ระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ

EXPERT SYSTEM FOR ASSIGNING PARKING SPACE

นายนิติกานต์ จันทร์อินทร์ รหัส 45360229

นายวสกร ภักดีสาร รหัส 45360393

นายอภิชาติ เงินอินต๊ะ รหัส 45360583

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2548

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน ระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ

ผู้ดำเนินโครงงาน นายนิติกานต์ จันทร์อินทร์ รหัส 45360229

นายวสกร ภักดีสาร รหัส 45360393

นายอภิชาติ เงินอินต๊ะ รหัส 45360583

อาจารย์ที่ปรึกษา ดร.สุรเชษฐ์ กานต์ประชา

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2548

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์

คณะกรรมการการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ

(ดร.สุรเชษฐ์ กานต์ประชา)

.....กรรมการ

(ดร.สมยศ เกียรติวนิชวิไล)

.....กรรมการ

(อาจารย์พนัส นัถฤทธิ์)

หัวข้อโครงงาน	ระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ		
ผู้ดำเนินโครงงาน	นายนิติกานต์	จันทร์อินทร์	รหัส 45360229
	นายวสกร	ภักดีสาร	รหัส 45360393
	นายอภิชาติ	เงินอินต๊ะ	รหัส 45360583
อาจารย์ที่ปรึกษา	ดร.สุรเชษฐ์	กานต์ประชา	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2548		

บทคัดย่อ

โครงงานนี้ได้ศึกษาและพัฒนาระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ เพื่อช่วยลดระยะเวลาในการหาที่จอดรถ และลดการใช้ทรัพยากรพลังงานเชื้อเพลิงลง ในโครงงานนี้ได้พัฒนาด้วยโปรแกรมภาษา วิชวลเบสิก เวอร์ชัน 6.0 โดยโปรแกรมแบ่งออกเป็นสองส่วน ส่วนแรกเป็นส่วนของการจำลองเหตุการณ์การเข้าและออกของรถในลานจอดรถ ส่วนที่สองคือส่วนของระบบที่ทำการแนะนำตำแหน่งที่ว่างแก่ผู้ขับ

Project title	Expert system for assigning parking space		
Name	Mr. Nitikan	Chanin	ID. 45360229
	Mr. Wasakon	Pakdeesan	ID. 45360393
	Mr. Aphichat	Ngerninta	ID. 45360583
Project advisor	Dr. Surachet	Kanprachar	
Major	Computer Engineering		
Department	Electrical and Computer Engineering		
Academic year	2005		

.....

Abstract

The project has been studied and developed for Expert System to identify car parking space. It can help decreasing the time for searching the parking space and consequently the energy to be used will be reduced. In this project, Microsoft Visual Basic 6.0 is adapted as a tool to help in terms of software programming. The program is divided into 2 parts; That is, simulating the incoming and out-going car for the car park and assigning the appropriate parking space to the incoming car.

กิตติกรรมประกาศ

โครงงานฉบับนี้คงไม่อาจสำเร็จลงได้ หากไม่ได้รับความช่วยเหลือ และการให้คำแนะนำปรึกษาจาก ท่านอาจารย์ ดร.สุรเชษฐ์ กานต์ประชา ขอขอบคุณมากครับ

ขอขอบคุณอาจารย์ทุกท่านในภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ มหาวิทยาลัยนเรศวร ที่ได้สั่งสอนและให้ความรู้แก่เรา ซึ่งสามารถนำไปประยุกต์ใช้ในการทำโครงงานนี้ได้อย่างมากมาย

และที่สุดคือพวกเราเอง ต้องขอบคุณซึ่งกันและกันที่เป็นเพื่อนร่วมงานที่ดี จนทำให้โครงงานนี้สำเร็จลงได้

นายนิติกานต์ จันทร์อินทร์
นายวสกร ภักดีสาร
นายอภิชาติ เงินอินต๊ะ

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของโครงงาน

ปัจจุบันในการจอดรถในอาคารใหญ่ๆ หรือในห้างสรรพสินค้าที่ไม่มีลานจอดรถกลางแจ้ง แต่มีที่จอดรถอยู่ภายในตัวอาคารแทนมักจะพบปัญหาในการหาที่จอดรถ เนื่องจากไม่ทราบว่ามีที่จอดที่ใดว่างบ้าง โดยทั่วไปแล้วผู้ใช้บริการมีวิธีการในการหาที่จอดรถ คือ ผู้ไปใช้บริการต้องขับรถเพื่อหาที่จอดรถเองจนกว่าจะพบที่จอดที่ว่าง และเป็นตำแหน่งที่พอใจ ซึ่งในกรณีที่ชั้นที่เพิ่งขับผ่านมาว่าง แต่ชั้นที่กำลังขับขึ้นไปจอดไม่ว่าง ทำให้ต้องขับกลับลงมาจอดชั้นที่ผ่านมาอีก ซึ่งอาจจะมีรถคันอื่นเข้ามาจอดก่อน เป็นเหตุให้สูญเสียเวลาในการขับหาที่จอดรถใหม่ และสิ้นเปลืองทรัพยากรพลังงานเชื้อเพลิงเป็นอย่างยิ่ง

จะเห็นได้ว่าในการจอดรถแต่ละครั้งสิ้นเปลืองเวลา หากผู้ขับหาที่จอดรถที่ว่างที่อยู่ใกล้ที่สุดไม่ได้ และยังส่งผลถึงการใช้ทรัพยากรพลังงานเชื้อเพลิงที่มากขึ้นตามไปด้วย ประกอบกับราคาน้ำมันเชื้อเพลิงปรับราคาสูงขึ้นอยู่ทุกขณะ การประหยัดเชื้อเพลิงจึงเป็นสิ่งสำคัญที่ไม่อาจมองข้าม

เพื่อแก้ปัญหาดังกล่าวปัญหาจึงได้ทำการศึกษาและพัฒนา ระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ (Expert system for assigning parking space) เพื่อช่วยแก้ปัญหาดังกล่าว

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อพัฒนาระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ ที่ช่วยลดระยะเวลาในการหาที่จอดรถของผู้ขับและช่วยลดความสิ้นเปลืองทรัพยากรพลังงานเชื้อเพลิง

1.2.2 ศึกษาการสร้าง และพัฒนาโปรแกรม ด้วย วิชวลเบสิก 6

1.2.3 พัฒนาทักษะในการออกแบบระบบและการเขียนโปรแกรม

1.2.4 ศึกษาการติดต่อสื่อสารระหว่าง คอมพิวเตอร์ และไมโครคอนโทลเลอร์

1.3 ขอบเขตของโครงงาน

1.3.1 วิเคราะห์และออกแบบระบบเพื่อช่วยตัดสินใจในการหาที่จอดรถแก่ผู้ขับ

1.3.2 พัฒนาโปรแกรม จากระบบที่ออกแบบขึ้น

1.3.3 พัฒนาโปรแกรม จำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ

1.3.4 แสดงผลที่ได้จาก Application ผ่านทางแผงวงจร LED

1.4 ขั้นตอนการดำเนินงาน

1.4.1 ศึกษาเหตุการณ์ต่างๆ ในการเข้าจอดในลานจอดรถของผู้ขับ

1.4.2 ค้นคว้าและศึกษาโปรแกรมที่จะนำมาออกแบบการจำลองเหตุการณ์การเข้าจอดรถในลาน จอดรถ

1.4.3 พัฒนาโปรแกรม เพื่อจำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ

1.4.4 วิเคราะห์และออกแบบระบบเพื่อช่วยตัดสินใจในการหาที่จอดรถแก่ผู้ขับ

1.4.5 พัฒนาโปรแกรม จากระบบที่ออกแบบขึ้น

1.4.6 ศึกษาการเชื่อต่อระหว่าง คอมพิวเตอร์ และไมโครคอนโทลเลอร์ ทางซีเรียลพอร์ท

1.4.7 เขียนโปรแกรมติดต่อระหว่าง คอมพิวเตอร์ และไมโครคอนโทลเลอร์ เพื่อแสดงผลของโปรแกรมออกทางแผงวงจร LED

1.4.8 ออกแบบ และสร้างแผงวงจร LED เพื่อใช้ในการแสดงผล

1.4.9 ทดสอบการทำงาน และแก้ไขข้อผิดพลาด

1.4.10 สรุปผลการทดลองและจัดทำรูปเล่มโครงงาน

1.5 แผนการดำเนินงาน

ตารางที่ 1.1 แสดงขั้นตอนการดำเนินงาน

กิจกรรม	ปี 2547		ปี 2548									
	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.
1. ศึกษาเหตุการณ์ต่างๆ ในการเข้าจอดรถของผู้ขับ	↔											
2. ค้นคว้าและศึกษาโปรแกรมที่จะนำมาออกแบบการจำลองเหตุการณ์การเข้าจอดรถ		↔										
3. พัฒนาโปรแกรม เพื่อจำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ		↔	↔	↔								
4. วิเคราะห์และออกแบบระบบเพื่อช่วยตัดสินใจในการหาที่จอดรถแก่ผู้ขับ				↔	↔							
5. พัฒนาโปรแกรม จากระบบที่ออกแบบขึ้น						↔	↔					
6. ศึกษาการเชื่อมต่อระหว่างคอมพิวเตอร์ และไมโครคอนโทลเลอร์ ทางซีเรียลพอร์ต							↔	↔				
7. เขียนโปรแกรมติดต่อระหว่าง คอมพิวเตอร์ และไมโครคอนโทลเลอร์ เพื่อแสดงผลออกทางแผง LED								↔	↔	↔		
8. ออกแบบ และสร้างแผงวงจร LED เพื่อใช้ในการแสดงผล										↔	↔	
9. ทดสอบการทำงาน และแก้ไขข้อผิดพลาด											↔	
10. สรุปผลการทดลองและจัดทำรูปเล่มโครงงาน												↔

1.6 ผลที่คาดว่าจะได้รับ

1.6.1 โปรแกรม ที่พัฒนาขึ้นสามารถช่วยในการตัดสินใจในการหาที่จอดรถแก่ผู้ขับได้

1.6.2 สามารถแสดงผลของโปรแกรม ผ่านทางแผงวงจร LED ได้

1.6.3 เข้าใจการพัฒนาโปรแกรม ด้วย วิชวลเบสิก 6

1.6.7 เข้าใจการเชื่อมต่อ ระหว่างคอมพิวเตอร์ และไมโครคอนโทลเลอร์ ผ่านทางซีเรียลพอร์ต และการออกแบบวงจรอิเลคทรอนิคส์

1.7 งบประมาณที่ใช้

1.7.1 ค่าจัดทำรายงาน

1.7.2 ค่าหมึกพิมพ์

1.7.3 ค่าแผ่นซีดี

รวมเป็นเงิน 3,000 บาท(สามพันบาทถ้วน)

บทที่ 2
ทฤษฎีและหลักการ

การที่จะสร้าง ระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ (Expert system for assigning parking space) นั้นจำเป็นจะต้องมีความรู้ในด้าน การพัฒนาแอพพลิเคชันด้วย Visual Basic เป็นอย่างดี เหตุผลที่ใช้ Visual Basic ก็เนื่องมาจาก Visual Basic สามารถสร้างแอพพลิเคชันบน Windows ได้อย่างรวดเร็ว และยังต้องใช้ความรู้ด้านไมโครโปรเซสเซอร์เพื่อที่จะสามารถควบคุมแผงวงจรได้อย่างมีประสิทธิภาพ

2.1 การพัฒนาแอพพลิเคชันสมัยใหม่ [1]

แต่ก่อนนั้นการสร้างแอพพลิเคชัน (Application) ด้วยการเขียนโปรแกรม โดยในอดีตนั้นนักเขียนโปรแกรม หรือโปรแกรมเมอร์ต่างต้องผ่านความยุ่งยากซับซ้อนของการเขียนโปรแกรมมาด้วยกันทุกคน ทำให้มองกันว่าการเขียนโปรแกรมเป็นเรื่องที่ซับซ้อน

การใช้งาน Windows นั้นเป็นเรื่องที่แสนจะง่าย แต่การสร้างแอพพลิเคชันให้ทำงานกับ Windows นั้นกลับทำได้ยาก แต่เดิมนั้นผู้ที่จะสร้างแอพพลิเคชันในการทำงานบน Windows จำเป็นต้องรู้การทำงานภายใน Windows เป็นอย่างดีเพื่อที่จะเขียนโปรแกรมได้อย่างถูกต้อง

2.1.1 การพัฒนาแอพพลิเคชันด้วย Visual Basic

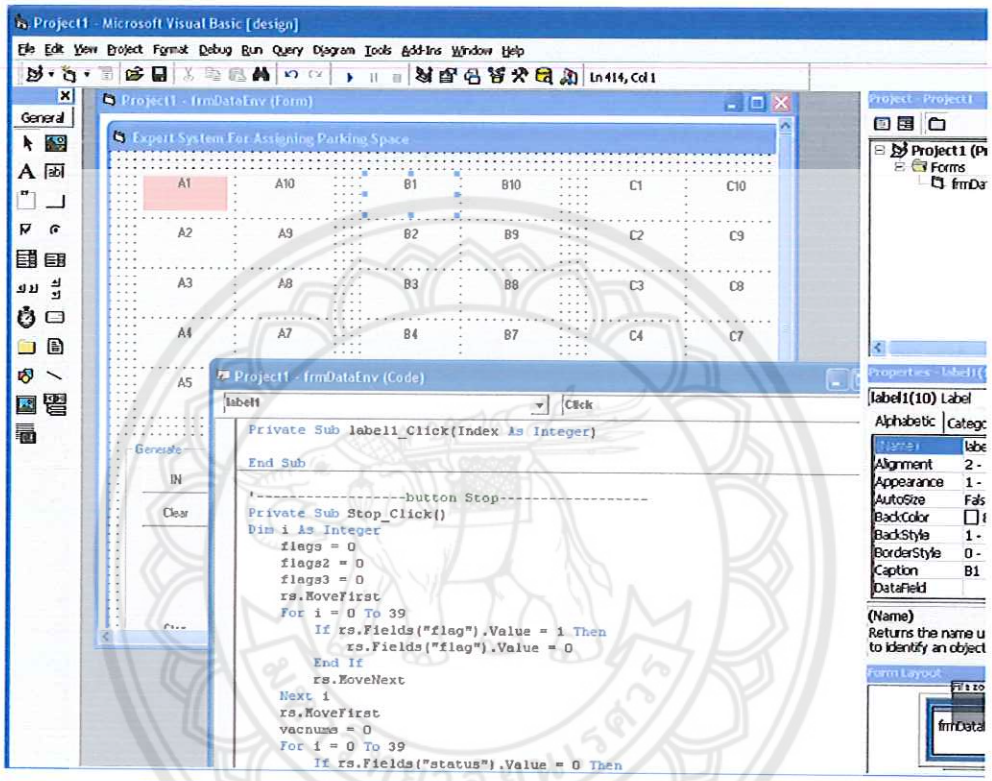
ภาษาในการเขียนโปรแกรมถูกสร้างขึ้นมาด้วยวัตถุประสงค์ที่แตกต่างกันแต่ละภาษามีจุดเด่นที่ต่างกันออกไป จึงเหมาะกับงานคนละประเภท เช่น ภาษา C เป็นภาษาที่นักเขียนโปรแกรมทั่วโลกนิยมกันมากเพราะมีความยืดหยุ่นสูง ทำงานได้เร็ว แต่โปรแกรมเมอร์ที่เขียนโปรแกรมเกี่ยวกับฐานข้อมูลกลับไม่นิยมใช้ สู้จะใช้ภาษา Basic ใน Visual Basic ไม่ได้

เพื่อที่เป็นทางเลือกในการสร้างแอพพลิเคชันที่หลากหลายบน Windows ทำได้อย่างรวดเร็ว ไมโครซอฟท์ได้แนะนำ Visual Studio ซึ่งเป็นเครื่องมือที่สำหรับสร้างแอพพลิเคชัน Windows ที่มีเครื่องมือให้โปรแกรมเมอร์ได้เลือกใช้งาน

2.1.2 จุดเด่นของ Visual Basic

2.1.2.1 สามารถสร้างแอพพลิเคชันได้อย่างรวดเร็ว

Visual Basic ได้รับการวางตัวให้เป็นเครื่องมือที่ช่วยให้สร้างแอพพลิเคชันได้อย่างรวดเร็วและง่ายเพื่อที่จะได้ลดเวลาในการสร้างแอพพลิเคชันลงซึ่งเรียกรูปแบบนี้ว่า Rapid Application Development หรือ RAD ดังรูปที่ 2.1

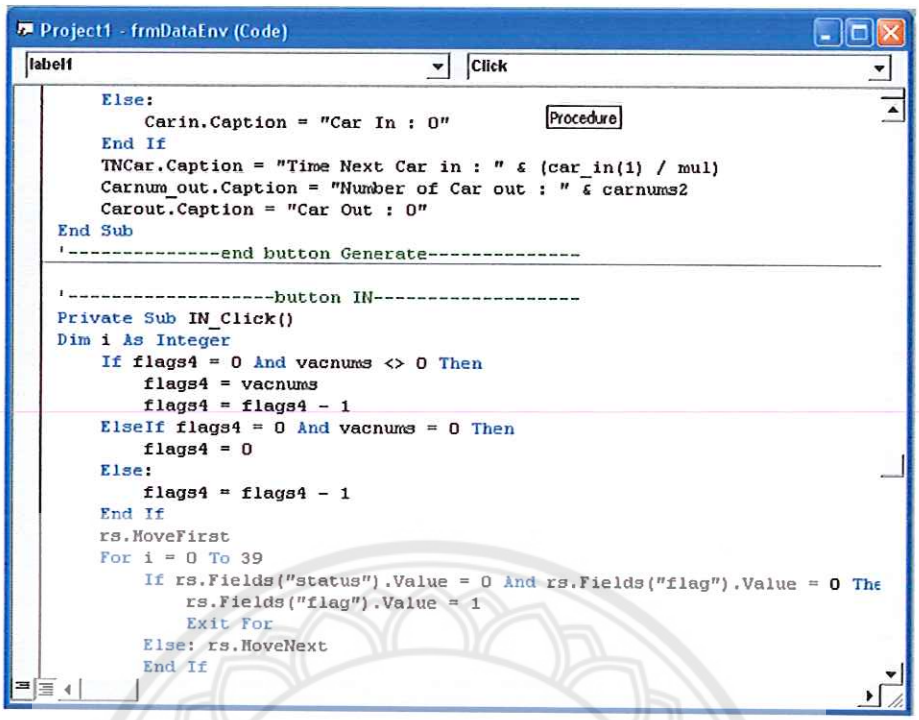


รูปที่ 2.1 ภาพแสดงโปรแกรม วิชวลเบสิก 6

ทั้งนี้เพราะมีการขจัดงานที่โปรแกรมเมอร์ต้องทำซ้ำๆ ออกไปขจัดสิ่งที่ไม่จำเป็นต้องรู้เกี่ยวกับการควบคุม Hardware การจักการภายในของ windows ออกไปเหลือเฉพาะที่ต้องโฟกัสเกี่ยวกับปัญหาของงานจริงๆแล้วเขียนโปรแกรมจัดการปัญหานั้น และส่วนอื่นก็ให้ Visual Basic จัดการ

2.1.2.2 ภาษาเขียนโปรแกรมที่ง่ายต่อการเริ่มเรียนรู้

ถ้ามีโอกาสที่จะเขียนโปรแกรมด้วย Visual Basic แล้วจะเห็นว่า ภาษา Basic นั้นอ่านง่ายเพราะใกล้เคียงกับภาษาที่เราใช้อยู่ในชีวิตประจำวันอ่านแล้วเข้าใจได้ง่ายกว่าภาษาอื่นๆ ทำให้ผู้ที่เริ่มเขียนโปรแกรมทำความเข้าใจได้ง่ายกว่าการเขียนโปรแกรมด้วยภาษาอื่น



รูปที่ 2.2 ภาพแสดงการเขียนโปรแกรมใน วิชวลเบเสิก 6

จากรูปที่ 2.2 เป็นการแสดงการเขียนโปรแกรมโดยใช้ Visual Basic แม้ว่าบรรดานักเขียนโปรแกรมหลายๆ คนจะว่าภาษา Basic ใน Visual Basic นั้นมีโครงสร้างไม่ค่อยดีนักเมื่อเทียบกับภาษาปาสคาล หรือ ภาษาจาวา แต่ต้องไม่ลืมว่าภาษา Basic ใช้เวลาในการเรียนรู้ที่เร็วกว่า

2.1.2.3 รวมเครื่องมืออำนวยความสะดวกในการเขียนโปรแกรม

นอกจากง่ายต่อการเรียนรู้แล้ว Visual Basic ยังมีเครื่องมือที่ช่วยในการเขียนโปรแกรมเป็นเรื่องที่ไม่ยุ่งยากอีกต่อไปเพราะจะมีเครื่องมือที่ช่วยให้ไม่ต้องจดจำไวยากรณ์ภาษาที่ยุ่งยาก ตรวจสอบอัตโนมัติว่าโปรแกรมที่เขียนนั้นถูกต้องหรือไม่ มีการแยกแยะส่วยของโปรแกรมอย่างเป็นระเบียบทำให้งานของโปรแกรมเมอร์ลดลงไปได้มาก

นอกจากจะมีเครื่องมือที่ช่วยในกรเขียนโปรแกรมแล้ว ยังมีเครื่องมือทดสอบแก้ไขโปรแกรม (Debugger) ที่เขียนขึ้นมาว่าทำงานได้อย่างถูกต้องหรือไม่ มีรับขอความช่วยเหลือ (Online Help) ไว้อ้างอิง และข้อความช่วยเหลือในจุดที่เราสงสัย

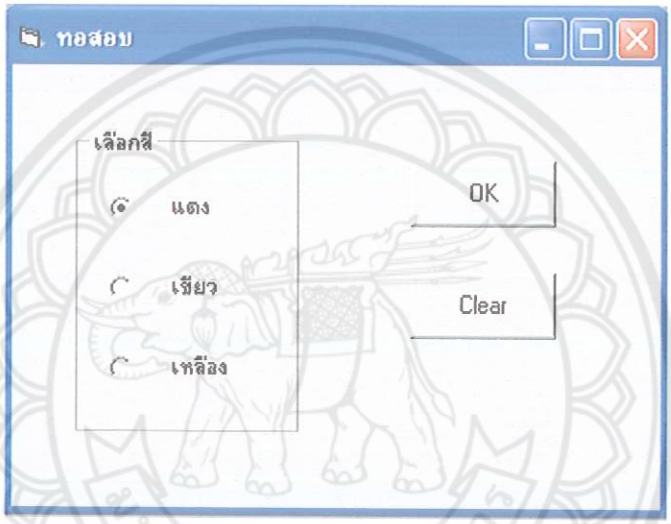
เครื่องมือทั้งหมดที่กล่าวมาถูกจัดรวมไว้ในสภาพแวดล้อมการทำงายเดียวกัน (Integrated Development Environment) ทำให้เรียกใช้งานได้สะดวกตั่งแต่เริ่มเขียนโปรแกรม ทดสอบโปรแกรม แก้ไข สร้างชุดติดตั่ง รวมทั้งระบบขอความช่วยเหลือ ซึ่งเราสามารถเพิ่มเติมเครื่องมือชนิดใหม่เข้าไปได้เรื่อยๆ หรือถอดเครื่องมือที่ไม่จำเป็นต่อการใช้งานออกไปเพื่อที่ประหยัดพื้นที่ของฮาร์ดดิสก์

2.1.3 รูปแบบการพัฒนาแอพพลิเคชันกับ Visual Basic [1]

เมื่อเรามองเห็นว่า Visual Basic จะช่วยให้เราสร้างแอพพลิเคชันบน windows ได้ง่าย และรวดเร็วแล้วเราลองมาสำรวจดูว่าแอพพลิเคชันที่สามารถใช้ Visual Basic สร้างขึ้นมามีรูปแบบใดบ้าง

2.1.3.1 พัฒนาแอพพลิเคชันแบบ ActiveX Control

การเขียนโปรแกรมที่รับค่าข้อมูลจากผู้ใช้แต่เดิมเราต้องมาเขียนโปรแกรมวาดหน้าจอเอง เขียนโปรแกรมวาดปุ่ม และช่องรับข้อความ รวมทั้งเขียนโปรแกรมเกี่ยวกับการจัดการข้อมูลจากผู้ใช้งานโดยช่องรับ ข้อมูล ปุ่มต่างๆ จะใช้ ActiveX Control จัดการ



รูปที่ 2.3 ภาพแสดงการใช้ ActiveX Control ใน วิชวลเบสิก 6

ดังตัวอย่างโปรแกรมในรูปที่ 2.3 เมื่อเราเลือกสีแล้วกดปุ่ม OK พื้นหลังก็จะเปลี่ยนสีไปตามที่เราได้เลือกไว้ จะเห็นได้ว่านอกจากการลดความซับซ้อนแล้วการที่เราใช้ ActiveX Control ในการเขียนโปรแกรมช่วยให้โปรแกรมที่เราเขียนกับโปรแกรมที่คนอื่นเขียนนั้นตั้งอยู่บนมาตรฐานเดียวกันทำให้การบำรุงรักษาเป็นไปได้ง่าย เพราะใครๆ ก็เข้าใจมาตรฐานของ ActiveX Control นี้ทำให้ไม่ต้องกังวลใจว่า โปรแกรมที่เขียนนั้นจะมีเฉพาะเราคนเดียวที่เข้าใจ

2.1.3.2 สร้างแอพพลิเคชันที่ใช้งานกับฐานข้อมูล

Visual Basic ได้ช่วยให้การสร้างแอพพลิเคชันกับฐานข้อมูลเป็นเรื่องที่ทำได้ง่าย เพราะมีเครื่องมีอำนวยความสะดวกในการเขียนโปรแกรมเพื่อที่ใช้งานในการจัดการข้อมูล ซึ่งไม่จำเป็นว่าจะต้องเป็นฐานข้อมูลแบบไหนทั้ง ฐานข้อมูลส่วนบุคคล ฐานข้อมูลผ่านเครือข่าย หรือ ฐานข้อมูลผ่านอินเตอร์เน็ต จากความสามารถที่หลายหลายของ Visual Basic นี้จึงทำให้เป็นตัวเลือกอันดับต้นๆ ของการสร้างแอพพลิเคชันที่เกี่ยวกับฐานข้อมูล

2.2 ไมโครโปรเซสเซอร์คืออะไร [2]

หน่วยประมวลผลกลางของคอมพิวเตอร์ (Central Processing Unit: CPU) จะถูกสร้างเป็นไอซี แบบ LSI ซึ่งสามารถคำนวณได้ภายใต้การควบคุมของโปรแกรม ซึ่งเราสามารถเรียกระบบไมโครโปรเซสเซอร์ได้อีกอย่างว่าหน่วยประมวลผลข้อมูล (Data Processing Unit) ดังนั้นหากจะสร้างระบบคอมพิวเตอร์จะต้องมีไอซีของไมโครโปรเซสเซอร์ ตัวไมโครโปรเซสเซอร์บางรุ่นมีหน่วยความจำประกอบอยู่ภายในตัวของมันเอง และบางรุ่นอาจจะมีระบบอินพุตและเอาต์พุตอยู่ในตัวเอง เรียกว่าไมโครคอนโทลเลอร์ (Microcontroller)

สำหรับคอมพิวเตอร์ที่เราใช้งานกันทั่วไป เช่น Personal Computer จะมีไมโครโปรเซสเซอร์ประกอบอยู่หลายรุ่น ซึ่งเป็นตัวที่บอกประสิทธิภาพของเครื่องด้วย เช่น 8088 80286 80386 80486 เป็นต้น หากมองไมโครโปรเซสเซอร์ง่ายๆ เราอาจมองเป็นไอซีดิจิตอลตัวหนึ่งจะทำงานได้จะต้องมีสัญญาณนาฬิกา ไปกระตุ้นเพื่อให้ไอซีไปอ่านรหัสคำสั่ง (Fetch) และปฏิบัติตามคำสั่ง (Execute) เพื่อให้ได้เอาต์พุตออกมาตามที่เราต้องการ ซึ่งคำสั่งทั้งหมดจะเก็บอยู่ในรูปแบบของเลขฐานสองสำหรับคำสั่งที่จะให้ไมโครโปรเซสเซอร์ทำงานนั้น จะเป็นชุดคำสั่ง (Instruction Set) ซึ่งออกแบบมาสำหรับซีพียูแต่ละรุ่น สำหรับการออกแบบไมโครโปรเซสเซอร์นั้นเราจะพิจารณาจากความสามารถในการประมวลผลข้อมูลวัดได้ 3 วิธี

1. ความยาวของเวิร์ดข้อมูล
2. ขนาดของหน่วยความจำของไมโครโปรเซสเซอร์อ้างได้
3. ความเร็วในการกระทำคำสั่ง

ความยาวของเวิร์ดข้อมูลที่ไมโครโปรเซสเซอร์จะประมวลผลมีตั้งแต่ 4 บิต 8 บิต 16บิต 32 บิต และ 64 บิต ถ้าเป็นการประมวลผล 16 บิต เราจะเรียกว่าเครื่อง 16 บิต ถ้าเป็นเครื่องขนาด 8 บิต เวิร์ดของข้อมูลจะมีขนาด 1 ไบต์ โดย 1ไบต์จะเท่ากับ 8บิต ถ้าเป็นเครื่องขนาด 16 บิต ก็จะเป็น 2 ไบต์ โดยการบิตที่ 8-15 เป็นบิตสูง ส่วนบิตที่ 0-7 จะเป็นบิตต่ำ

ถ้าคอมพิวเตอร์มีการประมวลผลแบบมีความยาวเวิร์ดสูง จะทำให้การประมวลผลแต่ละครั้งอ้างถึงตัวเลขได้มากกว่า ในอดีตไมโครโปรเซสเซอร์ขนาด 4 บิต จะถูกนำไปใช้อันดับแรกเพราะข้อมูลขนาด 4 บิตนี้สามารถใช้แทน BCD ได้ และต่อมามีการพัฒนาเป็น 8 บิต เนื่องจาก 8 บิตมีความยาวของเวิร์ดเป็น 2 เท่า ของ 4 บิตและ 8 บิตจะสามารถแทนตัวเลขใน BCD ได้ถึงสองตัว นอกจากนี้ 8 บิตยังมีตัวเลขพอที่จะเก็บเป็นตัวอักขระ 1 ตัวแทน ASCII อีกด้วย ถ้าเวิร์ดของข้อมูลมีขนาดสูงขึ้น เช่น ไมโครโปรเซสเซอร์ขนาด 32 บิต สามารถอ้างอิงถึงข้อมูลได้ถึง 4000 ล้านตำแหน่ง ซึ่งจะได้ความละเอียดถึง 1ส่วนใน 4000 ล้าน แต่เป็นไมโครโปรเซสเซอร์ขนาด 8 บิต จะมีความละเอียด 1 ส่วนใน 256 ส่วนเท่านั้น อีกประการหนึ่งของไมโครโปรเซสเซอร์ที่มีบิตสูงจะสามารถที่จะอ่านข้อมูลจากหน่วยความจำได้มากกว่า เช่น ไมโครโปรเซสเซอร์ขนาด 32 บิต จะอ่าน

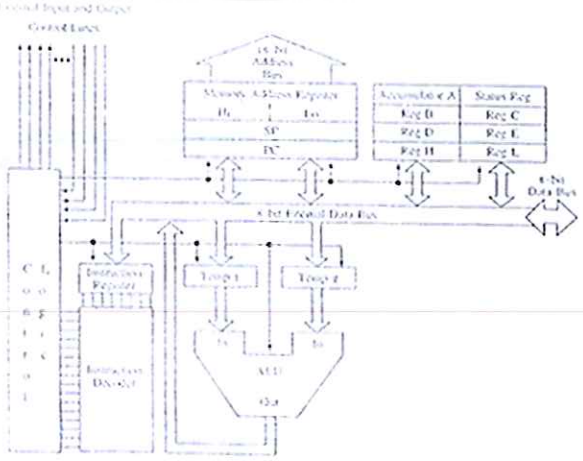
ข้อมูลได้มากกว่าแบบ 8 บิตถึง 4 เท่าเพราะ 32 บิต อ่านข้อมูลได้ทีละ 4 ไบต์แต่ 8 บิตอ่านข้อมูลได้ทีละไบต์

สำหรับหน่วยความจำที่ไมโครโปรเซสเซอร์อ้างถึง จะเป็นตัวบอกว่าเครื่องคอมพิวเตอร์ที่ใช้อยู่สามารถที่จะติดต่อกับหน่วยความจำได้มากที่สุดเท่าไร โดยปกติแล้วการพัฒนาไมโครโปรเซสเซอร์ให้ดีขึ้นนอกจากจะพัฒนาให้ความยาวของเวิร์ดสูงขึ้นแล้ว จะทำการพัฒนาให้อ่านข้อมูลในหน่วยความจำได้เร็วขึ้นด้วยขนาดของหน่วยความจำนี้จะใช้ตัวอักษร K หมายความว่าเป็นพัน M หมายความว่าเป็นล้าน G หมายความว่าเป็นพันล้าน

สำหรับความเร็วของไมโครโปรเซสเซอร์หมายถึง ความเร็วในการปฏิบัติตามคำสั่งจะวัดความเร็วของสัญญาณนาฬิกา (Clock Speed) ที่ให้กับไมโครโปรเซสเซอร์โดยทั่วไปจะมีหน่วยเป็นเมกะเฮิรตซ์หรือหวังความเร็วจากหน่วยคำสั่งที่ไมโครโปรเซสเซอร์ เอกซีคิวต์ได้ในหนึ่งวินาทีโดยทั่วไปจะใช้เป็นหน่วยล้านคำสั่งต่อหนึ่งวินาที ถ้าสัญญาณนาฬิกากับไมโครโปรเซสเซอร์สูง จะทำให้เวลาในการทำงานเร็วขึ้น เช่น ไมโครโปรเซสเซอร์ตัวหนึ่งถ้าบวกเลขจะต้องใช้สัญญาณนาฬิกาสองลูก ถ้าให้ความถี่ของสัญญาณนาฬิกาสูงขึ้นจะทำให้เลาของแต่ละลูกของสัญญาณสั้นลง

2.2.1 โครงสร้างภายในของไมโครโปรเซสเซอร์

ลักษณะภายในของไมโครโปรเซสเซอร์ ถ้าจะเขียนให้ละเอียดจะได้ดังรูปที่ 2.4 ซึ่งจะเห็นว่าภายในจะมีการแบ่งเป็นหน่วยย่อยๆ มากมายและมีการเชื่อมโยงระหว่างหน่วยความจำต่างๆ โดยใช้สายสัญญาณที่เรียกว่า บัส(BUS) รหัสจะถูกเขียนแทนด้วยลูกศรหนา ๆ ซึ่งแบ่งออกเป็นรหัสทางบัสเดียว (Directional Bus) บัสทางเดียวจะใช้ลูกศรหัวเดียว ส่วยบัสสองทางจะใช้ลูกศรสองหัว ซึ่งจะมีการส่งสัญญาณได้สองทิศทาง ระบบบัสที่เห็นในรูปเรียกว่าบัวภายใน ซึ่งเป็นการเชื่อมโยงสัญญาณภายใน CPU



รูปที่ 2.4 ภาพแสดงโครงสร้างภายในของไมโครโปรเซสเซอร์ [2]

ภายในไมโครโปรเซสเซอร์จะประกอบไปด้วยโครงสร้างที่สำคัญ 3 ส่วนคือ

1. หน่วยคำวนและลอจิก(Arithmetic/Logic Unit :ALU)
2. รีจิสเตอร์ (Register)
3. ส่วยควบคุม (Control Logic)

หน่วยคำวนแบบลอจิก ALU จะทำหน้าที่ประมวลผลข้อมูลในไมโครโปรเซสเซอร์ ซึ่งมีอินพุตสองพอร์ตส่งข้อมูลเข้าไปในหน่วยประมวลผล และได้เอาต์พุตออกมาหนึ่งค่า ซึ่งภายใน ALU นี้จะทำการคำนวณและกระทำทางลอจิกต่างๆ เอาต์พุตที่ได้จะเก็บไว้ในริจิสเตอร์พิเศษที่เรียกว่า Accumulator Register ส่วนของรีจิสเตอร์จะเป็นพื้นที่หนึ่งภายในไมโครโปรเซสเซอร์ที่ใช้ในการเก็บข้อมูลชั่วคราวในช่วงเวลาที่ทำการ Execute โปรแกรม สำหรับหน่วยควบคุมจำทำหน้าที่ควบคุมสัญญาณทั้งหมดภายในไมโครโปรเซสเซอร์

การอินเตอร์รัพท์ คือ การติดต่อเพื่อรับส่งข้อมูลกันระหว่างอุปกรณ์ภายนอกต่างๆของคอมพิวเตอร์ เช่น จอภาพ, แป้นพิมพ์, เครื่องพิมพ์, เมาส์ และ อื่นๆ กับ ไมโครโปรเซสเซอร์ ซึ่งจะมีการติดต่อกันอยู่เสมอๆ การที่จะทำให้ระบบมีประสิทธิภาพมากที่สุดนั้นก็คือ การมีการติด ต่อหรือการ อินเทอร์รัพท์ที่ดีนั่นเอง

2.2.2 การอินเตอร์รัพท์ [2]

ในระบบที่มีอุปกรณ์ต่างๆหลายชนิดนั้น บางครั้งเราอาจจะสงสัยว่าในขณะที่นาฬิกาของเครื่องเดินไปได้ตลอดเวลาพร้อมๆกับที่เราใช้งานอย่างอื่นโดยที่เวลาเดินไปอย่างไม่ผิดพลาด หรือว่าเครื่องได้อ่านแผ่นดิสก์อยู่ ในขณะเดียวกับที่เราพิมพ์ข้อมูลผ่านทางแป้นพิมพ์ได้ ซึ่งเสมือนว่าเครื่องสามารถทำงานได้หลายๆอย่างในเวลาเดียวกัน ซึ่งการที่จะทำได้เช่นนี้นั้น ซีพียูจำเป็นต้องมีวิธีการติดต่อกับอุปกรณ์ภายนอกต่างๆอย่าง มีประสิทธิภาพ

วิธีหนึ่งก็คือ ให้ซีพียูใช้เวลาส่วนใหญ่ในการทำโปรแกรมหลัก และหันมาสนใจอุปกรณ์ภายนอกก็ต่อเมื่ออุปกรณ์ภายนอกส่งสัญญาณ เข้ามาขัดจังหวะการทำงานที่ซีพียูทำอยู่ในขณะนั้น ซึ่งซีพียูมีสิทธิที่จะยอมรับหรือปฏิเสธการขัดจังหวะนั้นก็ได้ อาจเปรียบได้ว่าเรากำลังเขียนรายงานอยู่ แล้วมีเสียงโทรศัพท์ดังขึ้นมา เราก็จะต้องตัดสินใจว่าสิ่งใดมีความสำคัญมากกว่ากัน หากคิดว่าโทรศัพท์สำคัญกว่า ก็หยุดการ เขียนรายงานและไปรับโทรศัพท์ก่อน แล้วจึงกลับมาเขียนรายงานต่อ ดังนั้นในลักษณะนี้ หากว่าซีพียูยอมรับการขัดจังหวะนั้น ซีพียู จะหยุดจากการทำงานในโปรแกรมหลักและไปทำงานในโปรแกรมย่อยซึ่งอุปกรณ์นั้นๆร้องขอมา เมื่อเสร็จแล้ว ซีพียูก็จะกลับมาทำโปรแกรมเดิมที่ หยุดไปต่อ ไป ลักษณะเช่นนี้เรียกว่า การอินเตอร์รัพท์ (Interrupt)

2.2.2.1 ประเภทของการอินเตอร์รัพท์

การอินเตอร์รัพท์สามารถแบ่งได้เป็น 2 ประเภทใหญ่ๆดังนี้คือ

- **Non-mask able Interrupt (NMI)** คือ การอินเตอร์รัพท์ที่ซีพียูไม่สามารถปฏิเสธได้ หากมีการอินเตอร์รัพท์ประเภทนี้มา ซีพียูต้องหยุดการทำงานของโปรแกรมที่ทำอยู่ในขณะนั้นโดยไม่มีข้อแม้ และเปลี่ยนไปทำงานให้การอุปกรณ์ที่มีการส่งอินเตอร์รัพท์เข้ามา

- **Mask able Interrupt (INT)** คือ การอินเตอร์รัพท์ที่ซีพียูสามารถปฏิเสธได้ โดยปรกติ ซีพียูจะมีการกำหนดว่าในขณะนั้น ซีพียูจะอยู่ในภาวะ Disable Interrupt (DI) หรือ Enable (EI) ถ้าหากว่า ในขณะที่มีการอินเตอร์รัพท์เข้ามานั้น ซีพียูอยู่ในภาวะ Disable Interrupt แล้ว การอินเตอร์รัพท์นั้นก็จะได้รับการปฏิเสธไป แต่ถ้าเป็น Enable Interrupt การอินเตอร์รัพท์นั้นก็จะได้รับการสนองตอบ

นอกจากนี้ การอินเตอร์รัพท์ยังมีการจัดลำดับความสำคัญของแต่ละอินเตอร์รัพท์ที่เข้ามาอีกด้วย ดังจะกล่าวในหัวข้อถัดไปนี้

2.2.2.2 การจัดลำดับความสำคัญของการอินเตอร์รัพท์

ถึงแม้ว่าจะมีวงจรจากอุปกรณ์ภายนอกมากมายที่สามารถสร้างและส่งสัญญาณอินเตอร์รัพท์มาให้ซีพียูได้ แต่ซีพียูก็ยังสามารถแยกแยะได้ว่าวงจรใดเป็นผู้สร้างสัญญาณอินเตอร์รัพท์นั้นๆขึ้นมาได้ เนื่องจากแต่ละวงจรจะให้อินเตอร์รัพท์เวกเตอร์ที่ต่างกัน ทำให้ซีพียูกระโดดไปทำงานในอินเตอร์รัพท์เซอร์วิสรูทีนต่างๆกันโดนที่แต่ละโปรแกรมก็ถูกเขียนให้ทำงานบริการแก่อุปกรณ์นั้นๆ

แม้ว่าซีพียูจะสามารถแยกแยะได้ว่าอุปกรณ์ใดเป็นผู้ส่งสัญญาณอินเตอร์รัพท์แล้วก็ตาม แต่ในทางปฏิบัติแล้ว เมื่อมีอุปกรณ์มากกว่าหนึ่งอุปกรณ์ต้องการจะติดต่อกับซีพียูพร้อมๆกัน ปัญหาว่าซีพียูจะติดต่อกับอุปกรณ์ใดก่อนก็จะเกิดขึ้น จึงต้องมีการจัดให้ความสำคัญต่ออุปกรณ์ไม่เท่ากัน (Priority Arbitration)

อุปกรณ์ภายนอกจะไม่ได้ต่อสัญญาณอินเตอร์รัพท์ของตนเข้าโดยตรงกับซีพียู แต่จะต่อผ่านวงจรจัดลำดับความสำคัญ ถ้ามีอุปกรณ์หลายตัวต้องการ ที่จะส่งสัญญาณอินเตอร์รัพท์ (INT) เข้าซีพียูพร้อมๆกัน วงจรจัดลำดับความสำคัญจะส่งผ่านสัญญาณอินเตอร์รัพท์และ Interrupt Vector ของอุปกรณ์ที่มีความสำคัญ (Priority) สูงสุด< WBR>ไปยังซีพียู ซีพียูจะตอบรับและให้บริการอุปกรณ์ที่ส่งอินเตอร์รัพท์นั้นมาให้เสร็จสิ้นเสียก่อน แล้วจึงเริ่มให้ความสนใจแก่อินเตอร์รัพท์ของอุปกรณ์ที่มีความสำคัญรองลงไปตามลำดับ

การจัดระบบอินเตอร์รัพท์ไม่ว่าจะเป็นกลไกลการกระโดด, การตอบรับหรือกลไกลการเก็บสถานะ การจัดระบบดังกล่าวนี้ของซีพียูก็จะแตกต่างกันออกไปแล้วแต่บริษัทผู้ผลิตจะออกแบบตัวไมโครโปรเซสเซอร์มาเช่นไร ทั้งนี้ก็ขึ้นอยู่กับวัตถุประสงค์ของงานที่จะนำซีพียูตัวนั้น ๆ ไปใช้

2.2.2.3 การควบคุมอินเตอร์รัพท์

การควบคุมการอินเตอร์รัพท์คือการควบคุมซีพียูว่าจะให้ตอบรับอินเตอร์รัพท์หรือไม่ เนื่องจากในบางขณะที่ซีพียูกำลังทำงานที่ต่อเนื่องอยู่ เช่น โปรแกรมนาฬิกา ซึ่งหากมีการอินเตอร์รัพท์เข้ามาก็จะก่อให้เกิดความผิดพลาดของนาฬิกา ดังนั้นจึงต้องมีการบังคับไม่ให้มีการสนองตอบต่อการร้องขออินเตอร์รัพท์เข้ามา เมื่อโปรแกรมนาฬิกาเสร็จแล้ว จึงอนุญาตให้มีการตอบสนองได้

คำสั่งที่ใช้ในการเซตหรือรีเซตแฟลกอินเตอร์รัพท์นั้นเรียกว่า คำสั่งอินเตอร์รัพท์ ซึ่งได้แก่

- **DI** ซึ่งย่อมาจาก Disable Interrupt เป็นการเซตให้ซีพียูอยู่ในภาวะไม่ตอบสนอง
- **EI** ซึ่งย่อมาจาก Enable Interrupt ใช้เพื่อรีเซตแฟลกให้อยู่ในภาวะตอบสนอง

2.2.2.4 หลักการทำงานของอินเตอร์รัพท์

ในไมโครโปรเซสเซอร์แทบจะทุกประเภทนั้น การอินเตอร์รัพท์จะเกิดขึ้นโดยอุปกรณ์อินพุทเอาท์พุทได้ส่งสัญญาณอิเลคทรอนิกส์เพื่อไปเปลี่ยนระดับสัญญาณที่ขาใด< WBR>ขาหนึ่งของไมโครโปรเซสเซอร์ ซึ่งขานั้นจะถูกเรียกว่า ขาอินเตอร์รัพท์ (Interrupt Pin) เพื่อเป็นการแจ้งแก่ไมโครโปรเซสเซอร์ถึงการอินเตอร์รัพท์ โดยจะมีขั้นตอนต่างๆดังนี้

- วงจรภายนอกส่งสัญญาณอินเตอร์รัพท์มายังซีพียู
- หากซีพียูอยู่ในสภาวะ Enable Interrupt (EI) ก็จะตอบรับการอินเตอร์รัพท์ด้วยการส่งสัญญาณ Interrupt Acknowledge กลับไปยังวงจรภายนอก
- เซตซีพียูให้อยู่ในสภาวะ Disable Interrupt (DI) เพื่อป้องกันการอินเตอร์รัพท์ซ้อน
- หยุดการทำงานในโปรแกรมหลัก
- วงจรภายนอกส่งอินเตอร์รัพท์เวกเตอร์มาให้ซีพียูเพื่อบอกตำแหน่งที่จะให้ซีพียูกระโดดไป
- ซีพียูส่งค่าในรีจิสเตอร์ PC และรีจิสเตอร์ทั่วไป ไปเก็บไว้ในสแต็ก
- ไปทำงานในโปรแกรมย่อยเพื่อบริการอุปกรณ์ที่ส่งอินเตอร์รัพท์เข้ามาจนเสร็จ
- รับค่าที่ไปเก็บไว้ในสแต็กคืนสู่รีจิสเตอร์ PC และ รีจิสเตอร์ทั่วไปตามเดิม
- เซตซีพียูให้กลับสู่สภาวะ Enable Interrupt (EI)
- ทำงานในโปรแกรมหลักที่ค้างอยู่ต่อไป

2.2.2.5 ตำแหน่งของโปรแกรมย่อยในการอินเตอร์รัพท์

การที่ซีพียูจะรู้ได้ว่าโปรแกรมย่อยจากการอินเตอร์รัพท์มีตำแหน่งอยู่ที่ใดในหน่วยความจำนั้น ส่วนแรกขึ้นอยู่กับชนิดของไมโครโปรเซสเซอร์ นั้นๆ ผู้ผลิต

ไมโครโปรเซสเซอร์แต่ละชนิดหรือแต่ละรายก็จะมีการกำหนดลักษณะการอ้างถึงตำแหน่งแตกต่างกันออกไป โดยอาจจะกำหนดดังนี้

- ผู้ผลิตจะกำหนดไว้ว่าโปรแกรมย่อยของการอินเตอร์รัพท์จะต้องเก็บไว้ที่ตำแหน่งใดตำแหน่งหนึ่งตลอดทั้งโปรแกรม
- เขียนเป็นเพียงคำสั่งกระโดดเพียงคำสั่งเดียวเพื่อกระโดดไปยังตำแหน่งอื่นที่เก็บโปรแกรมย่อยไว้
- ต่อชิ้นส่วนฮาร์ดแวร์เพื่อชี้ตำแหน่งที่เก็บโปรแกรมย่อยการอินเตอร์รัพท์ในหน่วยความจำซึ่งอาจเก็บไว้ที่ใดก็ได้
- วิธีที่สองเป็นที่นิยมใช้ทั้งนี้เพราะการใช้วิธีนี้จะทำให้สามารถกำหนดช่วงของตำแหน่งได้กว้างกว่า ในขณะที่วิธีสุด< WBR>ท้ายเป็นวิธีที่เร็วกว่า แต่เป็นการสิ้นเปลืองฮาร์ดแวร์

2.2.2.6 การอินเตอร์รัพท์จากหลายอุปกรณ์

ในซีพียูจะมีขาที่ใช้รับสัญญาณอินเตอร์รัพท์เพียงขาเดียว ดังนั้นในกรณีที่มีอุปกรณ์หลายอุปกรณ์ร้องขอการอินเตอร์รัพท์เข้ามา ทุกๆอุปกรณ์ก็จะ ต้องป้อนสัญญาณอินเตอร์รัพท์ให้แก่ซีพียูที่ขาเดียวกันนี้ จึงมีปัญหาเกิดขึ้นคือ เมื่อมีสัญญาณอินเตอร์รัพท์เกิดขึ้น ซีพียูจะจำแนกได้อย่างไรว่า< WBR>สัญญาณที่เข้ามานั้นมาจากอุปกรณ์ตัวใด วิธีการจำแนกสามารถทำได้หลายวิธี แต่แบ่งวิธีที่สำคัญออกเป็น 2 วิธี คือ วิธีซอฟท์แวร์ เรียกว่า การโพล (Polling) และวิธีฮาร์ดแวร์ซึ่งต้องต้องต่อชิ้นส่วนอิเลคทรอนิกส์เข้าช่วย

การโพล หมายถึงการที่ซีพียูจะตรวจสอบการแฟลกการขออินเตอร์รัพท์ของหน่วยรับส่งข้อมูลเข้าออกที่ต่อกับอุปกรณ์แต่ละตัวตามลำดับเพื่อหา ว่าสัญญาณนั้นส่งมาจากอุปกรณ์ใด เมื่อตรวจพบแล้วก็จะทำตามขั้นตอนของอินเตอร์รัพท์ต่อ คือป้อนค่าจากรีจิสเตอร์ต่างๆ ภายในซีพียูไปเก็บไว้ที่สแต็ก จากนั้นไปทำงานตามโปรแกรมย่อยเพื่อให้บริการแก่อุปกรณ์ที่ขออินเตอร์รัพท์เข้ามา เมื่อจบแล้วก็จะป้อนค่าต่างๆจากสแต็กกลับ สู่รีจิสเตอร์ตามเดิม และกลับไปทำงานของโปรแกรมหลักที่ค้างไว้ การโพลจะช่วยค้นหาอุปกรณ์ที่ส่งสัญญาณอินเตอร์รัพท์และในขณะเดียวกันก็จะจัดลำดับความสำคัญก่อนหลัง (Priority) ของอินเตอร์รัพท์จากอุปกรณ์ต่างๆที่ร้องขอมา เพื่อในกรณีที่ทีการอินเตอร์รัพท์เข้ามาพร้อมๆกัน

วิธีฮาร์ดแวร์ วิธีฮาร์ดแวร์จะมีวงจรอิเลคทรอนิกส์เพิ่มเติมเพื่อช่วยค้นหาว่าสัญญาณอินเตอร์รัพท์ถูกส่งมาจากอุปกรณ์ไหน เมื่อซีพียูได้รับสัญญาณอินเตอร์รัพท์ (INT) ก็จะส่งสัญญาณตอบสนองอินเตอร์รัพท์ (INTA = Interrupt Acknowledge) ผ่านกลับไปให้แต่ละอุปกรณ์เพื่อตรวจสอบว่าอุปกรณ์ใดเป็นผู้ส่งสัญญาณอินเตอร์รัพท์มา เมื่อตรวจพบแล้วอุปกรณ์นั้นก็จะส่งตำแหน่งที่เก็บโปรแกรมย่อยของการอินเตอร์รัพท์นั้นให้แก่ซีพียู สัญญาณตอบสนองที่ซีพียูส่งออกไปนั้นจะอ่านอุปกรณ์ที่ละตัวตามลำดับก่อนหลังที่ได้กำหนดไว้ จนกว่าจะจะพบว่าอุปกรณ์

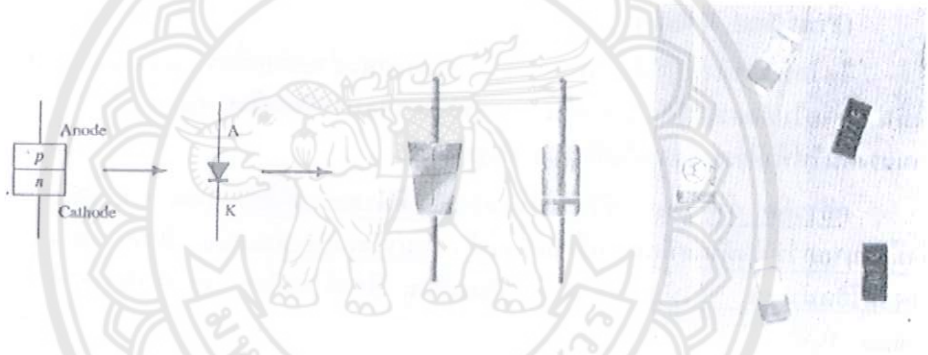
ใดเป็นผู้ส่งสัญญาณอินเตอร์รัพท์มา วงจรอิเลคทรอนิกส์ที่จะนำมาต่อเพิ่มเติมนั้นจะเป็นไอซีเฉพาะกิจที่บริษัทผู้ผลิตไมโครโปรเซสเซอร์ผลิตออกมาจำหน่ายด้วย

2.2.3 อุปกรณ์ที่ช่วยในการทำงานของไมโครโปรเซสเซอร์ [2]

เนื่องจากไมโครโปรเซสเซอร์อย่างเดียว ไม่สามารถที่จะทำงานได้จะต้องมีอุปกรณ์อื่นเข้ามาช่วยการทำงานของไมโครโปรเซสเซอร์ด้วยดังนี้

2.2.3.1 ไดโอด (DIODE)

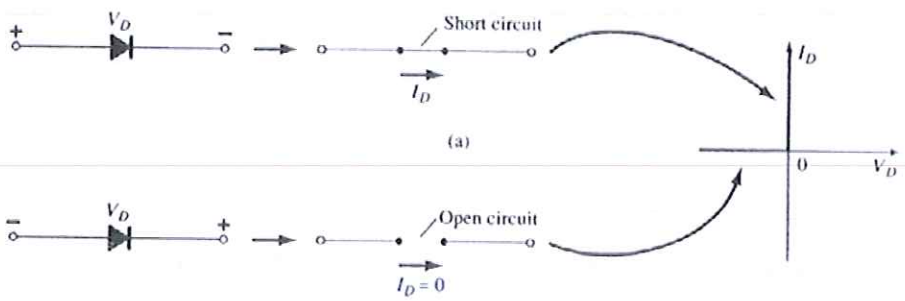
ไดโอด เป็นอุปกรณ์ที่ทำจากสารกึ่งตัวนำ p-n สามารถควบคุมให้กระแสไฟฟ้าจากภายนอกไหลผ่านตัวมันได้ทิศทางเดียว ไดโอดประกอบด้วยขั้ว 2 ขั้ว คือ แอโนด (Anode ; A) ซึ่งต่ออยู่กับสารกึ่งตัวนำชนิด p และ แคโทด (Cathode ; K) ซึ่งต่ออยู่กับสารกึ่งตัวนำชนิด n ดังรูปที่ 2.5



รูปที่ 2.5 ภาพแสดงอุปกรณ์ ไดโอด [2]

2.2.3.1.1 ไดโอดในทางอุดมคติ (Ideal Diode)

ไดโอดในอุดมคติมีลักษณะเหมือนสวิทช์ที่สามารถนำกระแสไหลผ่านได้ในทิศทางเดียว

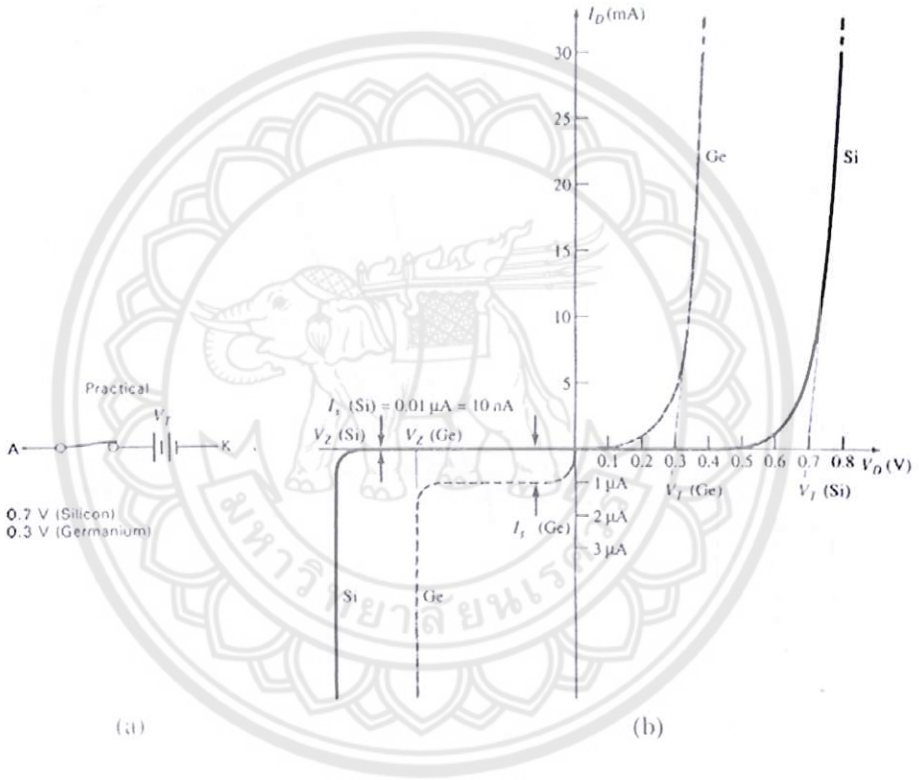


รูปที่ 2.6 ภาพแสดงการทำงานของ ไดโอด [2]

จากภาพถ้าต่อขั้วแบตเตอรี่ให้เป็นแบบไบอัสตรงไดโอดจะเปรียบเป็นเสมือนกับสวิทช์ที่ปิด (Close Switch) หรือไดโอดลัดวงจร (Short Circuit) Id ไหลผ่านไดโอดได้ แต่ถ้าต่อขั้วแบตเตอรี่แบบไบอัสกลับ ไดโอดจะเปรียบเป็นเสมือนสวิทช์เปิด (Open Switch) หรือเปิดวงจร (Open Circuit) ทำให้ Id เท่ากับศูนย์

2.2.3.1.2 ไดโอดในทางปฏิบัติ (Practical Diode)

ไดโอดในทางปฏิบัติมีการแพร่กระจายของพาหะส่วนน้อยที่บริเวณรอยต่ออยู่จำนวนหนึ่ง ดังนั้น ถ้าต่อไบอัสตรงให้กับไดโอดในทางปฏิบัติก็จะเกิด แรงดันเสมือน (Ge >= 0.3V; Si >= 0.7V) ซึ่งต้านแรงดันไฟฟ้าที่จ่ายเพื่อการไบอัสตรง ดังรูป 2.7



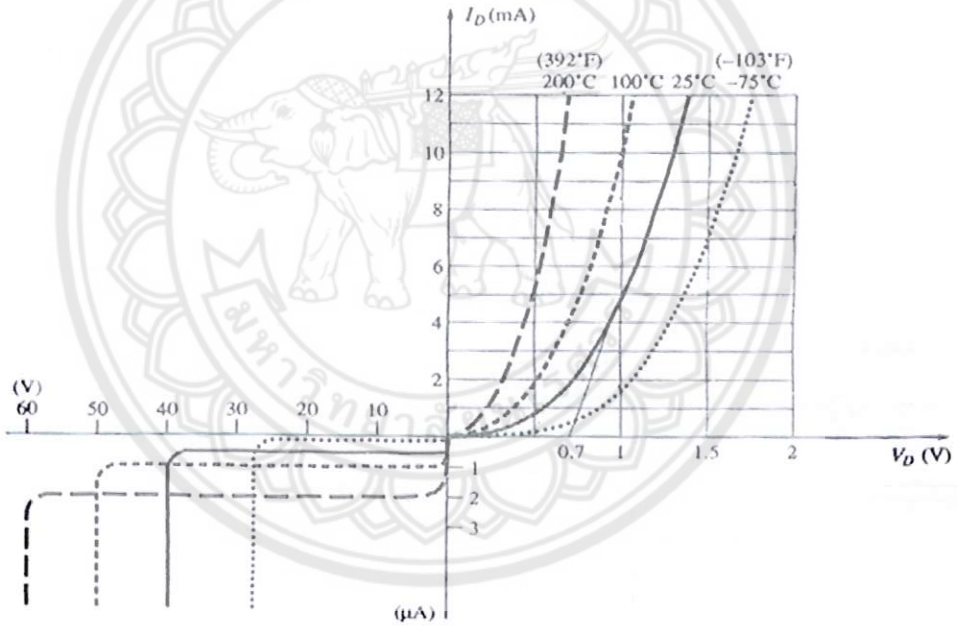
รูปที่ 2.7 ภาพแสดงการทำงานของ ไดโอด ภาพที่สอง [2]

ขนาดของแรงดันเสมือนจึงเป็นตัวบอกจุดทำงาน ดังนั้น จึงเรียก แรงดันเสมือน อีกอย่างหนึ่งว่า แรงดันในการเปิด (Turn-on Voltage ; Vt) กรณีไบอัสกลับ เราทราบว่า Depletion Region จะขยายกว้างขึ้น แต่ก็ยังมีพาหะข้างน้อยแพร่กระจายที่รอยต่ออยู่จำนวนหนึ่ง แต่ก็ยังมีกระแสรั่วไหลอยู่จำนวนหนึ่ง เรียกว่า กระแสรั่วไหล (Leakage Current) เมื่อเพิ่มแรงดันไฟฟ้าขึ้นเรื่อยๆ กระแสรั่วไหลจะเพิ่มขึ้นจนถึงจุดที่ไดโอดนำกระแสเพิ่มขึ้นมาก ระดับกระแสที่จุดนี้เรียกว่า กระแสอิ่มตัวย้อนกลับ (Reverse Saturation Current ; Is) แรงดันไฟฟ้าที่จุดนี้ เรียกว่า แรงดันพังทลาย (Breakdown Voltage) และถ้าแรงดันไปกลับสูงขึ้นจนถึงจุดสูงสุดที่ไดโอดทนได้

เราเรียกว่า แรงดันพังทลายซีเนอร์ (Zener Breakdown Voltage ; Vz) ถ้าแรงดันไบอัสกลับสูงกว่า Vz จะเกิดความร้อนอย่างมากที่รอยต่อของไดโอด ส่งผลให้ไดโอดเสียหายหรือพังได้ แรงดันไฟฟ้าที่จุดนี้เราเรียกว่า แรงดันพังทลายอวาแลนซ์ (Avalance Breakdown Voltage) ดังนั้น การนำไดโอดไปใช้งานจึงใช้กับการไบอัสตรงเท่านั้น

2.2.3.1.3 ผลกระทบของอุณหภูมิ (Temperature Effects)

จากการทดลองพบว่า Is ของ Si จะมีค่าเพิ่มขึ้นเกือบ 2 เท่า ทุกๆ ครั้งที่อุณหภูมิเพิ่มขึ้น 10 องศาเซลเซียสขณะที่ Ge มีค่า Is เป็น 1 หรือ 2 micro-amp ที่ 25 องศาเซลเซียส แต่ที่ 100 องศาเซลเซียสจะมีค่า Is เพิ่มขึ้นเป็น 100 micro-amp ระดับกระแสไฟฟ้าขนาดนี้จะเป็นปัญหาต่อการเปิดวงจรเรื่องจากได้รับการไบอัสกลับ เพราะแทนที่ Id จะมีค่าใกล้เคียงศูนย์ แต่กลับนำกระแสได้จำนวนหนึ่งตามอุณหภูมิที่เพิ่มขึ้น ดังรูปที่ 2.8



รูปที่ 2.8 ภาพแสดงการทำงานของ ไดโอด ภาพที่สาม [2]

2.2.3.1.4 ซีเนอร์ไดโอด (Zener Diode)

ซีเนอร์ไดโอดเป็นอุปกรณ์สารกึ่งตัวนำที่นำกระแสได้เมื่อได้รับไบอัสกลับ และระดับแรงดันไบอัสกลับที่นำซีเนอร์ไดโอดไปใช้งานได้เรียกว่า ระดับแรงดันพังทลายซีเนอร์ (Zener Breakdown Voltage; Vz) การใช้งานซีเนอร์ไดโอดเราจะต่อแบบไบอัสกลับ

กราฟแสดงคุณลักษณะของซีเนอร์ไดโอด จะเห็นได้ว่าขณะไบอัสซีเนอร์ไดโอด แรงดันไบอัสกลับ (Vr) จะมีค่าน้อยกว่า Vz เล็กน้อยไดโอดประเภทนี้เหมาะที่จะนำไปใช้

ควบคุมแรงดันที่โหลดหรือวงจรที่ต้องการแรงดันคงที่ เช่น ประกอบอยู่ในแหล่งจ่ายไฟเลี้ยง หรือโวลเทจเรกูเลเตอร์

2.2.3.1.5 แอลอีดี (Light Emitting Diode; LED)

LED เป็นไดโอดที่ใช้สารประเภทแกลเลี่ยมอาร์เซ็นไนต์ฟอสไฟต์ (Gallium Arsenide Phosphide ; GaAsP) หรือสารแกลเลี่ยมฟอสไฟต์ (Gallium Phosphide ; GaP) มาทำเป็นสารกึ่งตัวนำชนิด p และ n แทนสาร Si และ Ge สารเหล่านี้มีคุณลักษณะพิเศษ คือ สามารถเรืองแสงได้เมื่อได้รับไบอัสตรง การเกิดแสงที่ตัว LED นี้เราเรียกว่า อิเลคโทรลูมินิเซนต์ (Electroluminescence) ปัจจุบันนิยมใช้ LED แสดงผลในเครื่องมืออิเล็กทรอนิกส์ เช่น เครื่องคิดเลข นาฬิกา เป็นต้น

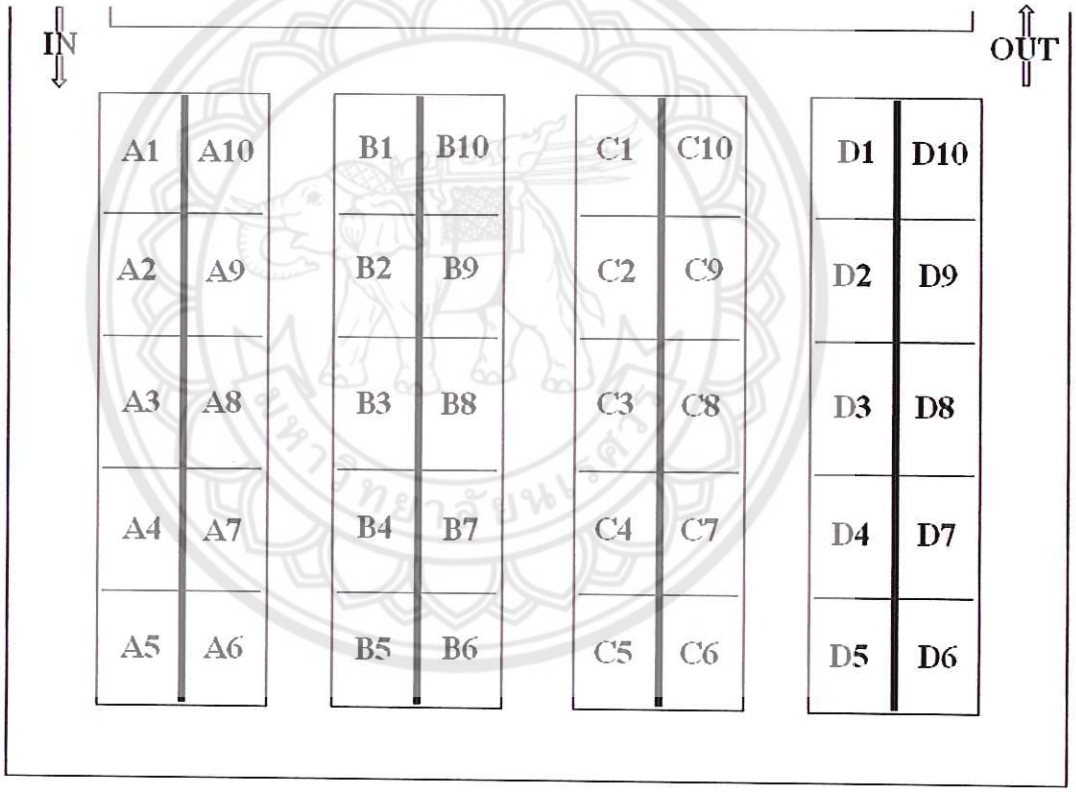
จากการศึกษาทางด้านการเขียนโปรแกรมและความรู้ทางด้านอุปกรณ์อิเล็กทรอนิกส์ทำให้สามารถเข้าใจหลักการเขียนโปรแกรมด้วย Visual Basic และยังได้ศึกษาเกี่ยวกับรูปแบบของอุปกรณ์อิเล็กทรอนิกส์ว่า อุปกรณ์อิเล็กทรอนิกส์ชนิดต่างๆมีคุณสมบัติเฉพาะตัวอย่างไรเพื่อที่จะได้สามารถเลือกใช้ให้เหมาะสมกับงานที่ต้องการได้อย่างมีประสิทธิภาพ

บทที่ 3
การพัฒนาระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ

เนื้อหาในบทนี้กล่าวถึงขั้นตอนการดำเนินการในส่วนต่างๆ โดยจะเป็นรายละเอียดในส่วนของการพัฒนาระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ (Expert system for assigning parking space)

3.1 ออกแบบลานจอดรถตัวอย่าง และสร้างฐานข้อมูล

3.1.1 ออกแบบลานจอดรถตัวอย่าง



รูปที่ 3.1 ภาพลานจอดรถที่ออกแบบขึ้น

ทำการออกแบบลานจอดรถตัวอย่างดังรูปที่ 3.1 เพื่อใช้ในการพัฒนาโปรแกรมระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ โดยออกแบบเป็น 4 ชั้น แต่ละชั้นมีตำแหน่งสำหรับจอดรถ 10 ตำแหน่ง

A 1-10 หมายถึง ที่จอดรถตำแหน่งที่ 1-10 ของชั้นที่ 1

B 1-10 หมายถึง ที่จอดรถตำแหน่งที่ 1-10 ของชั้นที่ 2

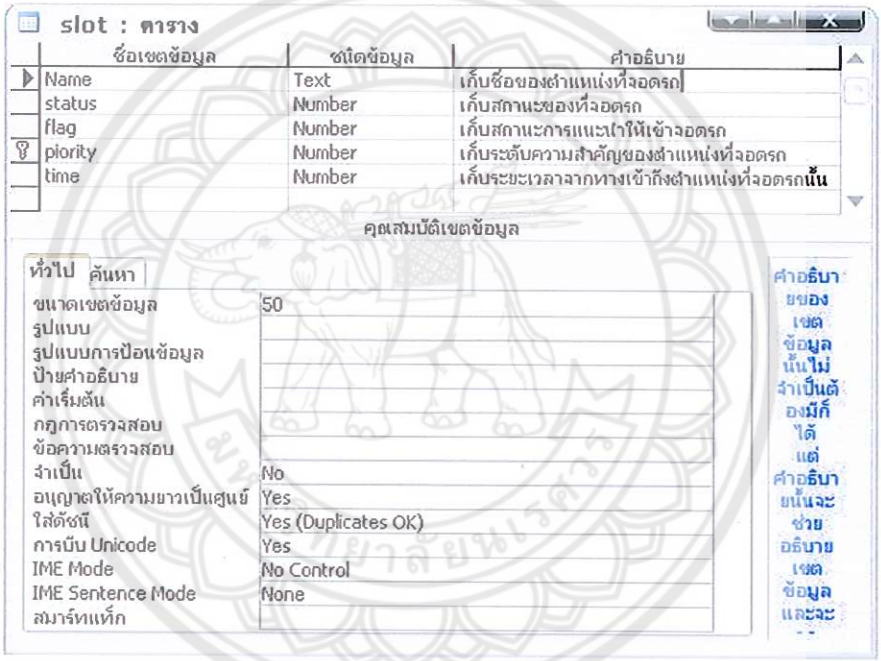
C 1-10 หมายถึง ที่จอดรถตำแหน่งที่ 1-10 ของชั้นที่ 3

D 1-10 หมายถึง ที่จอดรถตำแหน่งที่ 1-10 ของชั้นที่ 4

IN หมายถึง ทางเข้า

OUT หมายถึง ทางออก

3.1.2 สร้างฐานข้อมูลของลานจอดรถ



รูปที่ 3.2 ภาพแสดงฐานข้อมูลที่ออกแบบขึ้น

ออกแบบฐานข้อมูลดังรูปที่ 3.2 โดยกำหนดเขตข้อมูลไว้ 5 เขต ดังนี้

- Name เก็บชื่อของตำแหน่งที่จอดรถ
- status เก็บสถานะของที่จอดรถ
- flag เก็บสถานการณ์แนะนำให้เข้าจอดรถ
- priority เก็บระดับความสำคัญของตำแหน่งที่จอดรถ โดยระดับความสำคัญวัดจากระยะทางจากทางเข้าถึงตำแหน่งนั้น คือ ตำแหน่งที่อยู่ใกล้ทางเข้ามากที่สุดจะมีระดับความสำคัญมากที่สุด
- time เก็บระยะเวลาจากทางเข้าถึงตำแหน่งที่จอดรถนั้น

slot : ตาราง

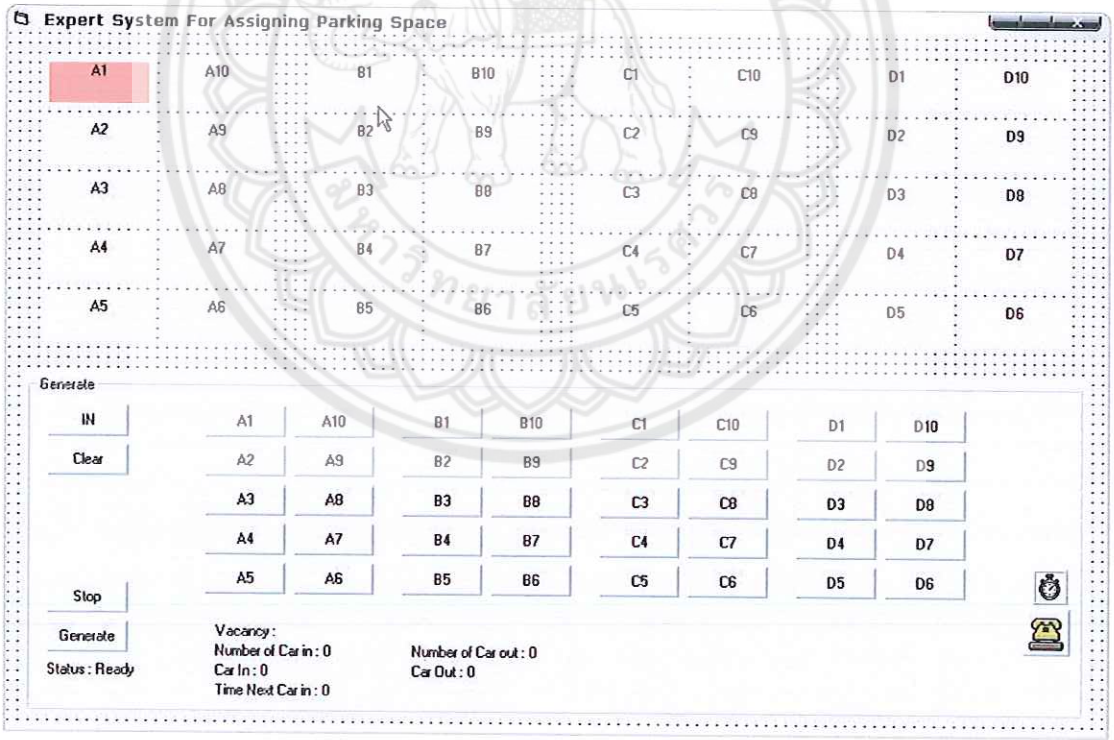
	Name	status	flag	piority	time
	A1	0	0	1	5
	A2	1	0	2	7
	A3	0	0	3	9
	A4	1	0	4	11
	A5	1	0	5	13
	A6	0	0	6	15
	A7	0	0	7	17
	A8	0	0	8	19
	A9	1	0	9	21
	A10	0	0	10	23

ระเบียน: 11 จาก 40

รูปที่ 3.3 ภาพแสดงตัวอย่างข้อมูลในฐานข้อมูลที่สร้างขึ้น

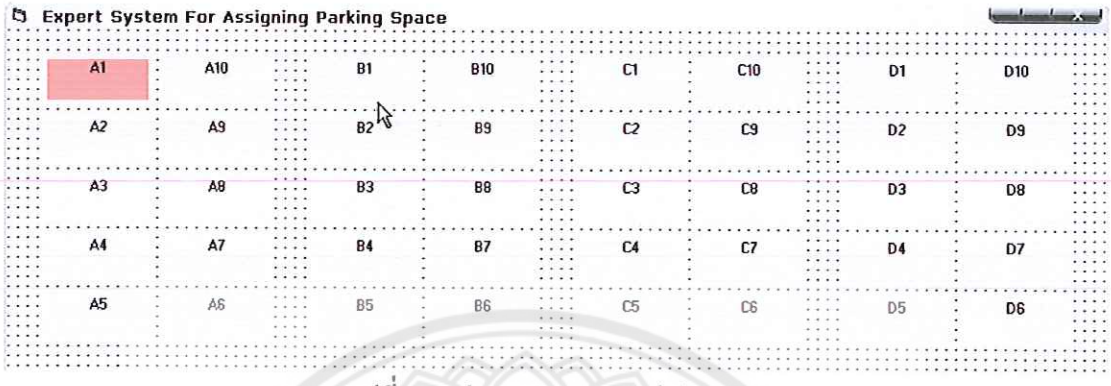
3.2 พัฒนาโปรแกรมจำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ และระบบช่วยตัดสินใจในการหาที่จอดรถแก่ผู้ขับ ด้วยวิชวลเบสิก 6

3.2.1 ออกแบบ แบบฟอร์มของโปรแกรม



รูปที่ 3.4 ภาพแสดงแบบฟอร์มของโปรแกรม

สร้างแบบฟอร์มของโปรแกรมดังรูปที่ 3.4 โดยแบ่งเป็น 2 ส่วน ส่วนบนคือส่วนที่แสดงสถานะต่างๆ ของลานจอดรถ และส่วนล่างเป็นส่วนของการจำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ



รูปที่ 3.5 ส่วนแสดงผลของโปรแกรม

จากรูปที่ 3.5 กำหนดให้

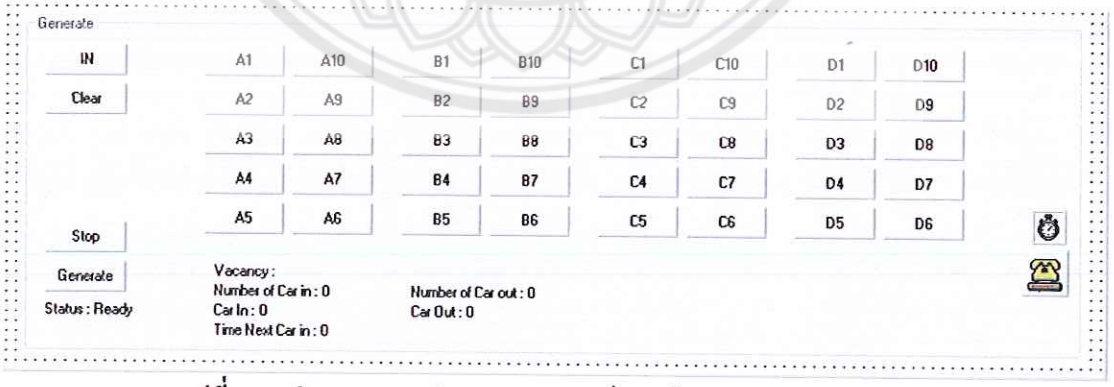
A 1-10 หมายถึง ที่จอดรถตำแหน่งที่ 1-10 ของชั้นที่ 1

B 1-10 หมายถึง ที่จอดรถตำแหน่งที่ 1-10 ของชั้นที่ 2

C 1-10 หมายถึง ที่จอดรถตำแหน่งที่ 1-10 ของชั้นที่ 3

D 1-10 หมายถึง ที่จอดรถตำแหน่งที่ 1-10 ของชั้นที่ 4

ส่วนของการจำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ



รูปที่ 3.6 ส่วนของการจำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ

จากรูปที่ 3.6 กำหนดให้

IN เป็นการจำลองเหตุการณ์ที่มีรถเข้ามาในลานจอดรถ

Clear เป็นการเอาค่าในส่วนที่แสดงผลออกให้เป็นค่าปกติทั้งหมด

Stop	เป็นการหยุด Generate
Generate	เป็นการจำลองเหตุการณ์ว่ามีรถเข้าในแบบสถานการณ์จริง
A 1-10	เป็นการจำลองเหตุการณ์ว่ามีรถจอดอยู่ในตำแหน่ง A 1-10
B 1-10	เป็นการจำลองเหตุการณ์ว่ามีรถจอดอยู่ในตำแหน่ง B 1-10
C 1-10	เป็นการจำลองเหตุการณ์ว่ามีรถจอดอยู่ในตำแหน่ง C 1-10
D 1-10	เป็นการจำลองเหตุการณ์ว่ามีรถจอดอยู่ในตำแหน่ง D 1-10

3.2.2 เขียนโปรแกรมจำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ และระบบช่วยตัดสินใจในการหาที่จอดรถแก่ผู้ขับ

การทำงานของโปรแกรมแบ่งเป็น 2 ส่วนคือ

3.2.2.1 การทำงานในส่วนของการจำลองเหตุการณ์การเข้าจอดรถในลานจอดรถ

โปรแกรมจะทำการจำลองเหตุการณ์โดยการสุ่มตัวเลขของจำนวนรถที่จะเข้า และออกจากลานจอดรถ โดยจำนวนตัวเลขของรถที่จะเข้ามาในลานจอดรถจะไม่เกินจากที่ว่างที่มีอยู่ในลานจอดรถ และจำนวนตัวเลขของรถที่ออกจากลานจอดรถจะไม่เกินจากจำนวนรถที่มีอยู่ในลานจอดรถ โดยรถที่จะเข้าและออกแต่ละคันนั้นจะถูกกำหนดเวลาในการเข้าและออกจากการสุ่มตัวเลขด้วยเช่นกัน หลังจากนั้นโปรแกรมก็จะทำการจับเวลาและจำลองเหตุการณ์จากตัวเลขดังกล่าว โดยการเปลี่ยนแปลงตัวเลขสถานะต่างๆ ในฐานข้อมูลของลานจอดรถ

3.2.2.2 การทำงานในส่วนของระบบช่วยตัดสินใจในการหาที่จอดรถแก่ผู้ขับ

โปรแกรมจะทำการดึงข้อมูลสถานะต่างของลานจอดรถจากฐานข้อมูลเพื่อตรวจสอบหาตำแหน่งที่ว่างที่มีอยู่ในลานจอดรถ เมื่อมีรถเข้ามาในลานจอดรถโปรแกรมจะทำการแนะนำที่ว่างให้แก่ผู้ขับ โดยโปรแกรมจะทำการตัดสินใจเลือกตำแหน่งที่ว่างที่ใกล้ที่สุดด้วยการพิจารณาเปรียบเทียบจากระดับความสำคัญของตำแหน่งที่ว่างทั้งหมด ซึ่งตำแหน่งที่ใกล้ที่สุดจะมีระดับความสำคัญมากที่สุด โปรแกรมจะทำการแนะนำผู้ขับโดยแสดงสัญญาณสีเหลือง ณ ตำแหน่งที่จะทำการแนะนำ

3.2.3 คำอธิบายโปรแกรมในส่วนที่สำคัญ

```
Private Const strconn = "Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=c:\park.mdb;Persist Security Info=false"
```

เป็นการกำหนดรูปแบบการเชื่อมต่อกับ Database และการกำหนดชื่อของ Database ที่ใช้

```
Private Sub IN_Click()
Dim i As Integer
If flags4 = 0 And vacnums <> 0 Then
        flags4 = vacnums
        flags4 = flags4 - 1
ElseIf flags4 = 0 And vacnums = 0 Then
        flags4 = 0
Else:
        flags4 = flags4 - 1
End If
        rs.MoveFirst
For i = 0 To 39
        If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
        rs.Fields("flag").Value = 1
Exit For
Else: rs.MoveNext
End If
Next i
        rs.MoveFirst
End Sub
```

เป็นฟังชันก์ที่ใช้ในการจำลองเหตุการณ์ที่มีรถเข้ามาในลานจอดรถ ฟังชันก์นี้จะทำการระบุตำแหน่งที่จอดรถที่เหมาะสมให้แก่ผู้ขับขี่

```
Sub toggle(i As Integer)
Dim j As Integer
 rs.MoveFirst
 For j = 0 To i - 2
    rs.MoveNext
 Next j

If rs.Fields("status").Value = 0 Then
        rs.Fields("status").Value = 1
        rs.Fields("flag").Value = 0
         time(j) = rs.Fields("time").Value * mul
 Else:
        If flags4 > 0 Then
        flags4 = flags4 + 1
End If
                rs.MoveFirst
End Sub
```

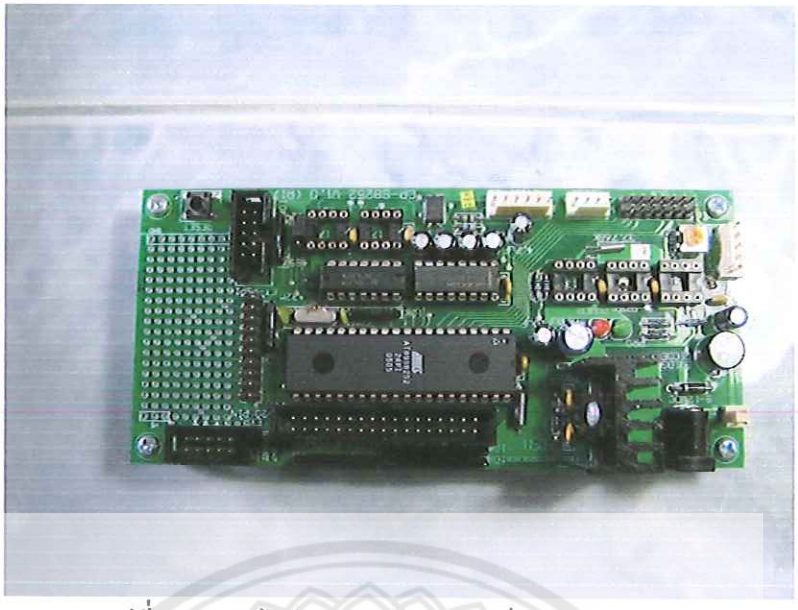
เป็นฟังชันก์ที่ใช้ในการกลับค่าตำแหน่งที่มีรถจอดอยู่หรือไม่มีรถจอดอยู่เพื่อใช้ในการจำลองเหตุการณ์ต่างๆ ที่เกิดขึ้นในลานจอดรถ

3.3 การพัฒนาในส่วนของการแสดงผลด้วยแผงวงจร LED

3.3.1 การออกแบบแผงวงจร ที่ใช้ในการแสดงผล

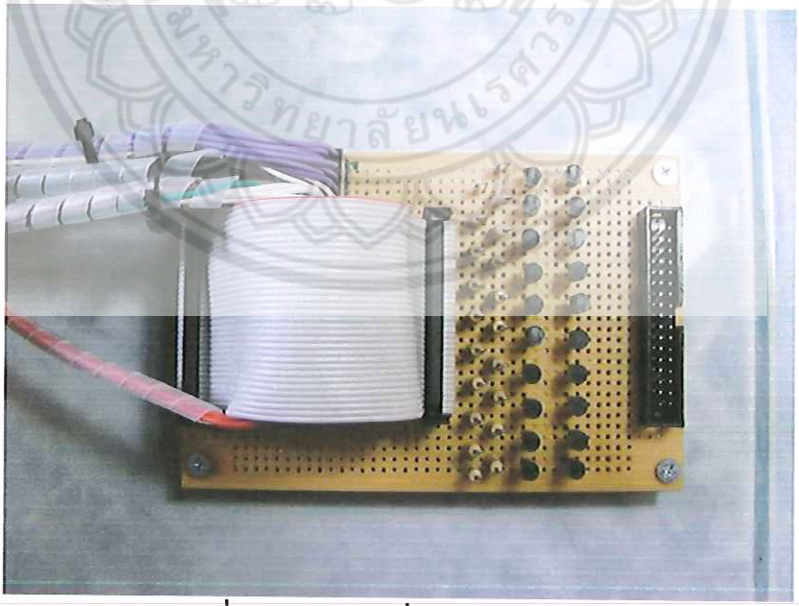
รูปที่ 3.7 ภาพแผงวงจร LED

แผงวงจร LED ใช้ในการแสดงผลซึ่งรับค่ามาจากคอมพิวเตอร์ ประกอบไปด้วย LED ที่สามารถแสดงได้ 3 สีจำนวน 40 ตัว โดยที่แต่ละตัวแสดงตำแหน่งที่จอดรถ 1 ตำแหน่ง โดยที่สัญญาณไฟ สีแดง หมายถึง ตำแหน่งนั้นได้มีรถจอดอยู่แล้ว สีเขียว หมายถึง แทนตำแหน่งที่ว่างอยู่ และ สีส้ม หมายถึง ตำแหน่งที่แนะนำให้ผู้ขับขี่ไปจอดรถ ดังรูปที่ 3.7



รูปที่ 3.8 ภาพไมโครคอนโทรลเลอร์ CP-S8252 V1.0

ดังรูปที่ 3.8 เป็นไมโครคอนโทรลเลอร์ชนิดหนึ่งซึ่งการใช้งานนั้นจะต้องทำการใส่คำสั่ง Assembly เข้าไปใน CHIP ก่อนที่จะทำงานและไมโครคอนโทรลเลอร์นี้จะรับคำสั่งจากคอมพิวเตอร์ผ่านทางพอร์ท RS232 และจะประมวลผลคำสั่งแล้วจึงส่งค่าไปให้แผงวงจรที่ใช่ในการขับ LED และส่งค่าไปต่อเพื่อแสดงผลผ่านทางแผงวงจร LED ต่อไป



รูปที่ 3.9 แผงวงจรที่ใช้ในการขับ LED

เพื่อป้องกันไม่ให้พอร์ทของไมโครคอนโทรลเลอร์ CP-S8252 V1.0 เกิดความเสียหายจากการจ่ายไฟเพื่อขับ LED จึงจำเป็นต้องสร้างวงจรที่ใช้จ่ายไฟเพื่อขับ LED ขึ้นมาดังรูปที่ 3.9

3.3.2 โปรแกรม Assembly ที่สำคัญในการควบคุมการทำงานของไมโครคอนโทรลเลอร์

```
MOV IE,#00000000B
MOV TMOD,#20H
MOV TL1,#0FDH
MOV TH1,#-5
```

เป็นการกำหนดตั้งค่าให้กับ Timer1 เพื่อใช้สร้าง Baud Rate ในการรับส่งข้อมูล

```
MOV SCON,#52H
SETB TR1
MOV P0,#00000000B
MOV P1,#00000000B
MOV P2,#01000000B
```

เป็นการกำหนดค่าให้กับ SCON Register โดยเลือกโหมด 1 และให้รับข้อมูล REN=1

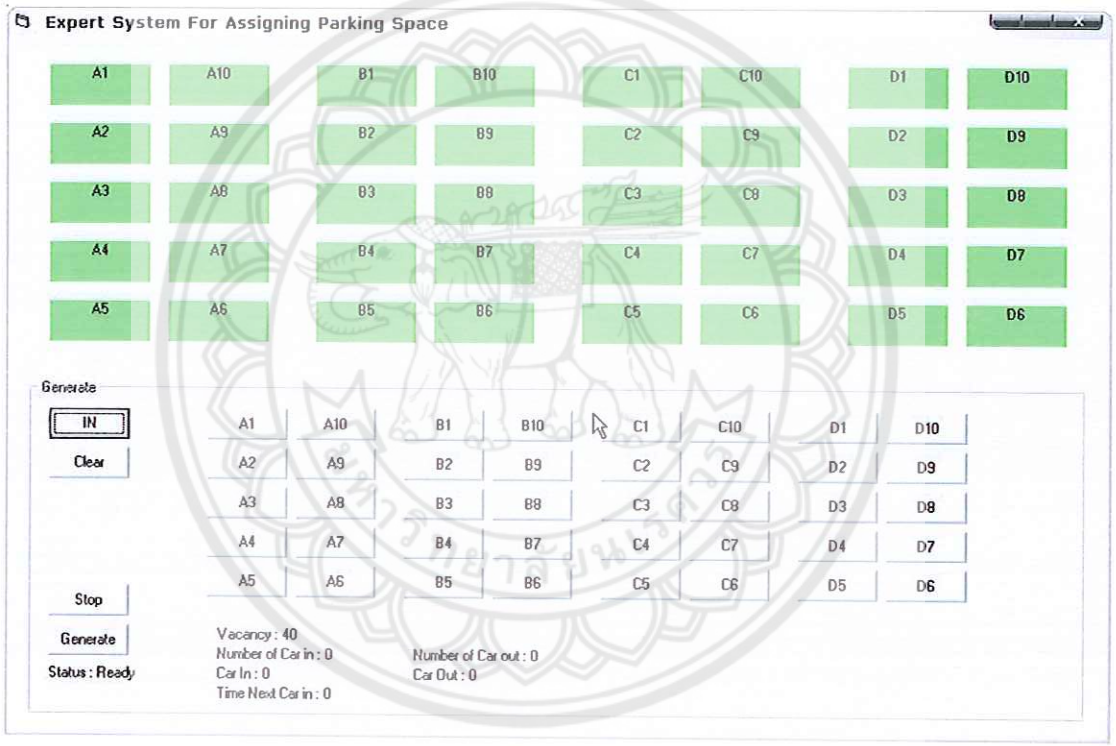
ขั้นตอนการดำเนินการดำเนินการในการพัฒนาโปรแกรมสามารถแบ่งการทำงานออกเป็นส่วนต่างๆ ดังนี้คือ ในส่วนการแสดงผลโปรแกรมจะแสดงตำแหน่งว่าตำแหน่งใดมีรถจอดบ้าง ส่วนของโปรแกรมจะประมวลว่ามีตำแหน่งใดบ้างที่มีรถจอดอยู่แล้วเก็บไว้ใน Database และส่วนที่ใช้แสดงผลบนแผงวงจร LED

เมื่อนำส่วนของโปรแกรมที่พัฒนาขึ้นมา ไมโครคอนโทรลเลอร์ และแผงวงจรที่ได้สร้างขึ้น มาทำงานร่วมกันโดยที่โปรแกรมจะส่งค่าให้ไมโครคอนโทรลเลอร์และไมโครคอนโทรเลอร์ก็จะส่งค่าไปให้แผงวงจรที่ใช้ในการขับ LED และแสดงผลต่อไปในแผงวงจร LED การทดลองการแสดงผลและจำลองเหตุการณ์นี้จะนำเสนอในบทที่ 4 ต่อไป

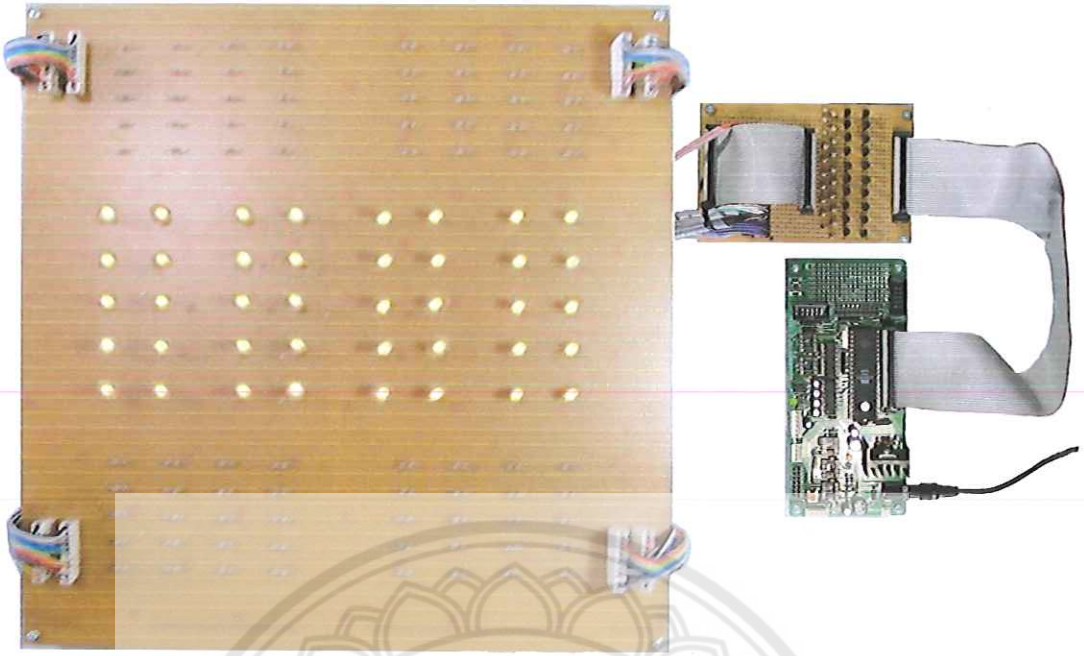
บทที่ 4

ผลการทดลอง

ในบทนี้เป็นการทำการทดลองเกี่ยวกับ ระบบอัจฉริยะสำหรับระบุตำแหน่งที่ว่างในการจอดรถ (Expert system for assigning parking space) ซึ่งเราได้ศึกษาและพัฒนาจากบทที่ผ่านมาแล้ว การทดลองในบทนี้จะเป็นการจำลองสถานการณ์ต่างๆ ที่สามารถเกิดขึ้นได้กับโปรแกรม เช่น การจำลองเหตุการณ์ ที่มีรถเข้ามาในลานจอดรถ การจำลองเหตุการณ์ที่มีรถออกจากลานจอดรถ และการเปลี่ยนสีของสัญญาณไฟเพื่อบอกสถานะของที่จอดรถ เป็นต้น

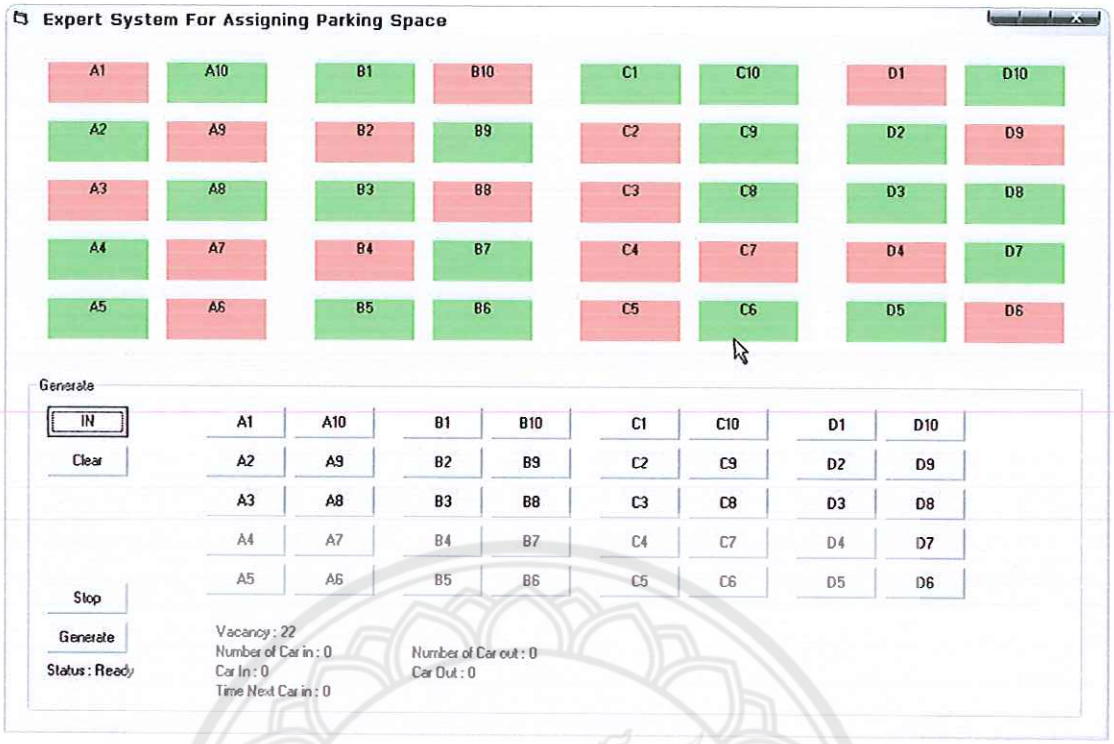


รูปที่ 4.1 ภาพแสดงโปรแกรมที่พัฒนาเสร็จบนจอคอมพิวเตอร์

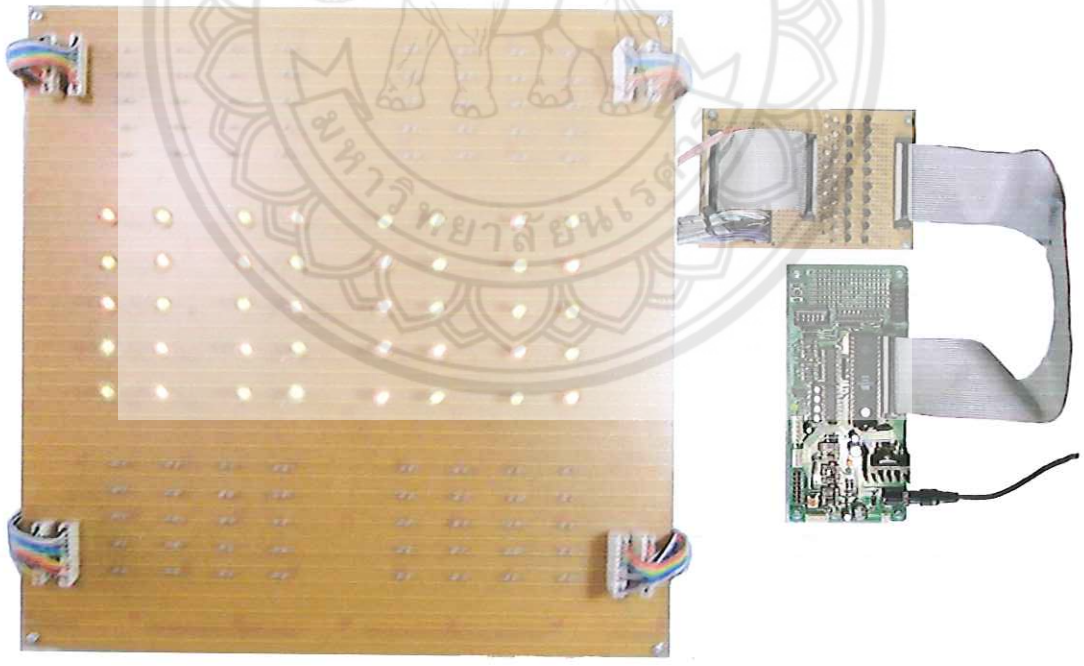


รูปที่ 4.2 ภาพแสดงทางแผงวงจร LED ที่ได้ทำขึ้น

จากรูปที่ 4.1 และ รูปที่ 4.2 เมื่อทำการเปิดโปรแกรม ตัวโปรแกรมจะทำการดึงข้อมูลจากฐานข้อมูล แล้วทำการประมวลผลและแสดงสถานะตำแหน่งต่างในลานจอดรถในส่วนของการแสดงผลด้านบน โดย สีเขียว หมายถึง ว่าง สีแดง หมายถึง ไม่ว่าง และสีเหลือง หมายถึง ตำแหน่งที่โปรแกรมแนะนำให้ผู้ขับนำรถไปจอด จะเห็นได้ว่าการแสดงผลบนจอคอมพิวเตอร์และแผงวงจร LED ที่ได้ทำขึ้นนั้นมีสถานะเหมือนกัน

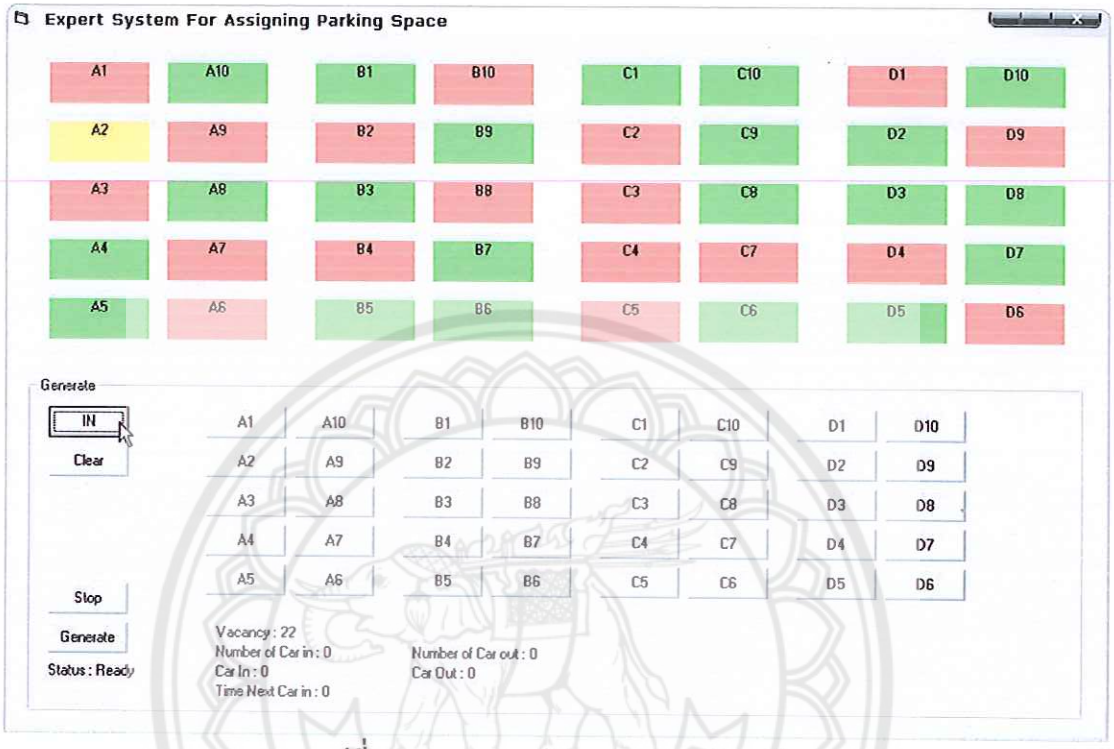


รูปที่ 4.3 ภาพแสดงโปรแกรมที่พัฒนาเสร็จภาพที่สอง

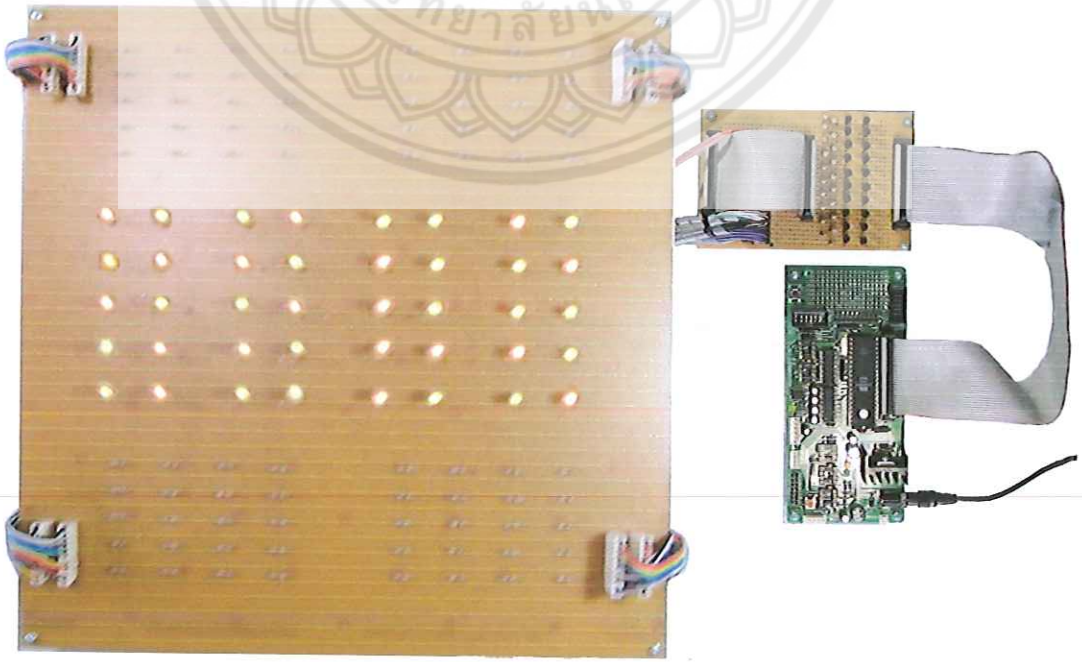


รูปที่ 4.4 ภาพแสดงทางแผงวงจรLED ภาพที่สอง

จากรูปที่ 4.3 และ รูปที่ 4.4 แสดงโปรแกรมหลังจากได้ทำการกำหนดให้มีรถเข้าจอดอยู่ในลานจอดรถโดยตำแหน่งสีแดง แทนตำแหน่งที่มีรถจอดอยู่แล้ว เช่น A1 A3 A6 A7 และ A9 เป็นต้น และตำแหน่งสีเขียวแทนตำแหน่งที่ยังไม่มีรถจอด เช่น A2 A4 A5 A8 และ A10 เป็นต้น

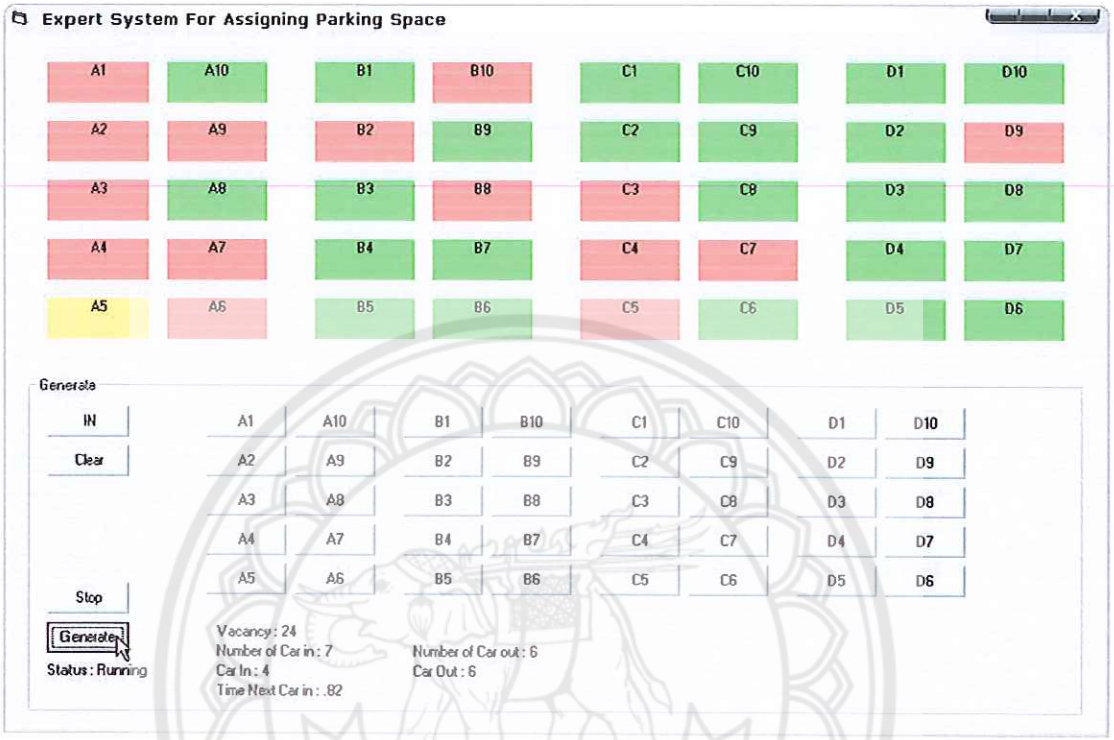


รูปที่ 4.5 ภาพแสดงโปรแกรมขณะมีรถเข้า

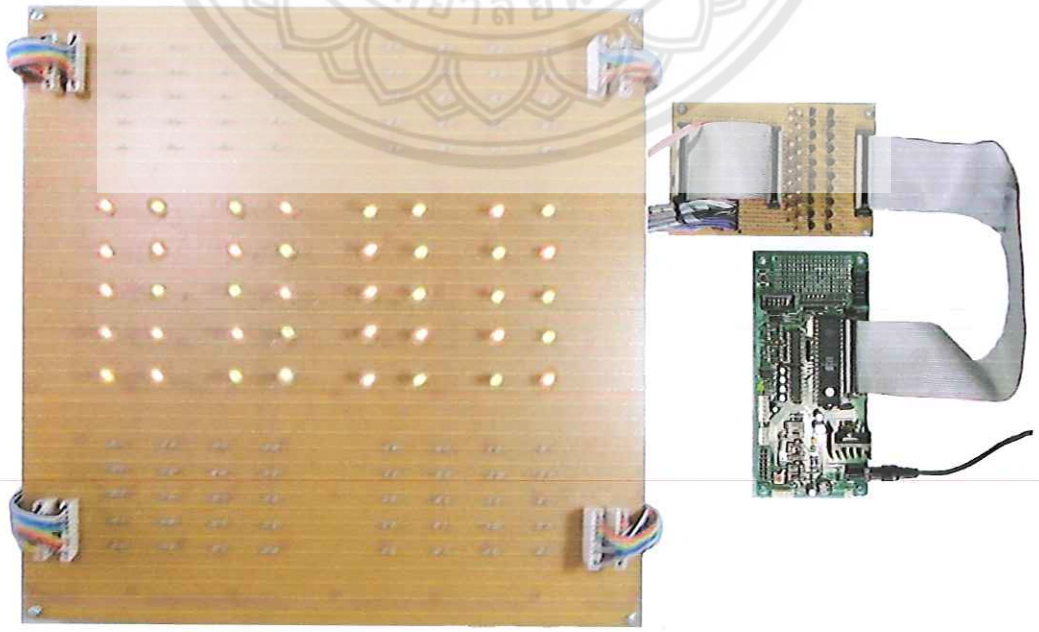


รูปที่ 4.6 ภาพแสดงทางแผงวงจรLED ขณะมีรถเข้า

เมื่อมีรถเข้ามาในลานจอดรถโปรแกรมจะทำการประมวลผล และแนะนำตำแหน่งที่ว่างที่ใกล้ที่สุดให้กับผู้ขับ ดังรูปที่ 4.5 และ รูปที่ 4.6 โดยสถานะของตำแหน่งที่เดิมเป็นสีเขียวจะเปลี่ยนกลายเป็นสีส้มกระพริบ เพื่อบอกว่าตำแน่งที่ใกล้ที่สุดคือตำแหน่งสัญญาณไฟกระพริบสีส้ม

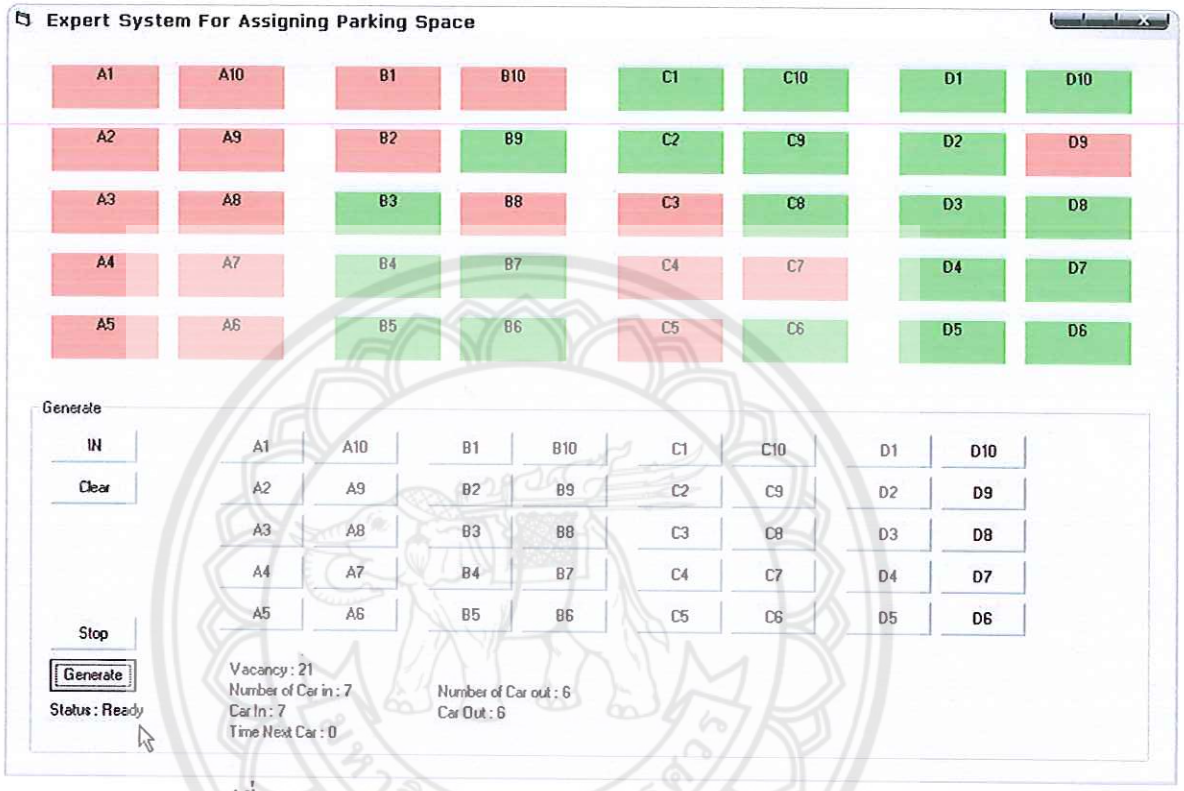


รูปที่ 4.7 ภาพแสดงโปรแกรมขณะทำการจำลองเหตุการณ์

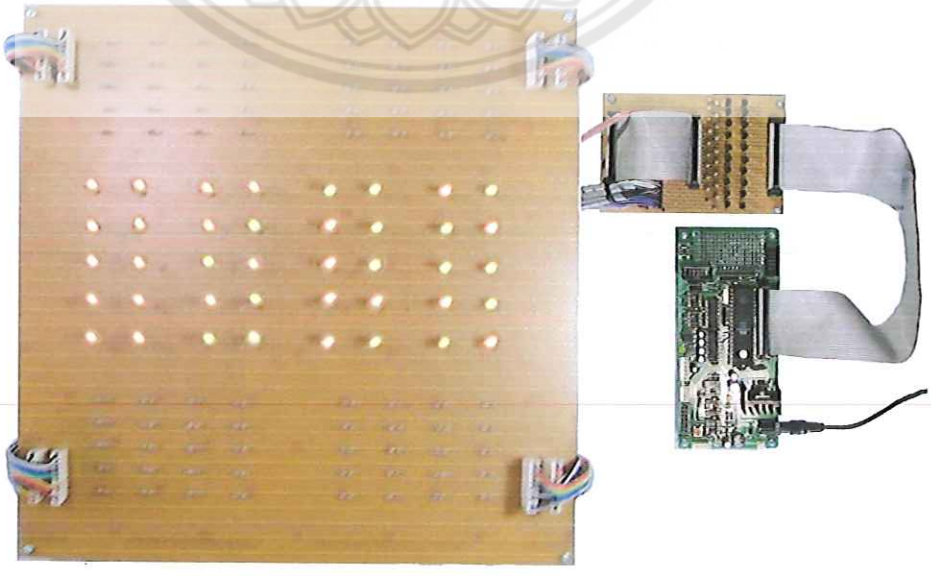


รูปที่ 4.8 ภาพแสดงทางแผงวงจรLED ขณะทำการจำลองเหตุการณ์

ขณะที่โปแกรมทำการจำลองเหตุการณ์โปรแกรมจะแสดงสถานะของโปรแกรมว่ากำลังทำงานอยู่ด้วย (Status: Running) ดังรูปที่ 4.9 และ รูปที่ 4.10 และเมื่อมีรถเข้ามาพร้อมกัน 2 คัน โปรแกรมก็จะทำการแนะนำตำแหน่งที่ใกล้ที่สุดให้กับผู้ขับขี่ทั้ง 2 คันพร้อมกันด้วยสัญญาณไฟกระพริบสีส้ม 2 ตำแหน่งที่ใกล้ที่สุด



รูปที่ 4.11 ภาพแสดงโปรแกรมหลังจากทำการจำลองเหตุการณ์



รูปที่ 4.10 ภาพแสดงทางแผงวงจรLEDหลังจากทำการจำลองเหตุการณ์

หลังจากทำการจำลองเหตุการณ์เสร็จสถานะของโปรแกรมจะแสดงให้ทราบว่าทำการจำลองเหตุการณ์เสร็จเรียบร้อยแล้ว(Status: Ready) ดังรูปที่ 4.11 และ รูปที่ 4.12 คือเมื่อมีรถจอดในตำแหน่งที่ได้แนะนำไว้แล้ว สีของสัญญาณไฟก็จะเปลี่ยนไปเป็นสีแดงเพื่อแสดงว่ามีรถจอดในตำแหน่งนี้แล้ว

จากการทดสอบผลการทำงานของโปรแกรมที่พัฒนาขึ้นพบว่า โปรแกรมสามารถแสดงผลได้อย่างถูกต้อง ในสถานการณ์ที่มีรถเข้ามาในลาดจอดรถโปรแกรมจะทำการจัดหาที่ว่างที่ใกล้ที่สุดในลานจอดรถให้ผู้ขับขี่ และเมื่อผู้ขับขี่เลือกที่จอดรถแล้วโปรแกรมก็จะเปลี่ยนสถานะของที่จอดรถตำแหน่งนั้นว่างมีรถจอดอยู่แล้ว ทำให้ง่ายในการหาที่จอดรถ

เมื่อได้ทำการทดสอบการทำงานร่วมกันระหว่างไมโครคอนโทรลเลอร์ และโปรแกรมไปแล้ว และในบทต่อไปจะเป็นการสรุปผลการทดลอง การระบุปัญหาที่เกิดขึ้นในการทำการทดลองและข้อเสนอแนะซึ่งอาจเป็นแนวทางในการพัฒนาระบบให้สามารถทำงานได้ดียิ่งขึ้นต่อไป

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุปผลการทดลอง

การทำงานของโปรแกรมที่ได้พัฒนาขึ้นจะแบ่งการทำงานออกเป็น 3 ส่วนคือ

- การทำงานในส่วนของการจำลองเหตุการณ์ ทำงานโดยการสุ่มจำนวนรถที่จะเข้าและออกจากลานจอดรถ และกำหนดค่าเวลาต่างๆ ขึ้นมา และทำการจับเวลาเพื่อดำเนินเหตุการณ์ที่จำลองขึ้นโดยเหตุการณ์ที่ถูกจำลองขึ้นจะไปทำการเปลี่ยนข้อมูลในฐานข้อมูลของลานจอดรถด้วย
- การทำงานในส่วนของระบบแนะนำที่ว่างแก่ผู้ขับ ทำงานโดยการดึงข้อมูลต่างๆจากฐานข้อมูลของลานจอดรถมาทำการประมวลผล และทำการแนะนำตำแหน่งที่ว่างที่ใกล้ที่สุดแก่ผู้ขับขี่ที่จะนำรถเข้ามาจอดในลานจอดรถ แสดงตำแหน่งอื่นๆในลานจอดรถ และส่งข้อมูลออกไปให้ไมโครคอนโทรลเลอร์ด้วย
- การทำงานในส่วนของไมโครคอนโทรลเลอร์โดยมีการรับข้อมูลมาจากคอมพิวเตอร์เพื่อนำค่านั้นมาแสดงผลในแผงวงจรที่ได้ทำขึ้นมา โดยข้อมูลที่ได้รับมาจะเป็นชุด การเชื่อมต่อนั้นติดต่อกับคอมพิวเตอร์ด้วยพอร์ต RS232แล้วต่อกับไมโครคอนโทรลเลอร์ CP-S8252 V1.0 เพื่อรับคำสั่ง

จากการประยุกต์ใช้ความรู้ในด้านโปรแกรมมิ่ง และการทำงานกับไมโครคอนโทรลเลอร์ในโครงงาน พบว่าการนำข้อมูลจากฐานข้อมูลมาแสดงผล การจำลองข้อมูลในการทำงาน และการรับค่าจากคอมพิวเตอร์มาแสดงผลในแผงวงจรที่ได้ทำขึ้นมา สามารถแสดงผลข้อมูลในลานจอดรถได้อย่างถูกต้อง

5.2 ปัญหาในการทดลอง

- การเขียนโปรแกรมจำลองเหตุการณ์ การเข้าออกของรถในลานจอดรถให้ครอบคลุมสถานการณ์ทุกสถานการณ์ที่สามารถเกิดขึ้นได้จริง ทำได้ยาก
- ในการตรวจสอบความถูกต้องในการทำงานของโค้ดแอสเซมบลีทำได้ลำบากเพราะต้องทำการเบิร์นลงไมโครคอนโทรลเลอร์และสั่งทำงานจริงก่อน ซึ่งใช้เวลาค่อนข้างมาก ทำให้ต้องใช้เวลามากในการปรับปรุง
- แผงวงจรที่ใช้ในการขยายพอร์ตและแผงวงจร LED ที่ได้ทำขึ้นมานั้นมีความเปราะบางมาก จึงต้องใช้ความระมัดระวังในการทำงานเป็นอย่างมา เพราะถ้าหากเกิดความเสียหายขึ้นจะทำให้ไม่สามารถแสดงผลได้อย่างถูกต้อง

5.3 ข้อเสนอแนะ

- โปรแกรมในโครงงานที่ทางผู้พัฒนาได้พัฒนาขึ้นใช้ วิชวลเบสิก 6 เนื่องจากง่ายต่อการออกแบบและพัฒนา หากมีการนำไปศึกษาพัฒนาต่อไป อาจเลือกใช้ภาษาอื่นในการพัฒนาเพื่อที่จะให้สามารถทำงานได้ซับซ้อนมากขึ้น
- หากสามารถพัฒนาให้สามารถใช้งานได้ในสถานที่จริง ก็จะเป็นการดีอย่างยิ่งเนื่องจากจะช่วยลดระยะเวลาในการจอดรถ และประหยัดทรัพยากรเชื้อเพลิงแก่ผู้ขับได้เป็นอย่างดี

บรรณานุกรม

[1] สัจจะ จรัสรุ่งรวีวร. **คู่มือการเขียนโปรแกรมละใช้งาน Visual Basic 6,** สำนักพิมพ์อิมโพเพรส : 2545

[2] ธีรวัฒน์ ประกอบผล. **การประยุกต์ใช้งานไมโครคอนโทรลเลอร์,** สำนักพิมพ์สมาคมส่งเสริมเทคโนโลยี(ไทย - ญี่ปุ่น) : 2546

ภาคผนวก

ส่วนของโปรแกรมในวิชวลเบสิก 6

```
Option Explicit
Dim time(0 To 39) As Integer
Dim time2(0 To 39) As Integer
Dim flag(0 To 39) As Integer
Dim car_in(0 To 39) As Integer
Dim vacpos(0 To 39) As Integer
Dim tmppos(0 To 39, 0 To 1) As Integer
Dim outpos(0 To 39, 0 To 1) As Integer
Dim position(0 To 39) As Integer
Dim vacnums, carnums, carnums2, flags, flags2, flags3, flags4, k, p, positions, tmp As Integer
Dim mul As Integer
Dim conn As New ADODB.Connection
Dim cmd As New ADODB.Command
Dim rs As New ADODB.Recordset
Dim channel As Integer
Private Const strconn = "Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=c:\park.mdb;Persist Security Info=false"

'-------------function gen pos out--------------
Sub nar(i As Integer)
Dim j As Integer
Dim k As Integer
k = i
   tmppos(i, 0) = outpos(round(Rnd * (39 - vacnums)), 0)
   For j = 0 To i - 1
      If tmppos(i, 0) = tmppos(j, 0) Then
         nar (k)
      Else:
      End If
   Next j
End Sub
'-----------end function gen pos out------------

'-------------function gen out time-------------
Sub ntm(i As Integer)
Dim j As Integer
Dim k As Integer
k = i
   outpos(i, 1) = (i * mul) + ((round(Rnd * 10) + 1) * mul)
   For j = 0 To i - 1
      If outpos(i, 1) = outpos(j, 1) Then
         ntm (k)
      Else:
      End If
   Next j
End Sub
'-----------end function gen out time-----------
```

```
'----------------function toggle----------------
Sub toggle(i As Integer)
Dim j As Integer
   rs.MoveFirst
   For j = 0 To i - 2
      rs.MoveNext
   Next j

If rs.Fields("status").Value = 0 Then
      rs.Fields("status").Value = 1
      rs.Fields("flag").Value = 0
      time(j) = rs.Fields("time").Value * mul
   Else:
      rs.Fields("status").Value = 0
      If flags4 > 0 Then
         flags4 = flags4 + 1
      End If
   End If
   rs.MoveFirst
End Sub
'-------------end function toggle---------------

'----------------button A1-D10------------------
Private Sub A1_Click()
toggle (1)
End Sub

Private Sub A2_Click()
toggle (2)
End Sub

Private Sub A3_Click()
toggle (3)
End Sub

Private Sub A4_Click()
toggle (4)
End Sub

Private Sub A5_Click()
toggle (5)
End Sub

Private Sub A6_Click()
toggle (6)
End Sub

Private Sub A7_Click()
toggle (7)
End Sub

Private Sub A8_Click()
toggle (8)
End Sub

Private Sub A9_Click()
toggle (9)
End Sub
```

```
Private Sub A10_Click()
toggle (10)
End Sub

Private Sub B1_Click()
toggle (11)
End Sub

Private Sub B2_Click()
toggle (12)
End Sub

Private Sub B3_Click()
toggle (13)
End Sub

Private Sub B4_Click()
toggle (14)
End Sub

Private Sub B5_Click()
toggle (15)
End Sub

Private Sub B6_Click()
toggle (16)
End Sub

Private Sub B7_Click()
toggle (17)
End Sub

Private Sub B8_Click()
toggle (18)
End Sub

Private Sub B9_Click()
toggle (19)
End Sub

Private Sub B10_Click()
toggle (20)
End Sub

Private Sub C1_Click()
toggle (21)
End Sub

Private Sub C2_Click()
toggle (22)
End Sub

Private Sub C3_Click()
toggle (23)
End Sub

Private Sub C4_Click()
toggle (24)
```

```
End Sub


Private Sub C5_Click()
toggle (25)
End Sub

Private Sub C6_Click()
toggle (26)
End Sub

Private Sub C7_Click()
toggle (27)
End Sub

Private Sub C8_Click()
toggle (28)
End Sub

Private Sub C9_Click()
toggle (29)
End Sub

Private Sub C10_Click()
toggle (30)
End Sub

Private Sub D1_Click()
toggle (31)
End Sub

Private Sub D2_Click()
toggle (32)
End Sub

Private Sub D3_Click()
toggle (33)
End Sub

Private Sub D4_Click()
toggle (34)
End Sub

Private Sub D5_Click()
toggle (35)
End Sub

Private Sub D6_Click()
toggle (36)
End Sub

Private Sub D7_Click()
toggle (37)
End Sub

Private Sub D8_Click()
toggle (38)
End Sub
```

```
Private Sub D9_Click()
toggle (39)
End Sub

Private Sub D10_Click()
toggle (40)
End Sub
'--------------end button A1-D10----------------

'-----------------button Clear------------------
Private Sub Clear_Click()
Dim i As Integer
   rs.MoveFirst
   For i = 0 To 39
      rs.Fields("status").Value = 0
      rs.Fields("flag").Value = 0
      time(i) = rs.Fields("time").Value * mul
      rs.MoveNext
   Next i
   rs.MoveFirst
   vacnums = 0
   flags = 0
   flags2 = 0
   flags3 = 0
   flags4 = 0
   tmp = 0
   For i = 0 To 39
      vacpos(i) = 0
      position(i) = 0
      time2(i) = 0
   Next i
   Vacnum.Caption = "Vacancy : " & vacnums
   Carnum.Caption = "Number of Car in : 0"
   Carin.Caption = "Car In : 0"
   TNCar.Caption = "Time Next Car in : 0"
   Carnum_out.Caption = "Number of Car out : 0"
   Carout.Caption = "Car Out : 0"
End Sub
'---------------end button Clear----------------

Private Sub Form_Load()
   Dim i As Integer
   channel = 1
   'Set connection to microcontroller
   MSComm1.Settings = "9600,N,8,1"
   MSComm1.CommPort = 1            'use com1
   'MSComm1.InputLen = 1
   MSComm1.PortOpen = True
   MSComm1.RThreshold = 1
   mul = 10
   With conn
      If .State = adStateOpen Then .Close
      .ConnectionString = strconn
      .ConnectionTimeout = 90
      .Open
   End With
```

```
    Dim sqlparking As String
    sqlparking = "SELECT * FROM slot"
    With rs
       If .State = adStateOpen Then .Close
          .ActiveConnection = conn
          .CursorLocation = adUseClient
          .LockType = adLockOptimistic
          .CursorType = adOpenForwardOnly
          .Open sqlparking
    End With

    For i = 0 To 39
       If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
          label1(i).BackColor = &H80FF80
          'MSComm1.Output = ptl(i) & "0"
       ElseIf rs.Fields("Status").Value = 0 And rs.Fields("flag").Value = 1 Then
          label1(i).BackColor = &H80FFFF
          'MSComm1.Output = ptl(i) & "1"
       Else: label1(i).BackColor = &H8080FF
          'MSComm1.Output = ptl(i) & "2"
       End If
       time(i) = rs.Fields("time").Value * mul
       flag(i) = rs.Fields("flag").Value
       rs.MoveNext
    Next i

    rs.MoveFirst
    vacnums = 0
    For i = 0 To 39
       If rs.Fields("status").Value = 0 Then
          vacnums = vacnums + 1
       End If
       rs.MoveNext
    Next i
    rs.MoveFirst
    Vacnum.Caption = "Vacancy : " & vacnums

End Sub

'----------------button Generate----------------
Private Sub Generate_Click()
Dim i As Integer
Dim j As Integer
    vacnums = 0 '-------------------------------------------------------------clear var
    flags = 0
    flags2 = 0
    flags3 = 0
    flags4 = 0
    For i = 0 To 39
       vacpos(i) = 0
       outpos(i, 0) = 0
       outpos(i, 1) = 0
       position(i) = 0
       time2(i) = 0
       car_in(i) = 0
    Next i
    positions = 0
    j = 0
```

```
rs.MoveFirst
For i = 0 To 39
   If rs.Fields("flag").Value = 1 Then
      rs.Fields("flag").Value = 0
   End If
   rs.MoveNext
Next i
rs.MoveFirst

For i = 0 To 39 '-----------------------------------------------------scan vacnums
   If rs.Fields("status").Value = 0 Then
      vacpos(positions) = i + 1
      positions = positions + 1
      vacnums = vacnums + 1
   Else:
      outpos(j, 0) = i + 1
      j = j + 1
   End If
   rs.MoveNext
Next i
rs.MoveFirst
If vacnums < 10 Then '-------------------------------------gen num of car in
   carnums = round(Rnd * vacnums)
   If vacnums = 0 Then
      flags = 0
   Else:
      flags = 1
   End If
Else:
   carnums = round(Rnd * 9) + 1
   flags = 1
End If
If (40 - vacnums) < 10 Then '-----------------------------gen num of car out
   carnums2 = round(Rnd * (40 - vacnums))
   If (40 - vacnums) = 0 Then
      flags3 = 0
   Else:
      flags3 = 1
   End If
Else:
   carnums2 = round(Rnd * 9) + 1
   flags3 = 1
End If
tmp = carnums2 - 1
car_in(0) = 0 '-------------------------------------------------gen time of car in
For i = 1 To carnums - 1
   car_in(i) = car_in(i - 1) + ((round(Rnd * 10) + 1) * mul)
Next i
For i = 0 To carnums2 - 1 '-------------------------------gen time&pos of car out
   nar (i)
   ntm (i)
Next i

For i = 0 To carnums2 - 1
   outpos(i, 0) = tmppos(i, 0)
Next i
p = 0
```

```
   Vacnum.Caption = "Vacancy : " & vacnums '---------------------------------caption
   Carnum.Caption = "Number of Car in : " & carnums
   If flags = 1 Then
      Carin.Caption = "Car In : 1"
   Else:
      Carin.Caption = "Car In : 0"
   End If
   TNCar.Caption = "Time Next Car in : " & (car_in(1) / mul)
   Carnum_out.Caption = "Number of Car out : " & carnums2
   Carout.Caption = "Car Out : 0"
End Sub
'--------------end button Generate--------------

'-------------------button IN-------------------
Private Sub IN_Click()
Dim i As Integer
   If flags4 = 0 And vacnums <> 0 Then
      flags4 = vacnums
      flags4 = flags4 - 1
   ElseIf flags4 = 0 And vacnums = 0 Then
      flags4 = 0
   Else:
      flags4 = flags4 - 1
   End If
   rs.MoveFirst
   For i = 0 To 39
      If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
         rs.Fields("flag").Value = 1
         Exit For
      Else: rs.MoveNext
      End If
   Next i
   rs.MoveFirst
End Sub
'-----------------end button IN-----------------


'-----------------button Stop-----------------
Private Sub Stop_Click()
Dim i As Integer
   flags = 0
   flags2 = 0
   flags3 = 0
   rs.MoveFirst
   For i = 0 To 39
      If rs.Fields("flag").Value = 1 Then
         rs.Fields("flag").Value = 0
      End If
      rs.MoveNext
   Next i
   rs.MoveFirst
   vacnums = 0
   For i = 0 To 39
      If rs.Fields("status").Value = 0 Then
         vacnums = vacnums + 1
      End If
      rs.MoveNext
   Next i
```

```
   rs.MoveFirst
   For i = 0 To 39
      vacpos(i) = 0
      position(i) = 0
      time2(i) = 0
   Next i
   Vacnum.Caption = "Vacancy : " & vacnums
   Carnum.Caption = "Number of Car in : 0"
   Carin.Caption = "Car In : 0"
   TNCar.Caption = "Time Next Car in : 0"
   Carnum_out.Caption = "Number of Car out : 0"
   Carout.Caption = "Car Out : 0"
End Sub
'----------------end button Stop----------------

Private Sub Timer1_Timer()
Dim i As Integer
Dim j As Integer
Dim l As Integer
Dim m As Integer
Dim n As Integer
   For i = 0 To 39
      Vacnum.Caption = "Vacancy : " & vacnums
      If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
         label1(i).BackColor = &H80FF80
'         MSComm1.Output = ptl(i) & "0"
      ElseIf rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 1 Then
         label1(i).BackColor = &H80FFFF
'         MSComm1.Output = ptl(i) & "1"
      Else:
         label1(i).BackColor = &H8080FF
'         MSComm1.Output = ptl(i) & "2"
      End If
      rs.MoveNext
   Next i
   rs.MoveFirst
   For i = 0 To 39
      flag(i) = rs.Fields("flag").Value
      rs.MoveNext
   Next i
   rs.MoveFirst
   For i = 0 To 39
      If flag(i) = 1 Then
         time(i) = time(i) - 1
         If time(i) = 0 Then
            rs.MoveFirst
            For j = 0 To i - 1
               rs.MoveNext
            Next j
            rs.Fields("flag").Value = 0
            time(i) = rs.Fields("time").Value * mul
         End If
      End If
   Next i
   m = 0
   n = 0
   vacnums = 0
   rs.MoveFirst
```

```
For i = 0 To 39
   vacpos(i) = 0
Next i
For i = 0 To 39
   l = 0
   For j = 0 To carnums - 1
      If i = position(j) - 1 Then
         l = 1
         Exit For
      End If
   Next j
   If rs.Fields("status").Value = 0 And l = 0 Then
      vacpos(m) = i + 1
      m = m + 1
      vacnums = vacnums + 1
   End If
   rs.MoveNext
Next i
rs.MoveFirst
If flags = 1 Then
   For j = 0 To carnums - 1
      If car_in(j) = 0 Then
         positions = j
         For i = 0 To 39
            If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
               rs.Fields("flag").Value = 1
               flags2 = 1
               If round(Rnd * 3) < 3 Then
                  position(positions) = i + 1
                  time2(positions) = (rs.Fields("time").Value - 1) * mul
               Else:
                  position(positions) = vacpos(round(Rnd * vacnums))
                  If position(positions) > 1 Then
                     rs.MoveFirst
                     For l = 0 To position(positions) - 2
                        rs.MoveNext
                     Next l
                     time2(positions) = (rs.Fields("time").Value - 1) * mul
                     rs.MoveFirst
                  Else:
                     time2(positions) = (rs.Fields("time").Value - 1) * mul
                  End If
               End If
               Exit For
            Else: rs.MoveNext
            End If
         Next i
         rs.MoveFirst
         k = j + 1
         TNCar.Caption = "Time Next Car in : " & (car_in(k) / mul)
         Carin.Caption = "Car In : " & k
      End If
      If j = carnums - 1 And car_in(carnums - 1) = 0 Then
         flags = 0
         TNCar.Caption = "Time Next Car : 0"
         Exit For
      End If
      car_in(j) = car_in(j) - 1
```

```
        TNCar.Caption = "Time Next Car in : " & (car_in(k) / mul)
      Next j

End If
  If flags2 = 1 Then
    For i = 0 To positions
      If time2(i) = 0 Then
        toggle (position(i))
      End If
      l = 0
      rs.MoveFirst
      For j = 0 To 39
        If rs.Fields("status").Value = 0 Then
          l = l + 1
        End If
        rs.MoveNext
      Next j
      rs.MoveFirst
      If i = carnums - 1 And l = vacnums Then
        flags2 = 0
        Exit For
      End If
      time2(i) = time2(i) - 1
    Next i
  End If
  If flags3 = 1 Then
    For i = 0 To carnums2 - 1
      If outpos(i, 1) = 0 Then
        j = outpos(i, 0)
        toggle (j)
        tmp = tmp - 1
        p = p + 1
      End If
      outpos(i, 1) = outpos(i, 1) - 1
      Carout.Caption = "Car Out : " & p
    Next i
    If tmp = -1 Then
      flags3 = 0
    End If
  End If
  If flags = 0 And flags2 = 0 And flags3 = 0 Then
    Stat.Caption = "Status : Ready"
  Else:
    Stat.Caption = "Status : Running"
  End If
  If flags4 = vacnums Then
    rs.MoveFirst
    For i = 0 To 39
      If rs.Fields("flag").Value = 1 Then
        rs.Fields("flag").Value = 0
      End If
    rs.MoveNext
    Next i
    rs.MoveFirst
    flags4 = 0
  End If
  For i = 0 To 39
    Vacnum.Caption = "Vacancy : " & vacnums
    If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
      label1(i).BackColor = &H80FF80
    ElseIf rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 1 Then
```

```
        label1(i).BackColor = &H80FFFF


      Else: label1(i).BackColor = &H8080FF
    End If
      rs.MoveNext
  Next i
  rs.MoveFirst
End Sub

Private Sub Timer2_Timer() 'Use to send data to mocro
  rs.MoveFirst
    sendCommand (channel)
  rs.MoveFirst
End Sub
Private Sub sendCommand(ByRef mchannel As Integer)
Dim i As Integer
  rs.MoveFirst
  Select Case mchannel
    Case 1
      MSComm1.Output = CStr(channel)
      For i = 0 To 9
        Vacnum.Caption = "Vacancy : " & vacnums
        If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
          label1(i).BackColor = &H80FF80
          MSComm1.Output = "0"
        ElseIf rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 1 Then
          label1(i).BackColor = &H80FFFF
          MSComm1.Output = "1"
        Else:
          label1(i).BackColor = &H8080FF
          MSComm1.Output = "2"
        End If
        rs.MoveNext
      Next i
      MSComm1.Output = "E"
    Case 2
      For i = 0 To 9
        rs.MoveNext
      Next i
      MSComm1.Output = CStr(channel)
      For i = 10 To 19
        Vacnum.Caption = "Vacancy : " & vacnums
        If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
          label1(i).BackColor = &H80FF80
          MSComm1.Output = "0"
        ElseIf rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 1 Then
          label1(i).BackColor = &H80FFFF
          MSComm1.Output = "1"
        Else:
          label1(i).BackColor = &H8080FF
          MSComm1.Output = "2"
        End If
        rs.MoveNext
      Next i
      MSComm1.Output = "E"
    Case 3
      For i = 0 To 19
        rs.MoveNext
      Next i
      MSComm1.Output = CStr(channel)
```

```
        For i = 20 To 29
           Vacnum.Caption = "Vacancy : " & vacnums
           If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
              label1(i).BackColor = &H80FF80
              MSComm1.Output = "0"
           ElseIf rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 1 Then
              label1(i).BackColor = &H80FFFF
              MSComm1.Output = "1"
           Else:
              label1(i).BackColor = &H8080FF
              MSComm1.Output = "2"
           End If
           rs.MoveNext
        Next i
        MSComm1.Output = "E"
     Case 4
        For i = 0 To 29
           rs.MoveNext
        Next i
        MSComm1.Output = CStr(channel)
        For i = 30 To 39
           Vacnum.Caption = "Vacancy : " & vacnums
           If rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 0 Then
              label1(i).BackColor = &H80FF80
              MSComm1.Output = "0"
           ElseIf rs.Fields("status").Value = 0 And rs.Fields("flag").Value = 1 Then
              label1(i).BackColor = &H80FFFF
              MSComm1.Output = "1"
           Else:
              label1(i).BackColor = &H8080FF
              MSComm1.Output = "2"
           End If
           rs.MoveNext
        Next i
        MSComm1.Output = "E"
   End Select

   channel = channel + 1
   If channel > 4 Then
      channel = 1
   End If
   rs.MoveFirst
End Sub

Private Sub Command1_Click()
On Error Resume Next
   Me.Timer2.Interval = Val(Me.Text1.Text)
End Sub

Private Sub Command2_Click()
   Me.Timer2.Enabled = Not (Me.Timer2.Enabled)
   If Me.Timer2.Enabled Then
      Me.Command2.Caption = "stop"
   Else
      Me.Command2.Caption = "start"
   End If
End Sub
```

โปรแกรมในส่วนของภาษา Assembly

```
ORG         0000H

; กำหนดตั้งค่าให้กับ Timer1 เพื่อใช้สร้าง Baud Rate ในการรับส่งข้อมูล

            MOV IE,#00000000B
            MOV TMOD,#00100000B
            MOV TL1,#0FDH
            MOV TH1,#-5

; กำหนดค่าให้กับ SCON Register โดยเลือกโหมด 1 และให้รับข้อมูล REN=1

            MOV SCON,#01010010B
            SETB TR1
            ;MOV P0,#00000000B          ; เซทค่าให้พอร์ตขนาน P0 เป็น 0 ทุกบิต
            CLR P0

            ;MOV P1,#00000000B          ; เซทค่าให้พอร์ตขนาน P1 เป็น 0 ทุกบิต
            CLR P1

            ;MOV P2,#00000000B          ; เซทค่าให้พอร์ตขนาน P2 เป็น 0 ทุกบิต ยกเว้น
                                        ; P2.7 เพื่อจ่ายให้ LE(74ls373) G1(74ls138)
            CLR P2

;ขั้นตอนการทำงานหลัก
INDEX:      LCALL SUB_RXD               ; ไปรับข้อมูลจาก Serial Port ที่ส่งมาจาก PC
                LJMP PARK_FL            ; เลือกชั้นที่ต้องการส่งค่าไปแสดงผล
FIRST:      LJMP POS_ONE                ; แสดงผลตำแหน่งที่ 1
                ;LCALL POS_TWO          ; แสดงผลตำแหน่งที่ 2
                ;LCALL POS_THREE        ; แสดงผลตำแหน่งที่ 3
```

```
                    ;LCALL POS_FOUR     ; แสดงผลตำแหน่งที่ 4
                    ;LCALL POS_FIVE     ; แสดงผลตำแหน่งที่ 5
                    ;LCALL POS_SIX      ; แสดงผลตำแหน่งที่ 6
                    ;LCALL POS_SEVEN    ; แสดงผลตำแหน่งที่ 7
                    ;LCALL POS_EIGHT    ; แสดงผลตำแหน่งที่ 8
                    ;LCALL POS_NINE     ; แสดงผลตำแหน่งที่ 9
                    ;LCALL POS_TEN      ; แสดงผลตำแหน่งที่ 10
                    ;LCALL END_CMD      ; รับตัวอักษร "E" แสดงการจบชุดคำสั่ง
                    ;LJMP INDEX

SUB_RXD:    JNB RI,$
                    CLR RI
                    MOV A,SBUF
                    ;MOV A,#30H
                    RET

PARK_FL:    CJNE A,#31H,FL_B
                    LJMP FL_A
                    RET

;----------------------กำหนดชั้นในการจ่ายไฟให้กับ led เพื่อทำการแสดงผล
;++++++++++++++++++++++++++++++++++++++++++++++Park Drive
;+              |    A    |    B    |    C    |    D    |
                     +
;+      P2.4   | 0    I    1    I    0    I    1    I
                +
;+      P2.6   | 0    |    0    |    1    |    1    |
                +
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++

FL_A:           MOV P2,#00001111B
                    MOV P0,#11111111B
```

```
                MOV P1,#11111111B
                ;CLR P0
                ;CLR P1
                ;CLR P2
                ;SETB P2.7
                LJMP FIRST

FL_B:       CJNE A,#32H,FL_C
                MOV P2,#00011111B
                MOV P0,#11111111B
                MOV P1,#11111111B
                ;CJNE A,#32H,FL_C
                ;CLR P0
                ;CLR P1
                ;CLR P2
                ;SETB P2.4
                LJMP FIRST

FL_C:       CJNE A,#33H,FL_D
                MOV P2,#01001111B
                MOV P0,#11111111B
                MOV P1,#11111111B
                ;CJNE A,#33H,FL_D
                ;CLR P0
                ;CLR P1
                ;CLR P2
                ;SETB P2.6
                LJMP FIRST

FL_D:       MOV P2,#01011111B
                MOV P0,#11111111B
                MOV P1,#11111111B
```

```
                    ;CLR P0
                    ;CLR P1
                    ;CLR P2
                    ;SETB P2.6
                    ;SETB P2.4
                    LJMP FIRST

POS_ONE:    LCALL SUB_RXD
                    LJMP LED1
                    RET
POS_TWO:    LCALL SUB_RXD
                    LJMP LED2
                    RET
POS_THREE:LCALL SUB_RXD
                    LJMP LED3
                    RET
POS_FOUR:   LCALL SUB_RXD
                    LJMP LED4
                    RET
POS_FIVE:   LCALL SUB_RXD
                    LJMP LED5
                    RET
POS_SIX:    LCALL SUB_RXD
                    LJMP LED6
                    RET
POS_SEVEN:  LCALL SUB_RXD
                          LJMP LED7
                          RET
POS_EIGHT:          LCALL SUB_RXD
                          LJMP LED8
                          RET
POS_NINE:           LCALL SUB_RXD
```

```
                    LJMP LED9
                    RET
POS_TEN:        LCALL SUB_RXD
                    LJMP LED10
                    RET

;++++++++++++++++++++++++++++++++++++++++++++LED DRIVE
;+      Green = 01  -+-  Yellow = 11 -+-  Red = 10                    +
;++++++++++++++++++++++++++++++++++++++++++++++++++++

;-------------------------------------------------------------------------LED1
LED1:  LJMP LED1_G
       EXIT1: RET

       LED1_G:     CJNE A,#30H,LED1_Y  ;30H = 0 green
                   SETB P0.0
                   CLR P0.1
                   CPL P0.0
                   CPL P0.1
                   LJMP POS_TWO

       LED1_Y:     CJNE A,#31H,LED1_R  ;31H = 1 yellow   if not equal 1(Yellow) jump
to LED1_R
                   SETB P0.0
                   SETB P0.1
                   CPL P0.0
                   CPL P0.1
                   LJMP POS_TWO

       LED1_R:     ;CJNE A,#32H,A1_N    ;32H = 2 red
                   CLR P0.0
                   SETB P0.1
```

```
                    CPL P0.0
                    CPL P0.1
                    LJMP POS_TWO
;-------------------------------------------------------------------------LED2
LED2:       LJMP LED2_G
      EXIT2: RET

      LED2_G:   CJNE A,#30H,LED2_Y  ;30H = 0 green
                    SETB P0.2
                    CLR P0.3
                    CPL P0.2
                    CPL P0.3
                    LJMP POS_THREE

      LED2_Y:   CJNE A,#31H,LED2_R  ;31H = 1 yellow    if not equal 1(Yellow) jump
to LED2_R
                    SETB P0.2
                    SETB P0.3
                    CPL P0.2
                    CPL P0.3
                    LJMP POS_THREE

      LED2_R:   ;CJNE A,#32H,A1_N     ;32H = 2 red
                    CLR P0.2
                    SETB P0.3
                    CPL P0.2
                    CPL P0.3
                    LJMP POS_THREE
;-------------------------------------------------------------------------LED3
LED3:       LJMP LED3_G
      EXIT3: RET
```

```
        LED3_G:     CJNE A,#30H,LED3_Y  ;30H = 0 green
                    SETB P0.4
                    CLR P0.5
                    CPL P0.4
                    CPL P0.5
                    LJMP POS_FOUR

        LED3_Y:     CJNE A,#31H,LED3_R  ;31H = 1 yellow   if not equal 1(Yellow) jump
to LED5_R
                    SETB P0.4
                    SETB P0.5
                    CPL P0.4
                    CPL P0.5
                    LJMP POS_FOUR

        LED3_R:     ;CJNE A,#32H,A1_N    ;32H = 2 red
                    CLR P0.4
                    SETB P0.5
                    CPL P0.4
                    CPL P0.5
                    LJMP POS_FOUR
;-------------------------------------------------------------------LED4
LED4:           LJMP LED4_G
        EXIT4: RET

        LED4_G:     CJNE A,#30H,LED4_Y  ;30H = 0 green
                    SETB P0.6
                    CLR P0.7
                    CPL P0.6
                    CPL P0.7
                    LJMP POS_FIVE
```

```
        LED4_Y:     CJNE A,#31H,LED4_R  ;31H = 1 yellow   if not equal 1(Yellow) jump
to LED5_R
                    SETB P0.6
                    SETB P0.7
                    CPL P0.6
                    CPL P0.7
                    LJMP POS_FIVE

        LED4_R:     ;CJNE A,#32H,A1_N    ;32H = 2 red
                    CLR P0.6
                    SETB P0.7
                    CPL P0.6
                    CPL P0.7
                    LJMP POS_FIVE
;-------------------------------------------------------------------LED5
LED5:           LJMP LED5_G
        EXIT5: RET

        LED5_G:     CJNE A,#30H,LED5_Y  ;30H = 0 green
                    SETB P1.0
                    CLR P1.1
                    CPL P1.0
                    CPL P1.1
                    LJMP POS_SIX

        LED5_Y:     CJNE A,#31H,LED5_R  ;31H = 1 yellow   if not equal 1(Yellow) jump
to LED5_R
                    SETB P1.0
                    SETB P1.1
                    CPL P1.0
                    CPL P1.1
                    LJMP POS_SIX
```

```
        LED5_R:     ;CJNE A,#32H,A1_N    ;32H = 2 red
                    CLR P1.0
                    SETB P1.1
                    CPL P1.0
                    CPL P1.1
                    LJMP POS_SIX
;-------------------------------------------------------------------------LED6
LED6:           LJMP LED6_G
        EXIT6: RET

        LED6_G:     CJNE A,#30H,LED6_Y  ;30H = 0 green
                    SETB P1.2
                    CLR P1.3
                    CPL P1.2
                    CPL P1.3
                    LJMP POS_SEVEN

        LED6_Y:     CJNE A,#31H,LED6_R  ;31H = 1 yellow   if not equal 1(Yellow) jump
to LED6_R
                    SETB P1.2
                    SETB P1.3
                    CPL P1.2
                    CPL P1.3
                    ;LJMP DELAY
                    LJMP POS_SEVEN

        LED6_R:     ;CJNE A,#32H,A1_N    ;32H = 2 red
                    CLR P1.2
                    SETB P1.3
                    CPL P1.2
                    CPL P1.3
```

```
                    LJMP POS_SEVEN
;-------------------------------------------------------------------------LED7
LED7:          LJMP LED7_G
        EXIT7: RET

        LED7_G:     CJNE A,#30H,LED7_Y  ;30H = 0 green
                    SETB P1.4
                    CLR P1.5
                    CPL P1.4
                    CPL P1.5
                    LJMP POS_EIGHT

        LED7_Y:     CJNE A,#31H,LED7_R  ;31H = 1 yellow   if not equal 1(Yellow) jump
to LED7_R
                    SETB P1.4
                    SETB P1.5
                    CPL P1.4
                    CPL P1.5
                    LJMP POS_EIGHT

        LED7_R:     ;CJNE A,#32H,A1_N    ;32H = 2 red
                    CLR P1.4
                    SETB P1.5
                    CPL P1.4
                    CPL P1.5
                    LJMP POS_EIGHT
;-------------------------------------------------------------------------LED8
LED8:          LJMP LED8_G
        EXIT8: RET

        LED8_G:     CJNE A,#30H,LED8_Y  ;30H = 0 green
                    SETB P1.6
```

```
                    CLR P1.7
                    CPL P1.6
                    CPL P1.7
                    LJMP POS_NINE

        LED8_Y:     CJNE A,#31H,LED8_R  ;31H = 1 yellow   if not equal 1(Yellow) jump
to LED8_R
                    SETB P1.6
                    SETB P1.7
                    CPL P1.6
                    CPL P1.7
                    LJMP POS_NINE

        LED8_R:     ;CJNE A,#32H,A1_N    ;32H = 2 red
                    CLR P1.6
                    SETB P1.7
                    CPL P1.6
                    CPL P1.7
                    LJMP POS_NINE
;-------------------------------------------------------------------LED9
LED9:           LJMP LED9_G
        EXIT9: RET

        LED9_G:     CJNE A,#30H,LED9_Y  ;30H = 0 green
                    SETB P2.0
                    CLR P2.1
                    CPL P2.0
                    CPL P2.1
                    LJMP POS_TEN

        LED9_Y:     CJNE A,#31H,LED9_R  ;31H = 1 yellow   if not equal 1(Yellow) jump
to LED9_R
```

```
                SETB P2.0
                SETB P2.1
                CPL P2.0
                CPL P2.1
                LJMP POS_TEN

    LED9_R:     ;CJNE A,#32H,A1_N    ;32H = 2 red
                CLR P2.0
                SETB P2.1
                CPL P2.0
                CPL P2.1
                LJMP POS_TEN
;-------------------------------------------------------------------LED10
LED10:      LJMP LED10_G
    EXIT10:     RET

    LED10_G:    CJNE A,#30H,LED10_Y      ;30H = 0 green
                SETB P2.2
                CLR P2.3
                CPL P2.2
                CPL P2.3
                LJMP END_CMD

    LED10_Y:    CJNE A,#31H,LED10_R;31H = 1 yellow   if not equal 1(Yellow) jump
to LED10_R
                SETB P2.2
                SETB P2.3
                CPL P2.2
                CPL P2.3
                LJMP END_CMD

    LED10_R:    ;CJNE A,#32H,A1_N    ;32H = 2 red
```

```
                    CLR P2.2
                    SETB P2.3
                    CPL P2.2
                    CPL P2.3
                    LJMP END_CMD

;-------------------------------------------------------------------------------End of command
END_CMD: LCALL SUB_RXD                                          ;รับค่าจนกว่าจะได้
ตัว "E" ซึ่งแสดงการจบชุดคำสั่ง
                    CJNE A,#45H,END_CMD    ; 45H = E
                    LCALL INDEX
END
```