บทที่ 2
ทฤษฎีเบื้องต้นและส่วนที่เกี่ยวข้องในโครงงาน

2.1 ความหมายและหลักการเบื้องต้นของไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (Microcontroller) คือ อุปกรณ์ทางอิเล็กทรอนิกส์ที่ทำหน้าที่เปรียบเสมือนคอมพิวเตอร์ขนาดเล็กที่ใช้ควบคุมการทำงานของเครื่องใช้ไฟฟ้า หรือระบบควบคุมทางอิเล็กทรอนิกส์ต่างๆ ให้มีความสามารถในการทำงานมากขึ้น ซึ่งการควบคุมดังกล่าวจะถูกกำหนดโดยโปรแกรมการทำงานที่เราเขียนลงไปในตัวไมโครคอนโทรลเลอร์ จึงทำให้เราสามารถนำไมโครคอนโทรลเลอร์มาประยุกต์ใช้ควบคุมการทำงานของอุปกรณ์ไฟฟ้ารอบตัวได้ เช่น ระบบอัตโนมัติของเครื่องซักผ้า ระบบอัตโนมัติของเครื่องปรับอากาศ เป็นต้น ซึ่งภายในไมโครคอนโทรลเลอร์จะมีโครงสร้างหลักอยู่ 5 ส่วน ดังต่อไปนี้

2.1.1 หน่วยประมวลผลกลาง (Central Processing Unit: CPU)

หน่วยประมวลผลกลาง คือ ส่วนที่ทำหน้าที่ประมวลผลสัญญาณที่เข้ามาในระบบ แล้วทำการส่งสัญญาณที่ได้จากการประมวลผลไปยังส่วนต่างๆเพื่อควบคุมการทำงานต่อไป ซึ่งส่วนที่เป็นหัวใจหลักภายใน CPU คือ หน่วยคำนวณทางคณิตศาสตร์และตรรกศาสตร์ (Arithmetic and Logic Unit: ALU) เป็นส่วนที่ทำหน้าที่คำนวณทางคณิตศาสตร์หรือทำการตัดสินใจแบบมีเงื่อนไขซึ่งจะมีการทำงานที่ซับซ้อน โดยลำดับในการทำงานของ CPU จะขึ้นอยู่กับการจัดลำดับคำสั่งในการทำงาน (Programming code) ที่ถูกบรรจุอยู่ภายในหน่วยความจำ

2.2.2 หน่วยความจำ (Memory Unit)

หน่วยความจำ คือ ส่วนที่ทำหน้าที่เก็บข้อมูล ภายในไมโครคอนโทรลเลอร์จะประกอบด้วยหน่วยความจำอยู่ 3 แบบ คือ หน่วยความจำโปรแกรม หน่วยความจำข้อมูลแบบแรม และหน่วยความจำข้อมูลอีอีพรอม ซึ่งมีรายละเอียดดังนี้

- หน่วยความจำโปรแกรม (Program memory) คือ หน่วยความจำที่ใช้เก็บข้อมูลโปรแกรมคำสั่งที่ผู้ใช้เขียนขึ้นมา หรือที่เรียกว่าโปรแกรมมอนิเตอร์ (Monitor program) เมื่อไม่มีไฟเลี้ยงจ่ายให้กับไมโครคอนโทรลเลอร์ ข้อมูลก็จะยังคงอยู่ไม่สูญหายไป CPU จะเข้ามาติดต่อเพื่ออ่านข้อมูลโปรแกรมหรือรหัสคำสั่งจากหน่วยความจำในส่วนนี้ แล้วนำไปประมวลผลเพื่อควบคุมการทำงานของระบบทั้งหมดต่อไป ชนิดของหน่วยความจำโปรแกรมที่ใช้ในไมโครคอนโทรลเลอร์จะมีอยู่ 3 แบบที่นิยมใช้ คือ แบบอีพรอม แบบอีอีพรอม และ แบบแฟลช ดังนี้

• แบบอีพรอม (EPROM: Erasable Programmable Read Only Memory) จะแบ่งออกเป็น 2 แบบ คือ แบบโปรแกรมได้ครั้งเดียว หรือที่เรียกว่า แบบ OTP (One Time

Programmable) และ แบบโปรแกรมได้หลายครั้ง ในแบบหลังนี้จะสามารถโปรแกรมใหม่ได้ 10-100 ครั้ง ซึ่งเวลาลบข้อมูลเก่าออกจะลบด้วยแสงอัลตร้าไวโอเล็ต

• แบบอีอีพรอม (EEPROM: Electrically Erasable Programmable Read Only Memory) จะทำการลบและเขียนข้อมูลใหม่ด้วยสัญญาณไฟฟ้า โดยที่สามารถโปรแกรมใหม่ได้ตั้งแต่ 100 ครั้งขึ้นไป และในบางตระกูลทำได้ถึง 1 ล้านครั้ง แต่ในปัจจุบันหน่วยความจำโปรแกรมแบบนี้ไม่เป็นที่นิยมใช้ในไมโครคอนโทรลเลอร์แล้ว เนื่องจากมีต้นทุนสูง

• แบบแฟลช (Flash) จะทำการลบและเขียนข้อมูลใหม่ด้วยสัญญาณไฟฟ้า โดยที่สามารถโปรแกรมใหม่ได้ตั้งแต่ 100 ครั้งขึ้นไป และในบางตระกูลทำได้ถึง 1 แสนครั้ง ขึ้นอยู่กับแรงดันที่ใช้ในการโปรแกรม หน่วยความจำโปรแกรมแบบแฟลชต่างกับแบบอีอีพรอมในการใช้งานตรงที่กระบวนการลบข้อมูล นั่นคือ แบบแฟลชจะไม่สามารถเลือกลบเฉพาะเจาะจงบางแอดเดรสหรือบางตำแหน่งได้ เมื่อต้องการลบข้อมูลจะต้องลบทั้งหมด ในปัจจุบันหน่วยความจำแบบนี้เป็นที่นิยมใช้กันมาก เนื่องจากราคาไม่สูง

- หน่วยความจำข้อมูลแบบแรม (RAM data memory, RAM: Random Access Memory) คือ หน่วยความจำที่ใช้เก็บข้อมูลทั้งในระหว่างและหลังจากการประมวลผล ยิ่งมีมากยิ่งทำให้การทำงานสะดวกมากขึ้น เพราะหน่วยความจำแรมมีอัตราเร็วในการอ่าน/เขียนข้อมูลสูงมาก และมีจำนวนครั้งในการอ่าน/เขียนไม่จำกัด เมื่อไม่มีไฟเลี้ยงจ่ายให้กับไมโครคอนโทรลเลอร์ ข้อมูลก็จะสูญหายไป ในพื้นที่ของหน่วยความจำข้อมูลแรมจะแบ่งออกเป็น 2 ส่วน คือ ส่วนของหน่วยความจำสำหรับใช้งานทั่วไป (General Purpose Register: GPR) ซึ่งใช้สำหรับเก็บผลลัพธ์และเงื่อนไขต่างๆของโปรแกรม และส่วนของรีจิสเตอร์ฟังก์ชันพิเศษ (Special Function Register: SFR) ซึ่งใช้สำหรับกำหนดเงื่อนไขการทำงานและบันทึกสถานะการทำงานของไมโครคอนโทรลเลอร์

- หน่วยความจำข้อมูลแบบอีอีพรอม (EEPROM data memory) คือ หน่วยความจำพิเศษที่ใช้เก็บข้อมูลที่ในไมโครคอนโทรลเลอร์บางเบอร์หรือบางตระกูลไม่มี นั่นคือ เมื่อไม่มีไฟเลี้ยงจ่ายให้กับไมโครคอนโทรลเลอร์ ข้อมูลล่าสุดก็จะยังคงอยู่ไม่สูญหายไป การอ่าน/เขียนข้อมูลจะใช้สัญญาณไฟฟ้า และจำนวนครั้งในการเขียนข้อมูลใหม่โดยปกติจะทำได้ 1 ล้านครั้งขึ้นไป

2.2.3. ส่วนเชื่อมต่อสัญญาณทางไฟฟ้า (Interface Unit)

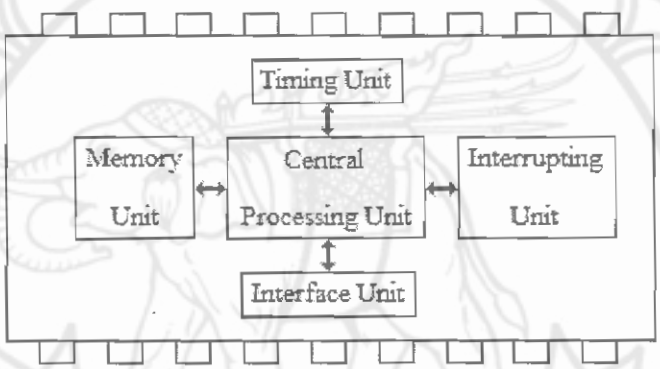
ส่วนเชื่อมต่อสัญญาณทางไฟฟ้า หรือเรียกอีกอย่างว่า ส่วนรับและแสดงผลข้อมูล (I/O: Input/Output) คือ ส่วนที่ทำหน้าที่ติดต่อสัญญาณระหว่างไมโครคอนโทรลเลอร์กับอุปกรณ์ภายนอก จะมีอยู่ 2 แบบ คือ อินพุตและเอาต์พุตแบบดิจิตอล (Digital I/O) ซึ่งจะรับข้อมูลและส่งข้อมูลด้วยสัญญาณแบบดิจิตอล (Digital signal) และ อินพุตและเอาต์พุตแบบอนาล็อก (Analog I/O) ซึ่งจะรับข้อมูลและส่งข้อมูลด้วยสัญญาณแบบอนาล็อก (Analog signal) อินพุตและเอาต์พุตแบบอนาล็อกจะมีอยู่ในไมโครคอนโทรลเลอร์บางรุ่นเท่านั้น

2.2.4 ส่วนกำเนิดสัญญาณนาฬิกา (Timing Unit)

ส่วนกำเนิดสัญญาณนาฬิกา คือ ส่วนที่ทำหน้าที่สร้างสัญญาณนาฬิกา โดยใช้วงจรเชื่อมต่อกับไมโครคอนโทรลเลอร์ที่เรียกว่า วงจรออสซิลเลเตอร์ (Oscillator circuit) ซึ่งมีอุปกรณ์หลักคือ คริสตอล (X-TAL) ที่ทำหน้าที่กำหนดช่วงเวลาในการประมวลผล (Executing time) ของหน่วยประมวลผลกลาง โดยจะส่งผลโดยตรงกับความเร็วในการประมวลผลของไมโครคอนโทรลเลอร์ นอกจากคริสตอลแล้ว เรายังสามารถใช้อุปกรณ์ที่เรียกว่า เซรามิคเรโซเนเตอร์ (Ceramic resonator) แทนได้ ซึ่งอุปกรณ์ทั้งสองจะแตกต่างกันตรงที่โหมดการทำงานของสัญญาณนาฬิกา

2.2.5 ส่วนอินเตอร์รัพท์สัญญาณ (Interrupting Unit)

ส่วนอินเตอร์รัพท์สัญญาณ คือ ส่วนที่ทำหน้าที่จัดลำดับความสำคัญในการทำงานในกรณีที่ไมโครคอนโทรลเลอร์ทำงานในลักษณะหลายๆงานพร้อมกัน (Multitasking)



รูปที่ 2.1 โครงสร้างหลักอย่างง่ายของไมโครคอนโทรลเลอร์

2.2 ไมโครคอนโทรลเลอร์ตระกูล PIC เบอร์ PIC16F627A

ไมโครคอนโทรลเลอร์ เมื่อแบ่งตามลักษณะการทำงานด้านการประมวลผล จะแบ่งออกเป็น 2 ประเภท คือ RISC (Reduced Instruction Set Computer) และ CISC (Complex Instruction Set Computer) ดังนี้

- ไมโครคอนโทรลเลอร์ประเภท RISC คือ ไมโครคอนโทรลเลอร์ที่มีโครงสร้างการทำงานที่มีจำนวนชุดคำสั่งน้อย แต่ละคำสั่งจะเป็นคำสั่งแบบง่ายๆ ความเร็วในการประมวลผลต่อ 1 คำสั่งจะมีค่าสูง เนื่องจากจะใช้สัญญาณนาฬิกาเพียง 1 ไซเคิลในการประมวลผลคำสั่งแบบง่ายๆ

- ไมโครคอนโทรลเลอร์ประเภท CISC คือ ไมโครคอนโทรลเลอร์ที่มีโครงสร้างการทำงานที่มีจำนวนชุดคำสั่งมาก แต่ละคำสั่งจะเป็นคำสั่งที่ซับซ้อน ความเร็วในการประมวลผลต่อ 1 คำสั่งจะช้ากว่าประเภท RISC เนื่องจากจะใช้สัญญาณนาฬิกามากกว่า 1 ไซเคิลในการประมวลผลคำสั่งที่ซับซ้อน

สำหรับไมโครคอนโทรลเลอร์ตระกูล PIC จะจัดอยู่ในประเภท RISC ซึ่งเป็นของบริษัทไมโครชิพ (Microchip) โดยที่ PIC ย่อมาจากคำว่า Peripheral Interface Controller ในส่วนของโครงงานนี้ จะใช้ไมโครคอนโทรลเลอร์ตระกูล PIC เบอร์ PIC16F627A แสดงอุปกรณ์ดังรูปที่ 2.2

2.2.1 รายละเอียดข้อมูลพื้นฐานของ PIC16F627A

รายละเอียดข้อมูลพื้นฐานของ PIC16F627A มีดังต่อไปนี้

- CPU เป็นแบบ RISC ที่มีคำสั่งใช้งานเพียง 35 คำสั่ง

- หน่วยความจำโปรแกรมแบบ Flash ขนาด 1 Kword (1K×14 bit) สามารถลบและเขียนข้อมูลใหม่ได้ 100,000 ครั้ง เก็บข้อมูลได้นาน 40 ปี

- หน่วยความจำข้อมูลแบบ RAM ชนิด SRAM ขนาด 224 byte

- หน่วยความจำข้อมูลแบบ EEPROM ขนาด 128 byte สามารถลบและเขียนข้อมูลใหม่ได้ 1,000,000 ครั้ง เก็บข้อมูลได้นาน 40 ปี

- มีจำนวนขาที่สามารถเชื่อมต่อกับอุปกรณ์ภายนอก (I/O) ได้สูงสุด 16 ขา

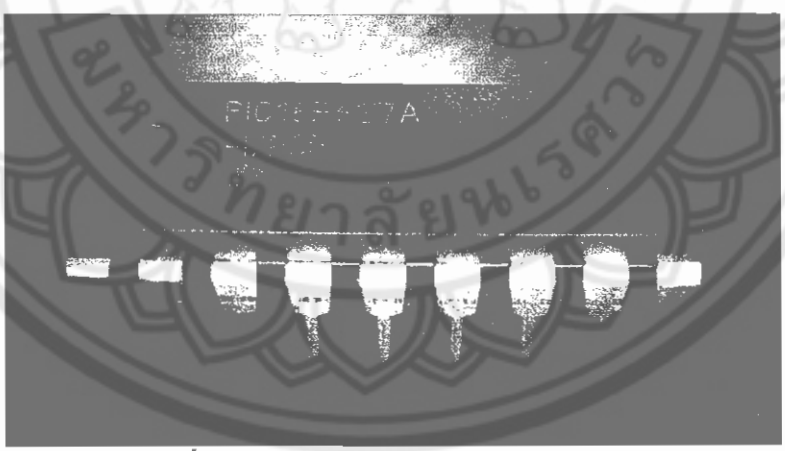
- ทำงานที่ไฟเลี้ยง (V_{DD}) ขนาด 3.0-5.5 V

- มีช่วงแรงดันในการทำงานที่กว้าง นั่นคือ 2.0-5.5 V

- ทำงานที่ความถี่ 0-20 MHz

- มีกำลังวัดด์สูงสุด 800 mW

- ราคาไม่สูง เหมาะกับงานที่มีขนาดไม่ใหญ่มาก



รูปที่ 2.2 ลักษณะภายนอกของ PIC16F627A

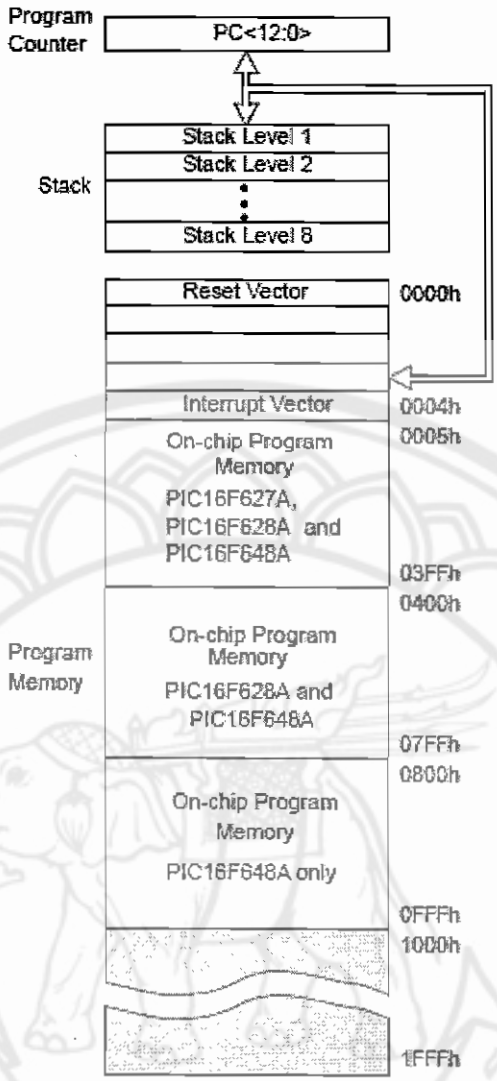
2.2.2 การจัดสรรหน่วยความจำและรีจิสเตอร์ของ PIC16F627A

- การจัดสรรหน่วยความจำโปรแกรม หน่วยความจำโปรแกรมของ PIC16F627A จะเป็นแบบ Flash ที่มีขนาด 1 Kword หรือ 1K×14 bit ซึ่งจะอยู่ในตำแหน่ง 0000h-03FFh ดังรูปที่ 2.3 เป็นหน่วยความจำที่มีขนาด 14 bit ในแต่ละแอดเดรส สำหรับแอดเดรสแรก 0000h จะถูกจัดให้มีหน้าที่

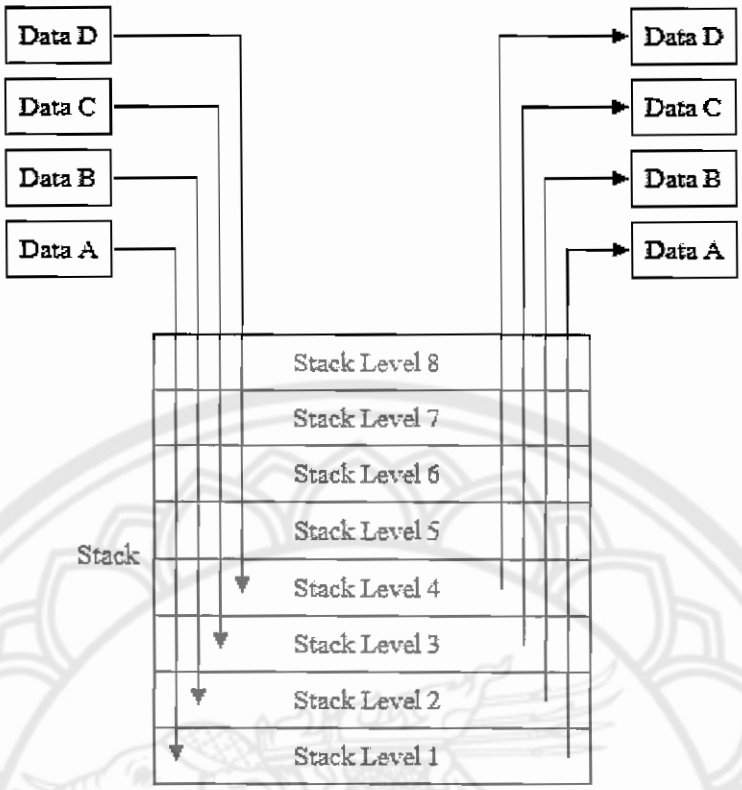
เก็บค่าแอดเดรสรีเซตเวกเตอร์ (Reset vector) และแอดเดรส 0004h จะถูกจัดให้มีหน้าที่เก็บค่าแอดเดรสอินเตอร์รัพท์เวกเตอร์

• โปรแกรมเคาน์เตอร์ (Program Counter: PC) คือ รีจิสเตอร์พิเศษที่ทำหน้าที่เป็นตัวชี้ตำแหน่งแอดเดรสของหน่วยความจำโปรแกรมที่ CPU จะต้องไปประมวลผลในลำดับถัดไป PC เป็นรีจิสเตอร์ขนาด 13 bit แบ่งเป็น 2 ส่วน ส่วนแรกคือรีจิสเตอร์ PCL มีแอดเดรสอยู่ที่ 02h ของหน่วยความจำข้อมูลแบบแรม รีจิสเตอร์นี้จะมีขนาด 8 bit เป็นข้อมูลในไบต์ต่ำ นั่นคือจะอยู่ที่ตำแหน่งบิตที่ 0-7 และส่วนที่สองคือรีจิสเตอร์ PCH มีขนาด 5 bit เป็นข้อมูลในไบต์สูง นั่นคือจะอยู่ที่ตำแหน่งบิตที่ 8-12 ซึ่งเราจะไม่สามารถเข้าถึงรีจิสเตอร์นี้ได้โดยตรง การปรับเปลี่ยนข้อมูลในรีจิสเตอร์ PCH ต้องกระทำผ่านรีจิสเตอร์ PCLATH ที่มีแอดเดรสอยู่ที่ 0Ah ของหน่วยความจำข้อมูลแบบแรม โดยปกติแล้ว ค่าของ PC จะเปลี่ยนแปลงโดยอัตโนมัติ ขึ้นอยู่กับผลการทำงานที่เกิดขึ้น นั่นคือ ค่าของ PC จะเพิ่มขึ้น 1 ทุกๆครั้งที่มีการประมวลผลคำสั่งเกิดขึ้น 1 ครั้ง ยกเว้นในกรณีที่ CPU ประมวลผลคำสั่งกระโดดไปยังโปรแกรมย่อยต่างๆหรือเกิดการอินเตอร์รัพท์ขึ้น ค่าของ PC ก็จะเปลี่ยนไปเป็นค่าที่ตำแหน่งปลายทางที่กระโดดไป

• สแต็ก (Stack) คือ หน่วยความจำพิเศษที่ทำหน้าที่เก็บข้อมูลที่ยังต้องการอยู่ของรีจิสเตอร์ PC ในขณะใดขณะหนึ่งเอาไว้ และเมื่อข้อมูลนั้นถูกนำมาเก็บไว้ใน Stack แล้ว ก็จะสามารถเปลี่ยนค่าข้อมูลในรีจิสเตอร์ PC ได้ทันที ต่อมาหลังจากที่ CPU ทำงานประมวลผลจากค่า PC ดังกล่าวเรียบร้อยแล้ว ก็จะกลับมาทำงานจากค่า PC เดิมที่เก็บเอาไว้กลับจาก Stack สำหรับ PIC16F627A จะมี Stack อยู่ 8 ระดับ การเก็บข้อมูลของ Stack จะมีลักษณะเป็นระดับ ข้อมูลที่เก็บเข้ามาก่อนจะถูกอ่านออกทีหลัง และข้อมูลที่เก็บเข้ามาทีหลังจะถูกอ่านออกก่อน โดยที่จะมีสแต็กพอยน์เตอร์ (Stack pointer) เป็นตัวชี้ตำแหน่งของสแต็ก โครงสร้างและกระบวนการทำงานของ Stack แสดงดังรูปที่ 2.4



รูปที่ 2.3 การจัดสรรหน่วยความจำโปรแกรมของ PIC16F627A



รูปที่ 2.4 โครงสร้างและกระบวนการทำงานอย่างง่ายของ Stack

- การจัดสรรหน่วยความจำข้อมูลแบบแรม หน่วยความจำข้อมูลแบบแรมของ PIC16F627A จะมีขนาด 224 byte โดยจะมีการแบ่งพื้นที่หน่วยความจำออกเป็น 4 ส่วน แต่ละส่วนจะเรียกว่า แบงก์ (Bank) ซึ่งแต่ละแบงก์จะมีขนาดสูงสุด 128 byte แสดงดังรูปที่ 2.5 แต่ละแบงก์จะมีการจัดสรรพื้นที่ดังนี้

- แบงก์ 0 มีช่วงแอดเดรสคือ 00h-7Fh
  - แอดเดรส 00h-1Fh เป็นพื้นที่ของ SFR
  - แอดเดรส 20F-7Fh เป็นพื้นที่ของ GPR ขนาด 96 byte
- แบงก์ 1 มีช่วงแอดเดรสคือ 80h-FFh
  - แอดเดรส 80h-9Fh เป็นพื้นที่ของ SFR
  - แอดเดรส A0h-EFh เป็นพื้นที่ของ GPR ขนาด 80 byte
  - **แอดเดรส F0h-FFh เป็นพื้นที่ที่เก็บข้อมูลเหมือนกับในแอดเดรส 70h-7Fh ในแบงก์ 0 เพื่อช่วยให้สามารถใช้ข้อมูลจากแอดเดรส 70h-7Fh ได้ง่ายขึ้น โดยไม่ต้องเปลี่ยนแบงก์**

- แบงก์ 2 มีช่วงแอดเดรสคือ 100h-17Fh

แอดเดรส 100h-10Fh เป็นพื้นที่ของ SFR

แอดเดรส 110h-11Fh เป็นพื้นที่ที่ไม่มีการใช้งาน

แอดเดรส 120h-14Fh เป็นพื้นที่ของ GPR ขนาด 48 byte

แอดเดรส 150h-16Fh เป็นพื้นที่ที่ไม่มีการใช้งาน

แอดเดรส 170h1-7Fh เป็นพื้นที่ที่เก็บข้อมูลเหมือนกับในแอดเดรส 70h-7Fh ในแบงก์ 0 เพื่อช่วยให้สามารถใช้ข้อมูลจากแอดเดรส 70h-7Fh ได้ง่ายขึ้น โดยไม่ต้องเปลี่ยนแบงก์

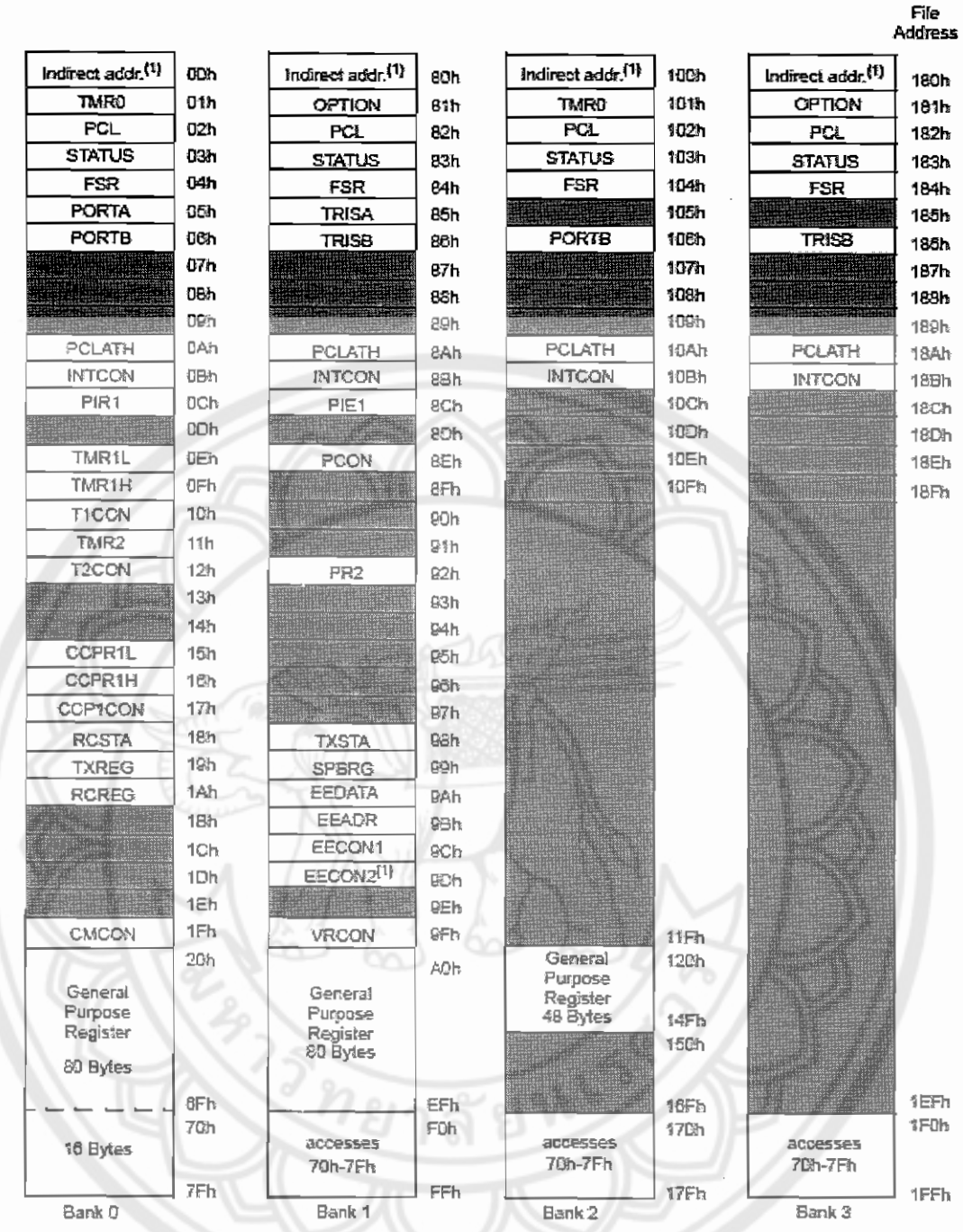
- แบงก์ 3 มีช่วงแอดเดรสคือ 180h-1FFh

แอดเดรส 180h-18Fh เป็นพื้นที่ของ SFR

แอดเดรส 190h-1EFh เป็นพื้นที่ที่ไม่มีการใช้งาน

แอดเดรส 1F0h-1FFh เป็นพื้นที่ที่เก็บข้อมูลเหมือนกับในแอดเดรส 70h-7Fh ในแบงก์ 0 เพื่อช่วยให้สามารถใช้ข้อมูลจากแอดเดรส 70h-7Fh ได้ง่ายขึ้น โดยไม่ต้องเปลี่ยนแบงก์

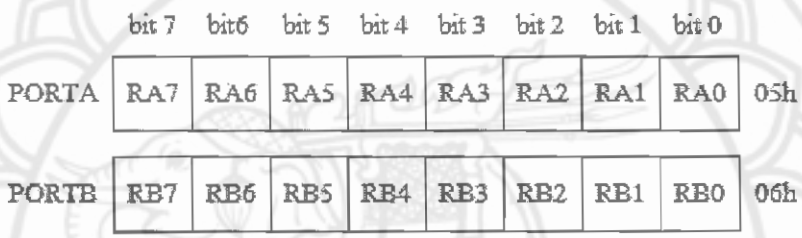
หมายเหตุ: จากรูปที่ 2.5 สำหรับพื้นที่ที่มีสีเทาทั้งหมดของหน่วยความจำข้อมูลแบบแรมจะเป็นพื้นที่ที่ไม่มีการใช้งาน



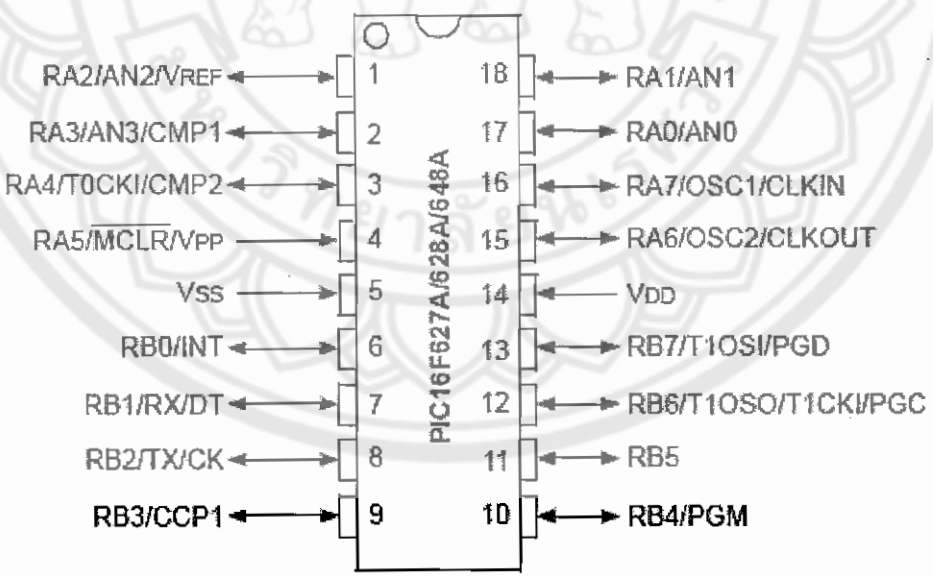
รูปที่ 2.5 การจัดสรรหน่วยความจำข้อมูลแบบแรมของ PIC16F627A

2.2.3 พอร์ตและและการทำงาน ของ PIC16F627A

ไมโครคอนโทรลเลอร์ PIC16F627A จะมีส่วนเชื่อมต่อสัญญาณทางไฟฟ้าหรือ I/O ที่เรียกว่าพอร์ต (PORT) ซึ่งจะมีให้ใช้อยู่ 2 พอร์ตคือ พอร์ต A (PORTA) และพอร์ต B (PORTB) ซึ่งต่างก็เป็นรีจิสเตอร์ขนาด 8 bit ชนิด SFR ของหน่วยความจำข้อมูลแบบแรม โดยที่ PORTA จะมีตำแหน่งอยู่ที่ 05h ส่วน PORTB จะมีตำแหน่งอยู่ที่ 06h ดังรูปที่ 2.5 PORTA เป็นรีจิสเตอร์ขนาด 8 bit โดยที่บิตแรกจะเรียกแทนด้วย RA0 บิตที่สองจะเรียกแทนด้วย RA1, RA2, ... ตามลำดับ และบิตสุดท้ายคือบิตที่แปดจะเรียกแทนด้วย RA7 ส่วน PORTB ก็จะมีชื่อเรียกแทนสำหรับแต่ละบิตในทำนองเดียวกัน ซึ่งจะมีโครงสร้างดังรูปที่ 2.6 แต่ละบิตของ PORTA และ PORTB นั้น คือบิตข้อมูลที่ใช้ในการส่งผ่านเพื่อเชื่อมต่อกับอุปกรณ์ภายนอกให้มีการทำงานเกิดขึ้น โดยที่แต่ละบิตดังกล่าวจะปรากฏอยู่ที่ขาตำแหน่งต่างๆของ PIC16F627A ดังรูปที่ 2.7



รูปที่ 2.6 โครงสร้างอย่างง่ายของรีจิสเตอร์ PORTA และ PORTB



รูปที่ 2.7 แผนภาพแสดงตำแหน่งขาของ PIC16F627A

ขาแต่ละขาที่แสดงถึงบิตข้อมูลของ PORTA และ PORTB สำหรับ PIC16F627A จะมีหน้าที่หลักดังนี้

สำหรับ PORTA

- RA0 จะอยู่ในตำแหน่งขาที่ 17 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RA1 จะอยู่ในตำแหน่งขาที่ 18 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RA2 จะอยู่ในตำแหน่งขาที่ 1 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RA3 จะอยู่ในตำแหน่งขาที่ 2 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RA4 จะอยู่ในตำแหน่งขาที่ 3 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RA5 จะอยู่ในตำแหน่งขาที่ 4 ทำหน้าที่เป็นขาอินพุตแบบดิจิตอลและเป็นขารีเซตหลัก (MCLR) แต่ในการทำงานปกติแล้ว ขา RA5 จะทำหน้าที่เป็นขารีเซตหลัก ซึ่งขานี้จะต้องต่ออยู่กับตัวต้านทานพูลอัพ (Pull-up resistor) ไว้ด้วย ความต้านทานพูลอัพ คือ ความต้านทานที่ช่วยรักษาระดับแรงดันของวงจรที่ต่ออยู่ แม้ว่าอุปกรณ์ภายนอกที่ต่ออยู่กับวงจรจะขาดการเชื่อมต่อ ซึ่งระดับแรงดันดังกล่าวจะอยู่ที่ประมาณ 5 V หรือมีค่าลอจิกเป็น “1” การรีเซตของไมโครคอนโทรลเลอร์ คือ การที่ไมโครคอนโทรลเลอร์กลับมาทำงานใหม่ที่ตำแหน่งเริ่มต้นของโปรแกรม ซึ่งจะช่วยแก้ปัญหาความผิดปกติหรือข้อผิดพลาดในการทำงานของไมโครคอนโทรลเลอร์ นั่นคือ ในการทำงานปกติของไมโครคอนโทรลเลอร์เราจะต้องป้อนแรงดันประมาณ 5 V หรือค่าลอจิก “1” ให้กับขานี้อยู่ตลอดเวลา แต่ถ้าเราต้องการรีเซตแล้ว จะต้องป้อนแรงดัน 0 V หรือค่าลอจิก “0” ให้กับขานี้
- RA6 จะอยู่ในตำแหน่งขาที่ 15 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล และเป็นขาที่เชื่อมต่อกับคริสตอล (X-TAL) ร่วมกับขา RA7 เพื่อสร้างสัญญาณนาฬิกาให้กับไมโครคอนโทรลเลอร์ แต่ในการทำงานปกติแล้ว ขา RA6 จะทำหน้าที่เป็นขาที่เชื่อมต่อกับคริสตอล
- RA7 จะอยู่ในตำแหน่งขาที่ 16 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล และเป็นขาที่เชื่อมต่อกับคริสตอล (X-TAL) ร่วมกับขา RA6 เพื่อสร้างสัญญาณนาฬิกาให้กับไมโครคอนโทรลเลอร์ แต่ในการทำงานปกติแล้ว ขา RA7 จะทำหน้าที่เป็นขาที่เชื่อมต่อกับคริสตอล

สำหรับ PORTB

- RB0 จะอยู่ในตำแหน่งขาที่ 6 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RB1 จะอยู่ในตำแหน่งขาที่ 7 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RB2 จะอยู่ในตำแหน่งขาที่ 8 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RB3 จะอยู่ในตำแหน่งขาที่ 9 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RB4 จะอยู่ในตำแหน่งขาที่ 10 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RB5 จะอยู่ในตำแหน่งขาที่ 11 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล

- RB6 จะอยู่ในตำแหน่งขาที่ 12 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล
- RB7 จะอยู่ในตำแหน่งขาที่ 13 ทำหน้าที่เป็นขาอินพุตและเอาต์พุตแบบดิจิตอล

ส่วนขาที่เหลืออีก 2 ขาคือ ไฟเลี้ยงและกราวนด์ (GND) ของPIC16F627A ดังนี้

- V_{DD} จะอยู่ในตำแหน่งขาที่ 14 ทำหน้าที่เป็นขาไฟเลี้ยงให้กับ PIC
- V_{SS} จะอยู่ในตำแหน่งขาที่ 5 ทำหน้าที่เป็นขากราวนด์ให้กับ PIC

การทำงานของ PORTA และ PORTB ในการที่จะควบคุมให้ขาใดขาหนึ่งเป็นอินพุตหรือ เอาต์พุตนั้น จะกำหนดได้ผ่านรีจิสเตอร์ขนาด 8 บิตชนิด SFR ของหน่วยความจำข้อมูลแบบแรมที่เรียกว่า TRISA และ TRISB ตามลำดับ โดยที่ TRISA จะมีตำแหน่งอยู่ที่ 85h ส่วน TRISB จะมีตำแหน่งอยู่ที่ 86h ดังรูปที่ 2.5 TRISA เป็นรีจิสเตอร์ขนาด 8 bit โดยที่บิตแรกจะเรียกแทนด้วย TRISA0 บิตที่สองจะเรียกแทนด้วย TRISA1, TRISA2, ... ตามลำดับ และบิตสุดท้ายคือบิตที่แปดจะเรียกแทนด้วย TRISA7 ส่วน TRISB ก็จะมีชื่อเรียกแทนสำหรับแต่ละบิตในทำนองเดียวกันซึ่งจะมีโครงสร้างดังรูปที่ 2.8

	bit 7	bit6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
TRISA	TRISA0	TRISA1	TRISA2	TRISA3	TRISA4	TRISA5	TRISA6	TRISA7	85h
TRISB	TRISB0	TRISB1	TRISB2	TRISB3	TRISB4	TRISB5	TRISB6	TRISB7	86h

รูปที่ 2.8 โครงสร้างอย่างง่ายของรีจิสเตอร์ TRISA และ TRISB

สำหรับ PORTA

ในการที่จะควบคุมให้ขาใดขาหนึ่งเป็นอินพุตหรือเอาต์พุตนั้น จะต้องกำหนดค่าลงใน TRISA นั่นคือ ถ้าต้องการให้ขาใดทำหน้าที่เป็นขารับสัญญาณอินพุต จะต้องกำหนดให้ TRISA ในบิตที่ตรงกันนั้นมีค่าลอจิกเป็น "1" ส่วนบิตที่เหลือกำหนดให้มีค่าลอจิกเป็น "1" หรือ "0" ก็ได้ตามความเหมาะสม หรือถ้าต้องการให้ขาใดทำหน้าที่เป็นขาส่งสัญญาณเอาต์พุต จะต้องกำหนดให้ TRISA ในบิตที่ตรงกันนั้นมีค่าลอจิกเป็น "0" ส่วนบิตที่เหลือกำหนดให้มีค่าลอจิกเป็น "1" หรือ "0" ก็ได้ตามความเหมาะสม

ตัวอย่างเช่น ถ้าเราต้องการให้ขา 18 (RA1) และขา 12 (RA3) ทำหน้าที่เป็นขารับสัญญาณ อินพุต จะทำการกำหนดค่าลงใน TRISA ดังรูปที่ 2.9

	bit 7	bit6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	05h
TRISA	(1/0)	(1/0)	(1/0)	0	1	0	1	0	85h

รูปที่ 2.9 การกำหนดค่าให้กับรีจิสเตอร์ TRISA

นั่นคือ เราจะต้องกำหนดค่าเป็น TRISA=0b00001010 สำหรับเลขฐานสอง (0b) หรือ TRISA=0x0A สำหรับเลขฐานสิบหก (0x) หรือ TRISA=10 สำหรับเลขฐานสิบ และการกำหนดค่าแบบนี้จะมีความหมายเหมือนกับว่าเราต้องการให้ขา 17 (RA0) ขา 1 (RA2) และขา 3 (RA4) ทำหน้าที่เป็นขาส่งสัญญาณเอาต์พุตเช่นเดียวกัน

สำหรับกรณีที่ขา 18 (RA1) และขา 12 (RA3) ทำหน้าที่เป็นขารับสัญญาณอินพุต เมื่อกำหนดค่าให้กับ TRISA แล้ว ขาทั้งสองนี้จะทำหน้าที่รับสัญญาณอินพุตได้โดยตรงจากสัญญาณภายนอก นั่นคือ ถ้ามีการจ่ายแรงดัน DC ประมาณ 5 V จากแหล่งจ่ายภายนอกให้กับขา 18 (RA1) และขา 12 (RA3) แล้ว เราจะสามารถอ่านค่าจาก PORTA ได้คือ RA1 และ RA3 จะมีค่าลอจิกเป็น "1" ส่วนบิตที่เหลือจะมีค่าลอจิกเป็น "0"

สำหรับกรณีที่ขา 17 (RA0) ขา 1 (RA2) และขา 3 (RA4) ทำหน้าที่เป็นขาส่งสัญญาณเอาต์พุต เมื่อกำหนดค่าให้กับ TRISA แล้ว ขาทั้งสามนี้จะทำหน้าที่ส่งสัญญาณเอาต์พุตได้โดยตรงจากการกำหนดค่าให้กับ PORTA นั่นคือ ถ้าเราต้องการส่งสัญญาณเอาต์พุตจากขาใดขาหนึ่งออกไป เราจะกำหนดค่าให้บิตนั้นมีค่าลอจิกเป็น "1" เช่น ถ้าเราต้องการส่งสัญญาณเอาต์พุตจากขา 17 (RA0) และขา 1 (RA2) ออกไปเท่านั้น เราจะกำหนดค่าให้กับ PORTA เป็น PORTA=0b00000101 สำหรับเลขฐานสอง (0b) หรือ PORTA=0x05 สำหรับเลขฐานสิบหก (0x) หรือ TRISA=5 สำหรับเลขฐานสิบ ดังรูปที่ 2.10

	bit 7	bit6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
PORTA	0	0	0	0	0	1	0	1	05h
TRISA	(1/0)	(1/0)	(1/0)	0	1	0	1	0	85h

รูปที่ 2.10 การกำหนดค่าให้กับรีจิสเตอร์ PORTA

หมายเหตุ : กรณี PORTA สำหรับขา 4 (RA5) ขา 15 (RA6) และขา 16 (RA7) จะมีหน้าที่เฉพาะของมันอยู่แล้วตามที่ได้กล่าวไปข้างต้น การใช้งานเป็นขา I/O จึงใช้ได้สูงสุดเพียง 5 ขาที่เหลือ ดังนั้นในการกำหนดค่าให้กับ TRISA เพื่อทำการเลือกขารับสัญญาณอินพุตหรือขาส่งสัญญาณเอาต์พุต จะกำหนดค่าให้กับบิตที่ตรงกันเป็นลอจิก "1" หรือ "0" ก็ได้ เราไม่ได้สนใจ นั่นคือในที่นี้เราจะกำหนดค่าให้กับบิต TRISA5, TRISA6 และ TRISA7 เป็นลอจิก "1" หรือ "0" ก็ได้

สำหรับ PORTB

ในการที่จะควบคุมให้ขาใดขาหนึ่งเป็นอินพุตหรือเอาต์พุตนั้น จะต้องกำหนดค่าลงใน TRISB ในทำนองเดียวกันกับกรณี PORTA นั่นคือ ถ้าต้องการให้ขาใดทำหน้าที่เป็นขารับสัญญาณอินพุต จะต้องกำหนดให้ TRISB ในบิตที่ตรงกันนั้นมีค่าลอจิกเป็น "1" ส่วนบิตที่เหลือกำหนดให้มีค่าลอจิกเป็น "1" หรือ "0" ก็ได้ตามความเหมาะสม หรือถ้าต้องการให้ขาใดทำหน้าที่เป็นขาส่งสัญญาณเอาต์พุต จะต้องกำหนดให้ TRISB ในบิตที่ตรงกันนั้นมีค่าลอจิกเป็น "0" ส่วนบิตที่เหลือกำหนดให้มีค่าลอจิกเป็น "1" หรือ "0" ก็ได้ตามความเหมาะสม

ตัวอย่างเช่น ถ้าเราต้องการให้ขา 9 (RB3) ขา 12 (RB6) และขา 13 (RB7) ทำหน้าที่เป็นขารับสัญญาณอินพุต จะทำการกำหนดค่าลงใน TRISB ดังรูปที่ 2.11

	bit 7	bit6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	06h
TRISB	1	1	0	0	1	0	0	0	86h

รูปที่ 2.11 การกำหนดค่าให้กับรีจิสเตอร์ TRISB

นั่นคือ เราจะต้องกำหนดค่าเป็น TRISB=0b11001000 สำหรับเลขฐานสอง (0b) หรือ TRISB=0xC8 สำหรับเลขฐานสิบหก (0x) หรือ TRISB=200 สำหรับเลขฐานสิบ และการกำหนดค่าแบบนี้จะมีความหมายเหมือนกับว่าเราต้องการให้ขา 6 (RB0) ขา 7 (RB1) ขา 8 (RB2) ขา 10 (RB4) และขา 11 (RB5) ทำหน้าที่เป็นขาส่งสัญญาณเอาต์พุตเช่นเดียวกัน

สำหรับกรณีที่ขา 9 (RB3) ขา 12 (RA6) และขา 13 (RB7) ทำหน้าที่เป็นขารับสัญญาณ**อินพุต เมื่อกำหนดค่าให้กับ TRISB แล้ว ขาทั้งสามนี้จะทำหน้าที่รับสัญญาณอินพุตได้โดยตรงจากสัญญาณ ภายนอก นั่นคือ ถ้ามีการจ่ายแรงดัน DC ประมาณ 5 V จากแหล่งจ่ายภายนอกให้กับขา 9 (RB3) ขา 12 (RB6) และขา (RB7) แล้ว เราจะสามารถอ่านค่าจาก PORTA ได้คือ RA1 และ RA3 จะมีค่าลอกจิกเป็น "1" ส่วนบิตที่เหลือจะมีค่าลอจิกเป็น "0"**

สำหรับกรณีที่ขา 6 (RB0) ขา 7 (RB1) ขา 8 (RB2) ขา 10 (RB4) และขา 11 (RB5) ทำหน้าที่เป็นขาส่งสัญญาณเอาต์พุต เมื่อกำหนดค่าให้กับ TRISB แล้ว ขาทั้งห้านี้จะทำหน้าที่ส่งสัญญาณเอาต์พุตได้โดยตรงจากการกำหนดค่าให้กับ PORTB นั่นคือ ถ้าเราต้องการส่งสัญญาณเอาต์พุตจากขาใดขาหนึ่งออกไป เราจะกำหนดค่าให้บิตนั้นมีค่าลอจิกเป็น "1" เช่น ถ้าเราต้องการส่งสัญญาณเอาต์พุตจากขา 6 (RB0) และขา 7 (RB1) ออกไปเท่านั้น เราจะกำหนดค่าให้กับ PORTB เป็น PORTB=0b00000011สำหรับเลขฐานสอง (0b) หรือ PORTA=0x03 สำหรับเลขฐานสิบหก (0x) หรือ TRISA=3 สำหรับเลขฐานสิบ ดังรูปที่ 2.12

	bit 7	bit6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
PORTB	0	0	0	0	0	0	1	1	06h
TRISB	1	1	0	0	1	0	0	0	86h

รูปที่ 2.12 การกำหนดค่าให้กับรีจิสเตอร์ PORTB

ในการกำหนดค่าให้กับรีจิสเตอร์ PORTA หรือ PORTB เพื่อทำการส่งสัญญาณเอาต์พุตออกไปเพียง 1 ขา เราสามารถทำการกำหนดค่าเพื่อเข้าถึงข้อมูลระดับบิตได้ โดยใช้รูปแบบคำสั่งคือ PORTA.F0, PORTA.F1, ..., PORTA.F7 แทนค่าของข้อมูลในบิต RA0, RA1, ..., RA7 ตามลำดับ สำหรับ PORTA และ PORTB.F0, PORTB.F1, ..., PORTB.F7 แทนค่าของข้อมูลในบิต RB0, RB1,..., RB7 ตามลำดับ สำหรับ PORTB ดังรูปที่ที่ 2.13

	bit 7	bit6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
PORTA	PORTA.F7	PORTA.F6	PORTA.F5	PORTA.F4	PORTA.F3	PORTA.F2	PORTA.F1	PORTA.F0	05h
PORTB	PORTB.F7	PORTB.F6	PORTB.F5	PORTB.F4	PORTB.F3	PORTB.F2	PORTB.F1	PORTB.F0	06h

รูปที่ 2.13 บิตต่างๆและชื่อเรียกที่สามารถเข้าถึงได้โดยตรงของ PORTA และ PORTB

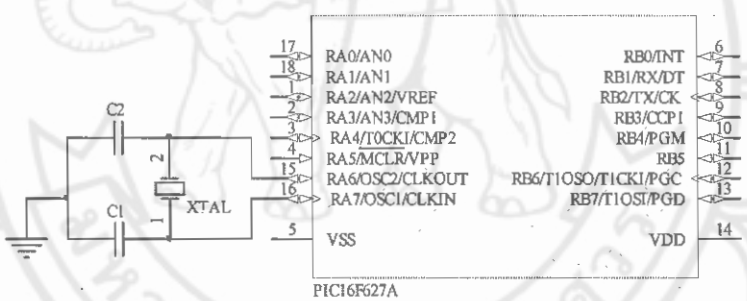
ตัวอย่างเช่น จากตัวอย่างที่แล้ว ถ้าเราต้องการส่งสัญญาณเอาต์พุตออกไปที่ขา 7 (RB1) ขาเดียวเท่านั้น เราจะกำหนดค่าให้กับ PORTB ได้โดยใช้รูปแบบคำสั่งคือ PORTB.F1=1

PIC16F627A สามารถเลือกโหมดของสัญญาณนาฬิกาได้ถึง 6 โหมดหลัก 8 โหมดย่อย ดังนี้

1. LP (Low Power Crystal)
2. XT (Crystal/Resonator)
3. HS (High Speed Crystal/Resonator)
4. RC (External Resistor/Capacitor) มี 2 โหมดย่อย
5. INTOSC (Internal Precision Oscillator) มี 2 โหมดย่อย
6. EC (External Clock In)

แต่การใช้งานในที่นี้ จะกล่าวถึงเพียง 3 โหมดแรก เนื่องจากเป็นกลุ่มของโหมดการใช้งานตามปกติ ซึ่งจะต้องมีการต่อวงจรเพื่อสร้างสัญญาณนาฬิกา โดยอุปกรณ์ในวงจรดังกล่าวสามารถเลือกใช้ได้ 2 อย่าง คือ คริสตอล (Crystal, X-TAL) และ เซรามิคเรโซเนเตอร์ (Ceramic Resonator) ดังนี้

กรณีใช้คริสตอล วงจรสร้างสัญญาณนาฬิกาจะแสดงดังรูปที่ 2.14 คริสตอลนั้นจะมีให้เลือกใช้หลายความถี่ ซึ่งค่าความถี่ของคริสตอลนี้จะมีผลต่อค่าของตัวเก็บประจุ C_1 และ C_2 ที่เราจะต้องเลือกใช้ในวงจร สามารถใช้งานได้ 3 โหมด ดังตารางที่ 2.1



รูปที่ 2.14 วงจรสร้างสัญญาณนาฬิกาโดยใช้คริสตอล

ตารางที่ 2.1 ขอบเขตของค่า C1 และ C2 ที่สามารถใช้ได้ กรณีใช้คริสตอล

Mode	Frequency	OSC1(C1)	OSC2(C2)
LP	32 kHz	15-30 pF	15-30 pF
	200 kHz	0-15 pF	0-15 pF
XT	100 kHz	68-150 pF	150-200 pF
	2 MHz	15-30 pF	15-30 pF
	4 MHz	15-30 pF	15-30 pF
HS	8 MHz	15-30 pF	15-30 pF
	10 MHz	15-30 pF	15-30 pF
	20 MHz	15-30 pF	15-30 pF

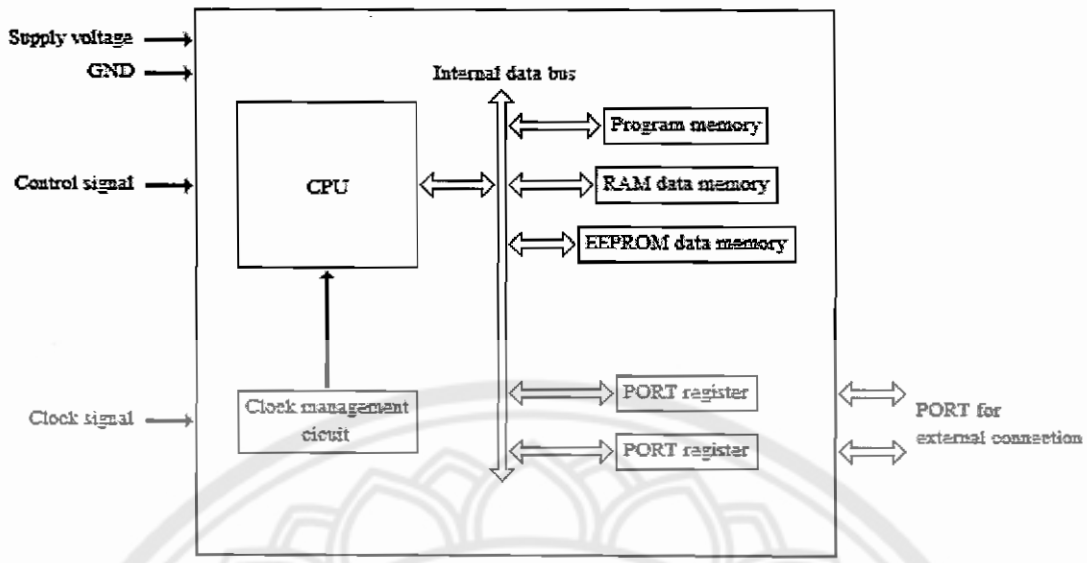
กรณีใช้เซรามิคเรโซเนเตอร์ เซรามิคเรโซเนเตอร์จะมีลักษณะคล้ายกับคริสตอล และวงจรสร้างสัญญาณนาฬิกาก็จะมีรูปแบบเหมือนดังรูปที่ 2.14 เซรามิคเรโซเนเตอร์นั้นก็จะมีให้เลือกใช้หลายความถี่เช่นเดียวกัน ซึ่งค่าความถี่นี้จะมีผลต่อค่าของตัวเก็บประจุ C_1 และ C_2 ที่เราจะต้องเลือกใช้ในวงจร สามารถใช้งานได้ 2 โหมด ดังตารางที่ 2.2

ตารางที่ 2.2 ขอบเขตของค่า C_1 และ C_2 ที่สามารถใช้ได้ กรณีใช้เซรามิคเรโซเนเตอร์

Mode	Frequency	OSC1(C1)	OSC2(C2)
XT	455 kHz	22-100 pF	22-100 pF
	2.0 MHz	15-68 pF	15-68 pF
	4.0 MHz	15-68 pF	15-68 pF
HS	8.0 MHz	10-68 pF	10-68 pF
	16.0 MHz	10-22 pF	10-22 pF

2.2.4 การทำงานโดยภาพรวมของ PIC16F627A

การทำงานของไมโครคอนโทรลเลอร์จะต้องประกอบด้วยส่วนต่างๆที่สำคัญแสดงดังรูปที่ 2.15 เมื่อ CPU ทำการติดต่อกับหน่วยความจำโปรแกรมเพื่ออ่านโปรแกรมการทำงานที่บรรจุอยู่จะต้องมีการอ้างตำแหน่งของหน่วยความจำโปรแกรมผ่านสายสัญญาณที่เรียกว่า บัสแอดเดรส (Address Bus) ขนาด 13 bit แล้วทำการอ่านข้อมูลโปรแกรมการทำงานในแอดเดรสนั้นๆผ่านสายสัญญาณที่เรียกว่าบัสข้อมูลโปรแกรม (Program Bus) ขนาด 14 bit จากนั้น CPU จึงทำการประมวลผล ซึ่งในระหว่างการประมวลผลนี้จะมีหน่วยความจำข้อมูลแรมเป็นที่พักของข้อมูล หรืออาจมองได้ว่าหน่วยความจำข้อมูลแรมเปรียบเสมือนกระดาษทดในการคำนวณ และในทำนองเดียวกันก็จะต้องมีการอ้างตำแหน่งของหน่วยความจำข้อมูลแบบแรมผ่านบัสแอดเดรสขนาด 8 bit แล้วทำการอ่านข้อมูลในแอดเดรสนั้นๆผ่านสายสัญญาณที่เรียกว่าบัสข้อมูล (Data Bus) ขนาด 8 bit เช่นเดียวกัน ดังรูปที่ 2.16

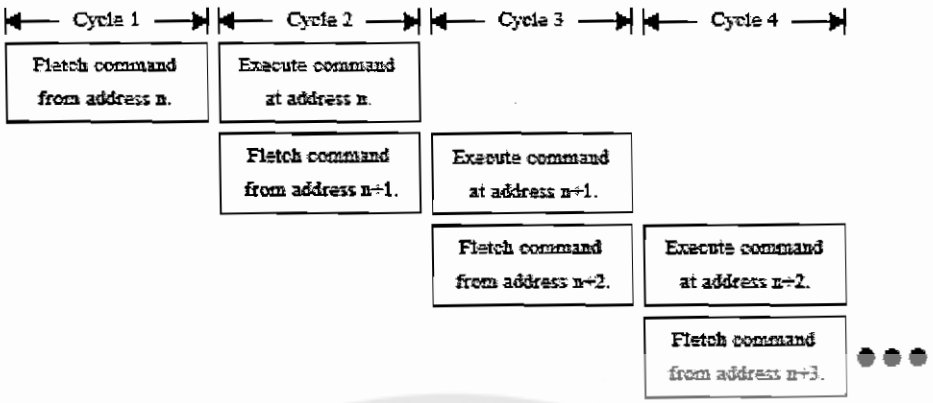


รูปที่ 2.15 โครงสร้างและส่วนประกอบหลักของไมโครคอนโทรลเลอร์

รูปที่ 2.16 โครงสร้างการติดต่อข้อมูลระหว่าง CPU กับ หน่วยความจำโปรแกรม และหน่วยความจำข้อมูลแบบแรม

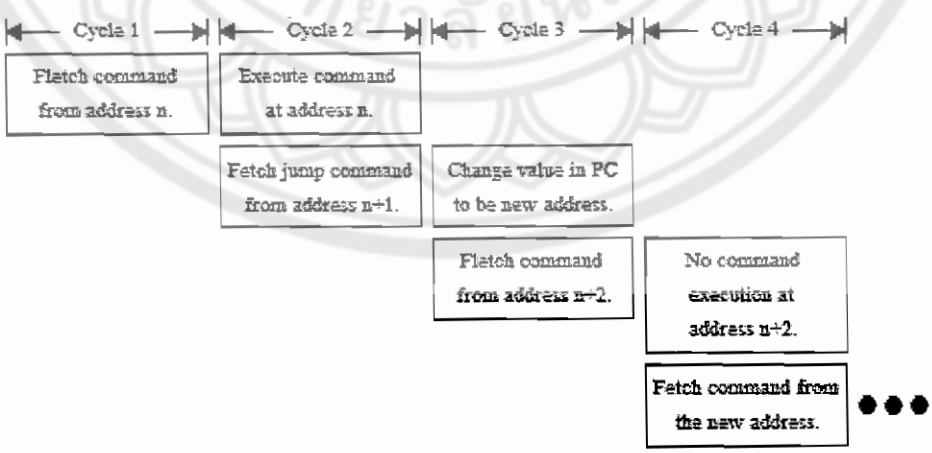
การประมวลผลของ PIC16F627A จะใช้กระบวนการทำงานที่เรียกว่าไปป์ไลน์ (Pipeline) ทำให้สามารถเฟตช์ (Fetch) คำสั่งถัดไปในขณะที่กำลังเอ็กซิคิวต์ (Execute) คำสั่งปัจจุบันอยู่ จึงทำให้ CPU สามารถประมวลผลได้ 1 คำสั่งภายในสัญญาณนาฬิกา 1 ลูกหรือ 1 ไซเคิล (Cycle) ซึ่งกระบวนการเฟตช์ (Fetch) คือ กระบวนการเรียกคำสั่งออกจากหน่วยความจำโปรแกรมแล้วทำการแปลคำสั่งนั้นให้เป็นรหัสเลขฐานสิบหกเพื่อให้ CPU เข้าใจ ส่วนกระบวนการเอ็กซิคิวต์ (Execute) คือ กระบวนการประมวลผลหรือกระทำคำสั่ง เพื่อให้เกิดผลลัพธ์ตามคำสั่งนั้นๆ การประมวลผลดังกล่าวอธิบายได้ดังนี้

เมื่อเริ่มต้นกระทำคำสั่งที่ 1 CPU จะเฟตช์คำสั่งนั้นจากหน่วยความจำโปรแกรมที่แอดเดรส n ในไซเคิลแรก จากนั้นจึงทำการเอ็กซิคิวต์คำสั่งที่แอดเดรส n ในไซเคิลถัดมา และในขณะเดียวกัน CPU ก็จะเริ่มต้นเฟตช์คำสั่งจากแอดเดรส n+1 เมื่อเอ็กซิคิวต์คำสั่งที่แอดเดรส n เสร็จแล้ว CPU ก็จะเอ็กซิคิวต์คำสั่งที่แอดเดรส n+1 ทันทีในไซเคิลถัดมา และในขณะเดียวกัน CPU ก็จะเริ่มต้นเฟตช์คำสั่งจากแอดเดรส n+2 ต่อไป กระบวนการทำงานจะเป็นแบบนี้ไปเรื่อยๆ ดังรูปที่ 2.17



รูปที่ 2.17 แผนภาพกระบวนการทำงานแบบไปป์ไลน์ของ PIC16F627A

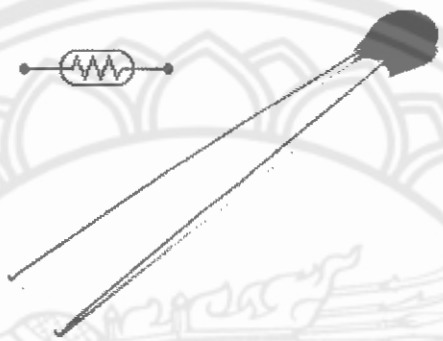
สำหรับการกระทำคำสั่งที่เป็นคำสั่งการกระโดด ก็จะมีขั้นตอนที่เพิ่มเข้ามา เมื่อทำการเอ็กซิคิวต์คำสั่งที่แอดเดรส n CPU ก็จะทำการเฟชต์คำสั่งจากแอดเดรส n+1 ในไซเคิลเดียวกันปรากฏว่าคำสั่งที่แอดเดรส n+1 เป็นคำสั่งการกระโดด ดังนั้นในไซเคิลถัดไปจึงยังไม่เกิดการเอ็กซิคิวต์ในทันที แต่จะเกิดการเปลี่ยนค่าของPC (Program Counter) ซึ่งเป็นรีจิสเตอร์ที่มีหน้าที่ระบุแอดเดรสถัดไปของคำสั่งที่ CPU จะไปประมวลผล ทำให้เกิดการกระโดดไปยังแอดเดรสใหม่และในขณะเดียวกัน CPU ก็จะทำการเฟตช์คำสั่งจากแอดเดรส n+2 ต่อไปตามขั้นตอนปกติ แต่เนื่องจากเกิดการกระโดดไปยังแอดเดรสใหม่ของ PC แล้ว ดังนั้นในไซเคิลถัดไปจึงยังไม่เกิดการเอ็กซิคิวต์คำสั่งที่แอดเดรส n+2 แต่จะเกิดการเฟตช์คำสั่งจากแอดเดรสใหม่ที่กระโดดเข้ามา จากนั้นก็จะเข้าสู่กระบวนการทำงานตามปกติต่อไปดังรูปที่ 2.18 จะเห็นได้ว่าการกระทำคำสั่งที่เป็นคำสั่งการกระโดดจะใช้สัญญาณนาฬิกาถึง 3 ไซเคิลต่อ 1 คำสั่ง แต่ในการกระทำคำสั่งตามปกติจะใช้สัญญาณนาฬิกาเพียง 1 ไซเคิลต่อ 1 คำสั่ง



รูปที่ 2.18 แผนภาพกระบวนการทำงานแบบไปป์ไลน์ของ PIC16F627A ในกรณีที่ CPU กระทำคำสั่งการกระโดด

2.3 เทอร์มิสเตอร์และการทำงาน

เทอร์มิสเตอร์ (Thermistor) คือ เซนเซอร์อุณหภูมิ (Temperature sensor) ชนิดหนึ่งที่ทำหน้าที่ตรวจจับอุณหภูมิโดยอาศัยความสัมพันธ์ระหว่างค่าความต้านทานไฟฟ้าของตัวมันเองกับอุณหภูมิแวดล้อม เทอร์มิสเตอร์เป็นอุปกรณ์สารกึ่งตัวนำที่ทำมาจากโลหะออกไซด์ เช่น แมงกานีส นิกเกิล โคบอลต์ ทองแดง และยูเรเนียม เป็นต้น สัญลักษณ์และตัวอย่างของเทอร์มิสเตอร์แสดงดังรูปที่ 2.19



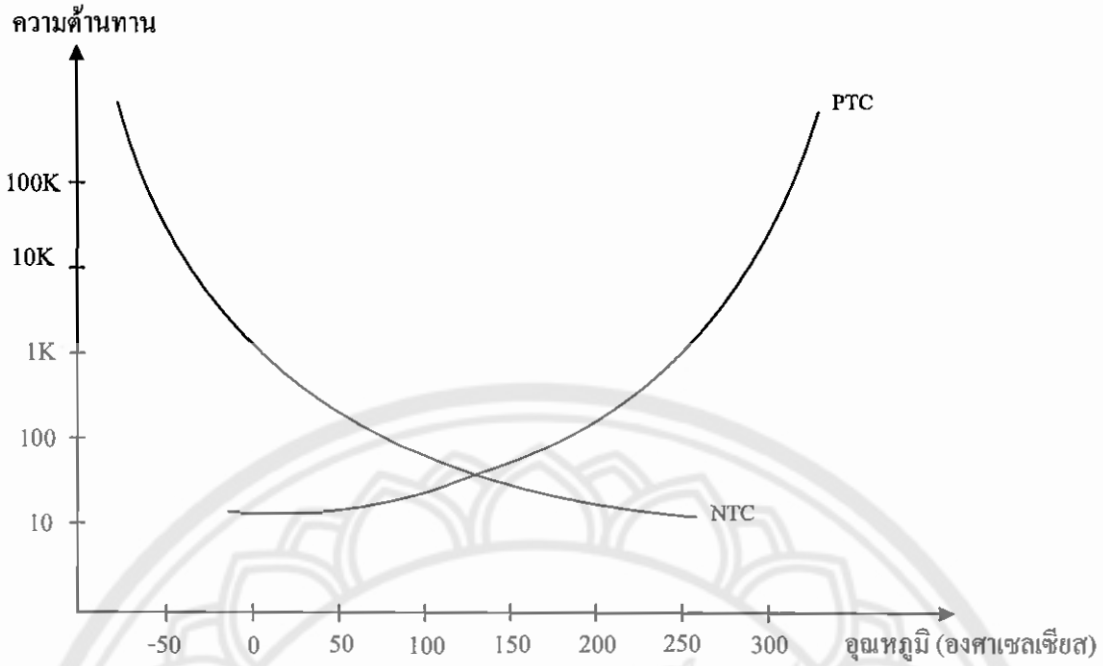
รูปที่ 2.19 สัญลักษณ์และตัวอย่างของเทอร์มิสเตอร์

2.3.1 ชนิดของเทอร์มิสเตอร์

โดยทั่วไปเทอร์มิสเตอร์จะแบ่งเป็น 2 ชนิด คือ แบบ NTC และ PCT ดังนี้

- เทอร์มิสเตอร์แบบ NTC (Negative Temperature Coefficient) คือ เทอร์มิสเตอร์ที่มีค่าความต้านทานแปรผกผันกับอุณหภูมิ นั่นคือ ถ้าอุณหภูมิสูงความต้านทานจะมีค่าน้อย และถ้าอุณหภูมิต่ำความต้านทานจะมีค่ามาก ซึ่งจากตัวอย่างของเทอร์มิสเตอร์ดังรูปที่ 2.19 จะเป็นแบบ NTC

- เทอร์มิสเตอร์แบบ PTC (Positive Temperature Coefficient) คือ เทอร์มิสเตอร์ที่มีค่าความต้านทานแปรผันตรงกับอุณหภูมิ นั่นคือ ถ้าอุณหภูมิสูงความต้านทานจะมีค่ามาก และถ้าอุณหภูมิต่ำความต้านทานจะมีค่าน้อย ตัวอย่างกราฟความสัมพันธ์ระหว่างความต้านทานกับอุณหภูมิของเทอร์มิสเตอร์แบบ NTC และแบบ PTC แสดงดังรูปที่ 2.20



รูปที่ 2.20 ตัวอย่างกราฟความสัมพันธ์ระหว่างความต้านทานกับอุณหภูมิของเทอร์มิสเตอร์แบบ NTC และ PTC

2.3.2 สมการความสัมพันธ์ระหว่างค่าความต้านทานกับอุณหภูมิของเทอร์มิสเตอร์แบบ NTC

สำหรับเทอร์มิสเตอร์แบบ NTC ความสัมพันธ์ระหว่างค่าความต้านทานกับอุณหภูมิ จะอยู่ในรูปของฟังก์ชันเอ็กโพเนนเชียล (Exponential function) ตามสมการดังนี้

$$R_t = R_0 \exp \beta \left(\frac{1}{T} - \frac{1}{T_0} \right) \qquad (2.1)$$

เมื่อ R_t คือ ค่าความต้านทานของเทอร์มิสเตอร์ที่อุณหภูมิ T เคลวิน (Kelvin, K)

R_0 คือ ค่าความต้านทานของเทอร์มิสเตอร์ที่อุณหภูมิอ้างอิง T_0 K

T คือ ค่าอุณหภูมิสัมบูรณ์ในหน่วย K

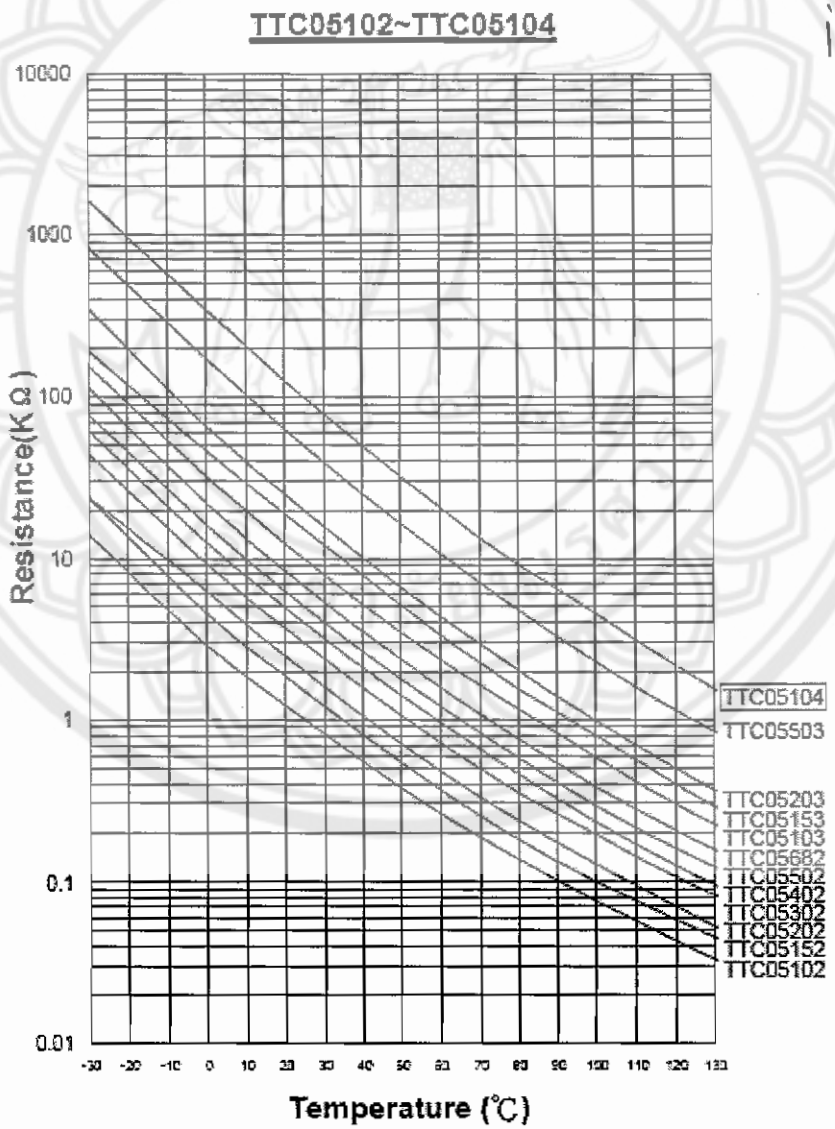
β คือ ค่าคงที่ของเทอร์มิสเตอร์ มีหน่วยเป็น K ซึ่งจะมีค่าขึ้นอยู่กับแต่ละรุ่นหรือแต่ละเบอร์ของเทอร์มิสเตอร์นั้นๆ

2.3.3 เทอร์มิสเตอร์ที่ใช้ในโครงงาน

สำหรับโครงงานนี้จะใช้เทอร์มิสเตอร์แบบ NTC รุ่น TTC-104 ซึ่งเป็นเทอร์มิสเตอร์ที่แสดงดังรูปที่ 2.19 โดยมีคุณสมบัติที่สำคัญจาก Data sheet ดังนี้

- มีเส้นผ่านศูนย์กลางขนาด 5 mm
- มีช่วงอุณหภูมิในการทำงานอยู่ที่ -30˚C ถึง 125˚C
- มีค่าความต้านทาน (R_0) 100 kΩ ที่อุณหภูมิ (T_0) 25˚C โดยมีค่าความคลาดเคลื่อนอยู่ที่ 5-10 %
- มีค่า β เท่ากับ 4,400 K

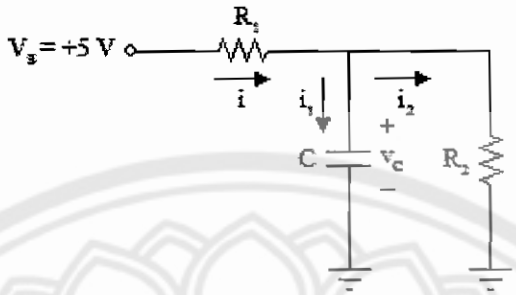
ส่วนกราฟความสัมพันธ์ระหว่างค่าความต้านทานกับอุณหภูมิ (R-T characteristic curve) จะแสดงดังรูปที่ 2.21



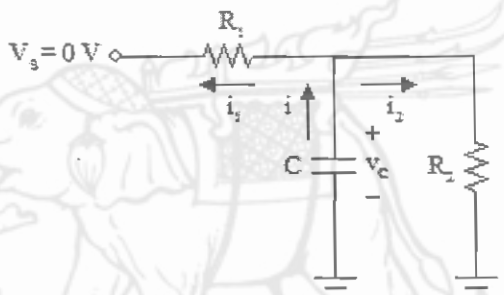
รูปที่ 2.21 R-T characteristic curve ของเทอร์มิสเตอร์แบบ NTC รุ่น TTC-104

2.4 การเก็บประจุและการคายประจุของตัวเก็บประจุ

สำหรับโครงงานนี้จะศึกษาการเก็บประจุและการคายประจุของตัวเก็บประจุในวงจรที่ประกอบด้วยตัวเก็บประจุ 1 ตัว และตัวต้านทาน 2 ตัว ดังรูปที่ 2.22 และ รูปที่ 2.23 ตามลำดับ ดังนี้



รูปที่ 2.22 วงจรในการเก็บประจุของตัวเก็บประจุ



รูปที่ 2.23 วงจรในการคายประจุของตัวเก็บประจุ

2.4.1 วิเคราะห์สมการการเก็บประจุและการคายประจุของตัวเก็บประจุในวงจร

- วิเคราะห์สมการการเก็บประจุของตัวเก็บประจุในวงจร พิจารณารูปที่ 2.22 ในการคำนวณ

จาก KCL จะได้

$$i = i_1 + i_2 \tag{2.2}$$

จาก KVL จะได้

$$i = \frac{V_S - v_C}{R_1} \tag{2.3}$$

$$i_1 = C\frac{dv_C}{dt} \tag{2.4}$$

และ

$$i_2 = \frac{v_C}{R_2} \tag{2.5}$$

แทนสมการที่ 2.3, 2.4 และ 2.5 ลงในสมการที่ 2.2 จะได้ว่า

$$\frac{d}{dt}v_C(t)+\left(\frac{R_1+R_2}{R_1R_2C}\right)v_C(t)-\frac{V_S}{R_1C}=0 \tag{2.6}$$

- วิเคราะห์สมการการคายประจุของตัวเก็บประจุในวงจร พิจารณารูปที่ 2.23 ในการคำนวณ จาก KCL จะได้

$$i=i_1+i_2 \tag{2.7}$$

จาก KVL จะได้

$$i=-C\frac{dv_C}{dt} \tag{2.8}$$

$$i_1=\frac{v_C}{R_1} \tag{2.9}$$

และ

$$i_2=\frac{v_C}{R_2} \tag{2.10}$$

แทนสมการที่ 2.8, 2.9 และ 2.10 ลงในสมการที่ 2.7 จะได้ว่า

$$\frac{d}{dt}v_C(t)+\left(\frac{R_1+R_2}{R_1R_2C}\right)v_C(t)=0 \tag{2.11}$$

จะเห็นได้ว่า สมการที่ 2.6 และ 2.11 จะอยู่ในรูปแบบที่คล้ายกัน นั่นคือจะอยู่ในรูปแบบของสมการเชิงอนุพันธ์อันดับที่หนึ่ง ดังนี้

$$\frac{d}{dt}x(t)+ax(t)+b=0 \tag{2.12}$$

เมื่อ x(t) คือ ฟังก์ชันของตัวแปร t

a และ b คือ ค่าคงที่ใดๆ

เราสามารถหาคำตอบของสมการที่ 2.12 ได้โดยใช้หลักการแยกตัวแปรและอินทิเกรต ดังนี้

แยกตัวแปรสมการที่ 2.12 จะได้

$$\frac{dx}{ax+b} = -dt \tag{2.13}$$

อินทิเกรตสมการที่ 2.13 จะได้

$$\int \frac{dx}{ax+b} = \int(-dt) + C$$

$$\int \frac{dx}{ax+b} = -t + C \tag{2.14}$$

เมื่อ C คือ ค่าคงที่สุดท้ายของการอินทิเกรต

จากสูตรของการอินทิเกรต

$$\int \frac{1}{u} du = \ln|u| + C_1 \tag{2.15}$$

เมื่อ C_1 คือ ค่าคงที่ของการอินทิเกรต

กำหนดให้

$$u = ax + b$$

เมื่อทำการหาอนุพันธ์ของ u เทียบกับตัวแปร x จะได้

$$\frac{du}{dx} = a$$

นั่นคือ

$$dx = \frac{du}{a}$$

ดังนั้น จะได้

$$\int \frac{dx}{ax+b} = \int \frac{1}{u} \frac{du}{a}$$

$$= \frac{1}{a} \int \frac{1}{u} du$$

$$= \frac{1}{a}\left(\ln|u| + C_1\right)$$

$$= \frac{1}{a}\ln|u| + C_2$$

$$\int \frac{dx}{ax+b} = \frac{1}{a}\ln(ax+b) + C_2 \qquad (2.16)$$

โดยที่

$$C_2 = \frac{C_1}{a}$$

จากสมการที่ 2.14 และ 2.16 จะได้ว่า

$$\frac{1}{a}\ln(ax+b) = -t + C$$

$$\ln(ax+b) = -at + aC$$

จากสูตรของลอการิทึมฐาน e จะได้ว่า

$$ax+b = e^{-at+aC}$$

จัดรูปสมการใหม่จะได้คำตอบคือ

$$x(t) = \frac{1}{a}\left(Ae^{-at} - b\right) \qquad (2.17)$$

โดยที่

$$A = e^{aC}$$

พิจารณาที่สภาวะเริ่มต้น แทน $t = 0$ ในสมการที่ 2.17 จะได้

$$x(0) = \frac{1}{a}(A - b)$$

ดังนั้น จะได้ว่า

$$A = ax(0) + b$$

แทน A กลับลงไปในสมการที่ 2.17 จะได้คำตอบคือ

$$x(t) = \left(x(0) + \frac{b}{a}\right)e^{-at} - \frac{b}{a} \qquad (2.18)$$

พิจารณาที่สภาวะอนันต์ จะได้ค่าสุดท้าย (Final value) ของ x(t) จากนิยามที่ว่า

$$x(\infty) = \lim_{t\to\infty} x(t)$$

ซึ่งพิจารณาจากสมการที่ 2.18 จะได้ว่า

$$x(\infty)=-\frac{b}{a} \tag{2.19}$$

แทนสมการที่ 2.19 ลงในสมการที่ 2.18 จะได้คำตอบคือ

$$x(t)=x(\infty)+\left(x(0)-x(\infty)\right)e^{-at} \tag{2.20}$$

ดังนั้น ในกรณีการวิเคราะห์หาสมการการเก็บประจุ เมื่อพิจารณาเปรียบเทียบสมการที่ 2.6 กับสมการที่ 2.12 จะได้ว่า

$$a=\frac{R_1+R_2}{R_1R_2C}$$

และ

$$b=-\frac{V_S}{R_1C}$$

แทนค่า a, b และ $x = v_C$ ลงในสมการที่ 2.18 จะได้สมการการเก็บประจุของตัวเก็บประจุในวงจรคือ

$$v_C(t)=\frac{R_2V_S}{R_1+R_2}+\left(v_C(0)-\frac{R_2V_S}{R_1+R_2}\right)e^{\frac{t(R_1+R_2)}{R_1R_2C}} \tag{2.21}$$

เมื่อ $v_C(0)$ คือ ค่าแรงดันเริ่มต้นที่ตกคร่อมตัวเก็บประจุ

และเมื่อพิจารณาสมการที่ 2.20 อีกครั้งจะได้ว่า

$$v_C(t)=v_C(\infty)+\left(v_C(0)-v_C(\infty)\right)e^{-\frac{t}{\tau}} \tag{2.22}$$

เมื่อ τ คือ ค่าคงที่เวลา (Time constant)

และเมื่อเปรียบเทียบสมการที่ 2.21 กับสมการที่ 2.22 จะได้ว่า

$$v_C(\infty)=\frac{R_2V_S}{R_1+R_2} \tag{2.23}$$

และ

$$\tau = \frac{R_1 R_2 C}{R_1 + R_2} \tag{2.24}$$

ส่วนในกรณีการวิเคราะห์หาสมการการคายประจุ เมื่อพิจารณาเปรียบเทียบสมการที่ 2.11 กับสมการที่ 2.12 จะได้ว่า

$$a = \frac{R_1 + R_2}{R_1 R_2 C}$$

และ

$$b = 0$$

แทนค่า a, b และ $x = v_C$ ลงในสมการที่ 2.18 จะได้สมการการคายประจุของตัวเก็บประจุในวงจรคือ

$$v_C(t) = v_C(0) e^{\frac{t(R_1+R_2)}{R_1 R_2 C}} \tag{2.25}$$

และเมื่อพิจารณาสมการที่ 2.20 อีกครั้งจะได้ว่า

$$v_C(t) = v_C(0) e^{-\frac{t}{\tau}} \tag{2.26}$$

และเมื่อเปรียบเทียบสมการที่ 2.25 กับสมการที่ 2.26 จะได้ว่า

$$v_C(\infty) = 0 \tag{2.27}$$

และ

$$\tau = \frac{R_1 R_2 C}{R_1 + R_2} \tag{2.28}$$

เช่นเดียวกัน

2.4.2 วิเคราะห์กราฟการเก็บประจุและการคายประจุของตัวเก็บประจุในวงจร

เราจะทำการวิเคราะห์กราฟการเก็บประจุและการคายประจุของตัวเก็บประจุในวงจรดังรูปที่ 2.22 และ 2.23 ตามลำดับ โดยกำหนดให้ค่าต่างๆมีค่าดังนี้

$$R_1 = 500\ \Omega \tag{2.29}$$

$$R_2 = 100\ k\Omega \tag{2.30}$$

และ

$$C = 0.1\ \mu F \tag{2.31}$$

สมการการเก็บประจุ จะหาได้จากสมการที่ 2.21 โดยการแทนค่า R_1, R_2 และ C ลงไป และกำหนดให้มีการจ่ายแรงดัน (V_S) 5 V ให้กับวงจร โดยที่สภาวะเริ่มต้น แรงดันที่ตกคร่อมตัวเก็บประจุ ($v_C(0)$) มีค่าเท่ากับ 0 V นั่นคือ

$$V_S = 5\ V \tag{2.32}$$

และ

$$v_C(0) = 0\ V \tag{2.33}$$

จะได้

$$v_C(t) = \frac{10^5 \times 5}{500 + 10^5} + \left(0 - \frac{10^5 \times 5}{500 + 10^5} \right) e^{\frac{t\left(500+10^5\right)}{500\times10^5\times10^{-7}}}$$

$$v_C(t) = 4.9751 - 4.9751e^{-20,100t} \tag{2.34}$$

สมการการคายประจุ จะหาได้จากสมการที่ 2.25 โดยการแทนค่า R_1, R_2 และ C ลงไป และกำหนดให้ที่สภาวะเริ่มต้น เป็นสภาวะหลังจากที่มีการเก็บประจุแล้วระยะหนึ่งจากสมการที่ 2.34 ดังนั้นแรงดันที่ตกคร่อมตัวเก็บประจุ ($v_C(0)$) จึงมีค่าเท่ากับหรือประมาณ 4.9751 V ซึ่งเป็นผลมาจากค่า $v(\infty)$ ในสมการที่ 2.22 นั่นคือ

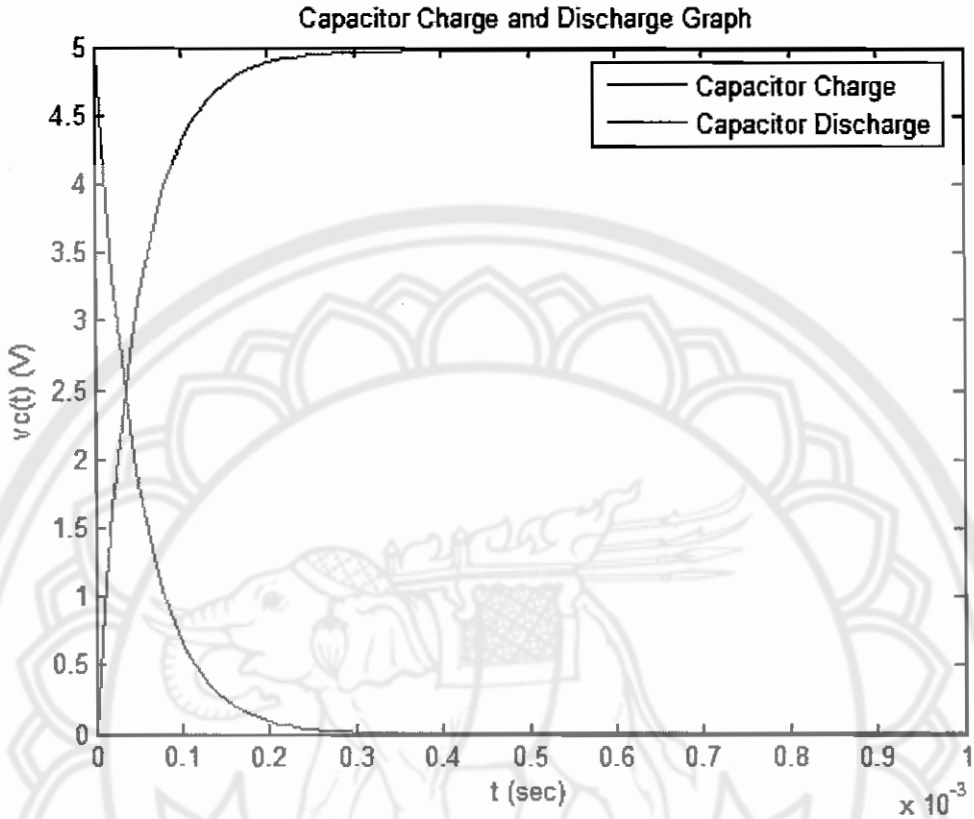
$$v_C(0) = 4.9751\ V \tag{2.35}$$

จะได้

$$v_C(t) = 4.9751e^{\frac{t\left(500+10^5\right)}{500\times10^5\times10^{-7}}}$$

$$v_C(t) = 4.9751e^{-20,100t} \tag{2.36}$$

นำสมการที่ 2.34 และ 2.36 มาเขียนกราฟเพื่อเปรียบเทียบกัน โดยใช้โปรแกรม MATLAB จะได้ดังรูปที่ 2.24



รูปที่ 2.24 กราฟการเก็บประจุและการคายประจุของตัวเก็บประจุในวงจร

จากกราฟ จะเห็นได้ว่ากราฟทั้งสองจะมีลักษณะสมมาตรกันตามแนวแกน y เนื่องจากเป็นสภาวะที่ต่อเนื่องกันและเป็นวงจรเดียวกันตามสมการที่ 2.34 และ 2.36 สำหรับการเก็บประจุของตัวเก็บประจุ จะมีการเก็บประจุอย่างรวดเร็วในช่วงเวลาแรกๆ นั่นคือในช่วงเวลา 0-0.2 ms หลังจากนั้นการเก็บประจุก็จะเริ่มเข้าสู่สภาวะคงที่ที่มีค่าแรงดันสุดท้ายที่ตกคร่อมตัวเก็บประจุ ($v(\infty)$) เท่ากับ 4.9751 V ตามสมการที่ 2.34 และสำหรับการคายประจุของตัวเก็บประจุ จะมีการคายประจุอย่างรวดเร็วในช่วงเวลาแรกๆ นั่นคือในช่วงเวลา 0-0.2 ms หลังจากนั้นการคายประจุก็จะเริ่มเข้าสู่สภาวะคงที่ที่มีค่าแรงดันสุดท้ายที่ตกคร่อมตัวเก็บประจุ ($v(\infty)$) เท่ากับ 0 V ตามสมการที่ 2.36

2.4.3 วิเคราะห์ความสัมพันธ์ระหว่างเวลาที่ใช้ในการคายประจุของตัวเก็บประจุกับค่าความต้านทาน R_2 ในวงจร

พิจารณาวงจรในการคายประจุของตัวเก็บประจุดังรูปที่ 2.23 ซึ่งในที่นี้จะกำหนดให้ R_1 และ C มีค่าคงที่ ส่วน R_2 จะแปรค่าได้ เนื่องจากในที่นี้เราจะกำหนดให้ R_2 เปรียบเสมือนกับเทอร์มิสเตอร์ที่สามารถเปลี่ยนค่าความต้านทานได้ตามอุณหภูมิที่เปลี่ยนไป เราจะวิเคราะห์ความสัมพันธ์ระหว่างเวลาที่ใช้ในการเก็บประจุ (t) กับ กับค่าความต้านทาน R_2 ในวงจรได้ดังนี้

พิจารณาสมการการคายประจุสมการที่ 2.25 เนื่องจากเราจะทำการหาระยะเวลาที่ตัวเก็บประจุคายประจุออกไปได้เกือบหมดหรือใกล้เคียง ดังนั้น จะกำหนดให้ $v_C(t)$ มีค่าเข้าใกล้ศูนย์แต่ไม่เท่ากับศูนย์ กำหนดให้

$$v_C(t) = 0.01 \tag{2.37}$$

แทนค่าในสมการที่ 2.25 จะได้

$$0.01 = v_C(0)e^{\frac{t(R_1+R_2)}{R_1R_2C}}$$

Take ลอการิทึมฐาน e ทั้งสองข้างของสมการ จะได้

$$\ln 0.01 = v_C(0)\ln e^{\frac{t(R_1+R_2)}{R_1R_2C}}$$

จากสูตรของลอการิทึมฐาน e จะได้ว่า

$$\ln 0.01 = -tv_C(0)\left(\frac{R_1+R_2}{R_1R_2C}\right)$$

ดังนั้น จะได้เวลาที่ใช้ในการเก็บประจุของตัวเก็บประจุคือ

$$t = -\frac{\ln 0.01}{v_C(0)}\left(\frac{R_1R_2C}{R_1+R_2}\right) \tag{2.38}$$

กำหนดให้

$$k = -\frac{\ln 0.01}{v_C(0)}R_1C$$

ซึ่งจะเห็นได้ว่า k จะมีค่าเป็นบวกเสมอ เนื่องจาก ln 0.01 จะมีค่าเป็นลบ นั่นคือ ถ้า $0<x<1$ แล้ว ln x จะ มีค่าเป็นลบเสมอ ส่วน $v_C(0)$, R_1 และ C จะมีค่าเป็นบวกเสมอ

แทน k ลงในสมการที่ 2.38 จะได้

$$t = k\frac{R_2}{R_1 + R_2} \tag{2.39}$$

เนื่องจากเราต้องการวิเคราะห์หาอัตราการเปลี่ยนแปลงของ t เทียบกับ R_2 นั่นคือ จะกำหนดให้ t เป็นฟังก์ชันของตัวแปร R_2 ดังนั้นสมการที่ 2.39 อาจเขียนได้ใหม่เป็น

$$t(R_2) = k\frac{R_2}{R_1 + R_2} \tag{2.40}$$

ดังนั้น เราสามารถหาอัตราการเปลี่ยนแปลงของ t เทียบกับ R_2 ได้จากอนุพันธ์ของ t เทียบกับ R_2 ดังนี้

$$\frac{\partial t}{\partial R_2} = k\frac{\partial}{\partial R_2}\frac{R_2}{R_1 + R_2}$$

จากสูตรของการหาอนุพันธ์ จะได้

$$\frac{\partial t}{\partial R_2} = k\frac{(R_1 + R_2)(1) - R_2(1)}{(R_1 + R_2)^2}$$

$$\frac{\partial t}{\partial R_2} = k\frac{R_1}{R_1^2 + 2R_1R_2 + R_2^2} \tag{2.41}$$

จากสมการที่ 2.41 จะเห็นว่า ไม่ว่า R_2 จะมีค่าเป็นบวกใดๆก็ตาม อัตราการเปลี่ยนแปลงของ t เทียบกับ R_2 หรือ $\partial t/\partial R_2$ ก็จะมีค่าเป็นบวกเสมอ นั่นคือจะได้ว่าฟังก์ชัน $t(R_2)$ เป็นฟังก์ชันเพิ่ม สำหรับทุกๆค่าบวกของ R_2 สรุปคือ จากสมการที่ 2.38 ที่แสดงถึงเวลาที่ใช้ในการคายประจุ ถ้ากำหนดให้ $v_C(0)$, R_1 และ C มีค่าคงที่แล้ว จะได้ว่าเวลาที่ใช้ในการคายประจุ (t) จะมีค่าแปรผันตรงกับค่าความต้านทาน R_2 นั่นคือ ถ้า R_2 มีค่ามากขึ้นแล้ว t ที่ใช้ก็จะมีค่ามากขึ้น และถ้า R_2 มีค่าน้อยลงแล้ว t ที่ใช้ก็จะมีค่าน้อยลงเช่นเดียวกัน