รายงานการวิจัย

เรื่อง

การสร้างระบบควบคุมการสวิทช์ขั้วไฟฟ้าสำหรับเครื่องวัดความต้านทานดิน

(Construction of Electrode Switching Controller System for Earth Resistivity Meter)

คณะผู้วิจัย:

หัวหน้าโครงการวิจัย : นายสมศักดิ์ เดียวสุรินทร์

ผู้ร่วมโครงการวิจัย:

1. ผศ.ดร.วรวุฒิ โลหะวิจารณ์
2. นาย แฉล้ม พรหมพัฒน์
3. นายวีระ ไทยสยาม

ภาควิชาฟิสิกส์ คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์

ได้รับทุนอุดหนุนการวิจัยจาก

คณะกรรมการจัดสรรทุนวิจัย คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์

อ.หาดใหญ่ จ. สงขลา

ปีงบประมาณ พ.ศ. 2543

ว.ค.

เลขหมู่ QC809.E15 ส64 2543 ฉ.1

Bib Key 225233

บทคัดย่อ

ได้ออกแบบและสร้างระบบจีโออีเล็คตริคโหนด ซึ่งเป็นเครื่องมือในการสวิทช์อีเล็คโตรดหลายขั้ว สำหรับใช้กับเครื่องวัดสภาพต้านทานดินใดๆ ระบบประกอบด้วยหน่วยควบคุมซึ่งต่อกับเครื่องวัดสภาพต้านทานดินด้วยสายไฟสี่สาย(สายกระแสสองเส้นและสายศักย์ไฟฟ้าสองเส้น)และต่อกับจีโออีเล็คตริคโหนดแบบเน็ตเวอร์ก 16 ชุดด้วยสายเคเบิ้ลเจ็ดสาย หน่วยควบคุมออกแบบโดยใช้ไมโครคอนโทรลเลอร์ PIC 16F877 ต่อกับคีย์แพดโทรศัพท์เพื่อเลือกโหนดที่ต้องการ มีหน่วยแสดงผลเป็นเซเว่นเซ็กเมนท์สองหลัก สวิทช์สี่ตัวสำหรับเลือกสายทั้งสี่ และ LED สี่ตัวแสดงผลสายที่ถูกเลือก แต่ละอีเล็คตริคโหนดออกแบบโดยใช้ไมโครคอนโทรลเลอร์ PIC16F84 กับสวิทช์รีเลย์สี่ตัว สายไฟสองสายใช้กับบัสRS-485ไดรเวอร์สำหรับการสื่อสารระหว่างหน่วยควบคุมกับแต่ละจีโออีเล็คตริคโหนด ระบบสามารถขยายเพื่อรองรับได้กับ 32 อีเล็คตริคโหนดและมีช่วงการวัดยาวถึง 3 กิโลเมตร

Abstract

A sixteen-node geo-electric system is presented. The multi-electrode switching array can be used with any earth-resistivity meter and consists of a control unit which interfaces resistivity meter by means of four wires (two current wires and two potential wires) with sixteen geo-electric nodes, connected to the control unit via a network of a seven-core cable. The control unit uses PIC16F877 interfaces with a telephone keypad to select the wanted electric node, two seven-segment LED for display, four switches for line selection and four LEDs for line indicator. Each electric node has been designed by using PIC16F84 and four relays for electrode switching. Two wires are used for the RS-485 bus driver between the control unit and each electric node for communication. The system can be expanded up to 32-electric nodes within a range of 3 kilometer.

สารบัญเรื่อง

	หน้า
กิตติกรรมประกาศ	iii
บทคัดย่อ(Abstract)	iv
บทนำ(Introduction)	v
วิธีการ(Methodology)	v
หน่วยควบคุม(Control Unit)	6
อีเล็คตริกโหนด(Electric node Unit)	8
ซอฟท์แวร์ควบคุม(Control Software)	8
โปรแกรมหลัก(main program)	8
โปรแกรมย่อยอินเตอร์รัป(Interupt service routine)	9
ทดสอบ(Test)	11
เอกสารอ้างอิง(Bibiography)	11
ภาคผนวก	
ลายพิมพ์วงจรควบคุมด้านหน้า	14
ลายพิมพ์วงจรควบคุมด้านหลัง	15
ลายพิมพ์วงจรอีเล็คตริกโหนดด้านหน้าและหลัง	16
ซอร์สโค้ดสำหรับโปรแกรมหน่วยควบคุม	17
ซอร์สโค้ดสำหรับโปรแกรมอีเล็คตริกโหนด	25

สารบัญภาพ

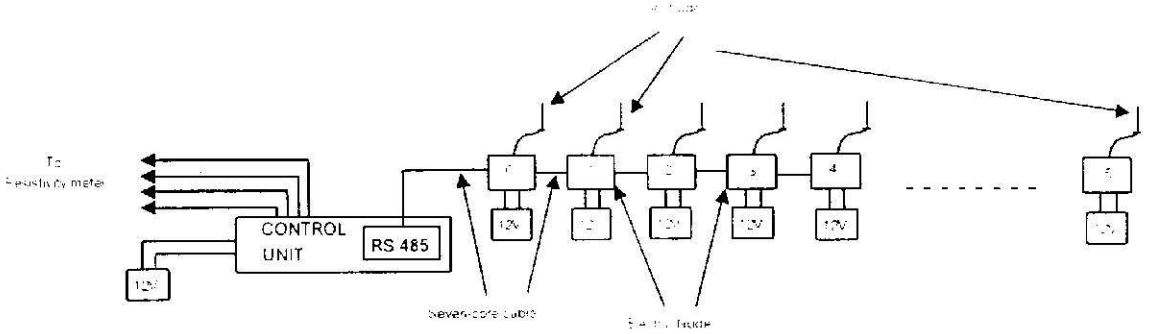
	หน้า
ภาพต้นแบบระบบจีโออีเล็คตริคโหนดซึ่งประกอบด้วยหน่วยควบคุมและอีเล็คตริกโหนด 1ชุด	11

สารบัญตาราง

	หน้า
ตารางเปรียบเทียบค่าความต้านทานที่อ่านได้จากเครื่อง ABEM Resistivity	12

บทนำ(Introduction)

หน่วยธรณีฟิสิกส์ ภาควิชาฟิสิกส์มีเครื่องวัดสภาพต้านทานไฟฟ้าของดินซึ่งใช้ในการสำรวจการตรวจหาชั้นน้ำบาดาล การตรวจหาลำดับชั้นดิน การตรวจหาความลึกของชั้นหินดานสำหรับงานก่อสร้าง การตรวจหารอยต่อของชั้นหินทางด้านธรณีวิทยาตามแนวราบ ในการวัดสภาพต้านทานไฟฟ้าจำเป็นต้องใช้ขบวนขั้วไฟฟ้า 4 ขั้วโดยขั้วไฟฟ้า 2 ขั้วสำหรับป้อนกระแสไฟฟ้า(I) ให้กับดิน และขั้วไฟฟ้าอีก 2 ขั้วสำหรับวัดความต่างศักย์ของดิน (ΔV) โดยสภาพต้านทานไฟฟ้าของดินสามารถคำนวณได้จากสมการ $\rho = G(\Delta V/I)$ เมื่อ G คือ geometrical factor ซึ่งเป็นฟังก์ชันของระยะห่างระหว่างขั้วไฟฟ้าต่างๆ ในการวัดเราต้องปักขั้วไฟฟ้าลงบนพื้นดินทั้งสี่จุดพร้อมกัน เมื่อต้องการวัดค่าอื่นๆจะต้องเพิ่มระยะห่างระหว่างขั้วกระแสไฟฟ้า หรือการเปลี่ยนตำแหน่งขั้วไฟฟ้าตามแนวราบ จำเป็นต้องใช้แรงงานคนช่วย โดยใช้คน 1 คนต่อ 1 ขั้วไฟฟ้า เพื่อความสะดวกและความรวดเร็วในการเปลี่ยนตำแหน่ง ในหนึ่งทีมสำหรับการวัดสภาพต้านทานไฟฟ้าของดินเราจำเป็นต้องมีผู้ช่วยงานสนามถึง 4คน



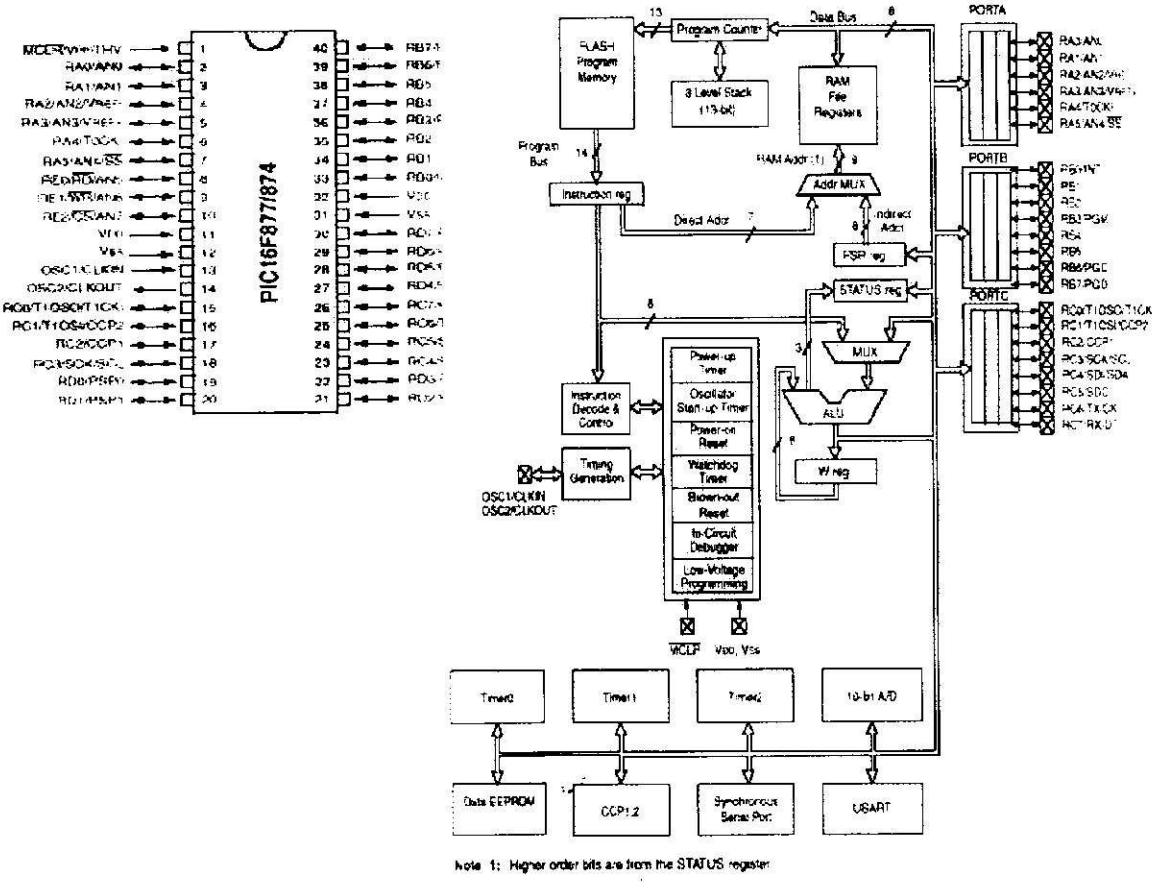
รูปที่ 1. บล็อกไดอะแกรมระบบจีโออีเล็คตริกโหนด

เพื่อลดจำนวนคนและเพิ่มประสิทธิภาพในกรณีที่ต้องการวัดสภาพต้านทานไฟฟ้าซ้ำหลายๆครั้งโดยไม่ต้องเคลื่อนย้ายขั้วไฟฟ้ากลับไปกลับมา การออกแบบระบบเคลื่อนย้ายขั้วไฟฟ้าอัตโนมัติหรือการใช้ระบบวัดแบบขั้วไฟฟ้าหลายขั้วเพื่อนำมาใช้กับเครื่องวัดสภาพต้านทานที่มีอยู่จึงเป็นสิ่งที่เป็นประโยชน์อย่างยิ่ง วิธีนี้จำนวนขั้วไฟฟ้าทั้งหมดที่ต้องการจะถูกปักลงบนพื้นดินครั้งเดียวตามแบบที่กำหนดไว้ เครื่องควบคุมสามารถเลือกขั้วไฟฟ้า 4 ขั้วใดๆตามต้องการ ระบบนี้ทำให้การออกสนามเพื่อวัดสภาพต้านทานไฟฟ้าใช้คนเพียงคนเดียว เรียกระบบนี้ว่าระบบจีโออีเล็คตริกโหนด (Geo-electric node system)

วิธีการ(Methodology)

ระบบจีโออีเล็คตริกโหนดมีบล็อกไดอะแกรมดังแสดงรูปที่1. ระบบประกอบด้วย 2ส่วนคือ หน่วยควบคุม(control unit)และอีเล็คตริกโหนด (electric node)16 ชุด หน่วยควบคุมต่อกับเครื่องวัดสภาพต้านทานไฟฟ้าด้วยสายไฟ 4 สาย สาย Aและ Bสำหรับกระแส สาย MและNสำหรับศักย์ไฟฟ้า และต่อกับอีเล็คตริกโหนดทั้ง 16 ชุดเป็นเน็ตเวิร์ค ด้วยเคเบิ้ล utp แบบ 8 สาย สาย4สายต่อผ่านแผงพิมพ์วงจรไปสู่ เครื่องวัดสภาพต้านทานไฟฟ้า 2 สายเป็นบัสของสัญญาณควบคุม สายกราวด์ 1 สาย และสายว่าง 1 สาย คอนเน็กเตอร์สำหรับแคเบิ้ลนี้เป็นแบบ โมดูลาแจ็ก โทรศัพท์(modular telephone jack)

หน่วยควบคุมทำหน้าที่เลือกสาย A B M หรือ N โดยสวิทช์แบบกดติดปล่อยดับ(push button switch) 4 ตัว และเลือกหมายเลขโหนด(electric node number) ด้วยคีย์แพ็ดโทรศัพท์ สายที่เลือกแสดงผลบน LED 4 ตัว และหมายเลขแสดงผลบนเซเว่นเซ็กเมนท์LEDสองหลัก สวิทช์อีกหนึ่งตัวทำหน้าที่ในการส่งข้อมูล(สวิทช์ Enter) อิเล็คตริกโหนดแต่ละตัวซึ่งมีหมายเลข(node number)เฉพาะของมันเอง(0-15) ทำหน้าที่รับหมายเลขจากหน่วยควบคุมพร้อมกันทั้ง 16 ชุด ถ้าหมายเลขที่ส่งมาตรงกับหมายเลขตัวมันเอง วงจรในอีเล็คตริกโหนดจะกระตุ้นรีเลย์สวิทช์ให้ทำงานทำให้ขั้วไฟฟ้าสำหรับอีเล็คตริกโหนดนั้นเชื่อมต่อเข้ากับสายใดๆในสี่สายที่ถูกเลือก ถ้าหมายเลขที่ส่งมาไม่ตรงกับหมายเลขของมันวงจรจะไม่กระตุ้นรีเลย์สวิทช์แต่จะเข้าสู่สลีปโหมด(sleep mode)เพื่อประหยัดพลังงาน



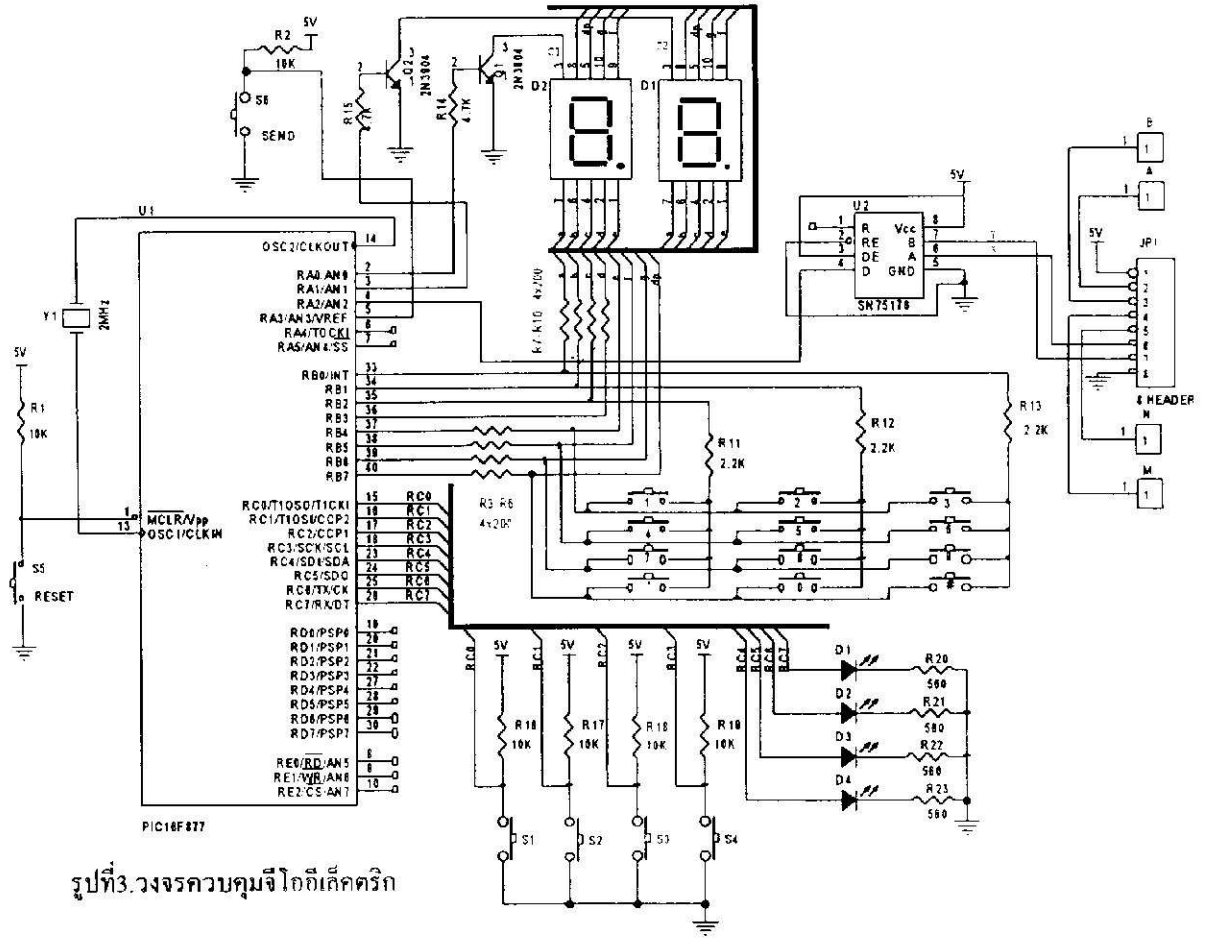
รูปที่2. ขาและบล็อกไดอะแกรมภายในของไมโครคอนโทรลเลอร์ PIC16F877สำหรับวงจรควบคุมหลัก

หน่วยควบคุม(Control Unit)

วงจรสำหรับหน่วยควบคุมมีไมโครคอนโทรลเลอร์ Microchip PIC 16F877 เป็นตัวควบคุมหลัก อุปกรณ์นี้เป็นอุปกรณ์ใหม่ที่บริษัทผลิตจำหน่ายเมื่อไม่นานมานี้ PIC 16F877 มี Flash EEPROM สำหรับเก็บโปรแกรมซึ่งสะดวกในการนำมาใช้สำหรับโครงงานนี้ เป็นอุปกรณ์ขนาด 40 ขา ไดอะแกรมขา16F877แสดงในรูปที่2. 1) ขา 32 และ 31 ต่อกับ แหล่งจ่ายไฟบวกและลบตามลำดับ แหล่งจ่ายไฟป้อนแรงดันระหว่าง2-6 โวลต์และโดย

ทั่วไปจ่ายกระแสประมาณ 2mAที่5 โวลต์ที่ 4MHz ขา 1 เป็น master clear ซึ่งปกติต้องเป็น high ยกเว้นเมื่อต้องการ clear ขา 15 และ 16 ต่อกับ crystal ออสซิเลเตอร์ 4MHz PIC 16F877 มี EEPROM 8K word ขนาด14-bit สำหรับเก็บคำสั่งและ 256 byte EEPROM เก็บข้อมูล มี15 special fuction รีจีสเตอร์ และ 36 byte รีจีสเตอร์อเนกประสงค์ มี I/Oพอร์ท 5 พอร์ท ซึ่งมีขาดังนี้ พอร์ทA(RA0-6) พอร์ทB (RB0-7) พอร์ทC(RC0-7) พอร์ทD(RD0-7) พอร์ทE (RE0-7) PIC มี คำสั่งเพียง 35 คำสั่งทำให้โปรแกรมง่ายต่อการเรียนรู้

PIC 16F877 ต่อกับอุปกรณ์ภายนอกเป็นวงจรควบคุมหลักดังรูปที่3. พอร์ทBติดต่อกับ เซเว่นเซ็กเมนท์LED สองหลักสำหรับแสดงหมายเลขของอีเล็คโตรดโหนด ขณะเดียวกันพอร์ทBก็ต่อกับ 3x4 คีย์แพ็ดโทรศัพท์สำหรับเลือกอีเล็คโตรดโหนดที่ต้องการติดต่อ พอร์ทBทำงานแบบมัลติเพล็กซ์คือเป็นเอาท์พุททั้งพอร์ทเมื่อแสดงผลแต่เป็นทั้งอินพุทและเอาท์พุทเมื่อต้องรับค่าจากคีย์แพ็ด ขา RB0-RB7ต่อกับ เซ็กเมนท์ a-gและ ตามลำดับ ขา RB3-RB0 ต่อกับคอลัมน์ 1-3 RB4-RB7ต่อกับแถว 1-4 ตามลำดับ พอร์ทAเป็นเอาท์พุทสำหรับ RA0และ RA1 เพื่อจุดเซเว่นเซ็กเมนท์หลักสิบและหลักหน่วย ตามลำดับ ขา RB2 เป็นขาส่งข้อมูลแบบอนุกรมไปยังอีเล็คตริกโหนดผ่านบัสไดรเวอร์ ไอซี SN75176B ซึ่งเป็น balanced line drivers สัญญาณอินพุทเข้าที่ขา4(D) และสัญญาณออกที่ขา6(A)และ7 (B) ผ่านคอนเน็กเตอร์เป็นบัสร่วมของอีเล็คตริกโหนดทั้ง16ชุด ขา RA3 เป็นอินพุทต่อกับสวิทช์เพื่อควบคุมการส่งข้อมูล พอร์ทC RC0- RC3ขาเป็นอินพุทต่อกับสวิทช์ ทั้งสี่(A B Mและ N) เพื่อเลือกสายกระแสหรือสายศักย์ไฟฟ้าของเครื่องวัดสภาพต้านทานไฟฟ้าที่จะต้องต่อกับอีเล็คโตรด ส่วนขา RC4- RC7 เป็นเอาท์พุทต่อกับ LED เพื่อแสดงผลสายทั้งสี่

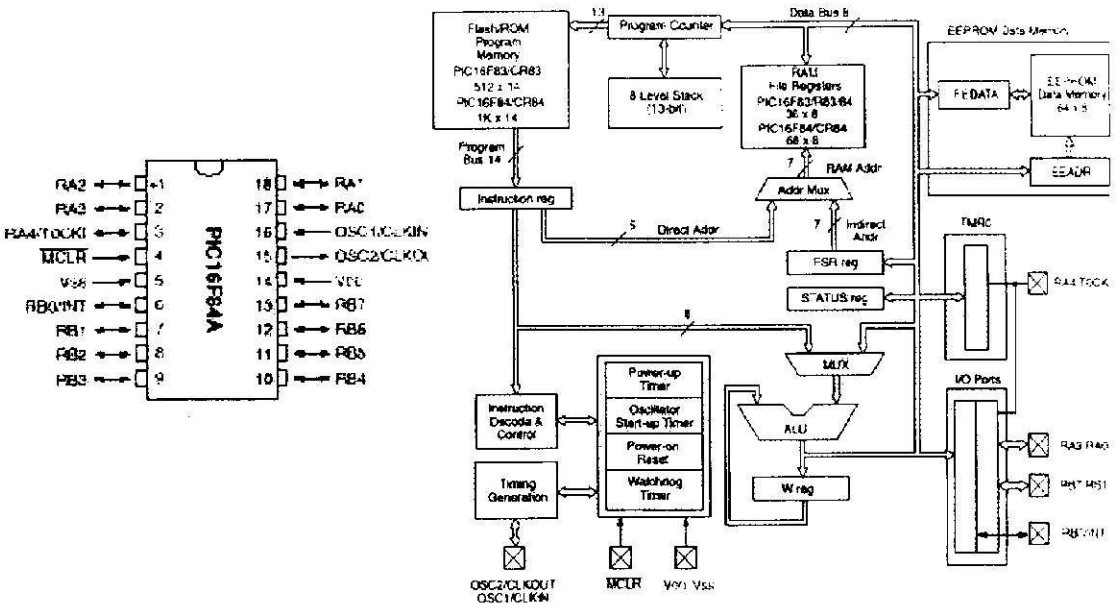


รูปที่3.วงจรควบคุมจีโออีเล็คตริก

อีเล็คตริกโหนด(Electric node Unit)

วงจรอีเล็คตริกโหนดมีไมโครคอนโทรลเลอร์ Microchip PIC16F84(ดูรูปที่4.)เป็นตัวควบคุมการทำงาน PIC16F84 เป็นไมโครคอนโทรลเลอร์ที่มีความซับซ้อนน้อกว่า PIC16F877 คือมีขาเพียง 18 ขา และFlash EEPROM สำหรับเก็บโปรแกรมก็น้อยกว่ามาก แต่ก็เพียงพอสำหรับหน้าที่ในการรับข้อมูลจากหน่วยควบคุมและกระตุ้นรีเลย์ให้ทำ หน้าที่ปิด-เปิด ขั้วไฟฟ้า อีเล็คตริกโหนดมีวงจรตามรูปที่5.

อีเล็คตริกโหนด รับข้อมูลจากบัสระบบด้วยบัสรีซีฟเวอร์ไอซี SN75176B เอาท์พุทรีซีฟเวอร์ R ขา1 ต่อกับอินพุทพอร์ทB RB5 ของ PIC16F84 ขา 11 RB0-RB3 เป็นเอาท์พุทต่อกับขาเบสทรานซิสเตอร์ 2N3904ทั้ง4เพื่อควบคุมรีเลย์สวิทช์ RELAY1 RELAY2 RELAY3และ RELAY4 ตามลำดับ LEDเปล่งแสงD5ต่อกับRB5ของพอร์ท Bเพื่อแสดงว่าอีเล็คตริกโหนดนั้นถูกเลือกจากหน่วยควบคุม



รูปที่4. ขาและบล็อกไดอะแกรมภายในของไมโครคอนโทรลเลอร์ PIC16F84สำหรับวงจรอีเล็คตริกโหนด

ซอฟท์แวร์ควบคุม(Control Software)

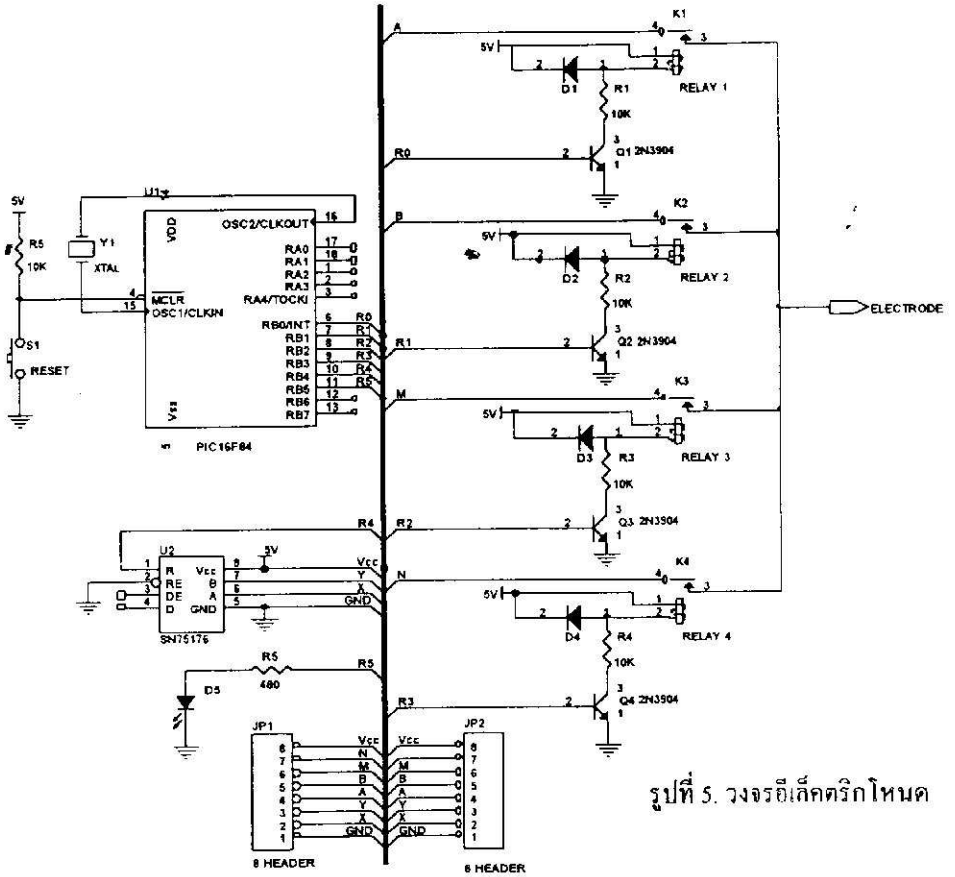
ซอฟท์แวร์โปรแกรม แบ่งออกเป็นสองส่วน คือ โปรแกรมสำหรับPIC16F877ในหน่วยควบคุมและโปรแกรมสำหรับPIC16F84ในอีเล็คตริกโหนด โปรแกรมในหน่วยควบคุมประกอบด้วยโปรแกรมหลัก (main Program)และโปรแกรมย่อยอินเตอร์รัป (interupt service rotine)ซึ่งควบคุมการส่งข้อมูลและแสดงผล โปรแกรมในอีเล็คตริกโหนดซึ่งควบคุมการรับข้อมูลและปิดเปิด RELAYทั้งสี่

โปรแกรมหลัก(main program)

โปรแกรมหลักเริ่มต้นด้วยการเตรียมพร้อมพอร์ทต่างๆโดยเซ็ทพอร์ทBเป็นเอาท์พุทสำหรับส่งข้อมูลไปแสดงผล พอร์ทAเป็นเอาท์พุทสำหรับเซเว่นเซ็กเมนท์LED พอร์ทC 4 บิทล่างเป็นอินพุทสำหรับรับข้อมูลจากสวิทช์ทั้ง 4 พอร์ทC 4 บิทบนเป็นเอาท์พุทสำหรับแสดงผลค่าของวิทช์ จากนั้นโปรแกรมจะเตรียมเซ็ทค่าต่างๆให้แก่Timer โดย

เซ็ทค่าprescaler หารด้วย 16 เซ็ทค่าเริ่มต้นใน TMR0 เซ็ท INTCON

ให้ Timer อินเตอร์รัปเมนโปรแกรมได้ เมนโปรแกรมจะถูกอินเตอร์รัปทุกๆ 5 มิลลิวินาที



รูปที่ 5. วงจรอีเล็คตริกโหนด

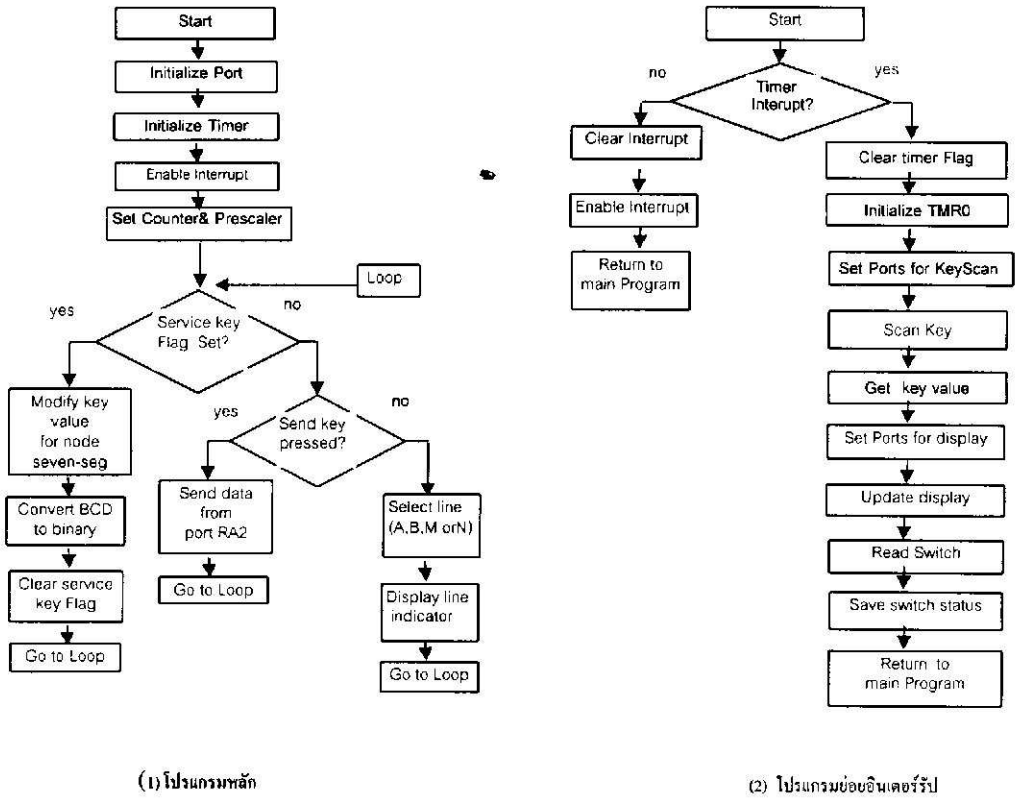
โปรแกรมทดสอบว่า Service key Flag เป็นเซ็ทหรือไม่ ถ้าFlagถูกเซ็ทโปรแกรมจะเปลี่ยนแปลงค่าที่ได้รับจากการกดคีย์แพ็ดเพื่อให้เหมาะสมต่อการแสดงผลบนเซเว่นเซ็กเมนท์LED ต่อไปก็แปลงข้อมูลจาก BCD ให้เป็น เลขฐานสองเพื่อส่งไปยังอีเล็คตริกโหนดแล้วเคลียร์Service key Flag และวนอยู่ในลูป แต่ถ้าFlagไม่ถูกเซ็ทโปรแกรมจะทดสอบว่าสวิทช์ที่ใช้ควบคุมในการส่งข้อมูลถูกกดหรือไม่ ถ้าถูกกดข้อมูลจะถูกส่งแบบอนุกรมผ่านขาRA2แล้วกลับเข้าลูป แต่ถ้าไม่ถูกกดโปรแกรมจะเลือกว่าLEDแสดงสาย A B M หรือ N ที่จะต้องแสดงผลซึ่งขึ้นอยู่กับสถานะที่เก็บไว้ในช่วง Interupt service routine แล้วกลับเข้าลูป

โปรแกรมย่อยอินเตอร์รัป(Interupt service routine)

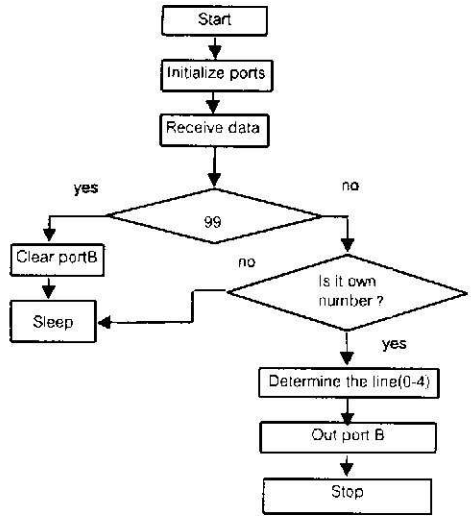
โปรแกรมเริ่มด้วยการทดสอบว่าเป็น timer อินเตอร์รัป ถ้าไม่ใช่โปรแกรมจะเคลียร์อินเตอร์รัปรีจีสเตอร์แล้วเซ็ทอินเตอร์รัปอีเนเบิ้ลแล้วกลับโปรแกรมหลัก แต่ถ้าใช่โปรแกรมจะเคลียร์ timer Flag ตั้งค่า TMR0 ใหม่ เซ็ทพอร์ทB สำหรับสแกนคีย์ทำการสแกนคีย์ หาค่าค่าคีย์ เซ็ทพอร์ทBสำหรับแสดงผล ปรับค่าใหม่สำหรับการแสดงผล อ่านค่าสวิทช์ เก็บสถานะสวิทช์ กลับโปรแกรมหลัก

โปรแกรมสำหรับอีเล็คตริกโหนด(Electric node program)

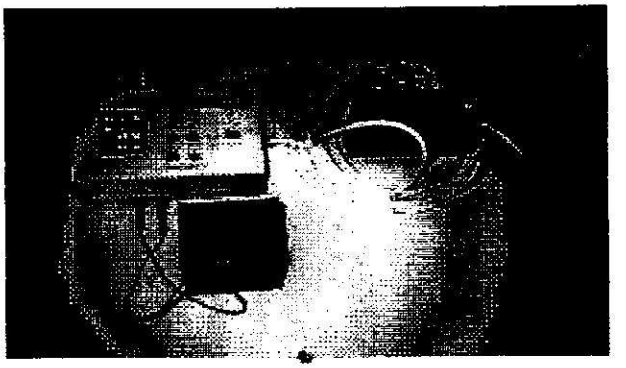
จากรูปที่ 7. โปรแกรมเริ่มต้นด้วยการเซ็ทพอร์ทแล้วทำการรับข้อมูล ถ้าข้อมูลเป็นเลข 99ซึ่งเป็นรหัสสำหรับรีเซ็ทพอร์ทBทำให้ขั้วไฟฟ้าไม่ต่อกับสายใดๆทั้ง 4 สาย ถ้าข้อมูลเป็นหมายเลขโหนดของมันโปรแกรมจะทำการหาว่าสายใดควรต่อกับขั้วไฟฟ้าแล้วส่งค่าออกที่พอร์ทB ถ้าข้อมูลเป็นหมายเลขอื่นโปรแกรมจะเข้าสู่โหมด sleep



รูปที่ 6. โฟลชาร์ทโปรแกรมการทำงานของหน่วยควบคุมจีโออีเล็คตริก
(1) โปรแกรมหลัก (2) โปรแกรมย่อยอินเตอร์รัป



รูปที่ 7. โฟลชาร์ทโปรแกรมการทำงานของจีโออีเล็คตริกโหนด



รูปที่ 8. ภาพต้นแบบระบบจีโออีเล็คตริคโหนดซึ่งประกอบด้วยหน่วยควบคุมและอีเล็คตริกโหนด 1ชุด

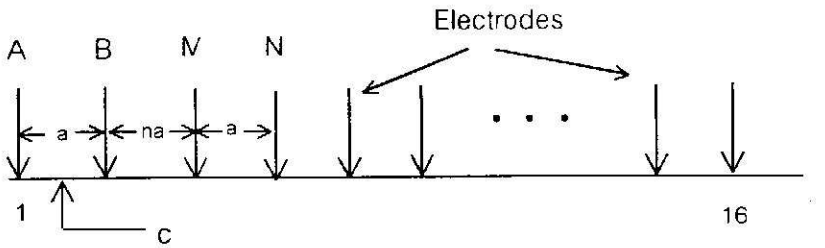
ทดสอบ(Test)

ได้นำระบบต้นแบบจีโออีเล็คตริกโหนด (รูปที่8)ไปทดสอบใช้ในสนามสำหรับการทดลองหา สภาพต้านทานดินของนักศึกษาวิชาฟิสิกส์ ปีที่3 บริเวณที่ทำการทดสอบใกล้ทางเท้าระหว่างคณะวิศวกรรมศาสตร์กับภาควิชาฟิสิกส์ มหาวิทยาลัยสงขลานครินทร์

การทดสอบเป็นการวัดค่าความต้านทานจากเครื่อง ABEM Resistivity Meter โดยเปรียบเทียบค่าที่อ่านแบบไม่มีระบบสวิทช์ขั้วกับค่าที่อ่านได้เมื่อมีระบบสวิทช์ขั้ว การจัดขั้วอิเล็คโตรดเป็นแบบ dipole (รูปที่ 9)โดยมีระยะห่างระหว่างขั้วกระแส(A,B)และขั้วศักย์ (M,N) a เท่ากับ 1 เมตร ขั้วศักย์ M อยู่ห่างจากขั้วกระแส Bเป็นระยะ na ตำแหน่ง c เป็นจุดกึ่งกลางระหว่างขั้ว Aและ B ผลการวัดแสดงอยู่ในตารางที่ 1

สรุป(Conclusion)

ผลการทำงานของระบบจีโออีเล็คตริกโหนดเป็นที่น่าพอใจ อย่างไรก็ตามค่าความต้านทานที่อ่านได้แตกต่างกันเล็กน้อยเมื่อเทียบกับการวัดแบบใช้สายต่อโดยตรงเมื่อไม่มีระบบจีโออีเล็คตริกโหนด ความผิดพลาดนี้เป็นผลมาจากการเกิดศักย์ไฟฟ้าเหนี่ยวนำในสายวัดศักย์ไฟฟ้า(M,N)เนื่องจากกระแสในสาย(A,B) เพราะสายที่ใช้ของระบบนี้เป็นแบบ unshield ความผิดพลาดนี้สามารถแก้ไขได้ด้วยการปรับปรุงระบบโดยเลือกใช้สายแบบ shield แต่ละคู่เพื่อไม่ให้มีสัญญาณรบกวนระหว่างคู่สาย แต่จะต้องเปลี่ยนคอนเน็กเตอร์เพราะสายสองชนิดนี้ใช้คอนเน็กเตอร์ไม่เหมือนกัน



รูปที่ 9. ภาพแสดงการจัดอิเล็คโตรดสำหรับการทดสอบการวัดความต้านทานแบบ dipole

ตำแหน่งจุดกึ่งกลางขั้วกระแส(c) (เมตร)	na (ระยะห่างระหว่างขั้ว A และขั้ว M)(เมตร)	ค่าความต้านทานเมื่อไม่มีระบบสวิทซขั้ว	ค่าความต้านทานเมื่อมีระบบสวิทซ์ขั้ว	เปอร์เซ็นต์ความแตกต่าง
	1a	17.78 Ω	14.0 Ω	-21.25
	2a	2.35 Ω	2.17Ω	-7.65
	3a	973 mΩ	930 mΩ	-4.42
	4a	570 mΩ	530 mΩ	-7.01
	5a	407 mΩ	413 mΩ	+1.47
	6a	87.7mΩ	74.4 mΩ	-15.1
3.0	6a	24.8 mΩ	18.8 mΩ	-24.19
	5a	131.1 mΩ	123 .6 mΩ	-5.72
	4a	616 mΩ	617 mΩ	+0.16
	3a	976 mΩ	905 mΩ	-7.27
	2a	2.37Ω	2.02 Ω	-14.76
	1a	16.99 Ω	10.99 Ω	-35.31
3.5	1a	12.53 Ω	9.50 Ω	-24.18
	2a	2.03 Ω	1.82Ω	-10.34
	3a	1.03 Ω	1.00 Ω	-2.91
	4a	236 mΩ	221 mΩ	-6.35
	5a	74.4 mΩ	75.3 mΩ	+1.20
	6a	54.2 mΩ	46.3 mΩ	-14.57
4.0	6a	68.6m Ω	53.8 mΩ	-15.01
	5a	69.8 mΩ	65.5 mΩ	-6.16
	4a	102.1 mΩ	85.9 mΩ	-15.86
	3a	435 mΩ	439 mΩ	+0.91
	2a	1.92 Ω	2.71 Ω	+41.14
	1a	6.38 Ω	7.14 Ω	+11.9

ตารางที่1 ตารางเปรียบเทียบค่าความต้านทานที่อ่านได้จากเครื่อง ABEM Resistivity

เอกสารอ้างอิง

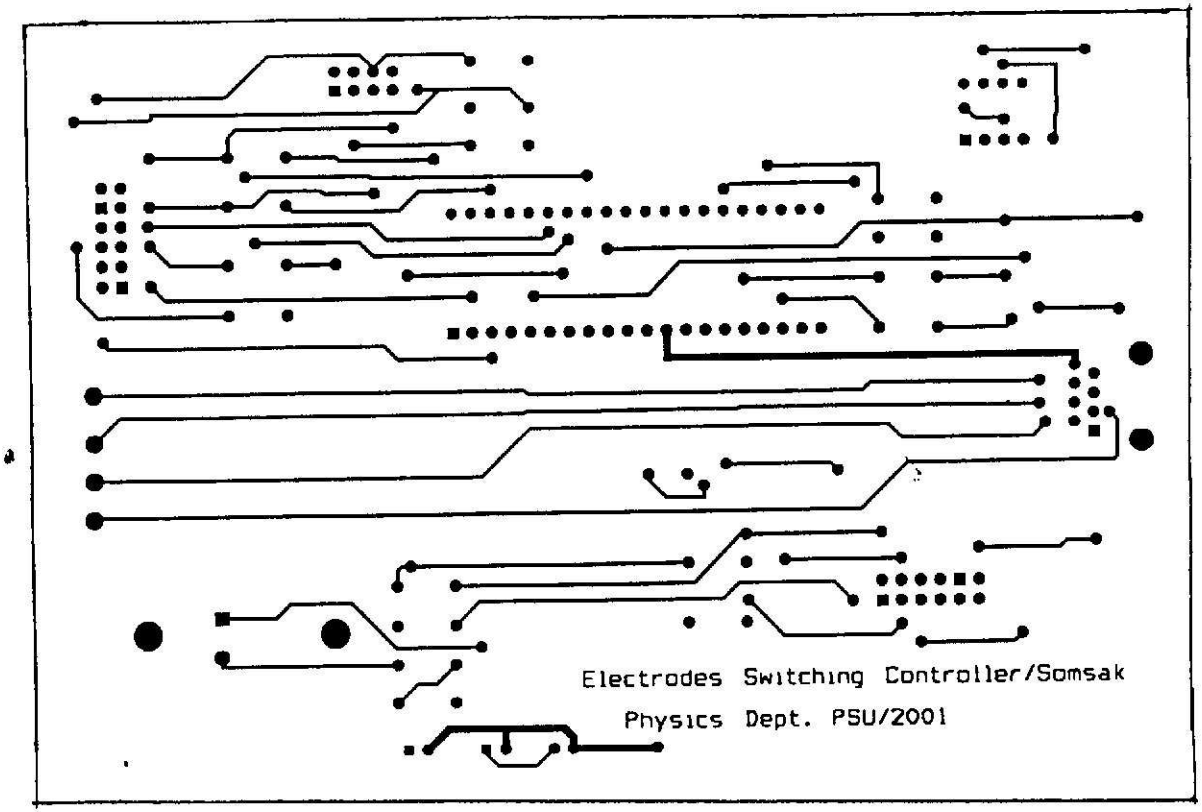
D'Souza Stan. 1977. Multiplexing LED Drive and4x4 Keypad Sampling. Applicatio nnoteAN529. Microchip Technology Inc.

Placherla,Amar.1977. SoftwareImplementation of Asynchronous Serial I/O. Application note AN555Microchip Technology Inc.

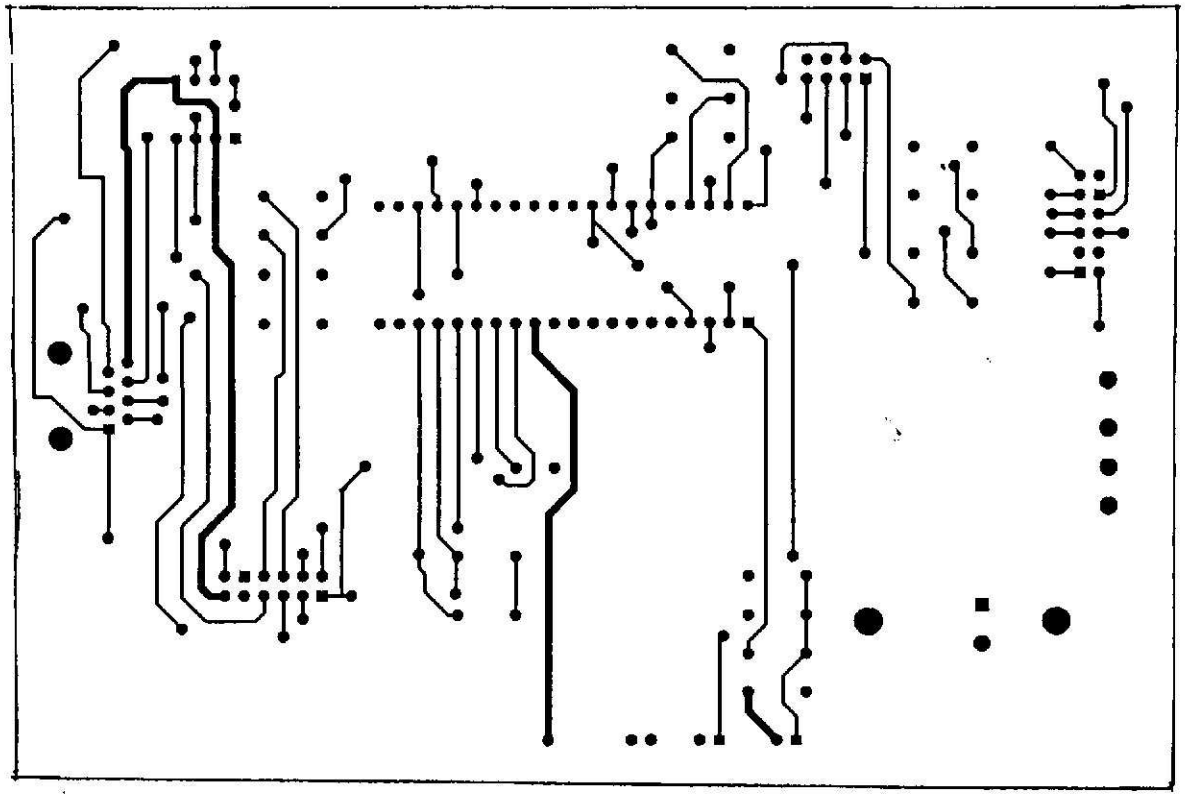
PIC16F877 28/40-Pin,8-BitCMOS FLASH Microconontrollers with10- bit A/D converter.1977 Databook. Microchip Technology Inc.

PIC16F84 18Pin, 8-Bit CMOS Flash/ - EEPROM Microcontrollers. 1977 DataBook Microchip Technology Inc.

ภาคผนวก



ภาพที่ 10. ลายพิมพ์วงจรควบคุมด้านหน้า



ภาพที่ 11. ลายพิมพ์วงจรควบคุมด้านหลัง

```
;*************************************************************************
;This program is contain in U1 which use method of multiplex two 7 segment
;LED digits and a 3X4 keypad using a PIC16F877.
;The two digits will start as '00' and when a key is hit
;it is displayed on the 7 segment leds as a decimal value 0 to 9. The last
;digit hit is always displayed on the right most led with the rest of
;the digit shifted to the left. The left  digit is deleted.
;The LEDs are updated every 20mS, the keypad is scanned at a rate of 20 mS.
;The TMR0 timer is used in internal interrupt mode to generate the 5 mS.
;
;
;
;       Program:          ESWC.ASM
;
;*************************************************************************
            LIST P=16F877
            ERRORLEVEL  -302
;
            #include     <p16F877.inc>

TempC    equ     0x20            ;temp general purpose regs
TempD    equ     0x21
TempE    equ     0x22
PABuf    equ     0x23
PBBuf    equ     0x24

Count    equ     0x26            ;count
MsdTime  equ     0x27            ;most significant Timer
LsdTime  equ     0x28            ;Least significant Timer
KeyFlag  equ     0x29            ;flags related to key pad
keyhit   equ     0       ;bit 0 --> key-press on
DebnceOn equ     1       ;bit 1 --> debounce on
Sflag    equ     2       ;no key entry = 0
ServKey  equ     3       ;bit 3 --> service key
NodeA    equ     4
NodeB    equ     5
NodeM    equ     6
NodeN    equ     7
Debnce   equ     0x30            ;debounce counter
NewKey   equ     0x31
WBuffer  equ     0x32
StatBuffer equ   0x33
OptionReg equ    1
PCL      equ     2
;

XmtReg      equ      36
Xcount      equ      37
Delaycnt    equ      38          ; delay for 104 us
mcount      equ      39
ncount      equ      41
;
Save_byte   equ      42
R0          equ      43            ; RAM Assignments
Save_temp   equ      44            ; temporary register
Var         equ      45
;
Digit1      equ     0x50
Digit2      equ     0x51
Digit3      equ     0x52
Digit4      equ     0x53

ByteTemp    equ     0x55
Adrtemp             equ     0x56
NDTemp      equ     0x58
NdCount     equ     0x59
TempSplit   equ     0x5A
BCD         equ     0x5B
BCD1        equ     0x5C
BCD2        equ     0x5D
NodeCount   equ     0x5E
VarA        equ     0x5F
SendCount   equ     0x60
LEDA        equ      4
LEDB        equ      5
```

```
LEDM        equ     6
LEDN        equ     7

;
;
            push    macro
            movwf   WBuffer         ;save w reg in Buffer
            swapf   WBuffer, F      ;swap it
            swapf   STATUS,W        ;get status
            movwf   StatBuffer      ;save it
            endm
;
pop         macro
            swapf   StatBuffer,W    ;restore status
            movwf   STATUS          ;       /
            swapf   WBuffer,W       ;restore W reg
            endm
;
            org     0

            goto    Start           ;skip over interrupt vector

            org     4
;It is always a good practice to save and restore the w reg,
;and the status reg during a interrupt.
            push
            call    ServiceInterrupts
            pop
            retfie
;
Start
            call    InitPorts
            call    InitTimers
loop
            call    WriteLED
            btfsc   KeyFlag,ServKey ;key service pending
            call    ServiceKey      ;yes then service
            btfsc   PORTA,3
            goto    loop
            call    Send

            goto    loop

;ServiceKey, does the software service for a keyhit. After a key service,
;the ServKey flag is reset, to denote a completed operation.
ServiceKey
            movf    NewKey,W        ;get key value
            movwf   TempE           ;save in TempE
            swapf   MsdTime,W       ;move MSD out
            andlw   B'11110000'     ;clr lo nibble
            movwf   MsdTime         ;save back
            iorwf   TempE,W         ;or with new lsd
            movwf   MsdTime
            call    BCDtoB
            call    Combine
            bcf     KeyFlag,ServKey ;reset service flag
            return

WriteLED
            clrf    PORTC
            btfss   KeyFlag,NodeA
            goto    CheckB
            bsf     PORTC,LEDA
            goto    ExitLED
CheckB      btfss   KeyFlag,NodeB
            goto    CheckM
            bsf     PORTC,LEDB
            goto    ExitLED
CheckM      btfss   KeyFlag,NodeM
            goto    CheckN
            bsf     PORTC,LEDM
            goto    ExitLED
CheckN      btfss   KeyFlag,NodeN
            goto    ExitLED
            bsf     PORTC,LEDN
ExitLED     return
```

```
Send
          bcf     INTCON,7
          movf    Save_byte ,W
          movwf   XmtReg
          movlw   .8
          movwf   Xcount          ;8 data bits
          bcf     PORTA,2         ;Send Start bit
          call    Delay1
X_next    bcf     STATUS,C
          rlf     XmtReg,F        ;Shift data bit
          btfsc   STATUS,C        ;Test the bit to be transmitted
          bsf     PORTA,2         ;Bit is one
          btfss   STATUS,C
          bcf     PORTA,2
          call    DelayX
          decfsz  Xcount,F        ; If count = 0, then transmit a stop bit
          goto    X_next          ; NO shift next bit
          bsf     PORTA,2         ; Send Stop Bit
          call    Delay1

          bsf     INTCON,7
          return

DelayX    movlw   .64
          goto    save

Delay1    movlw   .65
          goto    save

save      movwf   Delaycnt
redo_1    decfsz  Delaycnt,F
          goto    redo_1
          return

;

;
InitPorts
          bsf     STATUS,RP0      ;select pg 1
          movlw   0x06
          movwf   ADCON1
          movlw   B'00011000'
          movwf   TRISA           ;RA3,RA4 = input
          clrf    TRISB           ;make RB0-7 outputs
          bsf     TRISB,3
          movlw   0x0F            ;RC0-RC3 = iputs
          movwf   TRISC           ;RC4-RC7 = outputs
          bcf     STATUS,RP0      ;select page 0
          clrf    PORTB
          clrf    PORTC
          clrf    Save_byte
          clrf    KeyFlag

          movlw   .6
          movwf   PORTA

          return
;
;
;The clock speed is 4.096Mhz. Dividing internal clk. by a 32 prescaler,
;the rtcc will be incremented every 31.25uS. If rtcc is preloaded
;with 96, it will take (256-96)*31.25uS to overflow i.e. 5mS. So the
;end result is that we get a rtcc interrupt every 5mS.
InitTimers
          clrf    MsdTime         ;clr timers

          clrf    KeyFlag         ;clr all flags
          bsf     STATUS,RP0      ;select pg 1
          movlw   B'10000100'     ;assign ps to rtcc
          movwf   OptionReg       ;ps = 32
          bcf     STATUS,RP0      ;select pg 0
          movlw   B'00100000'     ;enable rtcc interrupt
          movwf   INTCON          ;
          movlw   .44             ;preload rtcc
          movwf   TMR0            ;start counter
```

Central Library
Prince of Songkla University

```
          retfie
;
ServiceInterrupts
          btfsc   INTCON,T0IF      ;rtcc interrupt?
          goto    ServiceTMR0      ;yes then service
          clrf    INTCON           ;else clr all int
          bsf     INTCON,T0IE
          return
;
ServiceTMR0
          movlw   .44              ;initialize rtcc
          movwf   TMR0
          bcf     INTCON,T0IF      ;clr int flag
          btfsc   PORTA,0          ;if msb on then do
          call    ScanKeys         ;do a quick key scan
          call    UpdateDisplay    ;update display
          call    ReadSwitch
          return
;
;
;ScanKeys, scans the 4X4 keypad matrix and returns a key value in
;NewKey (0 - B) if a key is pressed, if not it clears the keyhit flag.
;Debounce for a given keyhit is also taken care of.
;The rate of key scan is 20mS with a 4.096Mhz clock.
ScanKeys
          btfss   KeyFlag,DebnceOn ;debounce on?
          goto    Scan1            ;no then scan keypad
          decfsz  Debnce, F        ;else dec debounce time
          return                   ;not over then return
          bcf     KeyFlag,DebnceOn ;over, clr debounce flag
          return                   ;and return
Scan1
          call    SavePorts        ;save port values
          movlw   B'11110111'      ;init TempD
          movwf   TempD
ScanNext
          movf    PORTB,W         ;read to init port
          bcf     INTCON,RBIF      ;clr flag
          rrf     TempD, F         ;get correct column
          btfss   STATUS,C         ;if carry set?
          goto    NoKey            ;no then end
          movf    TempD,W          ;else output
          movwf   PORTB           ;low column scan line
          nop
          btfss   INTCON,RBIF      ;flag set?
          goto    ScanNext         ;no then next
          btfsc   KeyFlag,keyhit   ;last key released?
          goto    SKreturn         ;no then exit
          bsf     KeyFlag,keyhit   ;set new key hit
          movf   PORTB,W          ;read port
          movwf   TempE            ;save in TempE
          swapf   TempE,F
          call    GetKeyValue      ;get key value 0 - B
          movwf   NewKey
          movlw   0x0A
          subwf   NewKey,w
          btfsc   STATUS,Z
          goto    Akey
          incf    NewKey,f
          goto    Done
 Akey
          movlw   0x00
          movwf   NewKey           ;save as NewKey
Done       bsf     KeyFlag,ServKey ;set service flag
          bsf     KeyFlag,DebnceOn ;set flag
          movlw   2
          movwf   Debnce           ;load debounce time
SKreturn
          call    RestorePorts     ;restore ports
          return
;
NoKey
          bcf     KeyFlag,keyhit   ;clr flag
          goto    SKreturn
;
```

```
;GetKeyValue gets the key as per the following layout
;
;                  Col1    Col2    Col3
;                  (RB2)   (RB1)   (RB0)
;
;Row1(RB4)           0       1       2
;
;Row2(RB5)           3       4       5
;
;Row3(RB6)           6       7       8
;
;Row4(RB7)           9       A       B
;
GetKeyValue
          clrf    TempC
          btfss   TempD,2          ;first col.
          goto    RowValEnd
          incf    TempC, F
          btfss   TempD,1          ;second col.
          goto    RowValEnd
          incf    TempC,F          ;last col

RowValEnd
          btfss   TempE,0          ;top row?
          goto    GetValCom        ;yes then get 0,1,2
          btfss   TempE,1          ;2nd row?
          goto    Get345           ;yes the get 3,4,5,
          btfss   TempE,2          ;3rd row?
          goto    Get678           ;yes then get 6,7,8
Get9ab
          movlw   3
          addwf   TempC,F
Get678
          movlw   6
          addwf   TempC,F
          goto    GetValCom        ;do common part
Get345
          movlw   3
          addwf   TempC,F
GetValCom
          movf    TempC,W
          addwf   PCL, F
          retlw   0
          retlw   1
          retlw   2
          retlw   3
          retlw   4
          retlw   5
          retlw   6
          retlw   7
          retlw   8
          retlw   0a
          retlw   0a
          retlw   0a

;
;SavePorts, saves the porta and portb condition during a key scan
;operation.
SavePorts
          movf    PORTA,W          ;Get sink value
          movwf   PABuf            ;save in buffer
          bcf     PORTA,0          ;disable all sinks
          bcf     PORTA,1
          movf    PORTB,W          ;get port b
          movwf   PBBuf            ;save in buffer

          movlw   0xf7             ;make all high
          movwf   PORTB            ;on port b
          bsf     STATUS,RP0        ;select page 1
          bcf     OptionReg,7       ;enable pull ups
          movlw   B'11110000'       ;port b hi nibble inputs
          movwf   TRISB             ;lo nibble outputs
          bcf     STATUS,RP0        ;page 0
          return
;
;RestorePorts, restores the condition of porta and portb after a
```

```
;key scan operation.
RestorePorts
        movf    PBBuf,W         ;get port b
        movwf   PORTB
        movf    PABuf,W         ;get port a value
        movwf   PORTA
        bsf     STATUS,RP0      ;select page 1
        bsf     OptionReg,7     ;disable pull ups
        clrf    TRISA           ;make port a outputs
        bsf     TRISA,3         ;exept RA3
        clrf    TRISB           ;as well as PORTB
        bcf     STATUS,RP0      ;page 0
        return
;
UpdateDisplay
        movf    PORTA,W        ;present sink value in w
        bcf     PORTA,0         ;disable all digits sinks
        bcf     PORTA,1

        andlw   0x03
        movwf   TempC           ;save sink value in tempC

        bsf     TempC,2         ;preset for lsd sink
        rrf     TempC, F        ;determine next sink value
        btfss   STATUS,C        ;c=1?
        bcf     TempC,1         ;no then reset LSD sink
        btfsc   TempC,0         ;else see if Msd
        goto    UpdateMsd       ;yes then do Msd
UpdateLsd
        movf    MsdTime,W       ;get Lsd in w
        andlw   0x0f            ;
        goto    DisplayOut
UpdateMsd
        swapf   MsdTime,W       ;get 2nd Lsd in w
        andlw   0x0f            ;mask rest

DisplayOut
        call    LedTable        ;get digit output
        movwf   PORTB          ;drive leds
        movf    TempC,W         ;get sink value in w
        iorwf   PORTA,F
        return
;
;
LedTable
        addwf   PCL, F          ;add to PC low
        retlw   B'01110111'     ;led drive for 0
        retlw   B'00000110'     ;led drive for 1
        retlw   B'10110011'     ;led drive for 2
        retlw   B'10010111'     ;led drive for 3
        retlw   B'11000110'     ;led drive for 4
        retlw   B'11010101'     ;led drive for 5
        retlw   B'11110101'     ;led drive for 6
        retlw   B'00000111'     ;led drive for 7
        retlw   B'11110111'     ;led drive for 8
        retlw   B'11000111'     ;led drive for 9


ReadSwitch
        movf        PORTA,W
        movwf       VarA
        btfs        VarA,4
        bsf         KeyFlag,Sflag
        movf        PORTC,W
        movwf       Var
        btfsc       Var,0
        goto        SwB
        bsf  KeyFlag,NodeA
        bcf     KeyFlag,5
        bcf     KeyFlag,6
        bcf     KeyFlag,7
        goto    SwExit
SwB     btfsc   Var,1
        goto    SwM
        bsf     KeyFlag,NodeB
        bcf     KeyFlag,4
```

```
        bcf     KeyFlag,6
        bcf     KeyFlag,7
        goto    SwExit
SwM     btfsc   Var,2
        goto    SwN
        bsf     KeyFlag,NodeM
        bcf     KeyFlag,4
        bcf     KeyFlag,5
        bcf     KeyFlag,7
        goto    SwExit
SwN     btfsc   Var,3
        goto    SwExit
        bsf     KeyFlag,NodeN
        bcf     KeyFlag,4
        bcf     KeyFlag,5
        bcf     KeyFlag,6

SwExit  return

Combine btfss   KeyFlag,LEDA
        goto    CheckLEDB
        bsf     Save_byte,4
        bcf     Save_byte,5
        bcf     Save_byte,6
        bcf     Save_byte,7
CheckLEDB
        btfss   KeyFlag,LEDB
        goto    CheckLEDM
        bcf     Save_byte,4
        bsf     Save_byte,5
        bcf     Save_byte,6
        bcf     Save_byte,7
CheckLEDM
        btfss   KeyFlag,LEDM
        goto    CheckLEDN
        bcf     Save_byte,4
        bcf     Save_byte,5
        bsf     Save_byte,6
        bcf     Save_byte,7
CheckLEDN
        btfss   KeyFlag,LEDN
        goto    CheckExit
        bcf     Save_byte,4
        bcf     Save_byte,5
        bcf     Save_byte,6
        bsf     Save_byte,7
CheckExit
        return

NodeDisplay

        movlw   0x52
        movwf   FSR
        movf    INDF,W
        andlw   0x0F
        call    BtoBCD
        movwf   MsdTime
        bcf     KeyFlag,Sflag
        return

BtoBCD  clrf    BCD2
        movwf   BCD1
gtenth  movlw   .10
        subwf   BCD1,W
        BTFSS   STATUS,C
        goto    over
        movwf   BCD1
        incf    BCD2, F
        goto    gtenth
over
        movf    BCD1,W
        swapf   BCD2,F
        iorwf   BCD2,W
        return
BCDtoB
        movf    MsdTime,W
```

```
          movwf   R0
          clrf    Save_byte
          swapf    R0,W
          andlw    0F
          movwf   Save_byte
          call    mpy10a          ; result = 10a
          movfw   R0
          andlw   0F
          addwf   Save_byte,F
          return

mpy10a    bcf     STATUS,C        ; multiply by 2
          rlf     Save_byte,W
          movwf   Save_temp       ;Save_temp) = 2*N
          bcf     STATUS,C        ; multiply by 2
          rlf     Save_byte, F
          bcf     STATUS,C        ; multiply by 2
          rlf     Save_byte, F
          bcf     STATUS,C        ; multiply by 2
          rlf     Save_byte, F    ; (Save_byte) = 8*N

          movf    Save_temp,W
          addwf   Save_byte, F
          retlw   0               ; (H_byte,L_byte) = 10*N
          end
```