยอร์ทุกๆตัว จะต้องปิดด้วย End Sub เสมอ โดยจะแสดงว่าสิ้นสุดการทำทั้งหมดของโปรแกรมย่อยนี้แล้ว ซึ่งในหนึ่งหน้าต่างจอภาพใดๆที่ต้องการสร้างนั้นจะมีหลายออปเจ็กต์ ดังนั้นจะมีหลายโพรซีเยอร์ โดยจะต้องเขียนประกอบกันรวมกันเป็นหนึ่งโปรแกรม ซึ่งหลังจากเขียนโพรซีเยอร์เสร็จ คอมพิวเตอร์จะทำงานตามคำสั่งทีละบรรทัดจากบนลงล่าง และจะจบการทำงานเมื่อสิ้นสุดบรรทัดสุดท้าย คือเมื่อพบคำสั่ง End Sub นั่นเอง

ในวิชชวลเบสิก คำสั่งจะทำงานเมื่อมีเหตุการณ์จำเพาะเกิดขึ้นเท่านั้น อย่างเช่น ทุกๆครั้งที่คลิกเมาส์บนออบเจ็กต์ ชุดคำสั่งชุดเดียวกันของเหตุการณ์นี้จะถูกดำเนินการ และโปรแกรมจะสิ้นสุดเมื่อออบเจ็กต์เจาะจงว่าต้องจบการทำงาน

หน้าจอของการแสดงผลทางคอมพิวเตอร์



รูปที่ 3.14 หน้าจอการแสดงผลทางคอมพิวเตอร์

บทที่ 4

การทดลองและผลการทดลอง

ในบทนี้จะกล่าวถึงการทดลองและผลการทดลองของแหล่งจ่ายไฟฟ้าสำรอง ซึ่งได้แยกการทดลองและผลการทดลองออกเป็นส่วนๆ ดังนี้

สำหรับผลการทดลองในโครงงานแหล่งจ่ายไฟฟ้าสำรอง (UPS) นี้ จะแบ่งผลออกเป็น 3 ส่วนคือ

1) ผลการทดลองในส่วนของฮาร์ดแวร์ โดยจะแสดงผลการทดลองและรูปคลื่นสัญญาณในส่วนทั่วๆไปทั้งในสภาวะปกติและผิดปกติ เช่น

1) รูปคลื่นสัญญาณสภาวะของระบบไฟฟ้าปกติ

2) รูปคลื่นสัญญาณสภาวะที่โหลดได้รับพลังงานจากอินเวอร์เตอร์

2) ผลการทดลองในส่วนของไมโครคอนโทรลเลอร์ โดยจะแสดงรูปคลื่นสัญญาณการขับวงจรอินเวอร์เตอร์ทั้งในสภาวะปกติและผิดปกติ เช่น

1) รูปคลื่นสัญญาณที่ใช้ขับวงจรอินเวอร์เตอร์

2) รูปคลื่นสัญญาณที่ใช้ในการอัดประจุแบตเตอรี่

3) ผลการทดลองในส่วนของการแสดงผลทางคอมพิวเตอร์ ซึ่งจะแสดงผลการทดลองการรับค่าอินพุทจากไมโครคอนโทรลเลอร์ แสดงภาพหน้าจอในขณะทำงานทั้งในสภาวะที่ปกติและผิดปกติ เช่น

1) รูปแสดงการไหลของกำลังไฟฟ้าจากระบบไฟฟ้าปกติ

2) รูปแสดงการไหลของกำลังไฟฟ้าจากระบบไฟฟ้าตก-ไฟฟ้าเกิน

3) รูปแสดงการไหลของกำลังไฟฟ้าจากระบบไฟฟ้าดับ

จุดประสงค์ในการทดลอง

1) เพื่อทดลองแหล่งจ่ายไฟฟ้าสำรองต้นแบบที่ทำการสร้างขึ้นมา

2) เพื่อทำการจ่ายภาระได้ในสภาวะปกติและผิดปกติ

3) เปรียบเทียบผลที่เกิดขึ้นกับความเป็นจริง

4) เพื่อรวบรวมผลการทดลองเพื่อทำการปรับปรุงต่อไป

อุปกรณ์ที่ใช้ในการทดลอง

1) แหล่งจ่ายไฟฟ้าแบบปรับค่าได้ (Variac) ขนาด 5A 0-250 V

2) แหล่งจ่ายไฟฟ้าสำรอง (UPS)

3) ภาระเป็นหลอดไฟแบบหลอดไส้ 100 วัตต์ จำนวน 10 หลอด

4) แอมป์มิเตอร์ ขนาด 3Aac

5) โวลต์มิเตอร์ 0-250 Vac

6) ออสซิลโลสโคป

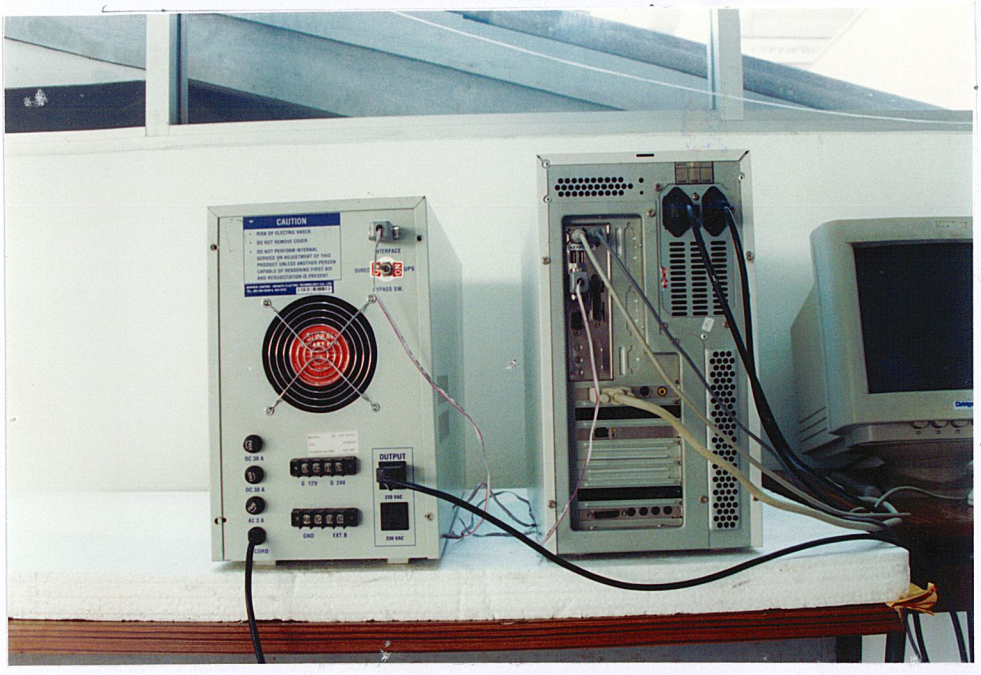
7) คอมพิวเตอร์ เพื่อทำการแสดงผล

4.1 ผลการทดลองในส่วนของฮาร์ดแวร์

4.1.1 แหล่งจ่ายไฟฟ้าสำรองต้นแบบที่สร้างขึ้น

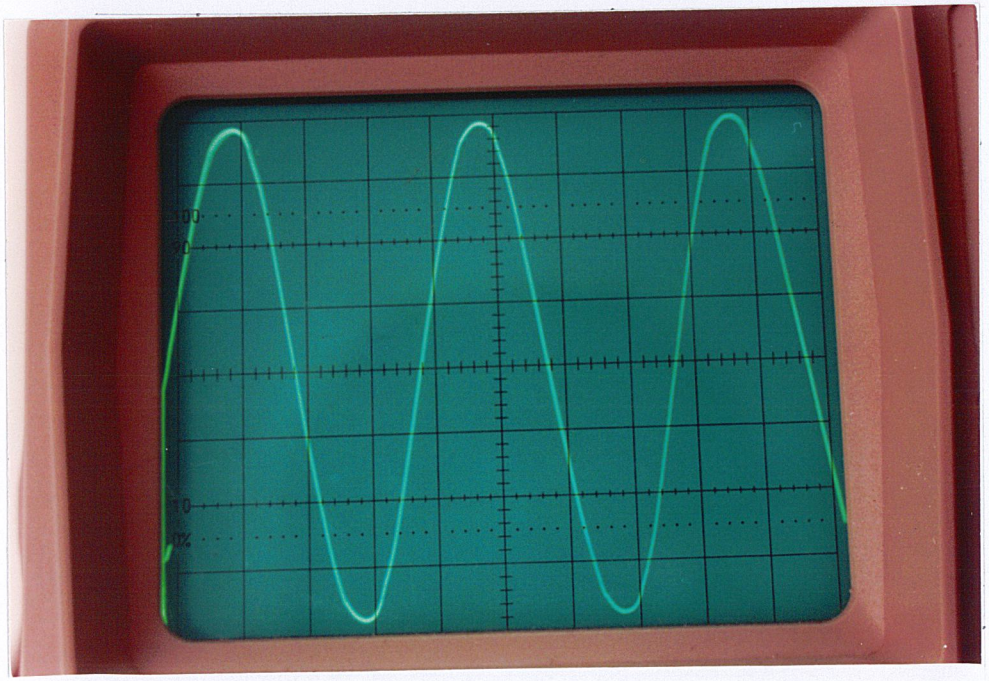
รูปที่ 4.1 แหล่งจ่ายไฟฟ้าสำรองต้นแบบ

4.1.2 รูปแบบการต่อแหล่งจ่ายไฟฟ้าสำรองในการทดลอง



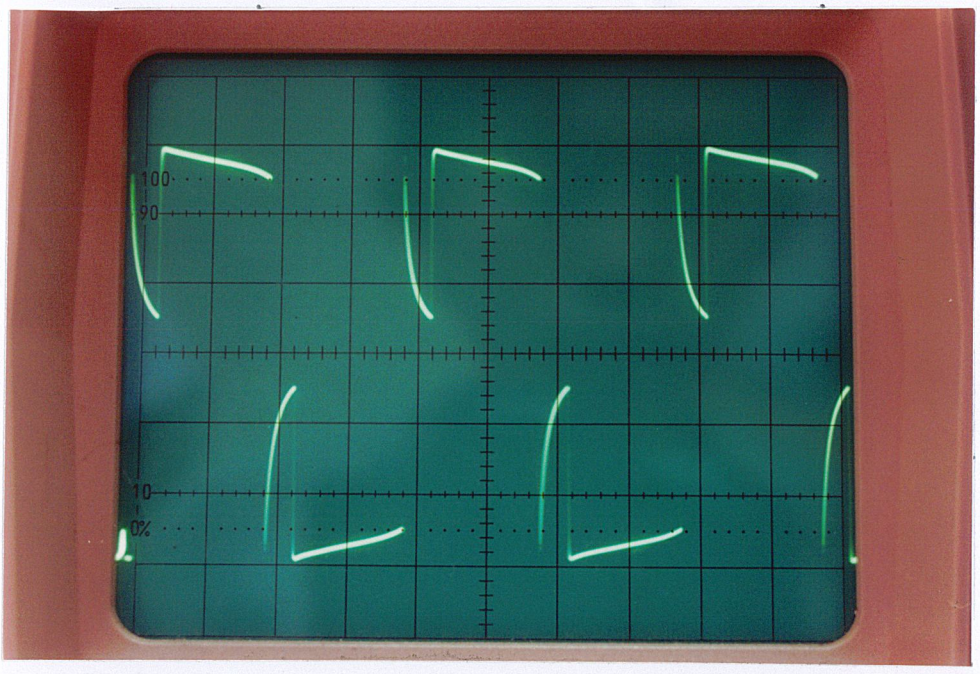
รูปที่ 4.2 การต่อแหล่งจ่ายไฟฟ้าสำรองในการทดลอง

4.1.3 ผลการทดลองแสดงสภาวะปรกติของระบบไฟฟ้า



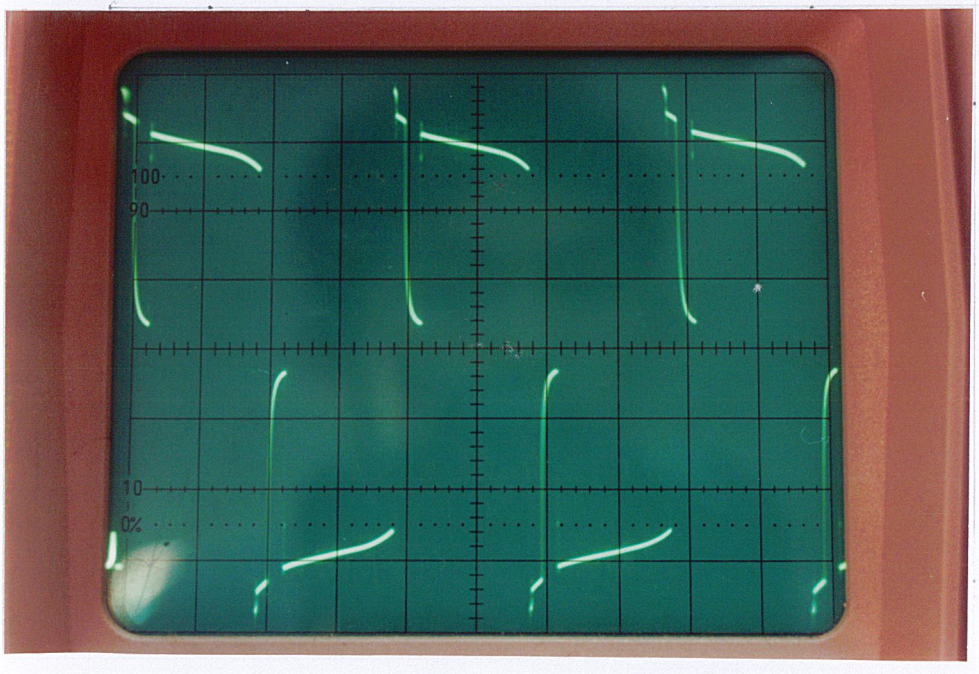
รูปที่ 4.3 รูปคลื่นสัญญาณสภาวะปรกติของระบบไฟฟ้า

4.1.4 ผลการทดลองในสภาวะที่โหลดได้รับกำลังงานไฟฟ้าจากอินเวอร์เตอร์



รูปที่ 4.4 รูปคลื่นสัญญาณสภาวะที่โหลดได้รับกำลังงานไฟฟ้าจากอินเวอร์เตอร์

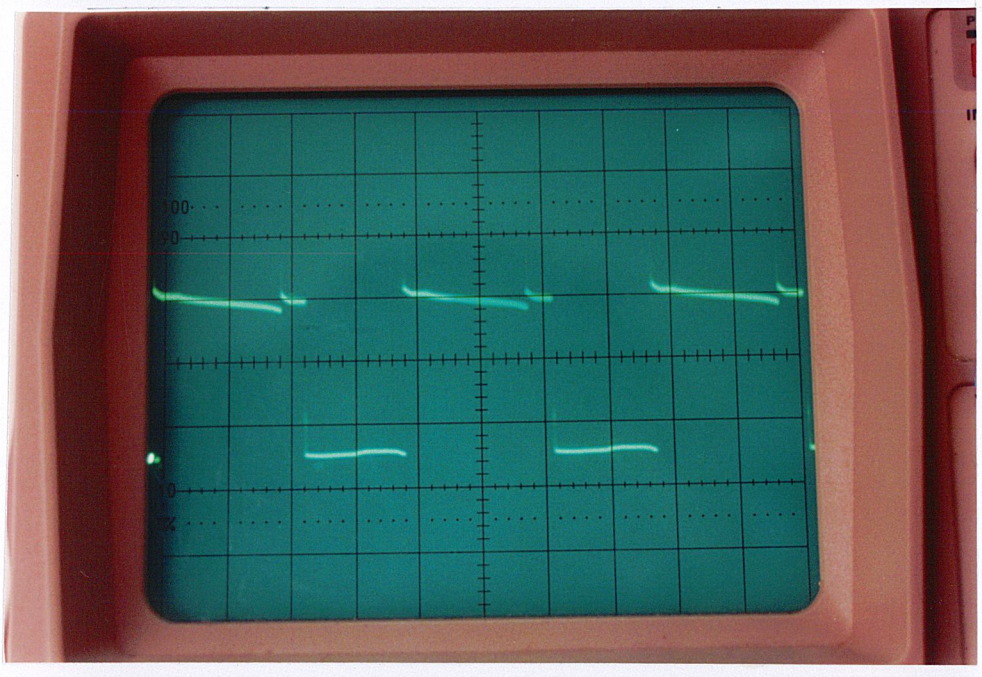
4.1.5 ผลการทดลองแสดงรูปคลื่นสัญญาณจากอินเวอร์เตอร์ขณะไม่มีโหลด



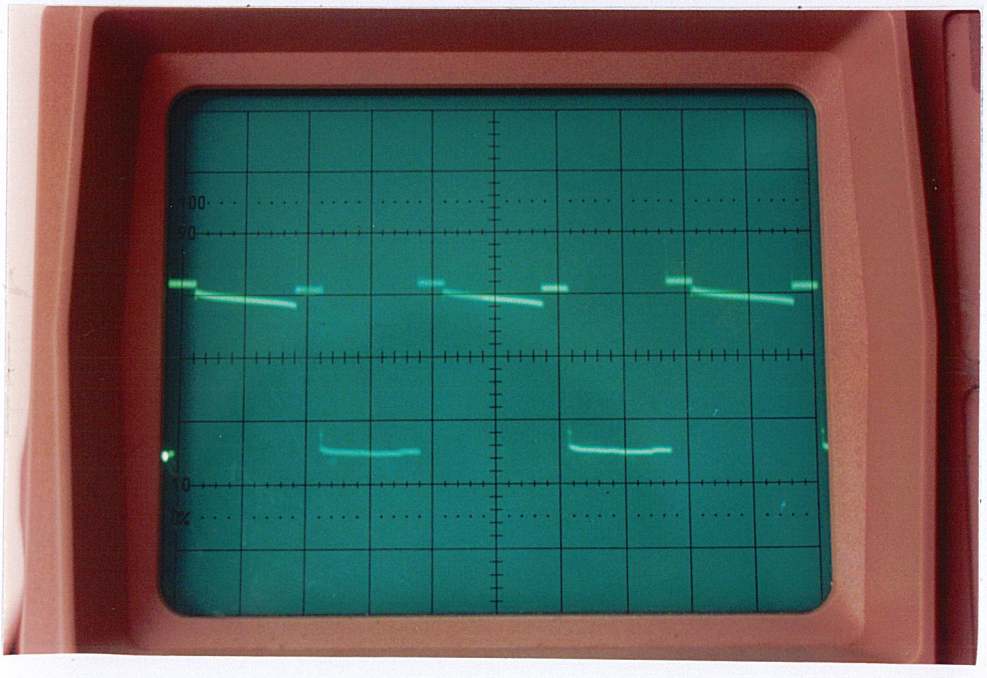
รูปที่ 4.5 รูปคลื่นสัญญาณจากอินเวอร์เตอร์ขณะไม่มีโหลด

4.2 ผลการทดลองในส่วนของไมโครคอนโทรลเลอร์

4.2.1 ผลการทดลองรูปคลื่นจากไมโครคอนโทรลเลอร์ที่ใช้ในการขับวงจรอินเวอร์เตอร์

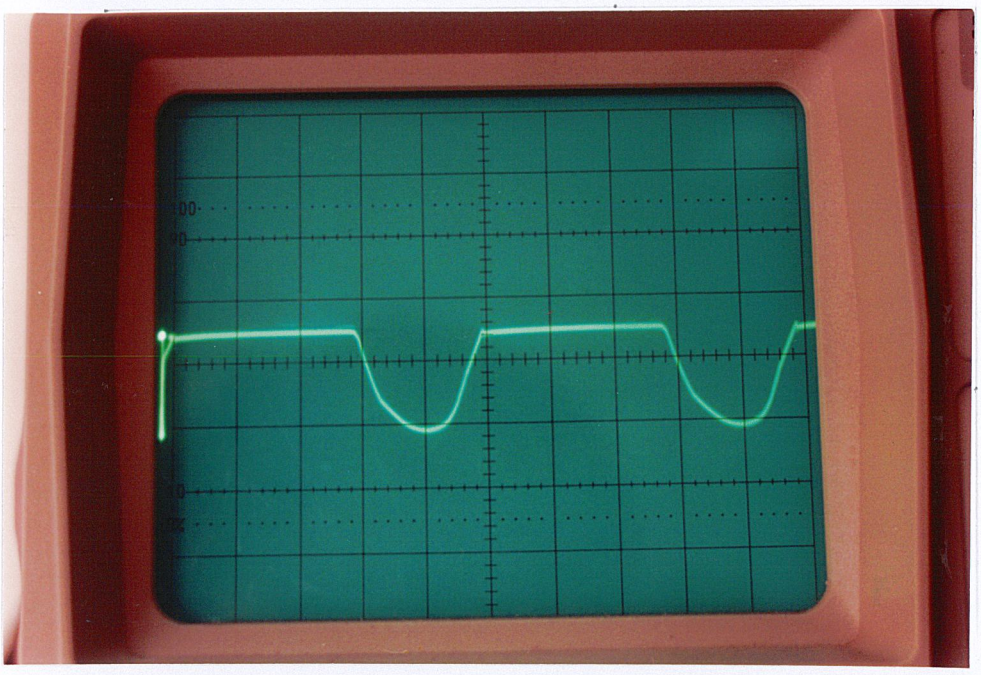


รูปที่ 4.6 รูปคลื่นสัญญาณจากไมโครคอนโทรลเลอร์ที่ใช้ขับวงจรอินเวอร์เตอร์ขณะไม่มีโหลด



รูปที่ 4.7 รูปคลื่นสัญญาณจากไมโครคอนโทรลเลอร์ที่ใช้ขับวงจรอินเวอร์เตอร์ขณะมีโหลด

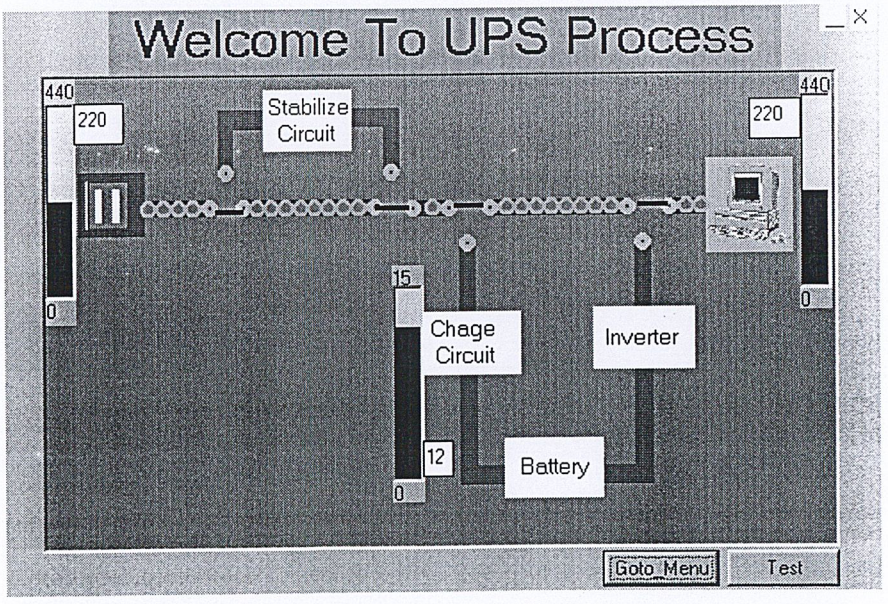
4.2.2 ผลการทดลองรูปคลื่นจากไมโครคอนโทรลเลอร์ที่ใช้ในการอัดประจุแบตเตอรี่



รูปที่ 4.8 รูปคลื่นสัญญาณกระแสอัดประจุแบตเตอรี่

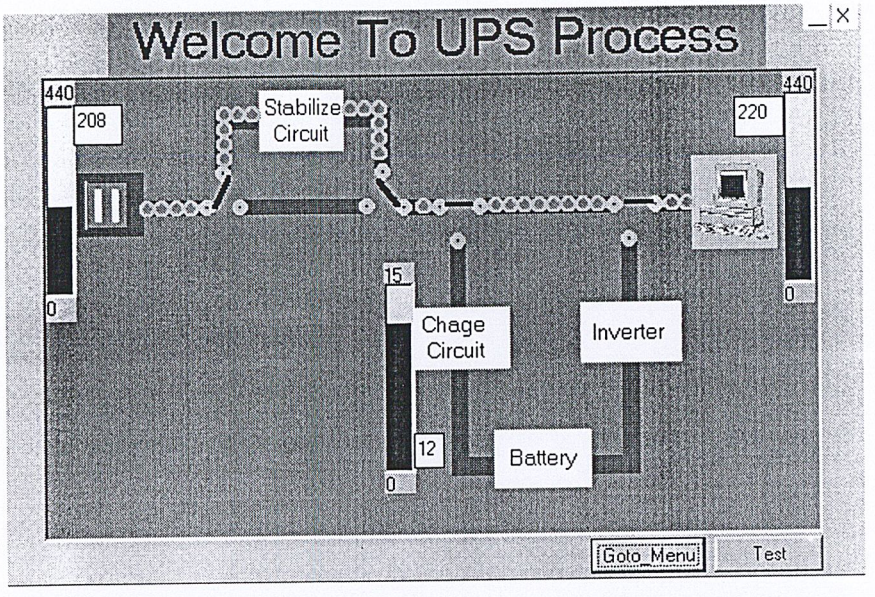
4.3 ผลการทดลองในส่วนการแสดงผล

4.3.1 ผลการทดลองการแสดงผลขณะสภาวะไฟฟ้าระบบปกติ



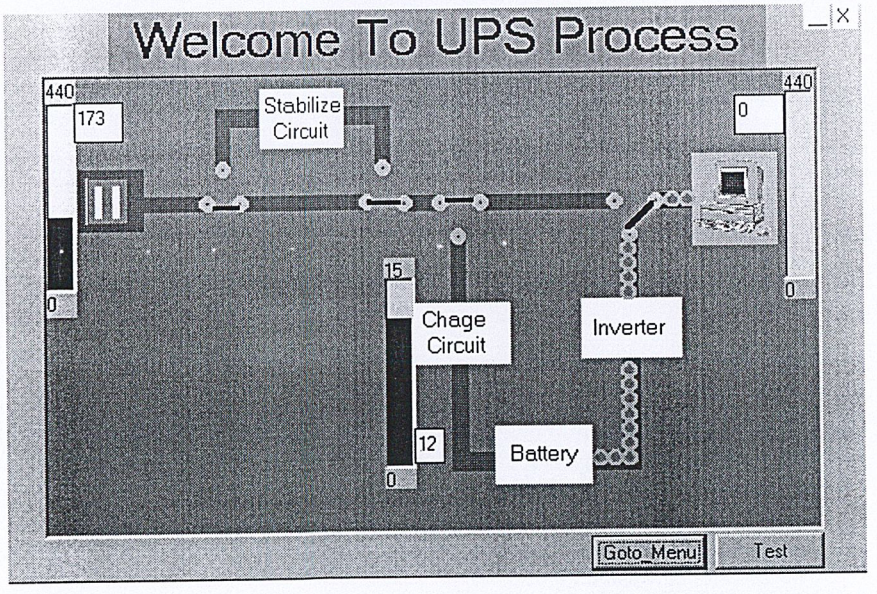
รูปที่ 4.9 การแสดงผลขณะสภาวะไฟฟ้าระบบปกติ

4.3.2 ผลการทดลองการแสดงผลขณะสภาวะไฟฟ้าผิดปกติ(ไฟฟ้าเกิน-ไฟฟ้าตก)



รูปที่ 4.10 การแสดงผลขณะสภาวะไฟฟ้าผิดปกติ(ไฟฟ้าเกิน-ไฟฟ้าตก)

4.3.3 ผลการทดลองการแสดงผลขณะสภาวะไฟฟ้าผิดปกติ(ไฟฟ้าดับ)



รูปที่ 4.11 การแสดงผลขณะสภาวะไฟฟ้าผิดปกติ(ไฟฟ้าดับ)

บทที่ 5
บทสรุป ปัญหา และแนวทางการแก้ไข

5.1 บทสรุป

แหล่งจ่ายไฟฟ้าสำรองที่สร้างขึ้นเพื่อศึกษาหลักการทำงานของแหล่งจ่ายไฟฟ้าสำรองนี้ อาจกล่าวได้ว่าจุดเด่นของโครงงานนี้คือการควบคุมการทำงานด้วยไมโครคอนโทรลเลอร์ และมีการแสดงผลเป็นกราฟฟิกบนคอมพิวเตอร์ ส่วนในด้านฮาร์ดแวร์จะไม่ค่อยมีอะไรเป็นจุดเด่นนัก แต่ทุกอย่างเมื่อผสมผสานเข้าด้วยกันแล้ว ผลที่ออกมาเป็นที่น่าพอใจ

5.2 ปัญหาและอุปสรรค

หลังจากที่ได้รับการสร้างเครื่องต้นแบบและทดลองต่อกับภาระค่าต่างๆ พบว่าการรักษาระดับแรงดันเอาท์พุตยังมีปัญหา โดยระดับแรงดันจะไม่คงที่เท่าที่ควรเหมือนเช่นในตอนไร้ภาระ เนื่องจากมีแรงดันตกคร่อมมากเกินไป เมื่อไมโครคอนโทรลเลอร์ทำการปรับแรงดันขึ้นอัตโนมัติโดยการขยายพัลส์ ค่าแรงดันที่ได้จะไม่เท่ากับแรงดันในตอนแรกดังที่กล่าวมาแล้ว ซึ่งเป็นเหตุให้กำลังเอาท์พุทไม่สามารถจ่ายได้ที่กำลังพิกัด

5.2.1 วงจรอินเวอร์เตอร์

ในส่วนนี้สามารถทำการแปลงไฟฟ้ากระแสตรงเป็นไฟฟ้ากระแสสลับได้ดี ซึ่งจะมีการผิดพลาดของรูปคลื่นเล็กน้อยโดยจะไม่เป็นรูปคลื่นสี่เหลี่ยมสักทีเดียวนัก สำหรับช่วงเวลาการเปิด (t_{on}) และเวลาปิด (t_{off}) ของรูปคลื่นที่ได้นั้น มีค่าน้อยมากประมาณ microsecond (μsec) สำหรับรูปคลื่นแรงดันเอาท์พุทจะมีค่าเป็น ideal ก็ในสภาวะที่มีภาระเพิ่มขึ้น เป็นเพราะว่าในตอนแรก อิมพีแดนซ์ส่วนใหญ่จะเป็นอินดัคตีฟ รีแอกแตนซ์ (Inductive Reactance) และ มีค่าความต้านทานภาระเล็กน้อย ดังนั้นค่าเพาเวอร์แฟคเตอร์ต่ำ รูปคลื่นจะมีกระแสในส่วนหลักน้อยกว่ากระแสทั้งหมด ทำให้รูปคลื่นไม่เป็นสี่เหลี่ยมเท่าที่ควร แต่เมื่อทำการเพิ่มภาระเข้าไป จะเป็นการเพิ่มค่าความต้านทานในวงจรมากขึ้น ภาระจะดึงกระแสจากอินเวอร์เตอร์มากขึ้น เพาเวอร์แฟคเตอร์ก็จะสูงขึ้น รูปคลื่นก็จะเป็นสี่เหลี่ยมมากขึ้น แต่ผลก็ไม่ได้แตกต่างกันมากนัก สำหรับการออกแบบเป็นวงจรพุช – พูล ซึ่งจะมีข้อเสียในเรื่องแรงดันตกคร่อมตัวมอสเฟทจะสูงหากเปรียบเทียบกับวงจรสวิทชิ่งแบบอื่น เช่น ฮาล์ฟ – บริดจ์ หรือ ฟูล – บริดจ์ ทำให้อุปกรณ์ทนแรงดันกระชากได้ต่ำในการต่อกับภาระที่มีความเหนี่ยวนำสูง และมีการเปลี่ยนแปลงของกระแสปริมาณกระแสมากเกิดขึ้นในช่วงเวลาทันทีทันใด

จะทำให้ตัวเหนี่ยวนำสร้างแรงดันสูงออกมาเพื่อคายพลังงานที่สะสมอยู่อาจมีผลทำให้มอสเฟทเสียหายได้ อย่างไรก็ดี วงจรสวิตชิ่งแบบพุช – พูล ยังมีข้อดีในเรื่องการออกแบบที่ง่าย และวงจรขับมอสเฟทใช้อุปกรณ์ที่น้อยกว่าชนิดอื่นๆในการพัฒนาต่อไปเพื่อที่จะหลีกเลี่ยงปัญหาที่อาจจะเกิดขึ้นจากแรงดันสูงที่ตัวเหนี่ยวนำสร้างขึ้นควรจะเลือกใช้วงจรสวิทชิ่งแบบอื่น ซึ่งจะทำให้พิกัดรวมทั้งความทนทานต่อสภาวะผิดปกติของวงจรเพิ่มขึ้น

5.2.2 วงจรอัดประจุแบตเตอรี่

สามารถใช้งานได้ดีในสภาวะที่แรงดันของแบตเตอรี่ไม่ต่ำจนเกินไปนัก ทั้งนี้เนื่องจากวงจรถูกออกแบบให้ประจุแบตเตอรี่ด้วยแรงดันคงที่เท่านั้น ในภาวะที่แรงดันของแบตเตอรี่ต่ำมากๆจะทำให้วงจรจ่ายกระแสสูงโดยไม่สามารถจำกัดกระแสอัดประจุได้ ซึ่งจะเป็นอันตรายต่อวงจรอัดประจุได้ แต่ถ้าหากเกิดเหตุการณ์ดังที่กล่าวมา การออกแบบวงจรอัดประจุควรคำนึงถึงการกำจัดกระแส หรือให้เครื่องมีการอัดประจุด้วยกระแสคงที่เพื่อยืดอายุการใช้งานของแบตเตอรี่

5.2.3 วงจรสวิตช์สับเปลี่ยน

เมื่อทำการทดสอบในสภาวะปกติและผิดปกติที่ภาระค่าต่างๆ ผลปรากฏว่าการทำตามเงื่อนไขต่างๆทำได้ค่อนข้างแม่นยำ เนื่องจากใช้รีเลย์เป็นตัวสับเปลี่ยน (Mechanical Switching)จึงไม่ค่อยมีปัญหาเหมือนเช่น กับการใช้อุปกรณ์อิเล็กทรอนิกส์ (Solid state Switching) ซึ่งจากการทำงานที่ถูกต้องเป็นผลมาจากการควบคุมที่แม่นยำด้วยไมโครคอนโทรลเลอร์ ทั้งรวดเร็ว เวลาในการสั่งการควบคุมใช้น้อยมาก ดังนั้นเป็นผลให้สวิตช์สับเปลี่ยนทำงานถูกต้อง

5.2.4 วงจรไมโครคอนโทรลเลอร์

ในส่วนนี้การทำงานได้สมบูรณ์ในทุกด้านทั้งเป็นการกำเนิดรูปคลื่นสัญญาณสี่เหลี่ยมเพื่อขับมอสเฟท การรับสัญญาณตรวจสอบจากอุปกรณ์ตรวจสอบต่างๆ รวมทั้งการสร้างสัญญาณควบคุมในส่วนต่างๆของฮาร์ดแวร์ เช่น ควบคุมการสับเปลี่ยนของสวิตช์สับเปลี่ยนในการทำงานที่สภาวะต่างๆ และที่สำคัญการทำงานที่รวดเร็วของตัวมันเอง เป็นผลให้การทำงานในด้านฮาร์ดแวร์และส่วนติดต่อกับคอมพิวเตอร์เพื่อแสดงผลก็ไม่มีปัญหา ซึ่งหากเปรียบเทียบกับการควบคุมโดยใช้วงจรอนาล็อก ก็อาจจะพบปัญหามากกว่านี้แน่นอน นี่เป็นข้อได้เปรียบอย่างมากเมื่อเทียบกับการใช้วงจรอนาล็อก สำหรับการส่งข้อมูลเป็นแบบอนุกรมให้กับคอมพิวเตอร์ ค่าที่ส่งนั้นผิดพลาดเพียงเล็กน้อยเท่านั้น ซึ่งไม่มีปัญหาในส่วนนี้

5.2.5 การแสดงผลบนคอมพิวเตอร์

ซึ่งส่วนนี้ทางกลุ่มผู้จัดทำได้ร่วมกันคิด ออกแบบ ในส่วนต่างๆ เช่น ส่วนหน้าจอแสดงผล (User Interface) การแสดงค่าพารามิเตอร์ให้มีการแสดงหลายๆค่าให้มากที่สุด เพื่อที่จะให้ผู้ใช้ได้ทราบสภาวะนั้นโดยพิจารณาจากค่าที่แสดงขึ้น เพื่อจะได้ทำการป้องกันไว้ก่อนหากเกิด

ผิดปกติขึ้น ดังนั้นค่าต่างๆที่ทำการส่งมาจากไมโครฯ จะทำการเข้าสูตรที่เป็นไปได้มากที่สุด สอดคล้องกับการออกแบบที่ดึงดูดผู้ใช้งาน ดังนั้นอาจกล่าวได้ว่า ในส่วนนี้เป็นการพัฒนาจากระบบแบบเดิมมาก ซึ่งการแสดงผลจะเป็นแบบกราฟฟิก ซึ่งโดยทั่วๆไปยังเป็นแบบมาตรวัดธรรมดาอยู่หรืออนาล็อก

5.3 แนวทางการพัฒนา

การปรับปรุงในอนาคตนั้นก็ควรจะมีการพัฒนารูปแบบแสดงผลให้หลากหลายมากขึ้น เช่น แสดงแบบกราฟ หรือเก็บเป็นข้อมูลเข้าหน่วยความจำ เพื่อต้องการที่จะดูผลเมื่อต้องการที่จะทราบสถิติที่เกิดขึ้น นี่คือแนวทางที่จะพัฒนาต่อไป

ดังนั้นสุดท้ายนี้ ข้อเสนอแนะจะเป็นเรื่องการทำงานที่มีแบบแผน มีจุดประสงค์ โดยการทำงานจะต้องแบ่งส่วนการทำงานออกให้ชัดเจน ยกตัวอย่างในโครงงานนี้จะแบ่งเป็น 5 ส่วนใหญ่ๆ คือ อินเวอร์เตอร์ อัดประจุแบตเตอรี่ สวิตซ์สับเปลี่ยน ไมโครคอนโทรลเลอร์ และการแสดงผลบนคอมพิวเตอร์ ซึ่งต้องแบ่งงานกันเพื่อศึกษาในแต่ละส่วน แล้วรวมความคิด เพื่อพิจารณาว่าแนวทางใดเป็นแนวทางที่เหมาะสมที่สุดแล้วจึงออกแบบ โดยมอบความรับผิดชอบสูงสุดให้กับบุคคลนั้นในการที่จะทำงานชิ้นนั้นให้สำเร็จโดยบุคคลที่เหลือคอยช่วยเหลือ ช่วยทำ เพื่อความสำเร็จของงาน ผลงานที่ออกมาบวกกับความมุ่งมั่น ความตั้งใจ จึงมีผลงานชิ้นนี้ในการนำเสนอในครั้งนี้

ภาคผนวก ก
โปรแกรมไมโครคอนโทรลเลอร์

```
;************************************************
;Program          : UPS  By MCS-51
;Description            : MAIN CONTROL
;Filename         : UCU_ORG.ASM
;                 DESING By DORATOM  12/10/43
;************************************************
;-------------- Define Port & Pin Name ----------
LEDX1       BIT         P2.7
LEDX2 BIT         P2.6
LEDR  BIT         P2.5
LEDG        BIT         P2.4
LED1  BIT         P2.3
LED2        BIT         P2.2
LED3  BIT         P2.1
LED4        BIT         P2.0

SW1         BIT         P0.0
SW2         BIT         P0.1

SCL         BIT         P1.1
SDA         BIT         P1.0

RELAY1      BIT         P1.6
RELAY2      BIT         P1.5
RELAY3      BIT         P1.4
RELAY4      BIT         P1.3
RELAY5      BIT         P1.2
SPK         BIT         P1.7

INVERT1     BIT         P0.2
INVERT2     BIT         P0.3
ONEWIRE     BIT         P0.4
CHARGE      BIT         P0.5
;-------------- Define User Register -----------
FLAG        EQU         02EH
I2C_ACK     BIT         FLAG.0
FLAG1 EQU         02FH
BUSY        BIT         FLAG1.0
BBTT        BIT         FLAG1.1
I2C_ADDR    EQU         030H
I2C_DATA    EQU         031H
CONTROL     EQU         032H
DA_DATA     EQU         033H
CHANNEL     EQU         034H
AD_DATA     EQU         035H
ONEWIRE_DATA EQU  036H
TEMP        EQU         037H
BUFFER      EQU         038H
BUFFER1     EQU         039H
```

```
BUFFER2     EQU         03AH
BUFFER3     EQU         03BH
TEMP1 EQU         03EH
DATA_RX     EQU         040H
DATA_TX     EQU         041H
ID1         EQU         10010000B
ID2         EQU         10010010B
;---------------- Main Program -----------------
            ORG   0000H
            SJMP SETUP
;------------------------------------------------
      ORG 0023H   ; RX INTERRUPT ADDRESS
                  JNB RI,$
                  MOV A,SBUF
                  CLR RI
                  MOV DATA_RX,A
                  RETI
;------------------------------------------------
SETUP:            MOV   P0,#00001011B
                  MOV   P1,#00000011B
                  MOV P2,#00000000B
                  MOV   P3,#00000011B
                  MOV DA_DATA,#000H
                  MOV TMOD,#20H     ; SET TIMER1 TO 8
                                    ; BIT AUTO RELOAD
                  MOV TH1,#0FDH     ; 9600 BPS TIMER1
                  SETB TR1          ; ON TIMER1
                  MOV IE,#10010000B ;ENABLE INTERRUPT
                                     ;(EA),SERIAL PORT
                                     ;INTERRUPT(ES)
                  MOV SCON,#01011000B ;MODE 1,RECIVE
                                       ;  ENABLE
                  MOV R2,#02H
                  MOV R1,#00H
                  MOV 7FH,#12

MAIN:             ACALL POW_ON            ; CHECK KEY POWER
                                          ;ON IN 4 SEC
;----------------  POWER ON NOW ----------------
CK_LOOP:          ACALL AD_READ
                  ACALL CHK_VIN
                  ACALL CHK_BAT
                  CJNE R2,#01H,NEXT
                  CLR LEDG
                  SETB LEDR
                  SETB P0.7
                  ACALL BEEP
                  AJMP NEXT1
NEXT:       CLR P0.7
                  CLR LEDR
```

```
                    SETB LEDG
                    MOV R1,#00H
NEXT1:              ACALL TX_DATA
                    MOV 0,#5
CHK_KEYX1:  JNB   SW2,CHK_LPX1
                    MOV  0,#5
                    AJMP CK_LOOP
CHK_LPX1:   ACALL DELAY_500MS
                    DJNZ R0,CHK_KEYX1
                    CLR LEDG
                    SETB SPK
                    ACALL DELAY_10MS
                    CLR SPK
                    AJMP SETUP
;---------------- CHECK VOLTAGE INPUT -----------
CHK_VIN:
V_MAX:              MOV A,BUFFER
                    MOV R0,#08fH
                    SUBB A,R0
                    JC v_maX1
                    SETB RELAY5
                    CLR RELAY4
                    CLR RELAY3
                    CLR RELAY2
                    MOV R2,#01H
                    RET
v_max1:             MOV A,BUFFER
                    MOV R0,#080H
                    SUBB A,R0
                    JC NORMAL
                    SETB RELAY5
                    SETB RELAY4
                    SETB RELAY3
                    SETB RELAY2
                    MOV R2,#02H
                    RET
NORMAL:             MOV A,BUFFER
                    MOV R0,#06FH
                    SUBB A,R0
                    JC V_LOW1
                    SETB RELAY5
                    CLR RELAY4
                    SETB RELAY3
                    SETB RELAY2
                    MOV R2,#02H
                    RET
v_low1:             MOV A,BUFFER
                    MOV R0,#055H
                    SUBB A,R0
                    JC V_LOW
```

```
                    CLR RELAY5
                    CLR RELAY4
                    SETB RELAY3
                    SETB RELAY2
                    MOV R2,#02H
                    RET
V_LOW:          SETB RELAY5
                    CLR RELAY4
                    CLR RELAY3
                    CLR RELAY2
                    MOV R2,#01H
                    RET
;------------------ CHECK BATTERY ---------------
CHK_BAT:
B_HIGH:       MOV A,BUFFER3
                    MOV R0,#0BFH
                    SUBB A,R0
                    JC B_MID
                    SETB LEDX1
              CLR LEDX2
              MOV 7FH,#12
              RET
B_MID:              MOV A,BUFFER3
                    MOV R0,#07FH
                    SUBB A,R0
                    JC B_LOW
                    CLR LEDX1
              SETB LEDX2
              MOV 7FH,#7
                    RET
B_LOW:              SETB LEDX2
                    SETB LEDX1
                    MOV 7FH,#1
                    RET
;---------------------- BEEP --------------------
BEEP:         CJNE R1,#00H,BEEP1
                    SETB SPK
                    SETB LED1
                    ACALL DELAY_100MS
                    CLR SPK
                    CLR LED1
                    MOV R1,7FH
                    RET
BEEP1:              DEC R1
                    RET
;------------------- READ ANALOG ---------------
AD_READ:            MOV CHANNEL,#000H
                    MOV R0,#BUFFER
                    MOV R4,#4
```

```
CONVERSION_LOOP: MOV A,CHANNEL
                 ADD A,#01000000B
                 MOV CONTROL,A
                 ACALL PCF8591_WR
                 ACALL PCF8591_RD
                 ACALL PCF8591_RD
                 MOV @R0,AD_DATA
                 INC R0
                 INC CHANNEL
                 DJNZ R4,CONVERSION_LOOP
                 RET
;--------------------- READ TEMP ----------------
TEMP_READ:       ACALL DS1820_RST   ; DS1820 RESET
                 ACALL DS1820_PRES  ; DS1820 PRESENCE
                 MOV ONEWIRE_DATA,#0CCH
                 ACALL DS1820_WR
                 MOV ONEWIRE_DATA,#044H;WRITE CONVERT COMMAND
                 ACALL DS1820_WR
                 SETB BUSY          ;SET BIT BUSY
PRES_CHK_LOOP:   ACALL DS1820_RST   ; DS1820 RESET
                 ACALL DS1820_PRES  ; DS1820 PRESENCE
                 JB BUSY,PRES_CHK_LOOP
                 NOP
                 NOP
                 NOP
                 NOP
                 ACALL DS1820_RST     ; DS1820 RESET
                 ACALL DS1820_PRES    ; DS1820 PRESENCE
                 MOV ONEWIRE_DATA,#0CCH
                 ACALL DS1820_WR
                 MOV ONEWIRE_DATA,#0BEH ; WRITE READ
                                        ;SCRATCHPAD COMMAND
                 ACALL DS1820_WR
                 ACALL DS1820_RD         ; READ DS1820
                 MOV TEMP,ONEWIRE_DATA   ;GET FIRST
                                          ;BYTE AS TEMP (L)
                 ACALL DS1820_RST   ; DS1820 RESET
                 ACALL DS1820_PRES  ; DS1820 PRESENCE
                 MOV A,TEMP
                 CLR C
                 RRC     A
DIGIT1:          MOV     B,#100
                 DIV     AB
                 MOV     A,B
                 MOV     B,#10
                 DIV     AB
                 MOV     TEMP1,A
                 RET
```

```
;-----------------------------------------------
TX_DATA:            CLR EA
                    MOV R4,#04H
                    MOV R0,#38H
TX_DATA1:           MOV A,@R0
                    MOV A,BUFFER2
                    CLR TI
                    MOV SBUF,A
                    JNB TI,$
                    CLR TI
                    INC R0
                    DJNZ R4,TX_DATA1
                    SETB EA
                    RET
;-----------------------------------------------
POW_ON:             MOV 0,#5
CHK_KEY:            JNB  SW2,CHK_LP
                    MOV  0,#5
                    AJMP CHK_KEY
CHK_LP:             ACALL DELAY_500MS
                    DJNZ R0,CHK_KEY
                    SETB LEDG
                    SETB SPK
                    ACALL DELAY_10MS
                    CLR SPK
                    RET
;-----------------------------------------------
POW_OFF:            MOV 0,#5
CHK_KEY1:           JNB  SW2,CHK_LP1
                    MOV  0,#5
                    AJMP CHK_KEY1
CHK_LP1:            ACALL DELAY_500MS
                    DJNZ R0,CHK_KEY1
                    CLR LEDG
                    SETB SPK
                    ACALL DELAY_10MS
                    CLR SPK
                    RET
;--------------- DS1820 DATA READ --------------
DS1820_RD:          MOV     R4,#8
                    CLR     A
DS1820_RD_LOOP:CLR ONEWIRE
                    NOP
                    NOP
                    SETB    ONEWIRE
                    NOP
                    NOP
                    NOP
                    NOP
                    MOV     C,ONEWIRE
```

```
                ACALL   ONEWIRE_DELAY
                RRC     A
                DJNZ    R4,DS1820_RD_LOOP
                MOV     ONEWIRE_DATA,A
                RET
-------------- DS1820 DATA WRITE --------------
DS1820_WR:      MOV     R4,#8
                MOV     A,ONEWIRE_DATA

DS1820_WR_LOOP: RRC     A
                JNC     DS1820_WR_L
                CLR     ONEWIRE
                NOP
                NOP
                NOP
                NOP
                SETB    ONEWIRE
                ACALL   ONEWIRE_DELAY
                AJMP    DS1820_WR_NX
DS1820_WR_L:    CLR     ONEWIRE
                ACALL   ONEWIRE_DELAY
                SETB    ONEWIRE
                NOP
                NOP
                NOP
                NOP
DS1820_WR_NX:   DJNZ    R4,DS1820_WR_LOOP
                RET
;----------------- DS1820 RESET----------------
DS1820_RST:     CLR     ONEWIRE
                ACALL   DELAY_1mS
                SETB    ONEWIRE
                MOV     R4,#8
                DJNZ    R4,$
                RET
;-------- DS1820 RECEIVE PRESENCE PULSE --------
DS1820_PRES:    MOV     R4,#8
DS1820_PRES_1:  MOV     R5,#0
DS1820_PRES_2:  JNB     ONEWIRE,DS1820_PRES_3
                DJNZ    R5,DS1820_PRES_2
                DJNZ    R4,DS1820_PRES_1
                RET
DS1820_PRES_3:  JNB     ONEWIRE,$
                MOV     R4,#8
                DJNZ    R4,$
                CLR     BUSY
                RET
;-----------------------------------------------
PCF8591_RD:     MOV I2C_ADDR,#ID1+1
                ACALL I2C_SLAVE
```

```
                    ACALL I2C_DATA_RD
                    MOV AD_DATA,I2C_DATA
                    ACALL I2C_NACK_BIT
                    ACALL I2C_STOP
                    RET
;-----------------------------------------------
PCF8591_WR:         MOV I2C_ADDR,#ID1
                    ACALL I2C_SLAVE
                    MOV I2C_DATA,CONTROL
                    ACALL I2C_DATA_WR
                    MOV I2C_DATA,DA_DATA
                    ACALL I2C_DATA_WR
                    ACALL I2C_STOP
                    RET
;-----------------------------------------------
I2C_DATA_WR:        PUSH ACC
                    SETB I2C_ACK
                    MOV A,I2C_DATA
                    MOV R5,#008
I2C_DATA_WR_1:      RLC A
                    MOV SDA,C
                    ACALL I2C_CLK
                    DJNZ R5,I2C_DATA_WR_1
                    SETB SDA
                    ACALL I2C_DELAY
                    JB SDA,I2C_DATA_WR_2
                    CLR I2C_ACK
I2C_DATA_WR_2:       CLR SCL
                    POP ACC
                    RET
;-----------------------------------------------
I2C_DATA_RD:        PUSH ACC
                    CLR A
                    MOV R5,#008
I2C_DATA_RD_1:      ACALL I2C_DELAY
                    SETB SCL
                    ACALL I2C_DELAY
                    MOV C,SDA
                    RLC A
                    CLR SCL
                    DJNZ R5,I2C_DATA_RD_1
                    MOV I2C_DATA,A
                    POP ACC
                    RET
;-----------------------------------------------
I2C_SLAVE:          PUSH ACC
                    SETB I2C_ACK
                    MOV A,I2C_ADDR
                    ACALL I2C_START
                    MOV R5,#008
```

```
I2C_SLAVE_1:        RLC A
                    MOV SDA,C
                    ACALL I2C_CLK
                    DJNZ R5,I2C_SLAVE_1
                    SETB SDA
                    ACALL I2C_DELAY
                    SETB SCL
                    ACALL I2C_DELAY
                    JB SDA,I2C_SLAVE_2
                    CLR I2C_ACK
I2C_SLAVE_2:        CLR SCL
                    POP ACC
                    RET
;-------------------------------------------------
I2C_START:          SETB SCL
                    SETB SDA
                    ACALL I2C_DELAY
                    CLR SDA
                    ACALL I2C_DELAY
                    CLR SCL
                    RET
;-------------------------------------------------
I2C_STOP:           CLR SDA
                    ACALL I2C_DELAY
                    SETB SCL
                    ACALL I2C_DELAY
                    SETB SDA
                          RET
;-------------------------------------------------
I2C_CLK:            ACALL I2C_DELAY
                    SETB SCL
                    ACALL I2C_DELAY
                    CLR SCL
                    RET
;-------------------------------------------------
I2C_NACK_BIT:       SETB SDA
                    ACALL I2C_DELAY
                    ACALL I2C_CLK
                    RET
;-------------------------------------------------
ONEWIRE_DELAY:      MOV     R6,#012H
ONEWIRE_DELAY1:     NOP
                    NOP
                    DJNZ    R6,ONEWIRE_DELAY1
                    RET
I2C_DELAY:          MOV 6,#00CH
I2C_DELAY_1:        NOP
                    NOP
                    DJNZ R6,I2C_DELAY_1
                    RET
```

```
DELAY_1MS:          MOV 6,#0E6H
DELAY_1MS_1:        NOP
                    NOP
                    DJNZ R6,DELAY_1MS_1
                    RET
DELAY_10ms:         MOV 7,#10
DELAY_10ms_1:       MOV 6,#0E6H
DELAY_10ms_2:       NOP
                    NOP
                    DJNZ R6,DELAY_10ms_2
                    DJNZ R7,DELAY_10ms_1
                    RET
DELAY_100ms:        MOV 7,#100
DELAY_100ms_1:      MOV 6,#0E6H
DELAY_100ms_2:      NOP
                    NOP
                    DJNZ R6,DELAY_100ms_2
                    DJNZ R7,DELAY_100ms_1
                    RET
DELAY_500MS:        MOV 5,#50
DELAY_500MS_1:      ACALL DELAY_10ms
                    DJNZ R5,DELAY_500MS_1
                    RET
DELAY_1S:           MOV 5,#100
DELAY_1S_1:         ACALL DELAY_10ms
                    DJNZ R5,DELAY_1S_1
                    RET
;-------------------------------------------------
END
```

รูปที่ ก.1 โปรแกรมควบคุมของไมโครคอนโทรลเลอร์ MCS51

ภาคผนวก ข

โปรแกรมแสดงผลทางคอมพิวเตอร์

หน้าจอต้อนรับ

ไฟล์ congrat.frm

คำสั่งที่ใช้ในโปรแกรม

```
Option Explicit
 Private Sub Timer1_Timer()
    Formmain.Show
    Unload Me
  End Sub
```

รูปที่ข.1 โปรแกรมต้อนรับ

โปรแกรม จับเวลาชัตดาวน์เครื่องคอมพิวเตอร์เมื่อมีแรงดันแบตเตอรี่ต่ำกว่า 10 โวลท์

ไฟล์ digits.frm

คำสั่งที่ใช้ในโปรแกรม

```
      Option Explicit
      Private Const EWX_SHUTDOWN = 1
      Private Const EWX_REBOOT = 2
      Private Const EWX_LOGOFF = 0
      Private Const EWX_FORCE = 4

      Private Declare Function ExitWindowsEx Lib "User32" _
        (ByVal uFlags As Long, _
        ByVal dwReserved As Long) _
        As Long
      Private Sub CmdOK_Click()
            Me.Hide
      End Sub
      Private Sub Form_Load()
          Dim i As Integer, xFactor As Integer, yFactor As Integer
          Unload Formmain
          Unload MainProgram
          clpDigits.Cols = 11
          clpPunctuation.Cols = 8
          clpDigits = imgDigits(0)
          clpPunctuation = imgPunctuation(0)
          imgClock(0) = clpDigits.GraphicCell(0)
          For i = 1 To 7
           Load imgClock(i)
           imgClock(i).Left = imgClock(i-1).Left+imgClock(i-
1).Width
           imgClock(i).Visible = True
             If i = 2 Or i = 5 Then
               imgClock(i) = clpPunctuation.GraphicCell(0)
             Else
```

```
imgClock(i) = clpDigits.GraphicCell(0)
      End If
      Next i

      For i = 0 To 7
              imgClock(i) = clpDigits.GraphicCell(10)
          Next i
      Call ShowCurrTime
      End Sub
      Private Sub ShowCurrTime()
          Dim i As Integer, buff As String, aChar As String
          Static count As Integer, COmin As Integer, COho As Integer
              If count > 20 Then
                  count = 0
                  COmin = COmin + 1
      'ใช้ในการทดลองเมื่อกำหนดให้เวลาทำงานได้ 20 วินาที
      '           If count > 20 Then
      'จะทำการชัดดาวน์เครื่องเมื่อเกิดไฟดับ

      '            ExitWindowsEx EWX_SHUTDOWN, 0
      '           End If
                       If COmin > 59 Then
                            COmin = 0
                            COho = COho + 1
                            If COho > 23 Then COho = 0
                       End If
              End If
          buff = Format$(COho & ":" & COmin & ":" & count,
"hh:mm:ss")
          For i = 1 To 8
                        aChar = Mid$(buff, i, 1)
                        If aChar = ":" Then
                             imgClock(i - 1) =
clpPunctuation.GraphicCell(2)
                        Else
                            imgClock(i - 1) = clpDigits.GraphicCell
(Asc(aChar) - Asc("0"))
                        End If
              Next i
              count = count + 1
      End Sub
      Private Sub Timer1_Timer()
          Call ShowCurrTime
      End Sub
```

รูปที่ ข.2 โปรแกรมจับเวลาปิดเครื่องคอมพิวเตอร์

ในส่วนของโปรแกรมของตัวโปรแกรมหลักซึ่งจะทำหน้าที่ในการรับค่าข้อมูลเข้าทางพอร์ทอนุกรม

ไฟล์ main.frm

คำสั่งที่ใช้ในโปรแกรม

```
Option Explicit
Private Const EWX_SHUTDOWN = 1
Private Const EWX_REBOOT = 2
Private Const EWX_LOGOFF = 0
Private Const EWX_FORCE = 4
Dim contrl As Boolean
Dim dataErr As Integer
 Dim status As Boolean
Private Declare Function Shell_NotifyIcon Lib "shell32" Alias
"Shell_NotifyIconA" (ByVal dwMessage As Long, pnid As
NOTIFYICONDATA) As Boolean
Private Declare Function SendMessage Lib "User32" _
                        Alias "SendMessageA" (ByVal hWnd As Long,
_
                                              ByVal wMsg As Long,
_
                                              ByVal wParam As
Long, _
                                              lParam As Any) As
Long

Private Declare Sub ReleaseCapture Lib "User32" ()
Private Type NOTIFYICONDATA
    cbSize As Long
    hWnd As Long
    uId As Long
    uFlags As Long
    ucallbackMessage As Long
    hIcon As Long
    szTip As String * 64
End Type

Private Const NIM_ADD = &H0
Private Const NIM_MODIFY = &H1
Private Const NIM_DELETE = &H2
Private Const WM_MOUSEMOVE = &H200
Private Const NIF_MESSAGE = &H1
Private Const NIF_ICON = &H2
Private Const NIF_TIP = &H4


Private Const WM_LBUTTONDBLCLK = &H203
Private Const WM_LBUTTONDOWN = &H201
Private Const WM_LBUTTONUP = &H202
Private Const WM_RBUTTONDBLCLK = &H206
Private Const WM_RBUTTONDOWN = &H204
Private Const WM_RBUTTONUP = &H205
```

```
Private Const WM_NCLBUTTONDOWN = &HA1
Private Const HTCAPTION = 2
Private t As NOTIFYICONDATA
Dim Checkdat As Boolean
Dim formDBclick As Boolean
Dim valuegraph(100) As Integer
Dim Setshutdown As Boolean
Dim SetTimeShut As Double
Dim Imgout_1 As Boolean
Dim Imgsub_1 As Boolean
Dim lbbatt As Boolean
Private Sub CmdMENU_Click()
Formmain.Hide
MainProgram.Visible = True
MainProgram.Caption = "Append New Files  : " & datavar(0)
End Sub
Private Sub CmdTEST_Click()
Dim i As Integer
If contrl = True Then
     contrl = False
      CmdTEST.Caption = "STOP"
      For i = 0 To 3
      Text1(i).Visible = True
       Next i
ElseIf contrl = False Then
       contrl = True
         CmdTEST.Caption = "TEST"
         For i = 0 To 3
         Text1(i).Visible = False
          Next i
End If
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As
Integer)
    t.cbSize = Len(t)
    t.hWnd = Me.hWnd
    t.uId = 1&
    Shell_NotifyIcon NIM_DELETE, t
    Unload MainProgram
End Sub
Private Sub ImgBatt_Click()
If lbbatt = True Then
 lbbatt = False
 picProgress1.Visible = False
 Text3.Visible = False
 LBat(0).Visible = False
 LBat(1).Visible = False
 Else
 lbbatt = True
 picProgress1.Visible = True
 Text3.Visible = True
 LBat(0).Visible = True
 LBat(1).Visible = True
```

```
End If
End Sub
Private Sub LMain_MouseMove(Button As Integer, Shift As Integer, _
                                    x As Single, y As Single)
         Dim lngReturnValue As Long
         If Button = 1 Then
                        Call ReleaseCapture
                         lngReturnValue = SendMessage
(Formmain.hWnd, WM_NCLBUTTONDOWN, _
                         HTCAPTION, 0&)
                    End If
     End Sub

Private Sub ImgOUt_Click()
 If Imgout_1 = True Then
 Imgout_1 = False
 picProgress2.Visible = False
 Text4.Visible = False
 Lload(0).Visible = False
 Lload(1).Visible = False
 Else
 Imgout_1 = True
 picProgress2.Visible = True
 Text4.Visible = True
 Lload(0).Visible = True
 Lload(1).Visible = True
 'If value(1) > 176 And value(1) < 231 Then
 'End If
 End If
  End Sub

Private Sub ImgSup_Click()
       If Imgsub_1 = True Then
       Imgsub_1 = False
       picProgress.Visible = False
       Text2.Visible = False
       LSub(0).Visible = False
       LSub(1).Visible = False
       Else
          Imgsub_1 = True
          picProgress.Visible = True
          Text2.Visible = True
          LSub(0).Visible = True
          LSub(1).Visible = True
       End If
End Sub
 Private Sub imgBatt_MouseMove(Button As Integer, Shift As Integer, _
   x As Single, y As Single)
      Dim lngReturnValue As Long
      If Button = 1 Then
            Call ReleaseCapture
```

```
lngReturnValue = SendMessage(Formmain.hWnd, WM_NCLBUTTONDOWN, _
            HTCAPTION, 0&)
End If
     End Sub
 Private Sub Lclose_Click()
   Unload MainProgram
   If MSComm1.PortOpen = True Then MSComm1.PortOpen = False
  Unload Formmain
  End
 End
End Sub
 Private Sub picProgress_MouseMove(Button As Integer, Shift As
Integer, _
                                  x As Single, y As Single)
         Dim lngReturnValue As Long
         If Button = 1 Then
                        Call ReleaseCapture
                         lngReturnValue = SendMessage
(Formmain.hWnd, WM_NCLBUTTONDOWN, _
                         HTCAPTION, 0&)
                    End If
     End Sub
 Private Sub picProgress1_MouseMove(Button As Integer, Shift As
Integer, _
                                  x As Single, y As Single)
         Dim lngReturnValue As Long
         If Button = 1 Then
                        Call ReleaseCapture
                         lngReturnValue = SendMessage
(Formmain.hWnd, WM_NCLBUTTONDOWN, _
                         HTCAPTION, 0&)
                    End If
End Sub
Private Sub picProgress2_MouseMove(Button As Integer, Shift As
Integer, _
                                  x As Single, y As Single)
         Dim lngReturnValue As Long
         If Button = 1 Then
                        Call ReleaseCapture
                         lngReturnValue = SendMessage
(Formmain.hWnd, WM_NCLBUTTONDOWN, _
                         HTCAPTION, 0&)
                    End If
     End Sub
Private Sub pctmain_MouseMove(Button As Integer, Shift As Integer,
_
                                  x As Single, y As Single)
         Dim lngReturnValue As Long
         If Button = 1 Then
                        Call ReleaseCapture
                         lngReturnValue = SendMessage
(Formmain.hWnd, WM_NCLBUTTONDOWN, HTCAPTION, 0&)
```

```
End If
     End Sub
'strabilizer
Function intTimer1()

Dim cx, cy, i
  value(3) = value(1)

For i = 0 To 60
 shape1(i).Visible = False
Next i
   L_SW1(0).Visible = False
   L_SW1(1).Visible = True
   L_SW2(0).Visible = False
   L_SW2(1).Visible = True
   L_SW3(0).Visible = True
   L_SW3(1).Visible = False
   L_SW4(0).Visible = True
   L_SW4(1).Visible = False
    For i = 0 To 3
     shape1(i).Visible = True
    Next i
    For i = 23 To 34
     shape1(i).Visible = True
    Next i
    For i = 12 To 22
     shape1(i).Visible = True
    Next i
End Function
'2 chage
Function intTimer2()
Dim cx, cy, i
'value(3) = 0
For i = 0 To 60
 shape1(i).Visible = False
Next i
   L_SW1(0).Visible = True
   L_SW1(1).Visible = False
   L_SW2(0).Visible = True
   L_SW2(1).Visible = False
   L_SW3(0).Visible = False
   L_SW3(1).Visible = True
   L_SW4(0).Visible = False
   L_SW4(1).Visible = True
   For i = 0 To 12
     shape1(i).Visible = True
    Next i
  For i = 35 To 60
      shape1(i).Visible = True
    Next i
 End Function
'inverting
Function intTimer3()
Dim cx, cy, i
```

```
    'value(3) = 0
For i = 0 To 60
 shape1(i).Visible = False
Next i
    L_SW1(0).Visible = True
    L_SW1(1).Visible = False
    L_SW2(0).Visible = True
    L_SW2(1).Visible = False
    L_SW3(0).Visible = True
    L_SW3(1).Visible = False
    L_SW4(0).Visible = False
    L_SW4(1).Visible = True
    For i = 48 To 60
     shape1(i).Visible = True
     Next i
     For i = 21 To 22
      shape1(i).Visible = True
     Next i
End Function
'4 bypass
Function intTimer4()
Dim cx, cy, i
value(3) = value(1)
For i = 0 To 60
 shape1(i).Visible = False
Next i
    L_SW1(0).Visible = True
    L_SW1(1).Visible = False
    L_SW2(0).Visible = True
    L_SW2(1).Visible = False
    L_SW3(0).Visible = True
    L_SW3(1).Visible = False
    L_SW4(0).Visible = True
    L_SW4(1).Visible = False
 For i = 0 To 22
     shape1(i).Visible = True
    Next i
  End Function
  Private Sub vBsh2_MouseMove(Button As Integer, Shift As
Integer, _
                                x As Single, y As Single)
         Dim lngReturnValue As Long
         If Button = 1 Then
                     Call ReleaseCapture
                      lngReturnValue = SendMessage
(Formmain.hWnd, WM_NCLBUTTONDOWN, _
                      HTCAPTION, 0&)
                End If
     End Sub
 Private Sub vBsh3_MouseMove(Button As Integer, Shift As Integer,
_
                                x As Single, y As Single)
```

```
Dim lngReturnValue As Long
         If Button = 1 Then
                        Call ReleaseCapture
                         lngReturnValue = SendMessage
(Formmain.hWnd, WM_NCLBUTTONDOWN, _
                        HTCAPTION, 0&)
                  End If
End Sub
Private Sub Form_Load()
Dim i As Integer
Dim datafile As Integer
Dim xFactor As Integer, yFactor As Integer
    clpDigits.Cols = 11
    clpPunctuation.Cols = 8
    clpDigits = imgDigits(0)
    clpPunctuation = imgPunctuation(0)
    imgClock(0) = clpDigits.GraphicCell(0)
    For i = 1 To 7
        Load imgClock(i)
        imgClock(i).Left = imgClock(i - 1).Left + imgClock(i -
1).Width
        imgClock(i).Visible = True
       If i = 2 Or i = 5 Then
            imgClock(i) = clpPunctuation.GraphicCell(0)
        Else
            imgClock(i) = clpDigits.GraphicCell(0)
        End If
    Next i
    For i = 0 To 7
        imgClock(i) = clpDigits.GraphicCell(10)
    Next i
Timer2.Enabled = False
contrl = False
Setshutdown = False
Imgout_1 = True
Imgsub_1 = True
lbbatt = True
status = False
Checkdat = False
picProgress.Visible = False
picProgress1.Visible = False
picProgress2.Visible = False
LSub(0).Visible = False
LSub(1).Visible = False
LBat(0).Visible = False
LBat(1).Visible = False
Lload(0).Visible = False
Lload(1).Visible = False
Text2.Visible = False
Text3.Visible = False
Text4.Visible = False
For i = 0 To 3
Text1(i).Visible = False
Next i
```

```
rungraph = 1
runtim = 1
datafile = FreeFile
 'formmain.Caption = "UPS Program  ID.ED.21"
     If App.Path = "a:\" Or App.Path = "A:\" Then
              IconDrive = App.Path & "UPS.ico"
               FileDrive = App.Path & "battery.dat"
               INIDrive = App.Path & "ups.ini"
            Else
              IconDrive = App.Path & "\" & "UPS.ico"
FileDrive = App.Path & "\" & "battery.dat"
          INIDrive = App.
Path & "\" & "ups.ini"
          End If
For i = 0 To 60
shape1(i).Visible = False '
Next i
Set mtexScript = mfilScript.OpenTextFile(INIDrive, ForReading,
True)
If mtexScript.AtEndOfStream Then
mtexScript.Close
file_open = FileDrive
 OpenFile file_open
   Else
    mtexScript.Close
     ReadDATAfile
     findfile
    file_open = FileDrive
 datavar(0) = file_open
 End If
 With picProgressSlide
       .Cls
        .BackColor = &H80000016
        .ForeColor = vbHighlight
        .Move 0, 0, picProgress.ScaleWidth,
picProgressSlide.Height
    End With
    With picProgressSlide1
         .Cls
         .BackColor = &H80000016
         .ForeColor = vbHighlight
         .Move 0, 0, picProgress1.ScaleWidth,
picProgressSlide1.Height
    End With
   With picProgressSlide2
         .Cls
         .BackColor = &H80000016
         .ForeColor = vbHighlight
         .Move 0, 0, picProgress2.ScaleWidth,
picProgressSlide2.Height
    End With
  ShowCurrTime
  LoadCommParameters
  intTimer4
```

```
End Sub
Private Sub form_MouseMove(Button As Integer, Shift As Integer, x
As Single, y As Single)
Dim response As Integer
    Static rec As Boolean, msg As Long
    msg = x / Screen.TwipsPerPixelX
 If rec = False Then
       rec = True
   Select Case msg
    Case WM_LBUTTONDOWN:
Case WM_LBUTTONDBLCLK:
                    Formmain.Show
                    formDBclick = False
    Case WM_RBUTTONDOWN:

If formDBclick = True Then
                    response = MsgBox("          Menu Close Program
[Y/N]", 4, "UPS Program", 0, 1)
                        If response = vbYes Then
                            Lclose_Click
                        End If
              Else
         End If
    End Select
   rec = False
 End If
End Sub
Private Sub LMINI_Click()
t.cbSize = Len(t)
    t.hWnd = Formmain.hWnd
    t.uId = 1&
    t.uFlags = NIF_ICON Or NIF_TIP Or NIF_MESSAGE
    t.ucallbackMessage = WM_MOUSEMOVE
          t.hIcon = Formmain.Icon
          t.szTip = "UPS Program" & Chr$(0)
     Shell_NotifyIcon NIM_ADD, t
    Formmain.Hide 'Make the Form invisible
    formDBclick = True
    'App.TaskVisible = False
End Sub

Private Sub Timer1_Timer()
Dim i As Integer, col As Integer
Dim val(1024) As Integer
Dim InBufferCount As Integer
LbTime.Caption = Format(Now, "dd/mm/yy")
 If MSComm1.PortOpen = False Then MSComm1.PortOpen = True
 For i = 1 To MSComm1.InBufferCount
 If i Then
     InBufferCount = 0
If MSComm1.InBufferCount > 250 Then
     MSComm1.PortOpen = False
End If
```

```
InBufferCount = 0
 If MSComm1.PortOpen = False Then MSComm1.PortOpen = True
 If MSComm1.InBufferCount Then
      val(i) = Asc(MSComm1.Input)
 End If
     value(1) = val(1) / &H7F * 220
     value(2) = val(2) / &H64 * 15
       If Checkdat = True Then
        CheckValueVIN
       End If
       If i = 2 Then
            picProgressSlide1.Height = picProgress1.ScaleHeight -
picProgress1.ScaleHeight * CSng(value(2)) / 17
            Checkdat = True
       End If
       If i = 3 Then
         If value(1) > 75 And value(2) > 10 Then
              Text4.Text = value(3)
          Else
             Text4.Text = value(3)
          End If
          picProgressSlide2.Height = picProgress2.ScaleHeight -
picProgress2.ScaleHeight * CSng(value(3)) / 442
     End If
            If i < 4 Then
                Text1(i - 1).Text = Hex(val(i))
                If i = 1 Then Text2.Text = value(1)
                If i = 2 Then Text3.Text = value(2)
                End If
    End If
Next i
ttt
ShowCurrTime
End Sub
Function CheckValueVIN()
picProgressSlide.Height = picProgress.ScaleHeight -
picProgress.ScaleHeight * CSng(value(1)) / 442
picProgressSlide2.Height = picProgress2.ScaleHeight -
picProgress2.ScaleHeight * CSng(value(3)) / 442
' +25% of 220 =275 Volt  ,  -25% of 220 = 165 Volt
If ((value(1) >= 0) And (value(1) < 176)) Then
   intTimer3 '1 stra
  value(3) = value(1)
ElseIf (value(1) >= 176) And (value(1) < 210) Then
 intTimer1 '2 chage
ElseIf (value(1) >= 210) And (value(1) < 231) Then '
 intTimer4 '3 invert
 value(3) = value(1)
ElseIf (value(1) >= 231) And (value(1) < 265) Then
intTimer1 '4 bypass
value(3) = value(1)
ElseIf value(1) >= 265 Then
 intTimer3
```

```
End If
If value(2) = 10 Then
    If MSComm1.PortOpen = True Then
        ' MSComm1.PortOpen = False
         'Timer1.Enabled = False
    End If
         dataErr = TmpMns
        ' Timer2.Enabled = True
End If
If (value(2) < 10) Then
         'MSComm1.PortOpen = False
       '  dataErr = TmpMns
       '   Timer2.Enabled = True
        ' Timer1.Enabled = False
End If
 If value(2) > 10 And (MSComm1.PortOpen = False) Then
     MSComm1.PortOpen = True
     Timer1.Enabled = True
     Timer2.Enabled = False
End If
End Function

 Private Sub Timer2_Timer()
 Dim response As Integer
 'ttt
 value(2) = value(2)
If status = False Then
If Formmain.MSComm1.InBufferCount = 0 Then
          response = MsgBox("5 Minute Supply To Down ", 4, "Save
Your Program", 0, 1)
                 If response = vbYes Then
                          dataErr = TmpMns
                          status = True
                          Setshutdown = True
                 Else
                    MsgBox "Battery is Low . Please Close Your
Computer", , "Program is Close"
                          Timer2.Enabled = False
                 End If
 End If
 End If
      DIGITAL.Show
End Sub
 Function ShowCurrTime()
   Dim i As Integer, buff As String, aChar As String
   Static count As Integer, COmin As Integer, COho As Integer
        If count > 59 Then
            count = 0
            COmin = COmin + 1
                        'If COmin > 1 Then
                        '        ExitWindowsEx EWX_SHUTDOWN, 0
                       ' End If
                 If COmin > 59 Then
```

```
COmin = 0
                        COho = COho + 1
                        If COho > 23 Then COho = 0
                    End If
End If
   buff = Format$(Now, "hh:mm:ss")
    For i = 1 To 8
                    aChar = Mid$(buff, i, 1)
                    If aChar = ":" Then
                         imgClock(i - 1) =
clpPunctuation.GraphicCell(2)
                    Else
                        imgClock(i - 1) = clpDigits.GraphicCell(Asc
(aChar) - Asc("0"))
                    End If
        Next i
        count = count + 1
End Function
```

รูปที่ ข.3 โปรแกรมหลัก

ในส่วนของการแสดงผลของการเก็บข้อมูลที่โปรแกรม

ไฟล์ mainprogram.frm

คำสั่งที่ใช้ในโปรแกรม

```
Option Explicit
Const DEF_FILEFILTER = "data Files (*.dat)|*.dat"
 Public ind As Integer
 Public Echo As Boolean
 Private iFlow%, bTmpEcho As Boolean
Private Sub cmdCancel_Click()
Dim a As Integer
If datavar(3) <> val(Mid(cboPort.Text, 4, 4)) Then
  datavar(3) = datavar(3)
            If datavar(3) > 0 Then
               cboPort.ListIndex = datavar(3) - 1
            Else
               datavar(3) = 0
               MainProgram.Cbotime.ListIndex = 0
             End If
End If
If datavar(2) <> Cbotime.Text Then
   datavar(2) = datavar(2)
   Select Case datavar(2)
    Case "SEC   ": a = 0
    Case "MINUTE": a = 1
    Case "HOUR  ": a = 2
    Case "DATE  ": a = 3
   End Select
   End If
   Cbotime.ListIndex = a
  MainProgram.Visible = False
  Formmain.Visible = True
End Sub
Private Sub CmdDown_Click(Index As Integer)
Dim i As Integer
Dim databat As String
Select Case Index
     Case 0:
              If ind > 0 Then
                ind = ind - 1
              End If
     Case 1:
If ind > 0 Then
                ind = ind - 1
              End If
Case 2:
              If ind > 0 Then
                ind = ind - 1
              End If
C
```

```
If ind > 0 Then
                ind = ind - 1
              End If
   End Select
     For i = 0 To 3
       Text1(i).Text = ind
       databat = ReadDATA(ind)
       TxtTime(i).Text = Mid$(databat, 1, 20)
       TxtBattery1(i).Text = val(Mid$(databat, 21 + i * 5, 5))
     Next i
     If ind <= 0 Then
     ind = 0
       For i = 0 To 3
          CmdDown(i).Enabled = False
          CmdUP(i).Enabled = True
       Next i
     End If
End Sub
Private Sub CmdOK_Click()
Dim OldPort%, NewPort%
Dim a As Integer
Dim settings As String, cbopa As String
        If Formmain.MSComm1.PortOpen = True Then
            Formmain.MSComm1.PortOpen = False
        End If
 datavar(1) = Trim$(cboBaudRate.Text) & "," & Left$
(CboParity.Text, 1) _
& "," & Trim$(cboDataBits.Text) & "," & Trim$(cboStopBits.Text)
datavar(2) = Cbotime.Text
   Select Case datavar(2)
    Case "SEC   ": a = 0
    Case "MINUTE": a = 1
    Case "HOUR  ": a = 2
    Case "DATE  ": a = 3
   End Select
Cbotime.ListIndex = a
datavar(3) = val(Mid(cboPort.Text, 4, 4))
If datavar(3) > 0 Then
   cboPort.ListIndex = datavar(3) - 1
Else
 cboPort.ListIndex = 0
 datavar(3) = 0
 End If
   Formmain.MSComm1.CommPort = datavar(3)
   Formmain.MSComm1.settings = datavar(1)
   If Formmain.MSComm1.PortOpen = False Then
        Formmain.MSComm1.PortOpen = True
   End If
Formmain.Show
Set mtexScript = mfilScript.OpenTextFile(INIDrive, ForWriting,
True)
         mtexScript.Write "[Drive]"
         mtexScript.WriteLine datavar(0)
         mtexScript.Write "[Set_Mode]"
```

```
mtexScript.WriteLine datavar(1)
         mtexScript.Write "[Set_Time]"
         mtexScript.WriteLine datavar(2)
         mtexScript.Write "[Set_Port]"
         mtexScript.WriteLine datavar(3)
         mtexScript.Close
        MainProgram.Visible = False
End Sub
Private Sub CmdUP_Click(Index As Integer)
Dim i As Integer
Dim LbTime As String
Select Case Index
     Case 0: ind = ind + 1
     Case 1: ind = ind + 1
     Case 2: ind = ind + 1
     Case 3: ind = ind + 1
   End Select
   For i = 0 To 3
      Text1(i).Text = ind
      LbTime = ReadDATA(ind)
      TxtTime(i).Text = Mid$(LbTime, 1, 20)
      TxtBattery1(i).Text = val(Mid$(ReadDATA(ind), 21 + i * 5,
5))
      CmdDown(i).Enabled = True
    Next i
If ReadDATA(ind) = "" Then
   For i = 0 To 3
      CmdDown(i).Enabled = True
      CmdUP(i).Enabled = False
    Next i
End If
End Sub
Private Sub Form_Load()
Dim i As Integer
Dim READDATA_IND As String
ind = 0
For i = 0 To 3
 option1(i).value = True
 Text1(i).Text = ind
 READDATA_IND = ReadDATA(ind)
 TxtTime(i).Text = Mid$(READDATA_IND, 1, 20)
 TxtBattery1(i).Text = val(Mid$(READDATA_IND, 21 + i * 5, 5))
 CmdDown(i).Enabled = False
 CmdUP(i).Enabled = True
Next i
row1 = 1
dlgFile.filename = file_open
MainProgram.Caption = "Append Fiels : " & datavar(0)
End Sub
Private Sub M1_2_Click()
Dim aaa As String
If Changed Then Newfile
    dlgFile.DialogTitle = "Open File"
    With dlgFile
```

```
.filename = file_open
          .Filter = DEF_FILEFILTER
          .FilterIndex = 0
          .Flags = FileOpenConstants.cdlOFNHideReadOnly
         .ShowOpen
 End With
On Error Resume Next
file_open = dlgFile.filename
    If Err = 32755 Then
        Exit Sub
     End If
     If dlgFile.filename = "" Then
    Exit Sub
 Else
 file_open = dlgFile.filename
 datavar(0) = file_open
 FileDrive = file_open
 OpenFile dlgFile.filename
  End If
End Sub
 Private Sub M1_3_Click()
MainProgram.Hide
Formmain.Show
End Sub

Private Sub option1_Click(Index As Integer)
Select Case Index
    Case 0: MSChart1.chartType = VtChChartType2dLine
    Case 1: MSChart2.chartType = VtChChartType2dLine
    Case 2: MSChart3.chartType = VtChChartType2dLine
    Case 3: MSChart4.chartType = VtChChartType2dLine
   End Select
 End Sub

Private Sub Option2_Click(Index As Integer)
Select Case Index
    Case 0: MSChart1.chartType = VtChChartType2dBar
    Case 1: MSChart2.chartType = VtChChartType2dBar
    Case 2: MSChart3.chartType = VtChChartType2dBar
    Case 3: MSChart4.chartType = VtChChartType2dBar
   End Select
 End Sub

Private Sub Option3_Click(Index As Integer)
Select Case Index
     Case 0: MSChart1.chartType = VtChChartType3dBar
     Case 1: MSChart2.chartType = VtChChartType3dBar
     Case 2: MSChart3.chartType = VtChChartType3dBar
     Case 3: MSChart4.chartType = VtChChartType3dBar
   End Select
End Sub
```

รูปที่ ข.3 โปรแกรมการเก็บข้อมูลและสถิติ

โปรแกรมโมดูลซึ่งทำหน้าที่เป็นตัวรวบรวมชุดคำสั่งที่สามารถเรียกใช้งานได้ทุก ๆ ฟอร์ม

ไฟล์ valuebat.bas

คำสั่งที่ใช้ในโปรแกรม

```
Option Explicit
Const DEF_FILEFILTER = "data Files (*.dat)|*.dat"
Private Declare Function GetTickCount Lib "kernel32" () As uLong
Private Type uLong ' unsigned long integer variable
    a As Byte
    b As Byte
    c As Byte
    d As Byte
End Type
Public uTmpAll As uLong
Public TmpAll As Double
Public TmpHrs As Double
Public TmpMns As Double
Public TmpScs As Double
Public TmpMil As Double
Public TmpDat As Double
'Public valuegraph(100) As Integer
Public runtim As Integer
Public value(0 To 1024) As Integer
Public rungraph As Integer
Public row(0 To 4, 0 To 10) As Integer
Public row1 As Integer
Public file_open As String
Public rmblnFileChanged As Boolean
Private m_sFileFilter As String
Public filestr As String

Public mfilScript As New Scripting.FileSystemObject
Public mtexScript As TextStream
Public IconDrive As String
Public FileDrive As String
Public INIDrive As String
Public datavar(0 To 100) As String

Public Const EWX_SHUTDOWN = 1
Public Const EWX_REBOOT = 2
Public Const EWX_LOGOFF = 0
Public Const EWX_FORCE = 4
Public Type Batterry
          battery1 As String * 5
          battery2 As String * 5
          battery3 As String * 5
          battery4 As String * 5
          data_time As String * 20
End Type
```

```
Public databat As Batterry

Public Function filename() As String
    filename = file_open
    datavar(0) = file_open
End Function
Public Property Get Changed() As Boolean
    Changed = rmblnFileChanged
End Property
Public Property Let Changed(Setting As Boolean)
    rmblnFileChanged = Setting
End Property
Public Function findfile()
Dim a As String, b As String, c As Integer, d As Integer
For c = 0 To 100
  If ((c < Len(datavar(0))) Or (c < 100)) Then
       b = Mid$(datavar(0), Len(datavar(0)) - c, 1)
       a = Right(datavar(0), c)
       If (b = "\") Or (b = "/") Then
       c = 101
       End If
  End If
Next c
If a = "" Then
a = "battery.dat"
End If
      If App.Path = "a:\" Or App.Path = "A:\" Then
                IconDrive = App.Path & "ups.ico"
                FileDrive = App.Path & a
                INIDrive = App.Path & "ups.ini"
            Else
              IconDrive = App.Path & "\" & "ups.ico"
              FileDrive = App.Path & "\" & a
              INIDrive = App.Path & "\" & "ups.ini"
            End If
End Function
Public Function GetText(file_open) As String
    If file_open = "" Then Exit Function
    Set mtexScript = mfilScript.OpenTextFile(file_open,
ForReading, True)
    If mtexScript.AtEndOfStream Then
        GetText = ""
    Else
        GetText = mtexScript.ReadAll
    End If
    mtexScript.Close
End Function
Public Function ReadDATA(poindata As Integer) As String
Dim datafile As Integer, indFile As Integer
Dim datavar1 As String
If file_open = "" Then Exit Function
datafile = FreeFile
```

```
Set mtexScript = mfilScript.OpenTextFile(datavar(0), ForReading,
True)
If mtexScript.AtEndOfStream Then
mtexScript.Close
 Set mtexScript = mfilScript.OpenTextFile(datavar(0), ForWriting,
True)
    mtexScript.Close
     Set mtexScript = mfilScript.OpenTextFile(INIDrive,
ForWriting, True)
         file_open = FileDrive
         mtexScript.Write "[Drive]"
         mtexScript.WriteLine datavar(0)
         mtexScript.Write "[Set_Mode]"
         mtexScript.WriteLine datavar(1)
         mtexScript.Write "[Set_Time]"
         mtexScript.WriteLine datavar(2)
         mtexScript.Write "[Set_Port]"
         mtexScript.WriteLine datavar(3)
         mtexScript.Close
       Exit Function
   Else
     Open file_open For Input As #datafile
             If Err.Number = 0 Then
                 For indFile = 0 To poindata
                    If Not EOF(datafile) Then
                    Line Input #datafile, datavar1
                    ReadDATA = datavar1
                    End If
                 Next indFile
              End If
          Close #datafile
   End If
End Function
 Public Function ReadDATAfile()
Dim datafile As Integer, indFile As Integer
Dim Setvar(0 To 100) As String
Dim a As String, b As String
Dim c As Integer, d As Integer
If INIDrive = "" Then Exit Function
datafile = 0
datafile = FreeFile
          Open INIDrive For Input As #datafile
            Do While Not EOF(datafile)    ' Check for end of file.
               Line Input #datafile, Setvar(indFile)
                For c = 1 To 100
                If (c < Len(Setvar(indFile)) And (c < 200)) Then
                   b = Mid$(Setvar(indFile), c, 1)
                      If (b = "[") Then
                           d = c
                      ElseIf (b = "]") Then
                           a = Mid$(Setvar(indFile), d,c+1)
a = Mid$(Setvar(indFile), Len(a), Len(Setvar(indFile)))
                                  datavar(indFile) = a
```

```
indFile = indFile + 1
                                    End If
                              End If
                         Next c
              Loop
           Close #datafile
If datavar(0) = "" Then
     findfile
    datavar(0) = file_open
  If datavar(1) = "" Then
    datavar(1) = "9600,N,8,1"
     If datavar(2) = "" Then
       datavar(2) = "MINUTE"
          If datavar(3) = "" Then
            datavar(3) = 1
           End If
      End If
   End If
OpenFile datavar(0)
End If
End Function
Public Function Newfile()
Dim intSave As Integer
    If Changed Then
    MainProgram.dlgFile.Filter = DEF_FILEFILTER
       intSave = MsgBox("File has changed, save changes?",
vbYesNo, "Save Changes?")
        If intSave = vbYes Then
            Newfile
       End If
    End If
  rmblnFileChanged = False
 MainProgram.Caption = "Append New Files  : " & file_open
End Function
Public Function OpenFile(filename)
Dim IconDrive As String, FileDrive As String, INIDrive As String
 If App.Path = "a:\" Or App.Path = "A:\" Then
             IconDrive = App.Path & "ups.ico"
             FileDrive = App.Path & filename
             INIDrive = App.Path & "ups.ini"
           Else
             IconDrive = App.Path & "ups.ico"
             FileDrive = App.Path & "\" & filename
             INIDrive = App.Path & "\" & "ups.ini"
           End If
    rmblnFileChanged = False
    datavar(0) = file_open
    FileDrive = file_open
    filename = file_open
    datavar(1) = "9600,N,8,1"
    datavar(2) = "MINUTE"
    datavar(3) = 1
```

```
Set mtexScript = mfilScript.OpenTextFile(INIDrive, ForWriting,
True)
          mtexScript.Write "[Drive]"
          mtexScript.WriteLine datavar(0)
          mtexScript.Write "[Set_Mode]"
          mtexScript.WriteLine datavar(1)
          mtexScript.Write "[Set_Time]"
          mtexScript.WriteLine datavar(2)
          mtexScript.Write "[Set_Port]"
          mtexScript.WriteLine datavar(3)
    mtexScript.Close
    Set mtexScript = mfilScript.OpenTextFile(filename,
ForWriting, True)
    mtexScript.Close
    MainProgram.Caption = "Append Fiels : " & datavar(0)
 End Function
Public Function FNrungraph()
Dim i As Integer, max As Integer, j As Integer
databat.battery1 = value(1)
databat.battery2 = value(2)
databat.battery3 = value(3)
databat.battery4 = value(4)
databat.data_time = Now
max = 10
MainProgram.MSChart1.RowCount = max
MainProgram.MSChart2.RowCount = max
MainProgram.MSChart3.RowCount = max
MainProgram.MSChart4.RowCount = max
    If (row1 >= max + 1) Then
         row1 = 1
    End If
 row(1, 0) = databat.battery1
 row(2, 0) = databat.battery2
 row(3, 0) = databat.battery3
 row(4, 0) = databat.battery4
 If row1 >= 2 Then
    For j = 1 To 4
        For i = max To 1 Step -1
             row(j, i) = row(j, i - 1)
        Next i
    Next j
 End If
 For i = max To 1 Step -1
  MainProgram.MSChart1.row = i
  MainProgram.MSChart1.data = row(1, i)
  MainProgram.MSChart2.row = i
  MainProgram.MSChart2.data = row(2, i)
  MainProgram.MSChart3.row = i
  MainProgram.MSChart3.data = row(3, i)
  MainProgram.MSChart4.row = i
  MainProgram.MSChart4.data = row(4, i)
  Next i
  row1 = row1 + 1
```

```
End Function
Public Function RungraphMain()
If file_open = "" Then Exit Function
Set mtexScript = mfilScript.OpenTextFile(file_open, ForReading,
True)
If mtexScript.AtEndOfStream Then
mtexScript.Close
 Set mtexScript = mfilScript.OpenTextFile(file_open, ForWriting,
True)
    mtexScript.Close
 Else
End If
Set mtexScript = mfilScript.OpenTextFile(file_open, ForAppending,
True)
                    mtexScript.Write databat.data_time & " "
                    mtexScript.Write databat.battery1 & " "
                    mtexScript.Write databat.battery2 & " "
                    mtexScript.Write databat.battery3 & " "
                    mtexScript.WriteLine databat.battery4 & " "
                    mtexScript.Close
End Function

Public Function ttt()
Dim a As String
Dim i As Integer
Static b As Integer
Dim TmpHrs1 As Double
Dim TmpMns1 As Double
Dim TmpScs1 As Double
Dim TmpMil1 As Double
Dim TmpDat1 As Double
TmpDat1 = TmpDat
TmpHrs1 = TmpHrs
TmpMns1 = TmpMns
TmpScs1 = TmpScs
TmpMil1 = TmpMil
    uTmpAll = GetTickCount
    TmpAll = ULng2Dbl(uTmpAll)
    TmpHrs = Fix(TmpAll / 3600000)
    TmpMns = Fix((TmpAll - (TmpHrs * 3600000)) / 60000)
    TmpScs = Fix((TmpAll - (TmpHrs * 3600000) - (TmpMns * 60000))
\ 1000)
    TmpMil = TmpAll - (TmpHrs * 3600000) - (TmpMns * 60000) -
(TmpScs * 1000)
    TmpDat = TmpHrs \ 24
    If Formmain.MSComm1.InBufferCount = 0 Then
       If TmpScs1 < TmpScs - 1 Then
         b = b + 1
         If b > 10 Then MsgBox "NO Response Data ON Port"
         Else
          b = 0
          End If
```

```
End If
If TmpScs1 <> TmpScs Then FNrungraph
Select Case datavar(2)
     Case "SEC    ":
                    If TmpScs1 <> TmpScs Then
                     RungraphMain
                     End If
     Case "MINUTE":
                    If TmpMns1 <> TmpMns Then
                     RungraphMain
                     End If
     Case "HOUR   ":
                    If TmpHrs1 <> TmpHrs Then
                    RungraphMain
                     End If
     Case "DATE   ":
                    If TmpDat1 <> TmpDat Then
                     RungraphMain
                    End If
   End Select
End Function
Private Function ULng2Dbl(In_u As uLong) As Double
    Dim TmpDblVal As Currency
    TmpDblVal = CCur(CCur(In_u.d) * CCur(16777216)) + CCur(In_u.c
* 65536) + CCur(In_u.b) * 256 + CCur(In_u.a)
    ULng2Dbl = TmpDblVal
End Function

Public Sub LoadCommParameters()
    Dim i%, settings As String, cbopa As String
    Dim a As Integer

  If Formmain.MSComm1.PortOpen = True Then
    Formmain.MSComm1.PortOpen = False
  End If
   For i% = 1 To 2
        MainProgram.cboPort.AddItem "Com" & CStr(i%)
    Next i%
    If datavar(3) = "" Then
    datavar(3) = 0
    End If
    If (datavar(3) > 0) Then
    MainProgram.cboPort.ListIndex = datavar(3) - 1
    Else
      datavar(3) = 0
      MainProgram.cboPort.ListIndex = 0
    End If
    MainProgram.cboBaudRate.AddItem "9600"
    MainProgram.cboBaudRate.ListIndex = 0

    MainProgram.cboDataBits.AddItem "8"
    MainProgram.cboDataBits.ListIndex = 0
```

```
MainProgram.cboStopBits.AddItem "1"
    MainProgram.cboStopBits.ListIndex = 0

    MainProgram.Cbotime.AddItem "SEC    "
    MainProgram.Cbotime.AddItem "MINUTE"
    MainProgram.Cbotime.AddItem "HOUR   "
    MainProgram.Cbotime.AddItem "DATE   "

   Select Case datavar(2)
     Case "SEC    ": a = 0
     Case "MINUTE": a = 1
     Case "HOUR   ": a = 2
     Case "DATE   ": a = 3
   End Select
   If a > 0 Then
        MainProgram.Cbotime.ListIndex = a
    Else
      MainProgram.Cbotime.ListIndex = 0
   End If
    MainProgram.CboParity.AddItem "None"
    MainProgram.CboParity.ListIndex = 0
     datavar(1) = Trim$(MainProgram.cboBaudRate.Text) & "," &
Left$(MainProgram.CboParity.Text, 1) _
     & "," & Trim$(MainProgram.cboDataBits.Text) & "," & Trim$
(MainProgram.cboStopBits.Text)
    Formmain.MSComm1.InputLen = 1
   On Error Resume Next
      If Formmain.MSComm1.PortOpen = False Then
       Formmain.MSComm1.PortOpen = True
  End If
End Sub
```

รูปที่ ข.4 โปรแกรมในส่วนของโมดูล

บรรณานุกรม

จิระ จริงจิตร , " หลักการและสภาพแวดล้อมของ Visual Basic", *เรียนลัด Visual Basic 2539*, พิมพ์ครั้งที่ 2 บริษัทซีเอ็ดยูเคชั่น จำกัด (มหาชน), สิงหาคม 2539 , หน้า 17 – 19

ชัยรัตน์ เกษมบุญศิริ และคณะ, "*แรกเริ่มการเรียนรู้เรื่อง Visual Basic สำหรับ Windows 95 2539*", บริษัทซีเอ็ดยูเคชั่น จำกัด (มหาชน) , 2539 , 600 หน้า

โชคชัย เตชพรรุ่ง ,*Visual Basic สำหรับ Windows 95 Step by Step*, บริษัทซีเอ็ดยูเคชั่น จำกัด (มหาชน) , 2539, 484 หน้า

ดร.วิริยะ พิเชษฐจำเริญ , " Un- interruptible Power System",*Thyristor Application,*

มงคล ทองสงคราม , "ฟอสเฟตกำลัง", *อิเล็กทรอนิกส์กำลัง 2540* , พิมพ์ครั้งที่ 3 ห้างหุ้นส่วนจำกัด วี.เจ.พริ้นติ้ง ,พฤศจิกายน 2540 , หน้า 17 – 21

วรวิทย์ ตันติโภคิน และเพื่อน , "*การเขียนโปรแกรมบนวินโดวส์ด้วย Visual Basic ภาคปฏิบัติ*", บริษัทซีเอ็ดยูเคชั่น จำกัด (มหาชน) , 2537, 566 หน้า

รศ. ศุลี บรรจงจิตร , "ระบบไฟฟ้าฉุกเฉิน/สำรอง" , *อุปกรณ์และการติดตั้งในงานระบบไฟฟ้า 2537*, พิมพ์ครั้งที่ 2 บริษัทซีเอ็ดยูเคชั่น จำกัด (มหาชน), สิงหาคม 2539 ,หน้า 17 – 19

สุวัฒน์ ดั่น , "พุช – พูล คอนเวอร์เตอร์" , *เทคนิคและการออกแบบสวิตชิ่งเพาเวอร์ซัพพลาย 2538*, พิมพ์ครั้งที่ 2 บริษัทเอนเทลไทย จำกัด ,มิถุนายน 2538 , หน้า 41 – 45

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายกำธร ธงทอง
วันเดือนปีเกิด	15 ธันวาคม พ.ศ. 2522
สถานที่เกิด	จังหวัดอุดรธานี
ภูมิลำเนาเดิม	178 ซอย 10 ถ.มิตรภาพ ต.โนนสูง อ.เมือง
	จ.อุดรธานี 41330
ที่อยู่ปัจจุบัน	300/108 ถ.ฉลองกรุง
	แขวงลำปลาทิว เขตลาดกระบัง
	กรุงเทพมหานคร 10520
เบอร์โทร	7373897

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนเทศบาล 3 บ้านเหล่า
มัธยมศึกษาตอนต้น	โรงเรียนอุดรพิทยานุกูล
ประกาศนียบัตรวิชาชีพ	วิทยาลัยเทคนิคอุดรธานี
ประกาศนียบัตรวิชาชีพชั้นสูง	วิทยาลัยเทคนิคอุดรธานี
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์
	ภาควิชา ครุศาสตร์วิศวกรรม
	คณะวิชา ครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	-
คติพจน์	วันนี้ต้องดีกว่า อดีต

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นางสาวนารีรัตน์ แย้มมา
วันเดือนปีเกิด	15 ตุลาคม พ.ศ. 2522
สถานที่เกิด	จังหวัดราชบุรี
ภูมิลำเนาเดิม	142 ถ.ทรงพล อ.โพธาราม
	จ.ราชบุรี 70120
ที่อยู่ปัจจุบัน	312/34 ม.เครือทิพย์
	แขวงลาดกระบัง เขตลาดกระบัง
	กรุงเทพมหานคร 10520
เบอร์โทร	739-0755

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนเจี้ยไช้
มัธยมศึกษาตอนต้น	โรงเรียนโพธาวัฒนาเสนี
ประกาศนียบัตรวิชาชีพ	วิทยาลัยเทคนิคราชบุรี
ประกาศนียบัตรวิชาชีพชั้นสูง	วิทยาลัยเทคนิคราชบุรี
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์
	ภาควิชา ครุศาสตร์วิศวกรรม
	คณะวิชา ครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	-
คติพจน์	ทำวันนี้ให้ดีที่สุดไม่ต้องไปหวังถึงวันข้างหน้า

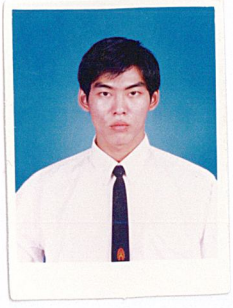
ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายภูวไนย สูนย์สาทร
วันเดือนปีเกิด	20 กรกฎาคม พ.ศ. 2520
สถานที่เกิด	จังหวัดสุราษฎร์ธานี
ภูมิลำเนาเดิม	24 ม. 1 ต. คลองไทร อ. ท่าฉาง จ. สุราษฎร์ธานี 84150
ที่อยู่ปัจจุบัน	16 ม. 8 แขวงคู้ฝั่งเหนือ เขตหนองจอก กรุงเทพมหานคร 10530
เบอร์โทร	(02) 988-6820 (02) 327-3120 (01)240-1117

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนมานิตานุเคราะห์
มัธยมศึกษาตอนต้น	โรงเรียนสุราษฎร์พิทยา
ประกาศนียบัตรวิชาชีพ	โรงเรียนสุราษฎร์เทคโนโลยี(ช่างอุตสาหกรรม)
ประกาศนียบัตรวิชาชีพชั้นสูง	วิทยาลัยเทคนิคมีนบุรี
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชา ครุศาสตร์วิศวกรรม คณะวิชา ครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	-
คติพจน์	เก็บฝันไว้เป็นพลัง...ทุกคิดทุกหวังก็คือพลังที่ดี

ประวัติผู้แต่ง



ชื่อผู้ทำปริญญานิพนธ์	นายณรงค์ บรรณเลิศ
วันเดือนปีเกิด	12 ตุลาคม พ.ศ. 2521
สถานที่เกิด	จังหวัดเชียงราย
ภูมิลำเนาเดิม	377/1 ถ.สันโค้งหลวง ซอย 12 ต.รอบเวียง อ.เมือง จ.เชียงราย 57000
ที่อยู่ปัจจุบัน	300/90 ม. 10 หมู่บ้านรุ่งอรุณ 1 แขวงลำปาทิว เขตลาดกระบัง กรุงเทพมหานคร 10520
เบอร์โทร	737-4477

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนอนุบาลเชียงราย
มัธยมศึกษาตอนต้น	โรงเรียนสามัคคีวิทยาคม
ประกาศนียบัตรวิชาชีพ	วิทยาลัยเทคนิคเชียงราย
ประกาศนียบัตรวิชาชีพชั้นสูง	วิทยาลัยเทคนิคเชียงใหม่
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชา ครุศาสตร์วิศวกรรม คณะวิชา ครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	-
คติพจน์	