ระบบสั่งงานหน้าจอโปรเจ็กเตอร์ด้วยปลายนิ้วเพื่อควบคุมหน้าจอวินโดว์แทนเมาส์

THE PROJECTOR TOUCH SCREEN CONTROL IN WINDOWS

โดย
นาย นรินทร์ อึ้งพานิช
นาย วิเชษฐ์ อารีราษฎร์
นาย อาคม แซ่ตั้น

ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมอิเล็กทรอนิกส์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2551

ระบบสั่งงานหน้าจอโปรเจ็กเตอร์ด้วยปลายนิ้วเพื่อควบคุมหน้าจอวินโดว์แทนเมาส์

THE PROJECTOR TOUCH SCREEN CONTROL IN WINDOWS

โดย

นาย นรินทร์ อึ้งพานิช รหัส 49015146

นาย วิเชษฐ์ อารีราษฎร์ รหัส 49015162

นาย อาคม แซ่ตั้น รหัส 49015179

อาจารย์ที่ปรึกษา

อาจารย์ ชินภัทร นันทจิวากรชัย

ปริญญานิพนธ์สำหรับปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2551

ปริญญานิพนธ์ ปีการศึกษา 2551
ภาควิชา อิเล็กทรอนิกส์
คณะ วิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
เรื่อง ระบบสั่งงานหน้าจอโปรเจ็กเตอร์ด้วยปลายนิ้วเพื่อควบคุมหน้าจอวินโดว์แทนเมาส์
(THE PROJECTOR TOUCH SCREEN CONTROL IN WINDOWS)

ผู้จัดทำ

1. นาย นรินทร์ อึ้งพานิช รหัส 49015146
2. นาย วิเชษฐ์ อารีราษฎร์ รหัส 49015162
3. นาย อาคม แซ่ตั้น รหัส 49015179

....

(ชินภัทร นันทจิวากรชัย)
อาจารย์ที่ปรึกษา

THE PROJECTOR TOUCH SCREEN CONTROL IN WINDOWS

Mr. Narin Ungpanich ID.49015146

Mr. Wichate Areraj ID.49015162

Mr. Arkom Saeton ID.49015179

Chinnapat Nantajiwakornchai Advisor

Educational Year 2008

Abstract

This thesis presents a system for accurately estimating a position of tapings on two dimension board. The position is estimated by using different time between each receivers that received tapping signal .The tapping time are counted by using a field program gate array (FPGA). The results will be transfer to MCS-51 to calculate the position x, y

ระบบสั่งงานหน้าจอโปรเจ็กเตอร์ด้วยปลายนิ้วเพื่อควบคุมหน้าจอวินโดว์แทนเมาส์

นาย นรินทร์ อึ้งพานิช รหัส 49015146

นาย วิเชษฐ์ อารีราษฎร์ รหัส 49015162

นาย อาคม แซ่ตั้น รหัส 49015179

อ.ชินภัทร นันทจิวากรชัย อาจารย์ที่ปรึกษา

ปีการศึกษา 2551

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้นำเสนอการหาตำแหน่งของการเคาะบนแผ่นบอร์ดในระนาบพิกัด 2 มิติ โดยอาศัยผลต่างของเวลาจากตำแหน่งที่เคาะเดินทางมายังตัวรับที่ตำแหน่งต่าง ๆ ซึ่งเราทราบตำแหน่งพิกัดของตัวรับที่แน่นอนและอาศัยหลักการเดินทางของคลื่นเสียงบนแผ่นบอร์ด ซึ่งผลต่างของเวลานี้จะใช้ FPGA เป็นตัววัดและผลต่างของเวลาที่ได้นี้จะถูกส่งให้กับ MCS-51 เพื่อคำนวณหาพิกัด X, Y เพื่อพัฒนาใช้ในการควบคุมหน้าจอ WINDOWS ต่อไป

สารบัญ

สารบัญภาพ

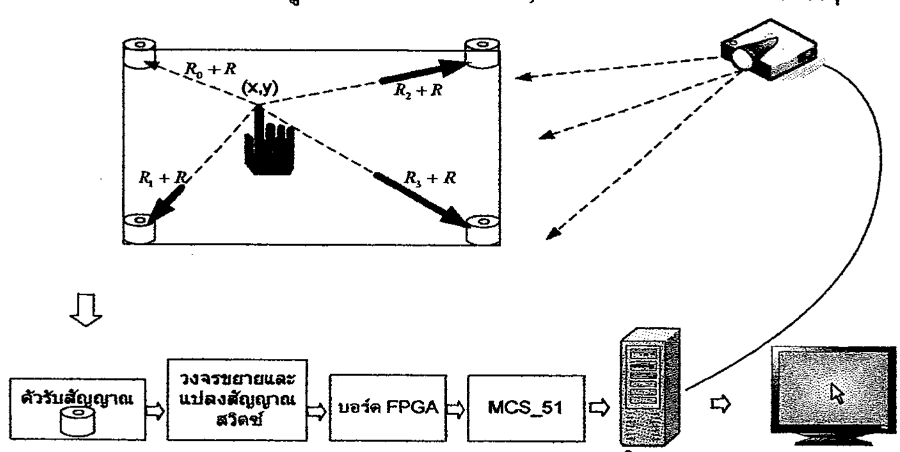
บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

ปัจจุบันเทคโนโลยีการใช้สื่อการเรียนการสอนหรือการนำเสนอผลงานในสถาบันการศึกษาหรือบริษัทต่างๆ ได้พัฒนาไปอย่างรวดเร็ว โดยสื่อที่ใช้งานง่ายและสะดวกกำลังเป็นที่นิยม คือ การใช้เครื่องโปรเจคเตอร์ฉายลงบนบอร์ดเพื่อนำเสนอข้อมูลจากเครื่องคอมพิวเตอร์ โดยข้อมูลที่ปรากฏลงบนบอร์ดนั้น จะมีหน้าตาเหมือนกับหน้าจอคอมพิวเตอร์ทุกประการ แต่มีขนาดใหญ่กว่า ดังนั้นเมื่อผู้นำเสนอยืนที่บริเวณหน้าบอร์ดก็สามารถนำเสนอผลงานได้ แต่หากต้องการเสนอข้อมูลถัดไปหรือเปลี่ยนแปลงข้อมูลก็ต้องกลับไปเปลี่ยนแปลงที่เครื่องคอมพิวเตอร์ เพื่อจัดการกับปัญหาดังกล่าวเราจึงได้เพิ่มความสะดวกสร้างโปรเจคชิ้นนี้ขึ้นมา โดยให้เราสามารถควบคุมหน้าจอวินโดว์ของเครื่องคอมพิวเตอร์ได้ โดยการติดต่อกับบอร์ดที่นำเสนออยู่ด้วยการใช้ปลายนิ้วเคาะลงไปที่ตำแหน่งต่างๆที่ต้องการลงบนบอร์ดที่ตรงกับตำแหน่งที่อ้างอิง โดยคอมพิวเตอร์เพื่อสั่งงานหน้าจอวินโดว์แทนการใช้เมาส์ นอกจากเพิ่มความสามารถในการติดต่อกับคอมพิวเตอร์โดย การเคาะแล้ว ยังสามารถประยุกต์ใช้กับงานในด้านต่างๆที่เกี่ยวข้องกับการเคลื่อนที่ของคลื่นจากการสั่นสะเทือนได้อีกมากมาย เช่น การตรวจจับพิกัดจากการเดินของผู้บุกรุกหรือ การหาตำแหน่งการยิงปืนใหญ่ในสนามรบ

1.2 วัตถุประสงค์

- เพื่อนำไปใช้ในการหาตำแหน่งพิกัดจากการสั่นสะเทือนโดยการเคาะในระนาบ 2 มิติ
- เพื่อศึกษาและเรียนรู้การทำงานของ FPGA, MCS-51 และโปรแกรมต่างๆ



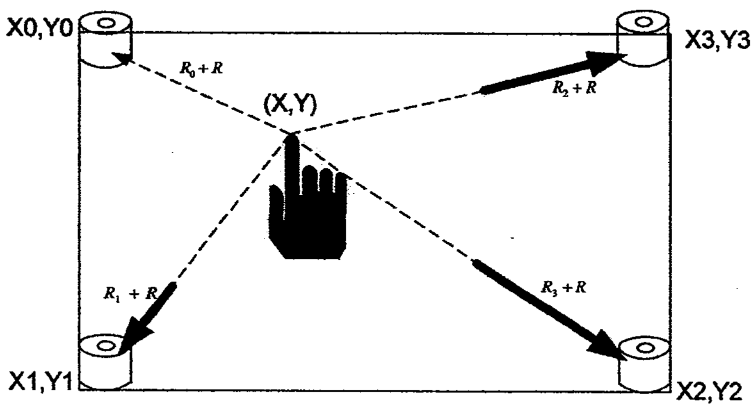
รูปที่ 1 แนวคิดในการออกแบบ

1.3 ส่วนประกอบของโครงงาน

ในโครงงานนี้จะประกอบไปด้วย 3 ส่วนหลักๆ คือ

1.3.1 ชุดรับสัญญาณจากการเคาะ

ซึ่งประกอบไปด้วยเปียโซอิเล็กทริกจำนวน 4 ตัว

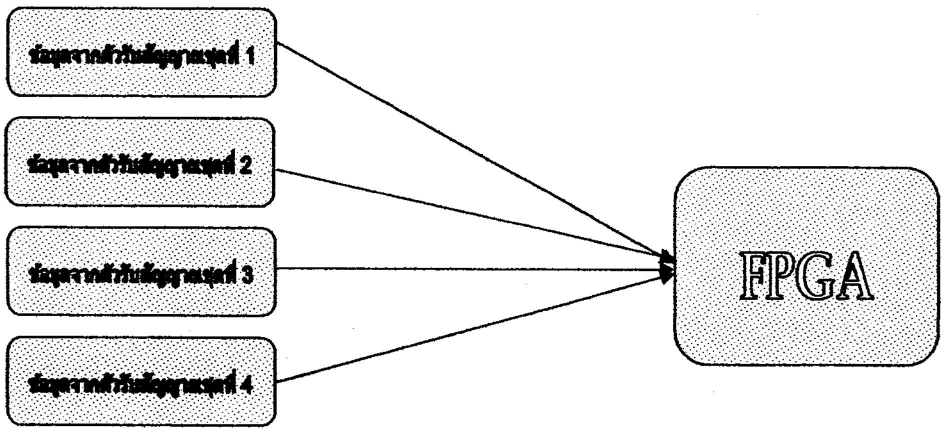


รูปที่ 1.1 แสดงการวางตำแหน่งของตัวรับ

ตัวรับสัญญาณจากการเคาะ

วงจรส่วนนี้ทำหน้าที่รับสัญญาณจากการเคาะที่ส่งออกมา โดยตัวรับจะแบ่งออกเป็น 4 จุด โดยวางตัวเป็นรูปสี่เหลี่ยมที่ตำแหน่งทราบค่าพิกัดที่แน่นอน ตัวรับแต่ละตัวจะได้รับสัญญาณที่เวลาแตกต่างกันขึ้นอยู่กับระยะทางเราจึงเอาหลักการนี้ไปเปรียบเทียบกับสัญญาณนาฬิกาที่ชุดนับสัญญาณต่อไป โดยสัญญาณที่ได้จากตัวรับซึ่งเป็นเปียโซอิเล็กทริกมีขนาดเล็กดังนั้นจึงมีวงจรขยายสัญญาณให้แรงขึ้นแล้วแปลงสัญญาณให้เป็นสัญญาณสวิตช์แล้วจึงส่งไปยังภาค FPGA

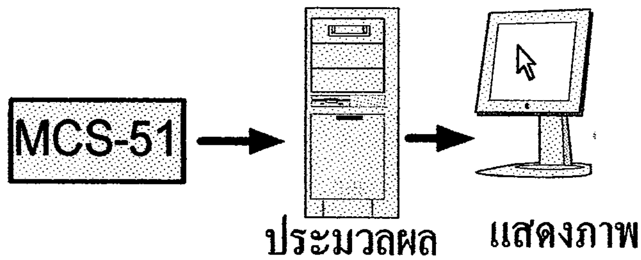
1.3.2 บอร์ดFPGA



รูปที่ 1.2 ภาพแสดงการทำงานของส่วน FPGA

FPGA สามารถรองรับวงจรดิจิตอลที่มีความสลับซับซ้อนได้เป็นอย่างดี นอกจากนี้ในด้านการออกแบบพัฒนาและทดสอบก็ทำได้ง่าย โดยได้นำบอร์ด FPGA เข้ามาเพื่อทำหน้าที่เป็นวงจร Counter นำผลต่างของเวลาที่ภาครับสัญญาณจากการเคาะมาเปรียบเทียบกับสัญญาณนาฬิกาบนบอร์ดFPGA เพื่อทำการนับ จะได้ผลข้อมูลที่เป็นจำนวน CLOCK ของตำแหน่งทั้ง 4 ตัว หากตำแหน่งที่เคาะอยู่ใกล้ตัวรับตัวใดระยะเวลาที่คลื่นใช้เดินทางจะมีค่าน้อยมีผลทำให้จำนวน CLOCK ในชุดนั้นนับได้น้อยสัมพันธ์กัน จากนั้นข้อมูลของแต่ละชุดจะส่งไปประมวลผลโดยใช้ไมโครคอนโทรลเลอร์ต่อไป

1.3.3 ภาคคำนวณ ประมวลผลและส่งเข้าสู่เครื่องคอมพิวเตอร์



รูปที่ 1.3 ภาพแสดงการทำงานของส่วนประมวลและแสดงผล

ภาคนี้ทำงานโดยการรับค่าจำนวนCounterที่นับได้จากบอร์ดFPGA เพื่อเข้ามาคำนวณพิกัด X, Y โดยประกอบด้วย MCS-51 ซึ่งเป็นไมโครคอนโทรลเลอร์ที่เราได้ทำการโปรแกรมป้อนสูตรทางคณิตศาสตร์เอาไว้แล้ว นอกจากนี้แล้ว MCS-51 ช่วยในการติดต่อกับคอมพิวเตอร์ให้ง่ายมากขึ้น โดยนำค่าพิกัดที่คำนวณได้ไปควบคุมการทำงานของเมาส์

1.4 ขอบเขตของโครงงาน

1. ศึกษาการใช้งานโปรแกรม Max Plus II เพื่อจำลองการทำงานของ FPGA
2. ศึกษาการใช้งานโปรแกรม Visual Basic 6.0
3. ศึกษาการใช้งานโปรแกรม Keil
4. ศึกษาการทำงานของวงจร MCS-51
5. ศึกษาถึงการออกแบบวงจรภาครับสัญญาณจากการเคาะโดยใช้โปรแกรม Proteus จำลองการทำงาน
6. ศึกษาถึงระยะทางที่ใช้เคาะที่มีผลต่อเฟสของสัญญาณ
7. ศึกษาการทำงานในส่วนของ FPGA ในภาคต่างๆ

บทที่ 2
ทฤษฎีที่ใช้ในการออกแบบ

2.1 ปรากฏการณ์ของคลื่น

2.1.1 คลื่น เกิดจากแหล่งกำเนิดคลื่นถ่ายทอดพลังงานออกไป โดยอนุภาคของตัวกลางไม่ได้เคลื่อนที่ไปด้วย เพียงแต่อนุภาคของตัวกลางมีการสั่นตามแนวสมดุล

2.1.2 ชนิดของคลื่น แบ่งได้หลายชนิดขึ้นอยู่กับว่าจะใช้เกณฑ์อะไรในการแบ่ง

1) แบ่งตามลักษณะของการสั่นของอนุภาค ได้เป็น 2 ประเภท

1.1) คลื่นตามยาว เช่น คลื่นเสียง, คลื่นอัดขยายในสปริงตามยาว

1.2) คลื่นตามขวาง เช่น คลื่นในเส้นเชือก, คลื่นน้ำ, คลื่นแม่เหล็กไฟฟ้า ฯลฯ

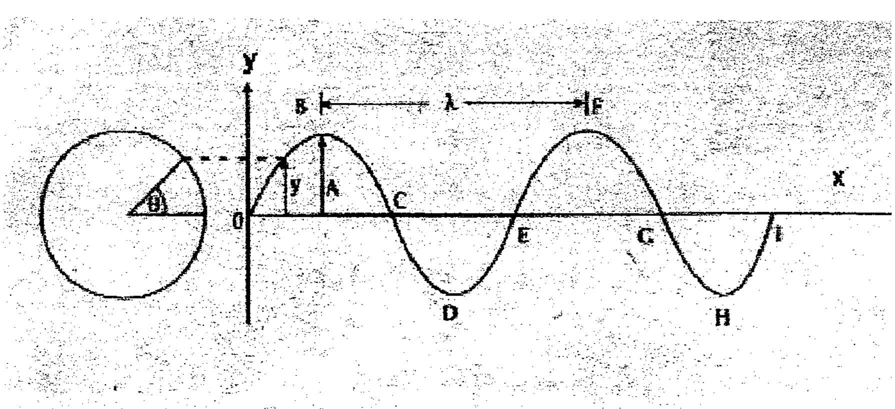
2) แบ่งตามลักษณะอาศัยตัวกลางหรือไม่ ได้ 2 ประเภท

2.1) คลื่นที่อาศัยตัวกลางในการเคลื่อนที่ หรือ เรียกว่า คลื่นกล เช่น คลื่นเสียง คลื่นน้ำ คลื่นในเส้นเชือก คลื่นของการสั่นของลวดสปริง ฯลฯ

2.2) คลื่นที่ไม่อาศัยตัวกลางในการเคลื่อนที่ หรือเรียกว่า คลื่นแม่เหล็กไฟฟ้า เช่น คลื่นแสง, คลื่นวิทยุ, ไมโครเวฟ ฯลฯ (คลื่นแม่เหล็กไฟฟ้ามีความเร็วแสง คือ $3*10^8$ m/s)

3) แบ่งตามลักษณะการเกิดคลื่น 1) คลื่นดล เกิดคลื่น 2-3 ลูก 2) คลื่นเป็นช่วง คลื่นดลที่เกิดขึ้นเป็นช่วงๆ 3) คลื่นต่อเนื่อง เกิดขึ้นตลอดอย่างต่อเนื่องอย่างสม่ำเสมอ

2.1.3 ส่วนประกอบของคลื่น พิจารณาจากรูปที่



รูปที่ 2.1 ลักษณะของคลื่น

สันคลื่น (ยอดคลื่น) คือ ตำแหน่งที่สูงสุดของคลื่น คือ จุด B และ F
ท้องคลื่น คือ ตำแหน่งที่อยู่ต่ำที่สุด คือ จุด D และ H
การขจัด คือ ระยะจากแนวสมดุลถึงจุดบนคลื่น คือระยะขจัด y ใดๆ
แอมพลิจูด (ช่วงกว้าง) คือ การขจัดที่มากที่สุด (A)
ความยาวคลื่น คือ ระยะระหว่าง จุดสองจุดที่มีเฟสเดียวกันบนลูกคลื่นที่อยู่ถัดกัน หรือระยะระหว่างสันคลื่นกับสันคลื่นที่อยู่ติดกัน (λ)
ความถี่ คือ จำนวนลูกคลื่นที่ผ่านจุดใดๆ ใน 1 หน่วยเวลา
คาบ คือ เวลาที่คลื่นหนึ่งลูกเคลื่อนที่ผ่านจุดๆ หนึ่ง (T)
เฟส คือ ตำแหน่งของจุดบนคลื่น หรือค่าของมุมที่บอกตำแหน่งของการขจัด

-เฟสตรงกัน จุดใดๆ บนคลื่นที่มีจุดต่างกัน $2n\pi$ เรเดียน เช่น 2π, 4π, 6π,.........

-เฟสตรงข้าม จุดใดๆ บนคลื่นที่มีเฟสต่างกัน $(2n+1)\pi$ เรเดียน เช่น π, 3π, 5π

-เฟสต่างกัน คือ จุดใดๆ บนคลื่นที่เฟสต่างกันใดๆ ที่ไม่ใช่เฟสตรงกันข้ามหรือตรงข้ามกัน

2.1.4 สูตรการคำนวณหาความเร็วของคลื่น

$$v = \frac{s}{t} \quad \text{หรือ} \quad v = f\lambda \qquad \text{-------------- (2.1)}$$

$$f = \frac{1}{t} \qquad \text{-------------- (2.2)}$$

เมื่อ v = ความเร็วของคลื่น (m/s)
s = ระยะทาง (m)
t = เวลาของการเคลื่อนที่ (s)
f = ความถี่ (Hz)
T = คาบเวลา (s)
λ = ความยาวคลื่น (m)

2.1.5 สมการทั่วไปของคลื่นการขจัด

$$y = A \sin \theta = A \sin \omega t \qquad \text{---------- (2.3)}$$

เมื่อ $\theta = \omega t = \dfrac{2\pi t}{T} = \dfrac{2\pi x}{\lambda}$

$$\omega = 2\pi f = \frac{2\pi}{T} \qquad \text{---------- (2.4)}$$

ความต่างเฟสระหว่างจุด 2 จุด $\Delta\theta = \dfrac{2\pi\Delta t}{T} = \dfrac{2\pi\Delta x}{\lambda}$

เมื่อ y = การขจัดใดๆ (m)

θ = มุมเฟส (เรเดียนหรือองศา)

A = แอมพลิจูด (m)

ω = ความเร็วเชิงมุม (rad / s)

t = เวลาใดๆ ที่คลื่นเคลื่อนที่ (s)

T = คาบเวลา (s / cycle)

x = ระยะทางใดๆ (m)

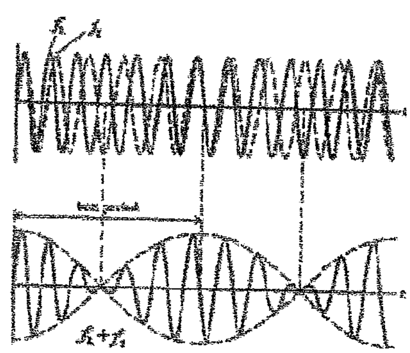
λ = ความยาวคลื่น (m)

2.1.6 คุณสมบัติโดยทั่วไปของคลื่นเสียง จะแสดงได้ 4 แบบ คือ

1) การแทรกสอดของเสียง (Interference)

การแทรกสอดของเสียงเกิดจากการรวมกันของคลื่น 2 คลื่นขึ้นไป ขณะเมื่อพบกันในตัวกลางเดียวกัน ซึ่งทำให้เกิดผลได้หลายลักษณะคือ

-*การบีตส์* (Beats) ของคลื่นเสียงเป็นปรากฏการณ์ที่เกิดจากการรวมคลื่นที่มีความถี่ต่างกัน หรือต่างเฟสกัน เคลื่อนที่ไปในตัวกลางเดียวกันแล้วรวมเป็นคลื่นใหม่ ซึ่งทำให้แอมพลิจูดเปลี่ยนไปดังแสดงในรูปที่2.2

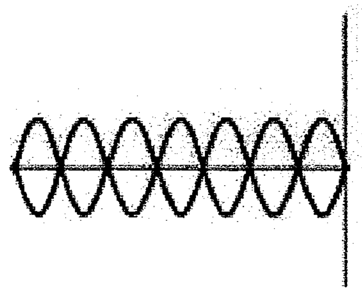


รูปที่ 2.2 แสดงการบีตของคลื่นเสียง 2 คลื่น

จากรูปที่ 2.2 เป็นการแสดงบีตส์ของคลื่น 2 คลื่น ซึ่งคลื่นที่มีแอมพลิจูดเปลี่ยนไปซึ่งประโยชน์ของการบีตส์ของคลื่นนั้น จะนำไปใช้ในการเปรียบเทียบความถี่ของคลื่นให้แสดงผลออกมาในลักษณะของแอมพลิจูดที่แตกต่างกันนี้

-*การเกิดคลื่นนิ่ง* (Standing Wave) เกิดจากการแทรกสอดของคลื่นชุดที่มีแอมพลิจูดเท่ากันและความถี่เท่ากัน แต่มีทิศทางตรงกันข้ามหรือเฟสตรงกันข้าม ซึ่งจะทำให้เกิดคลื่นนิ่งดังรูป เสียงจะมีความเข้มข้นสูงสุดที่ตำแหน่งต้นทาง. และเบาที่สุดที่ตำแหน่งปลายทาง. ซึ่งระยะระหว่างขั้วทั้ง 2 เท่ากับ

$$\mathbf{d = n\lambda/2;\ (n=1, 2, 3\ldots)} \qquad \text{------------ (2.5)}$$



รูปที่ 2.3 แสดงลักษณะการเกิดคลื่นนิ่ง

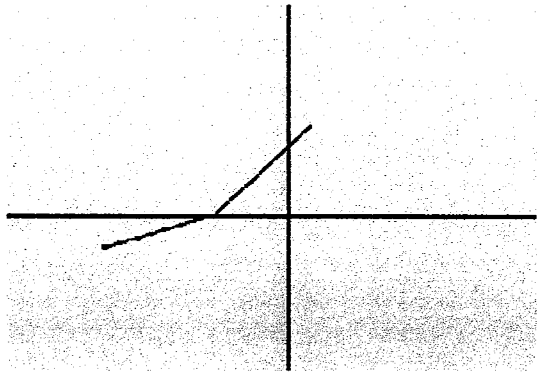
2) การเลี้ยวเบนของคลื่นเสียง (Diffraction)

คือ คลื่นเสียงจะเลี้ยวเบนอ้อมสิ่งกีดขวางที่มีลักษณะเป็นมุมหรือช่องแคบ ซึ่งปรากฏการณ์เช่นนี้พบในชีวิตประจำวันอยู่ตลอดเวลาเช่น ในกรณีที่เราได้ยินแตรรถที่อยู่คนละมุมถนนของตึก หรือการได้ยินเสียงลอดผ่านช่องเล็กๆ จากอีกห้องหนึ่ง ดังแสดงในรูป 2.4

รูปที่ 2.4 แสดงลักษณะของคลื่นเสียงที่ผ่านช่องแคบๆจะเสมือนเป็นแหล่งกำเนิดตัวใหม่

3) การหักเหของคลื่น (Refraction)

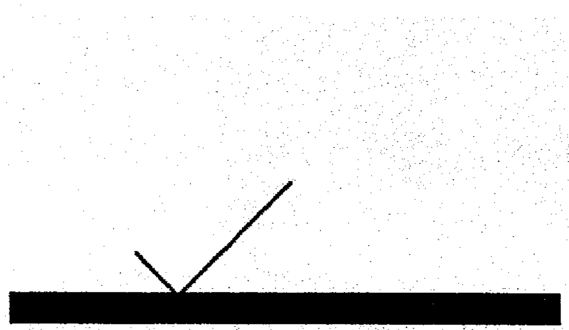
คลื่นเสียงเมื่อเดินทางผ่านตัวกลางที่มีความหนาแน่นต่างกันจะเกิดการหักเหของคลื่น ซึ่งทำให้ความเร็วของคลื่นเสียงเปลี่ยนไป โดยที่ความถี่ยังคงที่อยู่



รูปที่ 2.5 แสดงการหักเหของคลื่นเสียงในตัวกลางที่มีความหนาแน่นต่างกัน

4) การสะท้อนของคลื่นเสียง (Reflection)

คลื่นเสียงสามารถสะท้อนได้เมื่อตกกระทบตัวกลางโดยที่มุมสะท้อนและจะทำให้เกิดเสียงก้อง ซึ่งเสียงก้องที่สะท้อนกลับมาในเวลาที่มากกว่า 50 mS จะทำให้เราได้ยินเสียงนี้เป็นครั้งที่2



รูปที่ 2.6 การสะท้อนของคลื่น

2.1.7 ตารางความเร็วเสียงที่เคลื่อนที่ในวัตถุต่าง ๆ

Speed of Sound in Various Materials

solids	v (m/s)
aluminum	6420
beryllium	12,890
brass	4700
brick	3650
copper	4760
cork	500
glass, crown	5100
glass, flint	3980
glass, pyrex	5640
gold	3240
granite	5950
iron	5950
lead	2160
lucite	2680
marble	3810
rubber, butyl	1830
rubber, vulcanized	54
silver	3650
steel, mild	5960
steel, stainless	5790
titanium	6070
wood, ash	4670
wood, elm	4120
wood, maple	4110
wood, oak	3850

liquids	v (m/s)
alcohol, ethyl	1207
alcohol, methyl	1103
mercury	1450
water, distilled	1497
water, sea	1531

gases (STP)	v (m/s)
air, 000 °C	331
air, 020 °C	343
argon	319
carbon dioxide	259
helium	965
hydrogen (H_2)	1284
neon	435
nitrogen	334
nitrous oxide	263
oxygen (O_2)	316
water vapor, 134 °C	494

Sources: Unknown, but probably an old version of the CRC

2.2 สารเปียโซอิเล็กทริกเซรามิก

สารเปียโซอิเล็กทริกเซรามิกคือ สารเซรามิกชนิดหนึ่งมีสมบัติเปียโซอิเล็กทริก (piezoelectric effect) ซึ่งสมบัติเปียโซอิเล็กทริกนี้ถูกค้นพบครั้งแรกโดย Pierre Curie และ Jacques Curie เมื่อปี ค.ศ. 1880 ในการทดลองทางฟิสิกส์ของผลึกบางชนิดเช่น แร่ควอทซ์, Rochelle salt, ซิงก์เบรน (zincblende) และทัวมาลีน (tourmaline) ฯลฯ แต่เนื่องจากผลึกแบบเดี่ยว (single crystal) ยังมีความเสถียรต่ำต่ออุณหภูมิ จนกระทั่งปี ค.ศ. 1940 จึงได้มีการพัฒนาสารเปียโซอิเล็กทริกเซรามิก (Piezoelectric ceramics) ขึ้น ซึ่งมีข้อดีกว่าสารเปียโซอิเล็กทริกแบบเดี่ยว คือ

-ประสิทธิภาพการเปลี่ยนรูปพลังงาน Electromachanical สูง

-สามารถตกแต่งรูปทรงได้ตามต้องการโดย Mach inability

-สามรถปรับคุณสมบัติด้วยการปรับปรุงส่วนประกอบของเนื้อสาร (High degree of freedom in characteristics design)

-มีเสถียรภาพสูง

-สามารถนำไปผลิตเป็นอุตสาหกรรมจำนวนมากๆ ได้

ทำให้มีการพัฒนาสารเปียโซอิเล็กทริกเซรามิกไปประยุกต์ใช้งานทางด้านอัลตร้าโซนิกส์ทรานสดิวเซอร์ (Ultrasonic transducer) อย่างมากมาย เช่นเซนเซอร์ตรวจหาปลา (fish finders), เครื่องทำความสะอาดด้วยคลื่นอัลตร้าโซนิกส์, เครื่องเชื่อมพลาสติกด้วยเครื่องอัลตร้าโซนิกส์ ฯลฯ สำหรับการพัฒนาสารเปียโซอิเล็กทริกเซรามิกไปประยุกต์ใช้งาน การวิเคราะห์สมบัติทางด้านไฟฟ้าของสารนับเป็นสิ่งสำคัญเพื่อให้สารโซเปียอิเล็กทริกเซรามิกถูกนำไปใช้งานได้อย่างเต็มประสิทธิภาพ และวิธีการอธิบายสมบัติทางไฟฟ้าของสารที่นิยม ก็คือการใช้วงจรไฟฟ้าสมมูลแบบ Butterworth Van dyke หรือ BVD ซึ่งมีสมทางไฟฟ้าที่ใกล้เคียงกับสารเปียโซอิเล็กทริกเซรามิกในช่วงความถี่เรโซแนนซ์ โดยวงจร BVD สามารถคำนวณหาได้จากข้อมูลอิมพิแดนซ์ของสารเปียโซอิเล็กทริกในช่วงความถี่เรโซแนนซ์ (resonance frequency) นั่นเองเพื่อให้นักวิจัยและพัฒนาสารเปียโซอิเล็กทริกเซรามิกได้เข้าใจถึงพื้นฐานของสมบัติทางไฟฟ้าของสารเปียโซอิเล็กทริกเซรามิกก่อนนำไปประยุกต์ใช้งาน งานวิจัยนี้นำเสนอวิธีการหาวงจรBVD ของสารเปียโซอิเล็กทริกเซรามิกชนิดเลดเซอร์โคเนตไททาเนต (Lead-Zerconate-Titanate) หรือ PZT ซึ่งผลิตด้วยกระบวนการทางเซรามิกแบบดั้งเดิม ลักษณะเป็นแผ่นกลม ผิวหน้าด้านบนและด้านล่างของสารถูกเคลือบด้วยโลหะเงิน มีรูปแบบการสั่นแบบ thickness mode หรือรูปแบบการสั่นในทิศทางขนานกับความหนาของสาร PZT เมื่อสารถูกกระตุ้นด้วยแรงดันไฟฟ้ากระแสสลับ

2.2.1 คุณสมบัติของสารเปียโซอิเล็กทริกเซรามิก

สารเปียโซอิเล็กทริกเซรามิกเป็นสารประกอบที่มีโครงสร้างแบบเปอร์รอฟสไกต์ (Perovskite structure) จัดเรียงตัวกันเป็นผลึกแบบ polycrystalline ซึ่งต่างจากผลึกควอทซ์ที่มีลักษณะโครงสร้างเป็นผลึกเดี่ยว (single crystal) ตัวอย่างสารเปียโซอิเล็กทริกเซรามิกได้แก่ แบเรียมไททาเนต (Barium Titanate), สารเลดไททาเนต (Lead Titanate) และเลดเซอร์โคเนตไททาเนต เป็นต้น สมบัติเปียโซอิเล็กซิตี้ (piezoelectricity) ของสารสามารถแบ่งได้เป็น 2 ลักษณะ คือ สมบัติเปียโซอิเล็กทริกแบบทางตรง (direct effect) และสมบัติเปียโซอิเล็กทริกแบบทางอ้อม (reverse effect)

1) สมบัติเปียโซอิเล็กทริกแบบทางตรง (direct effect)

คือสมบัติของสารที่สามารถสร้างประจุไฟฟ้าออกมาที่ผิวหน้า เมื่อมีแรงกด (Stress) มากระทำต่อสาร ดังรูปที่ 2.7 โดยสัญลักษณ์ P แทนเวคเตอร์ของความเป็นขั้วไฟฟ้าในเนื้อสาร (polarization) T คือแรงที่มากระทำต่อพื้นที่หน้าตัดหรือแรงเครียด (strain) มีหน่วยเป็นนิวตันต่อตารางเมตร และ D คือประจุที่เกิดขึ้นภายในเนื้อสาร (dielectric displacement) จะได้ว่า

$$D = Q / A = dT \qquad \text{-------------- (2.6)}$$

โดยที่

d คือค่าคงที่เปียโซอิเล็กทริก (piezoelectric constant) ของสารมีหน่วยเป็นคูลอม ต่อนิวตัน (Coulombs / Newton)

Q คือ ประจุไฟฟ้ามีหน่วยเป็นคูลอม

A คือ พื้นที่หน้าตัดของเนื้อสารมีหน่วยเป็นเมตร

2) สมบัติเปียโซอิเล็กทริกแบบทางอ้อม (reverse effect)

เป็นสมบัติของสารที่สามารถยืดหรือหดตัวได้เมื่ออยู่ใต้สนามไฟฟ้า ดังรูปที่ ถ้าให้ S คือความเค้นที่เกิดภายในเนื้อสารและ E คือแรงดันไฟฟ้าที่มากระทำต่อพื้นที่หน้าตัด สามารถเขียนสมการได้ดังนี้

$$S = dE \qquad \text{----------------- (2.7)}$$

โดยสัญลักษณ์ d สามารถใช้แทนค่าคงที่เปียโซอิเล็กทริกทั้งแบบทางตรงและทางอ้อมได้ เนื่องจากเป็นอัตราส่วนของการเปลี่ยนรูปพลังงาน จากสมการ 2.1 และ 2.2 เมื่อพิจารณาถึงสมบัติของค่าความยืดหยุ่น (elastic) และค่าไดอิเล็กทริกของเนื้อสาร สามารถเขียนเป็นสมการเปียโซอิเล็กทริก คือ

$$D = dT + \varepsilon^{T}E \qquad \text{---------------- (2.8)}$$

$$S = sET + dE \qquad \text{---------------- (2.9)}$$

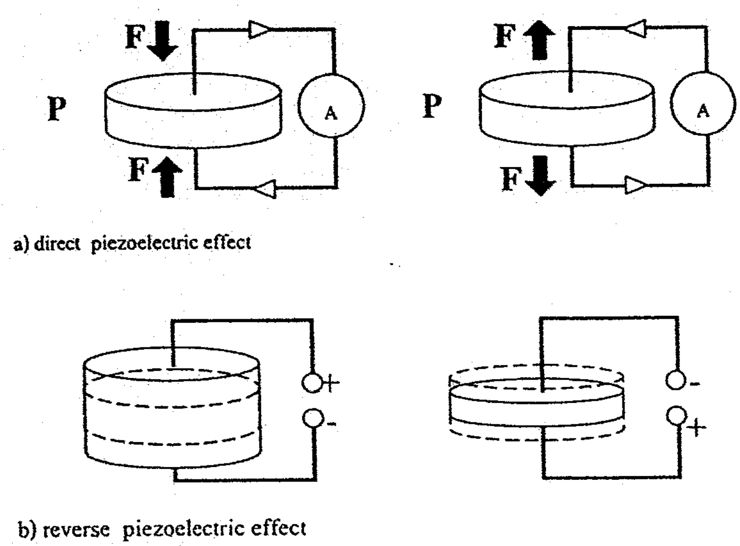
โดยที่

D คือค่า electric flux density หรือ electric displacement

S คือค่าความเค้น ($S = \Delta L/L$)

ε^{T} คืออัตราความจุกระแสไฟฟ้า (relative permittivity) หรือค่าคงที่ไดอิเล็กทริกของสาร (Free dielectric constant) (ซึ่งบางตำราใช้ K เป็นสัญลักษณ์) และสัญลักษณ์ตัวยกกำลัง T แทนสภาวะที่ไม่มีแรงกด

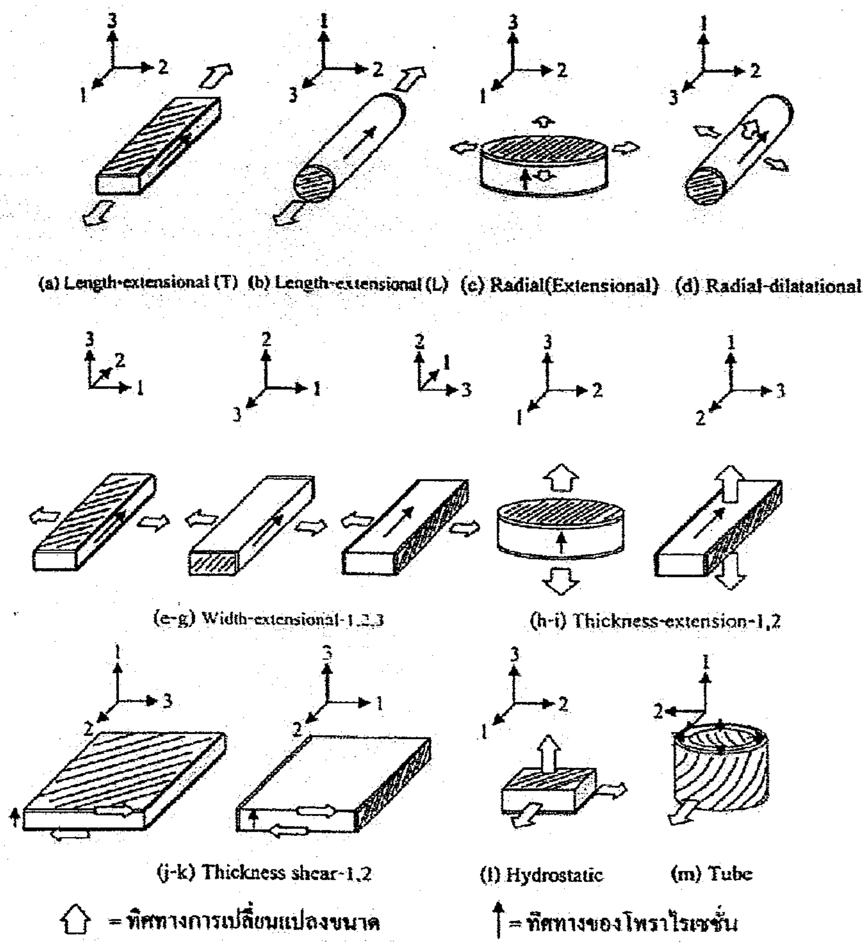
s^{E} คือสัมประสิทธิ์ความยืดหยุ่นของสาร (elastic coefficient) และสัญลักษณ์ตัวยกกำลัง E แทนสภาวะที่ไม่มีสนามไฟฟ้า



รูปที่2.7 แสดงสมบัติทางเปียโซอิเล็กทริกเซรามิก

2.2.2 โหมดการทำงานของเปียโซอิเล็กทริกเซรามิค

ในทิศทางการสั่นของสารเปียโซอิเล็กทริกเซรามิคเมื่อถูกกระตุ้นด้วนแรงดันไฟฟ้ากระแสสลับมีหลายรูปแบบขึ้นกับรูปทรงและชนิดของสารเปียโซอิเล็กทริกเซรามิค ดังแสดงในรูปที่ 2.8



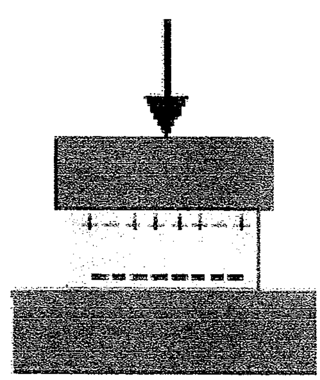
รูปที่ 2.8 แสดงโหมดการสั่นของสารเปียโซอิเล็กทริกเซรามิค

ตัวอย่างการนำสารเปียโซอิเล็กทริกเซรามิกไปใช้งานทางด้านคลื่นอุลตร้าโซนิกส์แบบทางตรงได้แก่ เครื่องจุดเตาแก๊ส, ตัวเซนเซอร์วัดแรงดัน แบบทางอ้อมได้แก่ เครื่องเชื่อมพลาสติกอุลตร้าโซนิกส์, เครื่องทำความสะอาดด้วยคลื่นอุลตร้าโซนิกส์ นอกจากนั้นสารเปียโซอิเล็กทริกยังสามารถนำไปใช้เป็นอุปกรณ์รับและส่งคลื่นอุลตร้าโซนิกส์ในตัวเดียวกันได้เช่น อุปกรณ์ตรวจหาปลา และโซนาทรานสดิวเซอร์ ซึ่งสารเปียโซอิเล็กทริกเซรามิคในแต่ละตัวอย่างข้างต้นจะมีรูปทรงชนิด และรูปแบบหรือโหมดการสั่นแตกต่างกันออกไป ตัวอย่างเช่นอุปกรณ์ทรานสดิวเซอร์ในเครื่องทำความสะอาดด้วยคลื่นอุลตร้าโซนิกส์ จะเป็นสารเปียโซอิเล็กทริกเซรามิกที่ทำจากสารเลดเซอร์โคเนตไททาเนต มีรูปแบบแบบการสั่นที่มีทิศทางแนวตั้งฉากกับผิวหน้าของสารหรือมีทิศทางการสั่นขนานกับความหนา เรียกว่า Thickness mode ดังรูปที่ 2.8 มีลักษณะเป็นแผ่นวงกลมและมีอัตราส่วนของขนาดเส้นผ่านศูนย์กลางต่อความหนาเท่ากับ 10 หรือมากกว่า

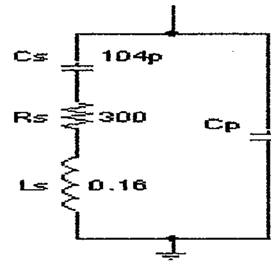
2.2.3 ปรากฏการณ์เปียโซอิเล็กทริก

เปียโซอิเล็กทริกเป็นปรากฏการณ์ธรรมชาติอย่างหนึ่ง ซึ่งทำให้พลังงานสามารถเปลี่ยนแปลงจากรูปหนึ่งไปเป็นอีกรูปหนึ่งได้ กล่าวคือถ้าป้อนแรงกลให้แก่ Solek Crystalline Dielectric ดังในรูปที่ 2.9 ก็จะทำให้เกิดความเค้น (Stress) ภายในคริสตอลและทำให้ผลึกของคริสตอลผิดรูปไปเช่นพวกควอทซ์ (Quartz) ผลก็คือประจุเปลี่ยนไป การผิดรูปร่างของแลททิสเป็นผลให้ความสัมพันธ์ระหว่างการแทนที่ (Displacement) ของประจุบวกและลบในแลททิสเปลี่ยนไป การแทนที่ของประจุภายในจะเท่ากับประจุภายนอกของขั้วที่ตรงข้ามของคริสตอล เรียกว่าผลของเปียโซอิเล็กทริก

การวัดประจุทำได้โดยการต่ออิเลคโทรด (Electrode) เข้าที่ผิวนอก แล้ววัดความต่างศักย์ระหว่างขั้วทั้งสอง ขนาด (Magnitude) และการมีขั้ว (Polarity) ของประจุบนผิวที่ถูกเหนี่ยวนำ (induce) เป็นสัดส่วนโดยตรงกับขนาดและทิศทางของแรงที่มากระทำ



รูปที่ 2.9 ผลของเปียโซอิเล็กทริก



รูปที่ 2.10 วงจรเสมือนของเปียโซอิเล็กทริก

C_c: คาปาซิสแตนท์ของชิ้นส่วนเปียโซอิเล็กทริก

R_c : Leagage Resistance ของเปียโซอิเล็กทริก ปกติจะมีค่าสูงประมาณ 10^8 - 10^{10}

ความต้านทานระหว่างเทอร์มินอลโดยทั่วไปจะเป็นความต้านทานทางโหลด R_L (ในช่วง $10^6 - 10^7$)

C_L : คาปาซิแตนท์ ของ Subsequent Stage(Load) ร่วมกับ Connection Cable

ที่ความถี่ปานกลางและความถี่สูงโวลท์เตจ E_0 ที่คร่อมโหลดหาได้โดย C_c และ C_L ค่าเหล่านี้มาได้โดยการใช้การแบ่งโวลท์เตจ โดยโวลตท์เตจ E_0 ได้จากสมการ

$$E_0 = E\,(C_c / (C_c + C_L))$$

โดยขึ้นกับความถี่ ถ้าเอาท์พุทโวลท์เตจมีค่ามากสามารถทำให้ลดลงได้โดยการเพิ่มค่า C_L เช่นการนำค่า C_c ต่อขนานเข้าไปที่ความถี่ต่ำโวลท์เตจ E_0 คร่อมโหลดหาได้โดยค่ารีแอคแตนท์ของ C_c และอิมพีแดนซ์ของ C_L และ R_L ที่ต่อขนานกันโวลท์เตจ E_0 ขึ้นกับความถี่และจะลดลงถ้าความถี่ลดลง การตอบสนองต่อความถี่สามารถปรับปรุงได้โดยการใช้ทรานสดิวเซอร์ที่มีค่าคาปาซิแตนท์สูงหรือโดยการเพิ่มความต้านทานโหลด RL

2.2.4 วัสดุเปียโซอิเล็กทริก

วัสดุเปียโซอิเล็กทริกที่ใช้กันอย่างกว้างขวาง เช่น Quartz , Tourmaline sulphate, Barium Tilanate และ Ziconnate (TZI) โดยทั่วไปแล้วพวกควอทซ์ และคริสตอลที่เป็นเปียโซอิเล็กทริกธรรมชาติมักจะมีขั้วของมันเองตามธรรมชาติ แต่พวกวัสดุเปียโซอิเล็กทริกที่สังเคราะห์ขึ้นมาเช่น แบเรียม ไททาเนท เซรามิค จะต้องนำมาทำการอบคริสตอลภายในแรงดัน และวางวัสดุที่ได้นี้ในสนามไฟฟ้าที่มีแรงดันของไฟฟ้ากระแสตรงมาก หลังจากชิ้นส่วนนี้ถูกนำในสนามไฟฟ้าแล้ว คริสตอลนี้จะมีขั้วตามแนวของทิศทางของสนาม และประพฤติตัวตามคุณสมบัติของเปียโซอิเล็กทริกสำหรับชิ้นส่วนที่ทำจากวัสดุสังเคราะห์นี้ ไม่มีข้อจำกัดทางขนาด โดยโครงสร้างของคริสตอลและยังสามารถทำให้มีรูปร่างและขนาดต่างๆ และทิศทางของขั้วก็จะถูกสร้างขึ้นระหว่างขั้นตอนการผลิต

2.2.5 ชนิดของเปียโซอิเล็กทริกทรานสดิวเซอร์

เปียโซอิเล็กทริกทรานสดิวเซอร์สามารถแบ่งได้เป็น 2 ชนิดคือ

- *แบบ Generation-Action Tranducer* ใช้เป็นตัวรับโดยแรงดันไฟฟ้าที่เกิดขึ้นจะหาได้จากแรงดันและความถี่ที่มากระทำต่อวัสดุเปียโซอิเล็กทริก

- *แบบ Motor-Action Transducer* ใช้เป็นตัวส่งโดยการเปลี่ยนแปลงของรูปร่างที่ทำให้เกิดคลื่นอัลตร้าโซนิค โดยจะขึ้นอยู่กับขนาดความสูงและความถี่ของแรงดันไฟฟ้าที่ให้ในทั้งสองกรณีค่าแรงดันไฟฟ้าที่เกิดขึ้นกับขนาดของวัสดุ

2.2.6 การทำงานของทรานสดิวเซอร์ตัวส่งตัวรับ

เมื่อเซรามิคได้รับสัญญาณแรงดันมาตกคร่อมจะทำให้ชิ้นสารเซรามิคโค้งงอทำให้เกิดการอัดอากาศโดยรอบเกิดเป็นคลื่นขึ้นมา ดังนั้นถ้าป้อนสัญญาณเป็นห้วงๆ (Electrically Pulse) จากการออสซิสเลทก็จะให้ชิ้นสารโก่งงอมากน้อยหรือทิศทางใดตามขนาดและทิศทางการเปลี่ยนแปลงขนาดของสัญญาณไฟฟ้าจากการออสซิสเลทนั้นออกไป โดยทั่วไปกำลังเอาท์พุทที่ออกมาจะตกประมาณ 10% ของกำลังไฟฟ้าที่ป้อนให้แต่เอาท์พุทจะสูงที่ค่านี้ โดยประมาณก็ต่อเมื่อความถี่ของสัญญาณออสซิสเลเตอร์ที่ป้อนเข้าชิ้นสารเซรามิค ตรงกับความถี่เรโซแนนท์เป็นความถี่ทางกลตามธรรมชาติของชิ้นสารเซรามิคนั้นๆ ส่วนที่ความถี่อื่นๆกำลังเอาท์พุทจะลดลงกว่านี้ ส่วนการทำงานของทรานสดิวเซอร์ตัวรับมักมีการทำงานตรงข้ามกับตัวส่ง คือเมื่อมีคลื่นเสียงที่มีความถี่ตรงกับความถี่เรโซแนนท์ของชิ้นสารเซรามิคเข้ามา จะทำให้ชิ้นสารโก่งตัวไปมาและเกิดสัญญาณแรงดันไฟฟ้าที่มีขนาดเล็กขึ้นคร่อมขั้วทั้งสองของตัวมัน

คุณสมบัติทั่วไปของทรานสดิวเซอร์แบบเปียโซอิเล็กทริกก็คือ มีความต้านทานไฟฟ้าตรงสูงมากอาจถึง 100 เมกกะโอห์ม เรียกว่า ถ้าเอามัลติมิเตอร์ธรรมดา ตั้งสเกลวัดค่าความต้านทานสูงๆ เข็มจะไม่กระดิกเลยแต่ขณะที่มันทำงานความต้านทานไฟฟ้าสลับจะลดลง

2.3 FPGA (Field Programmable Gate Array)

โครงสร้างภายในของ FPGA ประกอบไว้ด้วย แอร์เรย์ของลอจิกเกทต่างๆ มากมาย ซึ่งในปัจจุบันความจุเกทภายในตัวชิพ FPGA ได้เพิ่มขึ้นจากระดับไม่กี่พันตัวจนระดับล้านตัว ซึ่งสามารถรองรับวงจรดิจิตอลที่มีความสลับซับซ้อนได้เป็นอย่างดี นอกจากนี้ในด้านการออกแบบพัฒนาและทดสอบก็ทำได้ง่าย ซึ่งในปัจจุบันการออกแบบวงจรโดยใช้วิธี FPGA กำลังเป็นที่นิยมและมีแนวโน้มที่จะนำมาใช้งานมากขึ้นเรื่อย ๆ

2.3.1 FPGA แบ่งได้ตามเทคโนโลยีที่ใช้ในการโปรแกรมซึ่งมีอยู่ 2 แบบคือ

การโปรแกรมโดยการทำให้เกิดการเปลี่ยนแปลงทางกายภาพของตัวชิพ และการโปแกรมโดยการใช้หน่วยความจำ

1. การโปรแกรมโดยการทำให้เกิดการเปลี่ยนแปลงทางกายภาพ

*1.1 Fuse*เป็นวิธีการโปแกรมที่สามารถทำได้เพียงครั้งเดียวซึ่งหลังจากที่โปรแกรมแล้วจุดเชื่อมต่อจะขาดจากกัน

1.2Anti Fuse เป็นวิธีการโปแกรมที่คล้ายกับแบบ Fuse แต่ต่างกันที่หลังจากทำการโปรแกรมแล้ว จะเชื่อมต่อถึงกัน

2. การโปรแกรมโดยใช้หน่วยความจำ

2.1 EEPROM Based FPGA

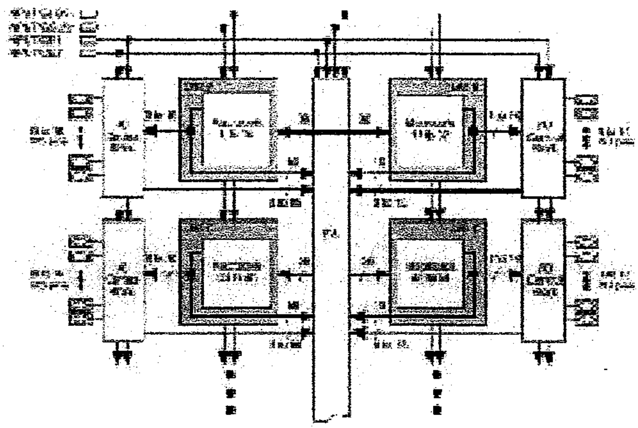
FPGA ที่ใช้การโปรแกรมแบบนี้มันเรียกว่า CPLD ซึ่งเทคโนโลยีที่ใช้จะเหมือนกับ EEPROM ทำให้มีความจุของเกทต่ำ โดยทั่วไปจะน้อยกว่า 20,000 เกท แต่ละข้อดีของ EEPROM Based FPGA คือสามารถเก็บข้อมูลที่โปรแกรมลงไปได้โดยไม่จำเป็นต้องมีไฟเลี้ยง และในการโปรแกรมจะใช้ทรานซิสเตอร์ 1 ตัวต่อ 1 บิต ซึ่งการโปรแกรมสามารถทำได้ประมาร 10,000 ครั้ง

2.2 SRAM Based FPGA

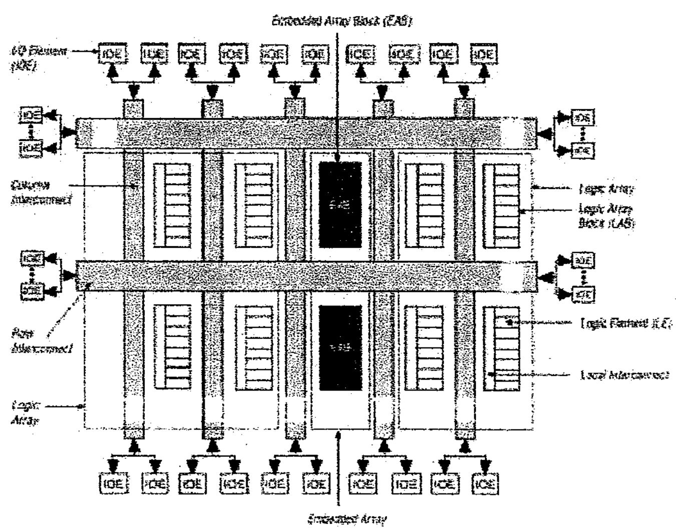
FPGA แบบนี้จะใช้เทคโนโลยีในการโปรแกรมเหมือนกับ SRAM (Static RAM) ทำให้สมารถโปแกรมซ้ำได้โดยไม่จำกัดจำนวนครั้ง นอกจากนี้ยังมีความจุของเกทในระดับปานกลางถึงสูงมาก (ประมาณ 10,000 – 1,000,000 เกท) ซึ่งข้อดีของ SRAM Based FPGA คือใช้เวลาในการโปรแกรมน้อย (ระดับ ns) การโปรแกรมทำได้ง่ายเทียบได้กับการเขียน SRAM ทั่วไป และเหมาะสำหรับการออกแบบวงจรที่มีความสลับซับซ้อนส่วนข้อเสียคือไม่สามารถเก็บโปรแกรมในภาวะที่ไม่มีไฟเลี้ยงได้ ดังนั้น FPGA ชนิดนี้จึงมักใช้ควบคู่กับ ROM เพื่อเก็บโปรแกรมและทำการโหลดโปรแกรมลงในตัวชิพในขณะที่เริ่มต้นใช้งาน

2.3.2 โครงสร้างภายในของ FPGA

ลักษณะโครงสร้างภายในของ FPGA จะเป็นอาร์เรย์ ของบล็อคลอจิกสามารถทำการโปรแกรมได้ดังรูปที่ 2.11 และ 2.12



รูปที่ 2.11 โครงสร้างภายใน FPGA ตระกูล MAX7000S



รูปที่ 2.12 โครงสร้างภายในของ FPGA ตระกูล FIEX10K

2.3.3 รายละเอียดของขั้นตอนในการออกแบบสามารถอธิบายได้ดังนี้

1.การสังเคราะห์วงจร (Logic Synthesis)

ในขั้นตอนนี้จะใช้ซอฟต์แวร์ในการสังเคราะห์วงจร (Synthesis Tools) ทำการสังเคราะห์พฤติกรรมของวงจรที่ได้จากการออกแบบด้วย Schematic หรือ VHDLซึ่งต้องทำการตรวจสอบด้วยว่าซอฟต์แวร์นั้นสนับสนุนเทคโนโลยี FPGA (FPGA Library) ที่ต้องการหรือไม่ ตัวอย่างเช่น FPGA ของบริษัท XLINX และบริษัท ALTIRA จะมีซอฟต์แวร์สังเคราะห์วงจรจะทำการแปลงโค้ด VHDL และทำการ Optimize เพื่อให้ได้วงจรตามเทคโนโลยี ที่เลือกใช้ ในการสังเคราะห์วงจรนั้น วงจรระดับเกท (Gate Level) จะไม่เหมาะสมกับโครงสร้างที่มีอยู่ในอุปกรณ์ FPGA นั้นๆจึงทำให้ผลที่ได้มีประสิทธิภาพและในขั้นตอนการสังเคราะห์วงจรนี้ ผู้ออกแบบสามารถกำหนดข้อบังคับสำหรับโมเดลแต่ละตัวได้ เช่น ข้อบังคับในเรื่องเวลา (Timing Constraints) หรือข้อบังคับในเรื่องของพื้นที่ (Area) หรือกำหนดชนิดและตำแหน่งของ I/O ซึ่งข้อบังคับเหล่านี้จะถูกนำไปใช้ในขั้นตอน Optimize เพื่อให้วงจรที่ได้เป็นไปตามที่กำหนด ส่วนสำคัญในการ Optimize คือการเทียบ (Mapping) โมเดลให้เข้ากับเทคโนโลยีที่ใช้เพื่อให้ได้วงจรที่เหมาะสมกับโครงสร้างและสถาปัตยกรรมภายในอุปกรณ์ FPGA เมื่อทำการสังเคราะห์วงจรเสร็จแล้ว ซอฟแวร์การสังเคราะห์วงจรก็จะมีการทราบผลว่าโมเดลที่ออกแบบไปนั้นเป็นอย่างไร เช่นมีค่าความหน่วง (Delay) เท่าใด ใช้ทรัพยากรต่างๆใน FPGA อะไรบ้าง เมื่อมาถึงขั้นตอนนี้ ผู้ออกแบบก็จะทราบว่าโมเดลเป็นไปตามข้อบังคับหรือไม่ ถ้าไม่ก็สังเคราะห์ใหม่จนกว่าจะเป็นไปตามที่กำหนด

2. การแบ่งวงจร (Partitioning)

ขั้นตอนนี้เป็นการแบ่งวงจรที่ได้จากการสังเคราะห์ เป็นส่วนย่อยๆสำหรับลงใน CLBs, IOBs หรือองค์ประกอบอื่นๆภายในอุปกรณ์ FPGA สำหรับเกณฑ์ที่ใช้ในการแบ่ง คือให้แต่ละส่วนที่จะแยกออกจากกันมีจำนวนสัญญาณที่เชื่อมต่อระหว่างกันน้อยที่สุดเท่าที่จะทำได้ เพื่อลดความหนาแน่นในตอนทำการเชื่อมต่อสัญญาณ (Routing) ในขั้นตอนนี้จะใช้ซอฟต์แวร์ทำ โดยซอฟต์แวร์จะเทียบส่วนประกอบของวงจรเช่น เกท (gate), ฟลิบ-ฟลอบ (flip-flop) ลงในทรัพยากรต่างๆ ที่มีอยู่ในอุปกรณ์ FPGA

หลังจากทำขั้นตอนนี้เสร็จแล้วผู้ออกแบบสามารถที่จะทราบว่าวงจรใช้จำนวนทรัพยากรภายในอุปกรณ์ FPGA ไปเท่าไร ส่วนข้อมูลทางเวลานั้นผู้ออกแบบจะทราบเฉพาะความหน่วงลอจิก (Logic Delay) ส่วนซอฟต์แวร์จะรวมเอาซอฟต์แวร์ย่อยอื่นๆ อีก เพื่อให้การทำ PPR (Partitioning Placement & Routing) เป็นไปอย่างต่อเนื่อง

3. การวางอุปกรณ์ (Placement)

ขั้นตอนนี้เป็นการเลือกทำเลที่ตั้งของแต่ละส่วนของวงจรที่ผ่านการแบ่งวงจร (Partitioning) มาแล้วว่าควรจะอยู่ ณ ตำแหน่งไหนในอุปกรณ์ FPGA เพื่อให้ได้ผลลัพธ์ดีที่สุด เช่นวงจรส่วนไหนควรอยู่ใกล้กันเพื่อจะได้ค้นหาเส้นทางได้ (route) ง่ายหรือช่วยลดความหน่วง จะเห็นได้ว่าตำแหน่งภายในอุปกรณ์ FPGA นั้นมีความสำคัญเพราะถ้าจัดวงจรลงในตำแหน่งที่ไม่เหมาะสมแล้วจะทำให้ความหน่วงเพิ่มขึ้นหรือ Router ทำการค้นหาเส้นทางสัญญาณได้ไม่หมด

การวางอุปกรณ์ที่ดีควรวางส่วนต่างๆให้อยู่ใกล้กันโดยเฉพาะส่วนที่มีการเชื่อมต่อสัญญาณด้วนกันนอกจากนั้นการกำหนดตำแหน่งขา I/O (I/O pin) ตามตำแหน่งขา I/O ของ FPGA บนแผ่น PCB ก็จะมีผลโดยตรงเลยคือซอฟต์แวร์จะวาง I/O ลงในตำแหน่งที่ผู้ออกแบบกำหนด ซึ่งบางครั้งตำแหน่งที่กำหนดไปไม่เหมาะสม ดังนั้นการกำหนดขา I/O ควรกำหนดตำแหน่งให้เหมาะสม หรือไม่ก็ให้ซอฟต์แวร์จัดการเอง

4. การเชื่อมต่อสัญญาณ (Routing)

ในขั้นตอนนี้เป็นการเชื่อมต่อสัญญาณระหว่างองค์ประกอบต่างๆภายในอุปกรณ์ FPGA ขั้นตอนนี้จำทำต่อเนื่องจากการวางอุปกรณ์ ในกรณีที่ทำการวางอุปกรณ์ไว้ไม่ดีซอฟต์แวร์ก็จะทำการเชื่อมต่อสัญญาณได้ไม่หมด (เนื่องจากจำนวนทรัพยากรสำหรับเชื่อมต่อสัญญาณนั้นมีอยู่จำกัด) หรือเกิดความหน่วงเกินค่าที่กำหนดในข้อบังคับ

ผู้ออกแบบสามารถทำขั้นตอนนี้ได้โดยใช้ซอฟท์แวร์หรือผู้ออกแบบเชื่อมต่อสัญญาณด้วยด้วยตนเองก็ได้ แต่ทางที่ดีควรใช้ซอฟต์แวร์ทำดีกว่า นอกจากนั้นการกำหนดข้อบังคับทางเวลา จะช่วยให้ผลที่ได้จากการเชื่อมต่อสัญญาณดีขึ้นได้

5. ความหน่วงด้านเวลา (Delay)

ในการทำ FPGA นั้นความหน่วงที่เกิดขึ้นเป็นความหน่วงที่เกิดจากการวางตำแหน่ง (layout) ของอุปกรณ์ ซึ่งผู้ออกแบบไม่สามารถเข้าไปแก้ไขได้ แต่สามารถทำมีความหน่วงน้อยที่สุดได้ สำหรับความหน่วงที่เกิดขึ้นนั้นแยกได้เป็นสองประเภทคือ

5.1.1 ความหน่วงทางลอจิก (Logic delay) เป็นความหน่วงภายในองค์ประกอบของอุปกรณ์ FPGA เอง

5.1.2 ความหน่วงที่เกิดจากการเชื่อมต่อสัญญาณ (Routing delay) เป็นความหน่วงที่เกิดจากการเชื่อมต่อสัญญาณระหว่างองค์ประกอบภายในอุปกรณ์ FPGA

โดยปกติแล้วค่าความหน่วงลอจิกไม่ควรเกิน 50 % ของค่าความหน่วงที่ยอมรับได้ เพราะความหน่วงที่เกิดจากการเชื่อมต่อสัญญาณมันจะมีค่ามากกว่าค่าความหน่วงลอจิก ดังนั้นในการวางอุปกรณ์ และเชื่อมต่อสัญญาณ ผู้ออกแบบควรกำหนดข้อบังคับทางเวลาเพื่อให้ซอฟต์แวร์ได้ทำงานอย่างมีประสิทธิภาพเพิ่มขึ้น และเพื่อให้ได้ผลลัพธ์ที่ดีขึ้น

ค่าความหน่วงที่ได้หลังจากการวางอุปกรณ์ และเชื่อมต่อสัญญาณแล้วจะมีค่าความหน่วงที่ค่อนข้างแน่นอน ซึ่งผู้ออกแบบสามารถทราบได้ว่าโมเดลที่ออกแบบนั้น เป็นไปตามข้อกำหนดหรือไม่

6. การจำลองการทำงานของวงจร(Simulation)

ในขั้นตอนนี้เป็นขั้นตอนที่สำคัญอีกขั้นตอนหนึ่ง เพราะเป็นขั้นตอนที่ผู้ออกแบบตรวจสอบฟังก์ชันการทำงานของโมเดลถูกต้องหรือไม่มีข้อผิด พลาดตรงไหนเพื่อจะได้ทำการแก้ไขให้ถูกต้อง ในขั้นตอนนี้ซอฟต์แวร์ที่ใช้สำหรับทำการจำลองการทำงานของวงจรที่ใช้อยู่ เช่น Model Sim ของบริษัท Model Technology หรือ Max Plus II ของบริษัท Altera ในการจำลองการทำงานของวงจรควรทำทุกครั้งหลังจากที่มีการทำแต่ละขั้นตอนหลักเสร็จแล้ว เพื่อจะได้ทราบว่าข้อผิดพลาดของโมเดลเกิดขึ้นตอนไหน จะได้แก้ไขข้อผิดพลาดตรงขั้นตอนนั้นๆ นั้นคือการทำงานจำลองการทำงานของวงจร ต้องทำทั้งหลังการเขียนโค้ด,การสังเคราะห์วงจร และการทำ PPR การจำลองการทำงานของวงจรหลังจากที่เขียนโค้ดเสร็จแล้วนั้น ผู้ออกแบบสามารถทราบได้แค่โมเดลทำงานถูกต้องหรือไม่เท่านั้น (function test) ยังไม่สามารถตรวจสอบการทำงานในเชิงเวลาได้ถูกต้อง ในการจำลองการทำงานของวงจรหลังจากที่สังเคราะห์เป็นวงจรแล้ว เพื่อตรวจสอบว่าฟังก์ชันการทำงานยังคงถูกต้องหรือไม่ และค่าความหน่วงที่เกิดขึ้นเป็นไปตามข้อบังคับหรือไม่ มีข้อผิดพลาดเกิดขึ้นหรือถ้ามีจะแก้ไขให้ถูกต้อง

ในการจำลองการทำงานของวงจรหลังจากที่ทำการวางอุปกรณ์การเชื่อมต่อสัญญาณ (Post layout simulation) แล้วก็มีความสำคัญเช่นกันเพราะผลที่ได้จากการจำลองการทำงานของวงจรในตอนนี้จะเป็นผลลัพธ์ของโมเดลเลย ซึ่งผู้ออกแบบนอกจากที่ตรวจสอบฟังก์ชันการทำงานแล้วยังต้องตรวจสอบคุณสมบัติอื่นๆ เช่น ความหน่วงที่ได้จากการทำ PPR ในรูปแบบค่าความหน่วงมาตรฐาน (Standard Delay Format: SDF) ว่าตรงตามที่กำหนดหรือไม่ หรือตรวจสอบว่า วงจรรวมสามารถใช้งานที่ความถี่สูงสุดเท่าไรนั้นเอง ในการจำลองการทำงานของ วงจรควรใช้ซอฟต์แวร์ตัวเดียวกันตลอดเพื่อจะได้เปรียบเทียบผลที่ได้จากขั้นตอนต่างๆ

7 .การโปรแกรมอุปกรณ์ FPGA (Configuration)

หลังจากที่โมเดลผ่านขั้นตอนต่างๆจนกรทั้งผ่านการทำ PPR(Partitioning ,Placement & Routing) แล้วนั้น ถึงตอนนี้ก็สามารถที่ดาวน์โหลด (Download)ลงในอุปกรณ์ FPGA ได้แล้ว ในการดาวโหลดนี้ก่อนอื่นต้องแปลงแบบวงจรรวมที่ได้เป็นข้อมูลวงจร (Configuration data) ซึ่งอยู่ในรูปของบิตสตรีม (Bit stream) ก่อนแล้วจึงดาวโหลดลงไปเพื่อให้อุปกรณ์ FPGA มีฟังก์ชันการทำงานตามโมเดลที่ผู้ออกแบบต้องการ ซึ่งในขั้นตอนนี้จะใช้วิธีที่แตกต่างกันออกไปสำหรับอุปกรณ์ FPGA ของแต่ละบริษัทผู้ผลิตคือ

ในกรณีที่เป็นอุปกรณ์ FPGA ชนิดที่ต้องโปรแกรมโดยวิธี SRAM นั้นในการใช้งานผู้ออกแบบจะต้องเก็บข้อมูลวงจรไว้ในหน่วยความจำประเภท EPROM หรือ Serial PROM ด้วยเพื่อจะใช้งานสะดวกขึ้น คือในการใช้งานโมเดลครั้งต่อไปไม่ต้องดาวน์โหลดข้อมูลวงจรจากเครื่องคอมพิวเตอร์อีก เพราะมีข้อมูลวงจรเก็บข้อมูลอยู่ในหน่วยความจำอยู่แล้ว แต่กรณีที่อุปกรณ์ FPGA เป็นชนิดที่โปรแกรมโดยใช้วิธี EPROM หรือ Anti fuse ก็ไม่จำเป็นต้องมีหน่วยความจำสำหรับเก็บข้อมูลวงจร เพราะว่าอุปกรณ์ FPGA ชนิดนี้เมื่อดาวน์โหลดข้อมูลวงจรลงไป ข้อมูลที่ดาวน์โหลดลงไปก็ยังคงอยู่ในอุปกรณ์ FPGA ชนิดนี้เมื่อดาวโหลดข้อมูลวงจรลงไป ข้อมูลที่ดาวน์โหลดลงไปก็ยังอยู่ในอุปกรณ์ FPGA และครั้งต่อไปก็ใช้งานโมเดลที่ออกแบบไว้ได้เลย

2.4 ไมโครคอนโทรลเลอร์ MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 เป็นไมโครคอนโทรลเลอร์ขนาด 8 บิตที่มี อุปกรณ์สนับสนุนประกอบอยู่ภายในหลายอย่าง ได้แก่ หน่วยความจำสำหรับเก็บข้อมูล หน่วยความจำสำหรับเก็บโปรแกรม ตัวตั้งเวลา/ตัวนับ อุปกรณ์รับส่งข้อมูลแบบอนุกรม เนื่องจากโครงสร้างของไมโครคอน- โทรลเลอร์มีอุปกรณ์สนับสนุนประกอบอยู่ภายในนี้เอง ทำให้การใช้งานง่ายขึ้นและมีประสิทธิภาพมากขึ้น โดยไม่ต้องมีการเชื่อมต่ออุปกรณ์ภายนอกเพิ่มเติมมากเหมือนกับตัวไมโครโปรเซสเซอร์ทั่วไป นอกจากนี้หากเราต้องการใช้งานไมโครคอนโทรลเลอร์ร่วมกับอุปกรณ์อื่นเพิ่มเติม เช่น ไอซี 8255 หรือหน่วยความจำภายนอก เรายังสามารถนำมาเชื่อมต่อเพิ่มเติมเข้ากับไมโครคอนโทรลเลอร์ได้อีกด้วย

2.4.1 โครงสร้างภายในของไมโครคอนโทรลเลอร์ MCS-51

โครงสร้างภายในพื้นฐานของไมโครคอนโทรลเลอร์ MCS-51 แสดงในรูปที่ 2.13 ประกอบด้วยอุปกรณ์ต่างๆ ดังนี้

-หน่วยประมวลผลกลางขนาด 8 บิต

-หน่วยประมวลผลสำหรับข้อมูลแบบบิต (BOOLEAN PROCESSOR)

-ความสามารถในการอ้างตำแหน่งของหน่วยความจำโปรแกรม 64 กิโลไบต์

-ความสามารถในการอ้างตำแหน่งของหน่วยความจำข้อมูล 64 กิโลไบต์

-หน่วยความจำโปรแกรมภายในขนาด 4 กิโลไบต์ แบบ อีพรอม (เบอร์ 8451)

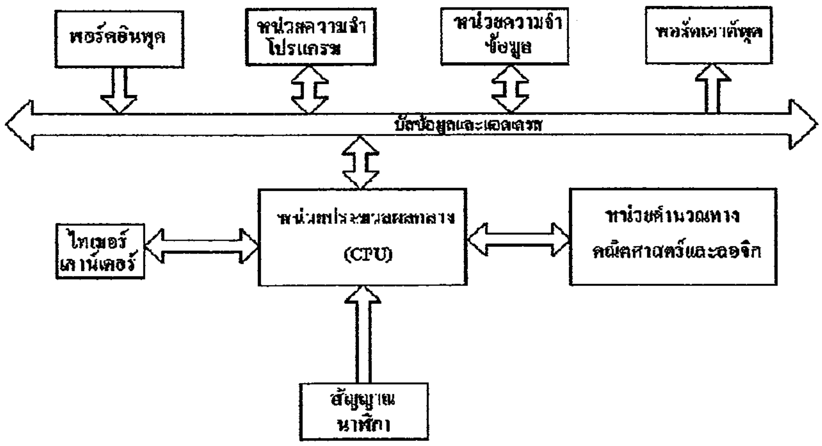
-หน่วยความจำแบบ แรม ภายในจำนวน 128 ไบต์

-พอร์ตอินพุต/เอาต์พุตแบบขนานจำนวน 32 เส้น ซึ่งสามารถแยกทำงานได้อย่างอิสระ

-วงจรนับ/จับเวลาขนาด 16 บิต จำนวนสองวงจร

-วงจรสื่อสารแบบอนุกรมแบบดูเพล็กเต็ม(FULL DUPLEX)

-วงจรควบคุมการอินเตอร์รัปต์จากแหล่งกำเนิดสัญญาณ 6 ประเภท พร้อมการกำหนดลำดับวงจรผลิตสัญญาณนาฬิกาภายในซึ่งโครงสร้างการทำงานทั้งหมดของไมโครคอนโทรลเลอร์จะ อาศัยหลักการทำงานที่เกี่ยวข้องกัน โดยอาศัยหลักการทำงานที่เป็นไปตามโครงสร้างเสมอ

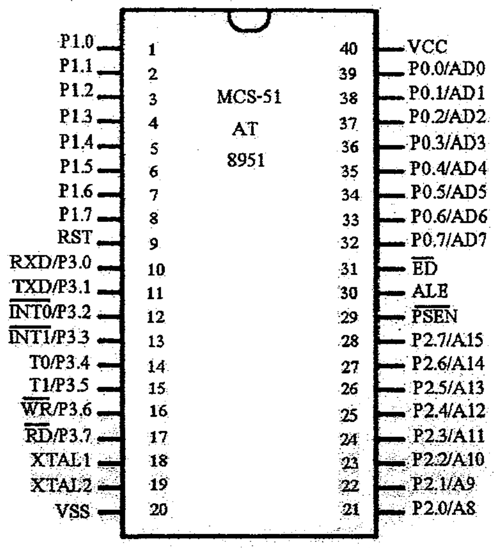


รูปที่ 2.13 แสดงโครงสร้างภายในของไมโครคอนโทรลเลอร์ MCS-51

โดยมากแล้วไมโครคอนโทรลเลอร์ตระกูลนี้มักจะมีรูปร่างของไอซีเป็นแบบขนาด 40 ขา ดังแสดงในรูปที่ 2.14 ซึ่งแต่ละขาสัญญาณจะมีหน้าที่ที่ระบุชัดเจนตามสัญลักษณ์ชื่อย่อที่กำกับในแต่ละขาอย่างไรก็ตามจะมีบางขาสัญญาณที่อาจจะมีหน้าที่ได้มากกว่าหนึ่งอย่าง (ซึ่งเขียนกำกับไว้ว่าALTERNATE FUNCTION ในรูปที่ 2.14) ซึ่งจะไม่สามารถใช้งานในเวลาเดียวกันได้ ตัวอย่างเช่นขาสัญญาณบิต 0 ของพอร์ต 3 (ใช้ตัวย่อเป็น P3.0) อาจจะใช้เป็นขาสัญญาณเอาต์พุตหรืออินพุตตาม ปกติ ภายในไมโครคอนโทรลเลอร์ MCS-51 ซึ่งประกอบด้วยหน่วยการทำงานต่างๆ ภายในไอซีMCS-51 จำนวนมาก โดยแต่ละบล็อกซึ่งเป็นวงจรควบคุมรีจิสเตอร์ (REGISTER) หรือหน่วยความจำภายในของไอซี MCS-51 จะถูกเชื่อมต่อเข้าด้วยกันผ่านทางเส้นสัญญาณที่เรียกว่าบัสข้อมูลภายใน รีจิสเตอร์และหน่วยความจำเหล่านี้จะถูกนำไปใช้ระหว่างการประมวลผลคำสั่ง หน้าที่ของโปรแกรมที่ผู้ใช้สร้างขึ้นมาก็เป็นการควบคุมการรับหรือส่งข้อมูลระหว่างรีจิสเตอร์เหล่านี้ ซึ่งอาจจะมีการดำเนินการร่วมกับหน่วยการดำเนินงานประมวลผลทางคณิตศาสตร์และลอจิก หรือเรียกว่าARITHMATIC AND LOGIC UNIT :ALU

2.4.2 โครงสร้างหน่วยความจำภายในไมโครคอนโทรลเลอร์ MCS-51

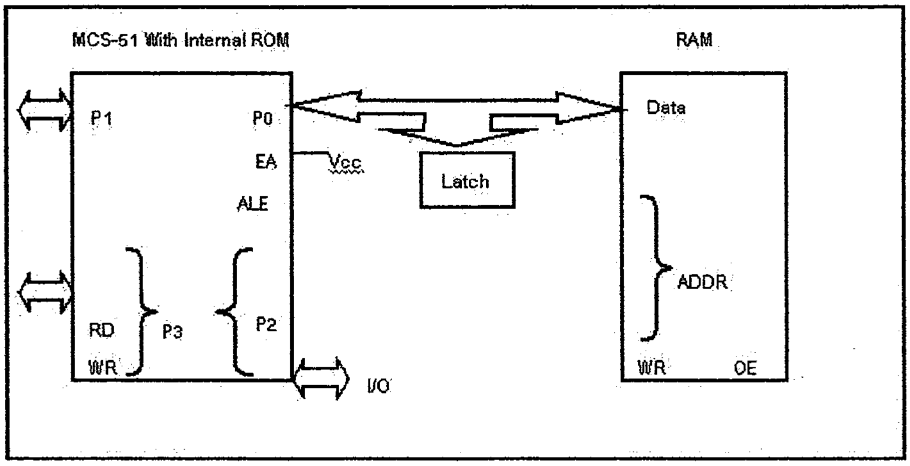
ไมโครคอนโทรลเลอร์ MCS-51 แยกการจัดการหน่วยความจำออกเป็นสองส่วนอย่างชัดเจน คือ หน่วยความจำโปรแกรม (PROGRAM MEMORY) และหน่วยความจำข้อมูล (DATA MEMORY) หน่วยความจำทั้งสองนี้ มีหน้าที่แตกต่างไปจากกัน และใช้วิธีการอ้างแอดเดรสสัญญาณการติดต่อแยกออกจากกัน



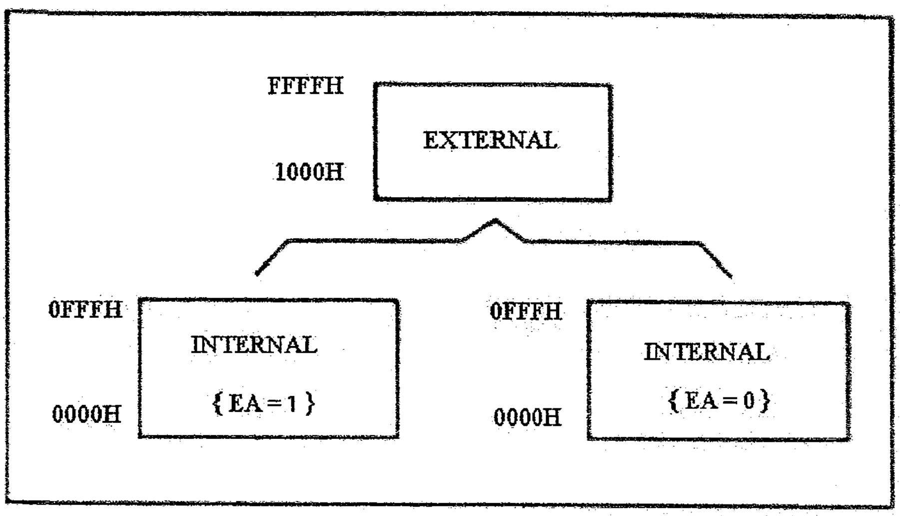
รูปที่2.14 แสดงรูปร่างและการจัดวางขาต่างๆ ของไมโครคอนโทรลเลอร์ MCS-51

2.4.3 หน่วยความจำโปรแกรม

หน่วยความจำโปรแกรมของไมโครคอนโทรลเลอร์ MCS-51 เป็นบริเวณหน่วยความจำสำหรับเก็บข้อมูลและคำสั่งใช้งานต่างๆ ซึ่งแม้ว่าจะไม่มีการจ่ายกระแสไฟฟ้าให้กับระบบข้อมูลเหล่านี้ก็ยังคงอยู่ไม่สูญหายโครงสร้างของหน่วยความจำโปรแกรมมีลักษณะเช่นเดียวกับหน่วยความจำที่บรรจุอยู่ใน ไมโครคอนโทรลเลอร์ MCS-51 ของหน่วยความจำประเภทต่างๆ เช่น หน่วยความจำแบบรอม (READ ONLY MEMORY) หรืออีพรอม(ERASABLE PROGRAMABLE READ ONLY MEMORY) ในไมโครคอนโทรลเลอร์ MCS-51 สามารถอ่านข้อมูลหน่วยความจำโปรแกรมนี้ได้สูงสุดไม่เกิน 64 กิโลไบต์ และแยกประเภทของหน่วยความจำโปรแกรมเป็น 2 ลักษณะ ตามตำแหน่งของหน่วยความจำนั้นคือ หน่วยความจำโปรแกรมภายใน(INTERNAL PROGRAM MEMORY)ซึ่งเป็นหน่วยความจำรอม หรืออีพรอม ที่อยู่ภายในตัวไอซีของไมโครคอนโทรลเลอร์เอง และหน่วยความจำโปรแกรมภายนอก (EXTERNAL PROGRAM MEMORY) ซึ่งเป็นการใช้ไอซีหน่วยความจำมาทำหน้าที่เป็นหน่วยความจำโปรแกรมของระบบ



รูปที่ 2.15 แสดงการใช้หน่วยความจำสำหรับเก็บโปรแกรม



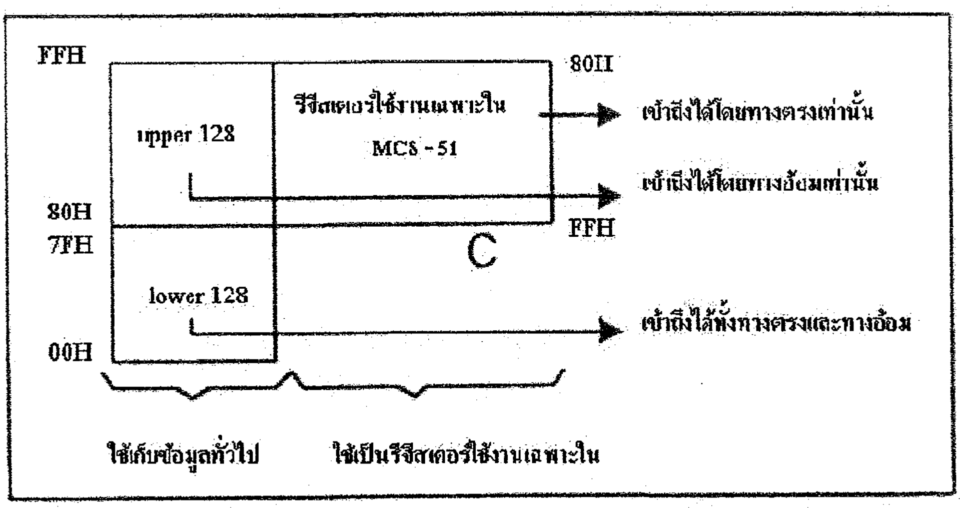
รูปที่ 2.16 แสดงการจัดพื้นที่ของหน่วยความจำโปรแกรมภายในและภายนอก

ไมโครคอนโทรลเลอร์เบอร์ต่างๆ ของตระกูล 8051 นี้สามารถขยายให้ใช้งานในหน่วยความจำภายนอกได้ทั้งสิ้น โดยกรณีที่มีหน่วยความจำโปรแกรมภายในอยู่แล้วการอ้างตำแหน่งแอดเดรสที่มีทั้ง ในหน่วยความจำโปรแกรมภายใน และภายนอกนั้นจะต้องทำการควบคุมระดับลอจิกของสัญญาณ ในขณะนั้นด้วยขนาดหน่วยความจำโปรแกรมภายในของไมโครคอนโทรลเลอร์เบอร์ต่างๆ ภายในตระกูล 8051จะแตกต่างกันออกไป เพื่อความเหมาะสมกับการนำไปใช้งานลักษณะต่างๆ

-8051และ8052มีหน่วยความจำแบบรอม4 และ 8 กิโลไบต์

-8751 มีหน่วยความจำแบบ อีพรอม ขนาด 4 กิโลไบต์ข้อมูลที่จัดเก็บภายในนี้ ซึ่งสามารถใช้แสงอุลตร้าไวโอเลตลบ และนำกลับไปบรรจุโปรแกรมใหม่ได้อีกครั้งหนึ่ง

-8031 และ 8032 ไม่มีหน่วยความจำโปรแกรมอยู่ภายในตัวไอซี ดังนั้นในการนำไปใช้งานจึงจำเป็นต้องอาศัยหน่วยความจำโปรแกรมภายนอกเสมอ

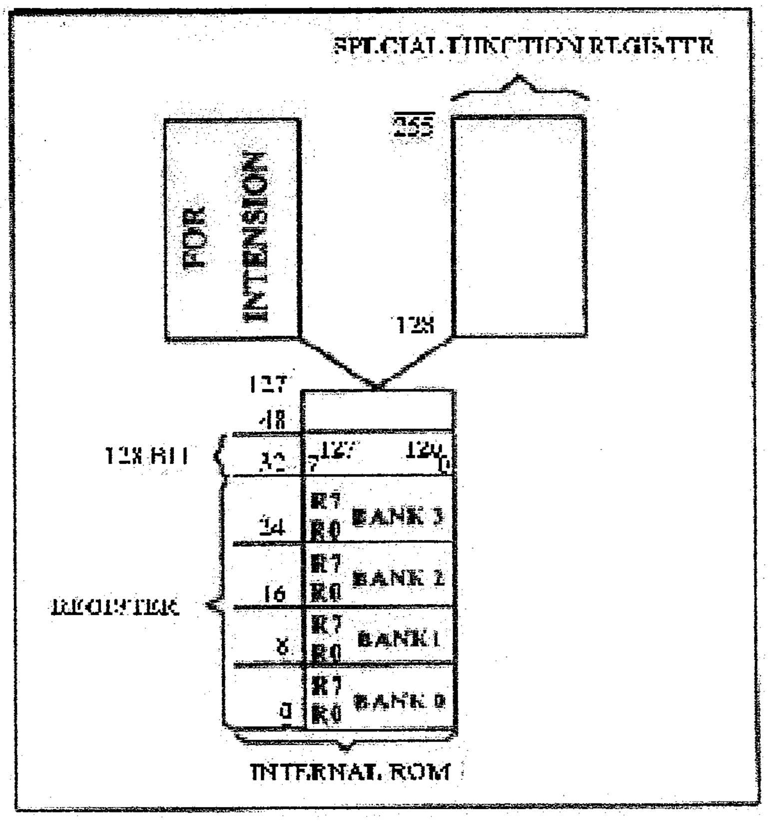


รูปที่ 2.17 แสดงหน่วยความจำสำหรับเก็บข้อมูลภายในไมโครคอนโทรลเลอร์ MCS-51

2.4.4 หน่วยความจำข้อมูล

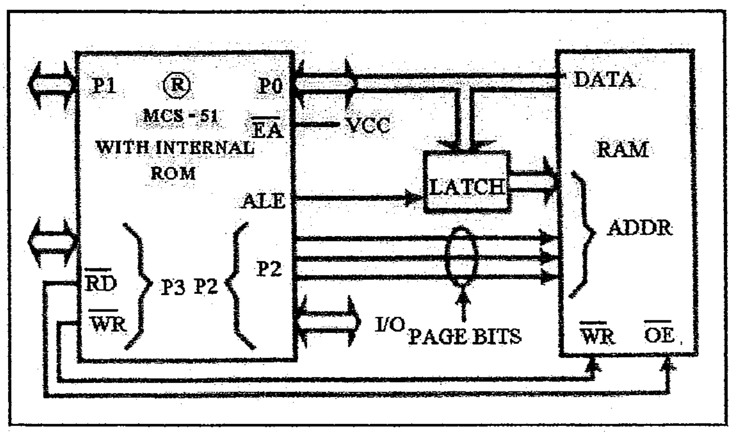
หน่วยความจำข้อมูล (DATA MEMORY) ซึ่งโดยพื้นฐานแล้วเป็นหน่วยความจำแรม สามารถเขียนหรืออ่านข้อมูลได้ (READ OR WRITE MEMORY) ใช้สำหรับเก็บข้อมูลหรือตัวแปรที่เกิดขึ้นในขณะที่กำลังประมวลผลโปรแกรมไว้เป็นการชั่วคราว ซึ่งโดยพื้นฐานแล้วหน่วยความจำข้อมูลจัดเป็นหน่วยความจำแรมแบบสแตติก ดังนั้นเมื่อไม่มีการจ่ายไฟฟ้าให้กับระบบก็จะมีผลทำให้ข้อมูลที่จัดเก็บ ไว้ภายในหน่วยความจำนี้สูญไป พื้นที่ของหน่วยความจำ

ข้อมูลของไมโครคอนโทรลเลอร์ MCS-51 มีได้สูงสุดไม่เกิน 64 กิโลไบต์ และแยกประเภทออกเป็นสองลักษณะตามตำแหน่งที่ตั้งของหน่วยความจำนั้น ตามลักษณะของหน่วยความจำโปรแกรมภายใน ซึ่งก็เป็นแรมที่อยู่ภายในตัวไอซีในตระกูลของไมโครคอนโทรลเลอร์ และหน่วยความจำข้อมูลภายนอกซึ่งเป็นการใช้ไอซีหน่วยความจำแรมมาเพิ่มเติมเข้าไปในวงจรลักษณะเดียวกับ การนำไอซีอีพรอมมาใช้งานเป็นหน่วยความจำโปรแกรมนั่นเอง



รูปที่ 2.18 แสดงการจัดหน่วยความจำข้อมูล

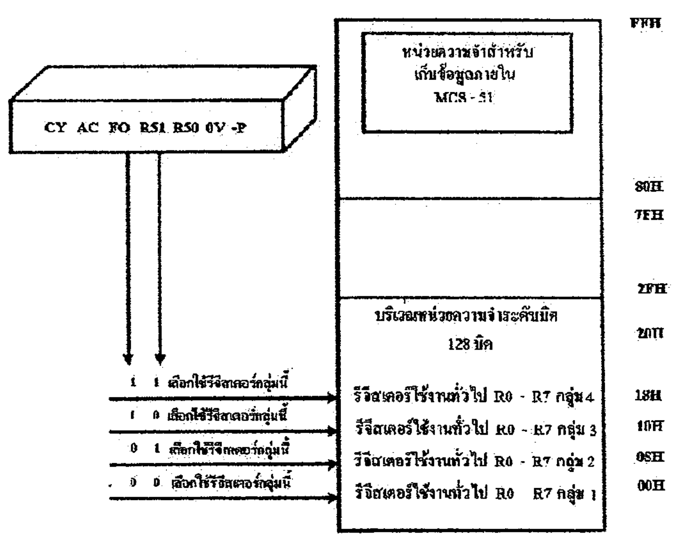
โดยที่หน่วยความจำสำหรับเก็บข้อมูลของไมโครคอนโทรลเลอร์MCS-5นี้สามารถแบ่งออกเป็น 2 ส่วนคือ ในส่วนที่เป็นหน่วยความจำสำหรับเก็บข้อมูลภายในไอซี และหน่วยความจำสำหรับเก็บข้อมูลภายนอกไมโครคอนโทรลเลอร์ MCS-51 ทุก ๆ เบอร์จะมีหน่วยความจำเก็บข้อมูลทั่ว ๆ ไปภายในไอซีอย่างน้อยคือ 128 ไบต์ ไปจนถึง 256 ไบต์ ทั้งนี้ขึ้นกับเบอร์ของไอซี หน่วยความจำสำหรับเก็บข้อมูลภายในไอซีในบริเวณ 128 ไบต์เรียกว่า LOWER 128 และในบริเวณ 128 ไบต์หลังที่มีเพิ่มในบางเบอร์มีชื่อเรียกว่า UPPER



รูปที่ 2.19 แสดงการต่อกับหน่วยความจำข้อมูลภายนอกไอซี

2.4.5 รีจิสเตอร์ที่เกี่ยวกับไมโครคอนโทรลเลอร์ MCS-51

รีจิสเตอร์ในกลุ่มนี้ จะเป็นรีจิสเตอร์ขนาด 16 บิตที่ใช้งาน เพื่อเก็บข้อมูลของตัวแอดเดรสเป็นสำคัญโดยค่าที่อยู่ภายในแอดเดรสนี้จะนำไปเป็นค่าของข้อมูลที่ส่งออกไปทางบัสแอดเดรส ในส่วนของไมโครคอนโทรลเลอร์ เพื่อบอกตำแหน่งที่ต้องการติดต่อ รีจิสเตอร์ที่จัดในกลุ่มนี้ประกอบด้วยรีจิสเตอร์ใช้งานทั่วไป (GENERAL-PURPOSE REGISTERS) รีจิสเตอร์ในกลุ่มนี้จัดเป็นพื้นที่หน่วยความจำที่ใช้ในการสนับสนุนในการประมวลผล การทำงานจากหน่วยประมวลผลทางคณิตศาสตร์และลอจิก (ALU) เพื่อให้สามารถจัดการข้อมูลให้เร็วที่สุด นอกจากนี้โปรแกรมที่ไม่ได้ใช้คำสั่งเหล่านี้ก็ยังใช้เป็นการเก็บข้อมูลตัวแปรภายในโปรแกรม จะเห็นได้ว่าชื่อของรีจิสเตอร์ไม่ว่าจะอยู่ในรีจิสเตอร์แบงก์ใด ก็จะมีชื่อว่า R0 ถึง R7 เหมือนกันทั้งสิ้น ดังนั้นในการใช้งานผู้ใช้จะต้องให้ความระมัดระวังว่า ต้องการรีจิสเตอร์นั้นๆ จากแบงก์ใดๆ ซึ่งการกำหนดเลือกแต่ละกลุ่มของรีจิสเตอร์นี้ก็ทำได้ง่าย เพียงการกำหนดค่าของบิตที่อยู่ภายในแฟลก (PSW) เท่านั้นอย่างไรก็ตาม โดยทั่วไปก็มักจะมีการใช้งานรีจิสเตอร์ R0 ถึง R7 เฉพาะในแบงก์ 0 เท่านั้น ดังนั้นพื้นที่ของแบงก์อื่นๆ ที่เหลือก็สามารถนำมาใช้ในลักษณะของหน่วย ความจำแรม



รูปที่ 2.20 แสดงการเลือกใช้รีจิสเตอร์ใช้งานทั่วไป R0-R7 ในแต่ละกลุ่ม

- รีจิสเตอร์หน้าที่พิเศษ เป็นรีจิสเตอร์หน้าที่พิเศษ (SFR) เป็นรีจิสเตอร์สำหรับ การควบคุมหน้าที่ และการทำงานของอุปกรณ์หรือพอร์ตของไมโครคอนโทรลเลอร์ MCS-51 ทั้งหมดตำแหน่งของรีจิสเตอร์เหล่านี้จะจัดอยู่ในบริเวณแอดเดรส 80H–FFH การใช้งานรีจิสเตอร์หน้าที่พิเศษเหล่านี้สามารถทำได้ทั้งการระบุชื่อของรีจิสเตอร์ หรือตำแหน่งแอดเดรส ที่เป็นของรีจิสเตอร์นั้นก็ได้การจัดพื้นที่หน่วย ความจำสำหรับรีจิสเตอร์หน้าที่พิเศษเหล่านี้ โดยมีข้อสังเกตว่ารีจิสเตอร์ที่อยู่ในตำแหน่งแอดเดรสที่มีจำนวนเป็นทวีคูณของค่า 8 จะสามารถอ้างถึงในระดับบิตได้ด้วย (นั่นคือแอดเดรส 80H 88H 90H A0H A8H B0H B8H D0H E0H และ F0H)

- แอกคิวมูเลเตอร์ (ACCUMULATOR) หรือ ACC เป็นรีจิสเตอร์ขนาด 8 บิต ทำหน้าที่ในการเก็บข้อมูลที่จะส่งให้กับหน่วยทำงานภายในหน่วยประมวลผลกลาง และเก็บผลลัพธ์ที่ได้จากการทำงานเท่านั้น การทำงานของรีจิสเตอร์นั้นมีลักษณะเช่นเดียวกับตัวแอกคิวมูเลเตอร์ของโปรเซสเซอร์ทั่วไป การใช้งานในโปรแกรมซึ่งใช้เรียกเป็น รีจิสเตอร์ A

2.4.6 ชุดคำสั่งของไมโครคอนโทรลเลอร์ MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 ประกอบด้วยคำสั่งทั้งหมดจำนวนมาก ซึ่งสามารถจะจัดกลุ่มคำสั่งเหล่านี้ตามลักษณะและหน้าที่การทำงานที่คล้ายคลึงกัน เพื่อความสะดวกต่อการศึกษาทำความเข้า ใจและใช้งาน ดังนี้

2.4.6.1 กลุ่มการถ่ายเทข้อมูล คือ กลุ่มคำสั่งในการโอนย้ายข้อมูล ทำหน้าที่ในโอนย้ายข้อมูลระหว่างรีจิสเตอร์ หรือหน่วยความจำภายในแรม โดยมีรายละเอียดดังนี้ ชุดคำสั่งในการถ่ายเทแรม ภายในนั้น ซึ่งเวลาที่ใช้ในหนึ่งคำสั่งนั้นจะเป็นเวลา เมื่อขณะที่ความถี่ในการทำงานของหน่วยประมวลผลกลางที่ความถี่ 12 เมกะเฮิรตซ์ และรายละเอียดของแต่ละคำสั่งมีดังนี้ MOV :จะทำงานในลักษณะเป็นการถ่ายเทข้อมูลที่มีขนาดเป็นไบต์ หรือ บิตก็ได้ จากแหล่งกำเนิดเข้าสู่ตัวรับข้อมูลในฟิลด์ โอเปอร์แรนด์ PUSH:จะทำงานโดยเพิ่มค่ารีจิสเตอร์ SP ก่อนแล้วจึงทำการถ่ายเทข้อมูล 1 ไบต์จากแหล่ง กำเนิดไปบริเวณสแต็กตามตำแหน่งที่รีจิสเตอร์ SP กำหนด POP:การถ่ายเทข้อมูลขนาด 1ไบต์ จากบริเวณตำแหน่งที่รีจิสเตอร์ SP กำหนดไปยังรีจิสเตอร์ที่โอเปอร์แรนด์ กำหนดและหลังจากนั้นรี จิสเตอร์ SP จะลดค่าลง XCH:คำสั่งแลกเปลี่ยนไบต์ระหว่างแหล่งกำเนิดโอเปอร์แรนด์กับรีจิสเตอร์ AXCHD คำสั่งในการแลกเปลี่ยนขนาดนิบเบิลทางอันดับต่ำของแหล่งกำเนิดโอเปอร์แรนด์กับนิบเบิลอันดับต่ำลงของแอกคิวมูเลเตอร์ ตัวอย่างเช่น ทำการเลื่อนข้อมูลไป 2 ไบต์ทางขวามือ ซึ่งจะมี 2 วิธีคือ ใช้คำสั่ง MOV หรือใช้คำสั่ง XCH รายละเอียดการใช้คำสั่งทั้ง 2 แบบ

2.4.6.2 กลุ่มคำสั่งทางคณิตศาสตร์ เช่น การบวก ลบ คูณ และหารข้อมูลภาย ในตัวรีจิสเตอร์ต่างๆ ช่วงเวลาการทำงานของแต่ละคำสั่งนั้นจะกำหนดที่ความถี่ของสัญญาณนาฬิกาที่ 12 เมกะเฮิรตซ์ คำสั่งทางคณิตศาสตร์ส่วนใหญ่ใช้เวลา 1 ms ยกเว้นคำสั่ง INC DPTR ซึ่งใช้เวลา 2 ms โดยที่คำสั่งการคูณและหารใช้เวลา 4 ms โดยมีรายละเอียดการใช้คำสั่งมีดังนี้ INC:เป็นการบวกหนึ่งกับโอเปอร์แรนด์และใส่ค่าใหม่กลับเข้าที่ตัวโอเปอร์แรนด์นั้น ๆ DEC:เป็นการลบออกจากตัวเลขที่อยู่ในแหล่งกำเนิด โอเปอร์แรนด์ และนำผลลัพธ์ที่ได้มาเก็บไว้ที่ตัวโอเปอร์แรนด์นั้น ADD:เป็นการบวกในแอกคิวมูเล-เตอร์เข้ากับค่าในแหล่งกำเนิดโอเปอร์แรนด์ ADDC:เป็นการบวกค่าต่างๆ ในแอกคิวมูเลเตอร์เข้ากับค่าในแหล่งกำเนิดโอเปอร์แรนด์และบวกกับบิตทดด้วย SUBB: เป็นการนำเลขที่แหล่งกำเนิดโอเปอร์แรนด์ ลบออกจากตัวเลขในA และนำค่าบิตตัวทดมาลบออกอีกและผลลัพธ์ที่ได้นำมาใส่ลงในแอกคิวมูเลเตอร์ A MUL:เป็นการคูณแบบไม่คิดตัวเครื่องหมายของตัวเลขที่อยู่ใน แอกคิวมูเลเตอร์กับ เลขในรีจิสเตอร์ B แล้วได้ผลลัพธ์ 2 ไบต์ นำมาเก็บไว้ที่ AB โดย A จะรับอันดับต่ำส่วน B จะรับอันดับสูง DIV:เป็นคำสั่งในการหารแบบไม่คิดเครื่องหมายที่อยู่ในแอกคิวมูเลเตอร์แล้วหารตัวเลขในรีจิสเตอร์ B แล้วนำผลลัพธ์ไปเก็บในแอกคิวมูเลเตอร์และเศษของการหารตัวเลขจะเก็บไว้ในรีจิสเตอร์ B DA:สำหรับการบวกกันทางตัวเลข BCD เป็นการปรับค่ารวมซึ่งเป็นผลมาจากการบวกกันทางไบนารี่ของระบบตัวเลข BCD ขนาด 2 หลักสองจำนวนการปรับค่าตัวเลขผลรวมด้วยการใช้คำสั่ง DA จะได้ผลลัพธ์กลับมาที่แอกคิวมูเลเตอร์รายละเอียดการใช้คำสั่ง

2.4.6.3 กลุ่มคำสั่งทางตรรกศาสตร์หรือแบบลอจิก ทำหน้าที่เกี่ยวกับการประมวลผลแบบลอจิก ต่างๆ เช่นการ AND OR หรือ EX-OR ระหว่างข้อมูลในรีจิสเตอร์ A นั่นเองโดยมีการใช้คำสั่งดังนี้ CPL:เป็นการใช้คำสั่งกลับค่าหรือคอมพลีเมนต์ ข้อมูลในแอกคิวมูเลเตอร์จะไม่มีผลใดๆต่อค่าของแฟลก หรือการอ้างถึงตำแหน่งแอดเดรสนั้นตามบิตนั้นๆ RL, RLC, RR, RRC, SWAP: ทั้ง 5 คำสั่งนี้เป็นคำสั่งในการทำงานการวนบิตบนตัวของแอกคิวมูเลเตอร์ ซึ่ง RL เป็นการวนบิตทางขวา, RLC เป็นการทำการวนทางซ้ายผ่านบิตทด , RRC เป็นการวนขวาผ่านบิตทด และ SWAP เป็นการวนซ้ายสี่ครั้ง ANL:เป็นการ ADD กันทางตรรกศาสตร์ ระหว่างแหล่งกำเนิดสองโอเปอร์แรนด์ ซึ่งจะสั่งให้ทำงานในรูปแบบของตรรกศาสตร์ทางข้อมูลขนาดเป็นไบต์หรือบิตกลุ่มคำสั่งแบบบูลีนหรือแบบบิตซึ่งเป็นความสามารถ ของไมโครคอนโทรลเลอร์ MCS-51 ที่จะดำเนินการประมวลผลแบบบิตแทนที่จะเป็นข้อมูล ทั้งไบต์เช่นปกติ โดยมีชุดคำสั่งที่จัดการโดยตรง ทุกคำสั่งจะเข้าถึงข้อมูลโดยตรงในระดับบิต โดยมีการบิตแอดเดรสได้ตั้งแต่ 00H - 7FH ในพื้นที่ 128 บิต หน่วยความจำข้อมูลภายในและบิตแอดเดรส 80H-FFH ในบริเวณกลุ่มรีจิสเตอร์ฟังก์ชั่นพิเศษ (SFR)

2.4.6.4 กลุ่มคำสั่งในการกระโดดไปยังตำแหน่งต่างๆภายในโปรแกรม ซึ่งจะเปลี่ยนลำดับของการประมวลผลภายในโปรแกรมไปยังส่วนต่างๆแทนที่จะดำเนินการไปเป็นลำดับ ต่อเนื่อง โดยที่คำสั่ง JMP จะแบ่งเป็น 3 ลักษณะ คือ SJMP, LJMP, AJMP ซึ่งในแต่ละคำสั่งจะมีข้อแตกต่างของการกระโดดไปยังแอดเดรสไกลสุดที่ต่างกันคำสั่ง JMP ซึ่งเป็นแบบโมนีซิกที่สามารถจะใช้ได้โดยมีรายละเอียดการใช้งานของคำสั่งดังต่อไปนี้ SMP:จะเป็นการกระโดดแบบการย้ายอันดับตำแหน่งของแอดเดรสตำแหน่งเดิมซึ่งจะสามารถกระโดดได้ -128 ถึง +127 ไบต์ AJMP:ลักษณะแบบนี้จะสามารถกระโดดได้ไกลสุดประมาณ 2 กิโลไบต์ ซึ่งจะใช้หน่วยความจำเพียง 2 ไบต์เท่านั้น ในการกำหนด LJMP:ลักษณะแบบนี้จะสามารถกระโดดได้ไกลสุดประมาณ 64 กิโลไบท์ ซึ่งจะใช้หน่วยความจำเพียง 3 ไบต์เท่านั้นในการกำหนด JMP @A+DPTR:เป็นการควบคุมการกระโดดไปยังโปรแกรมที่ต้องการเฉพาะภายในส่วนต่างๆ

2.4.7 โครงสร้างการอินเตอร์รัปต์ของไมโครคอนโทรลเลอร์ MCS-51

จากแผนภาพโครงสร้างระบบอินเตอร์รัปต์ของไมโครคอนโทรลเลอร์ MCS-51 สัญญาณที่เข้ามาทำ การอินเตอร์รัปต์ MCS-51 นั้นเกิดขึ้นได้ 5 ลักษณะตามตารางข้อมูล ในรูปที่ 2.6 โดยจะเห็นได้ว่าสามารถที่จะกำหนดเลือกเพื่อยินยอม (หรืออีนาเบิล : ENABLE) และห้าม (หรือดิสเอเบิล:DISABLE) ไม่ให้มีการอินเตอร์รัปต์แต่ละประเภทได้ โดยการกำหนดบิตของข้อมูลที่เกี่ยวข้อง ซึ่งมักจะอยู่ภายในรีจิสเตอร์ TCONและSCON นอกจากนี้ยังมีตำแหน่งบิตภายในรีจิสเตอร์ IE(INTERRUPT ENABLE REGISTER) ซึ่งทำหน้าที่เสมือนกับเป็นสวิตซ์หลักที่เกี่ยวข้องกับสัญญาณอินเตอร์รัปต์ทั้งหมด หากว่ากำหนดไม่ให้เกิดการอินเตอร์รัปต์ แล้วการกำหนดบิตเพื่อห้ามหรือยินยอมของแต่ละอินเตอร์รัปต์ก็จะไม่มีผลใดๆ เกิดขึ้นยังแสดงให้เห็นว่าสัญญาณอินเตอร์รัปต์ แต่ละประเภทยังสามารถกำหนดระดับความสำคัญ (PRIORITY) ของการอินเตอร์รัปต์ได้สองลักษณะ คือระดับความสำคัญสูงหรือต่ำ (HIGH OR LOW PRIORITY) กล่าวคือ ขณะที่กำลังประมวลผลอยู่ภายในส่วนของโปรแกรมย่อยบริการอินเตอร์รัปต์ ของสัญญาณที่มีระดับความสำคัญต่ำอยู่ ก็อาจจะถูกขัดจังหวะให้ไปประมวลผลของสัญญาณอินเตอร์-รัปต์ที่มีระดับความสำคัญสูงกว่า แต่หากว่าเป็นสัญญาณอินเตอร์รัปต์ที่มีระดับความสำคัญต่ำเช่นเดียวกันแล้วก็ต้องรอให้เสร็จสิ้นการประมวลผลที่ ดำเนินการอยู่ก่อน

2.4.8 การรีเซต

โดยความหมายของการรีเซตเป็นการบังคับให้มีการเริ่มต้นใหม่อีกครั้งหนึ่ง ซึ่ง มักจะกระทำโดยการกำหนดสภาวะของสัญญาณที่ขารีเซตของไอซี MCS-51ให้เป็นระดับลอจิกที่เหมาะสมเท่านั้น การรีเซตด้วยวิธีนี้ถือว่าเป็นการอินเตอร์รัปต์อย่างหนึ่งได้ แต่จะมีลักษณะต่างออกไปจากการอินเตอร์รัปต์ของสัญญาณนี้ได้ ซึ่งมีศัพท์เฉพาะเรียกว่า NON-MASKABLE INTERRUPT นอกจากนี้การดำเนินการของโปรแกรมก็แตกต่างออกไปด้วย โดยจะไม่มีการเก็บค่าของคำสั่งที่กำลังจะไปทำในลำดับต่อไปภายในรีจิสเตอร์ PC เมื่อมีการรีเซตเกิดขึ้นโปรแกรมจะถูกสั่งให้กระโดดไปยังแอดเดรส 0000 ทันที ซึ่งตำแหน่งนี้จะเป็นตำแหน่งเริ่มต้นของการทำงานของไมโครคอนโทรลเลอร์ MCS-51 เมื่อเริ่มจ่ายไฟให้กับระบบเมื่อใดก็ตามที่มีการรีเซตเกิดขึ้นค่าสภาวะต่างๆ ภายในไมโครคอนโทรลเลอร์จะถูกกำหนดกลับไปเป็นค่าเริ่มต้นใหม่อีกครั้ง

2.5 การใช้งานไมโครคอนโทรลเลอร์กับพอร์ตอนุกรม

การติดต่อกับพอร์ตอนุกรมนั้นจะยากกว่าการ ติดต่อกับกับพอร์ตขนาน การสื่อสารของอุปกรณ์ที่ต่อกับพอร์ตอนุกรมจะถูกเปลี่ยน (Convert) เป็นสัญญาณแบบขนาน แล้วจึงนำไปประมวลผลต่อ ซึ่งจะใช้ Universal Asynchronous Receive / Transmitter (UART) เป็นตัวทำหน้าที่ส่วนทางด้านโปรแกรม ก็มีรีจิสเตอร์ที่ต้องจัดการมากกว่า Standard Parallel Port (SPP) อีกหลายตัว

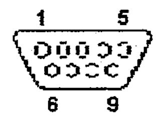
ข้อดีของพอร์ตอนุกรม

ระยะของสายอนุกรมสามารถมีความยาวได้มากกว่าสายของขนานมาก ทั้งนี้เพราะสัญญาณของพอร์ตอนุกรม ซึ่งส่วนใหญ่ใช้มาตรฐาน RS-232C จะมีค่า –3 Volt ถึง –15 Volt สำหรับ Logic “1” หรือ “Mark” และมีค่า +3 Volt ถึง +15 Volt สำหรับ Logic “0” หรือ “Space” (สำหรับช่วง +3 Volt ถึงถึง -3 Volt เป็นช่วง Undefined) ส่วนสัญญาณของสายขนานนั้น Logic “1” จะมีค่า +5 Volt และ logic “0” จะมีค่า 0 Volt ทำให้สัญญาณของสายอนุกรมสามารถรับการสูญเสียของสาย (Cable loss) ได้มากกว่าสัญญาณของสายขนานปกติสายขนานจะไปได้เพียง 5 ฟุต ส่วนสาย RS-232 จะไปได้ถึง 50 ฟุตที่ความเร็วสูงสุดของมัน

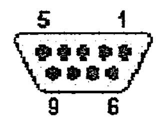
สายอนุกรมจะใช้จำนวนสายไฟน้อยกว่าสายขนานถ้าต่อในลักษณะ Null Modemจะใช้สายเพียง 3 เส้น ขณะที่แบบขนาน จะต้องใช้สาย 19 ถึง 25 เส้น

การสื่อสารแบบไร้สายเช่นการใช้ Infra Red การส่งพร้อมกันทีละ 8 บิต แบบขนาน จะทำให้ไม่สามารถแยกแยะได้ว่า Bit ใดเป็น Bit0 หรือ Bit1 ... เป็นต้น ปัจจุบันอุปกรณ์ IrDA มีความเร็วไม่ต่ำกว่า 115.2K Baud แต่มี Pulse length เพียง 3/16 ของ RS-232 เพื่อประหยัดพลังงาน เพราะส่วนมากใช้ในอุปกรณ์แบบพกพาเช่น Laptop หรือ Palmtop

ปัจจุบันไมโครคอนโทรลเลอร์ มักมีการผนวกพอร์ตการสื่อสารแบบอนุกรมไว้ด้วย เพราะใช้จำนวนขาน้อยกว่าแบบขนาน RS-232C กำหนด Baud rate ไว้ไม่เกิน 20K Baud ปัจจุบันได้แก้ไขให้รองรับกับเทคโนโลยีใหม่ได้ จึงมีการปรับปรุงถึง RS-232E ซึ่งมีรายละเอียดอีกหลายอย่าง



(Male at the computer side)



(Female at the cable side)

Pin	Name	RS232	V.24	Dir	Description
1	CD	CF	109	←	Carrier Detect
2	RXD	BB	104	←	Receive Data
3	TXD	BA	103	→	Transmit Data
4	DTR	CD	108.2	→	Data Terminal Ready
5	GND	AB	102	—	Signal Ground
6	DSR	CC	107	←	Data Set Ready
7	RTS	CA	105	→	Request to Send
8	CTS	CB	106	←	Clear to Send
9	RI	CE	125	←	Ring Indicator

รูปที่ 2.21 ตารางแสดงถึงหน้าที่ของขาต่าง ๆ

Abbreviation	Full Name	Originator	Function
TD	Transmit Data	DTE	Serial data output (TXD) from DTE.
RD	Receive Data	DCE	Serial data input (RXD) to DTE.
CTS	Clear To Send	DCE	Tell DTE that DCE is ready to exchange data.
(D)CD	(Data) Carrier Detect	DCE	Carrier from remote DCE is detected.
DSR	Data Set Ready	DCE	Tell DTE that DCE is ready to establish a link.
DTR	Data Terminal Ready	DTE	Tell DCE that DTE is ready to establish a link.
RTS	Ready To Send	DTE	Tell DCE that DTE is ready to exchange data.
RI	Ring Indicator	DCE	Ringing signal from the phone line is detected.

รูปที่ 2.22 แสดงสัญญาณต่างๆที่ส่งในรูปแบบอนุกรม

2.5.1 หน้าที่ของสัญญาณต่าง ๆ มีดังนี้

Protective ground เป็นจุดที่ต่อกับตัวเปลือกของอุปกรณ์ และไม่ต่อกับสัญญาณใด ๆ ในระบบเพื่อใช้ต่อลงดิน เป็นการป้องกันอันตรายจากไฟลัดวงจร และใช้เป็น Shield ป้องกันการรบกวน

Signal ground หรือ Common return เป็นจุดสำคัญที่สุดที่ต้องมีในระบบ RS-232 เพราะเป็นจุดอ้างอิงของทุกสัญญาณ ยกเว้น Protective ground

Transmit data เป็นสัญญาณข้อมูลที่ส่งจาก DTE ไปยัง DCE ในขณะที่ไม่ได้ส่งข้อมูลมันจะมีสถานะเป็นLogic "1" หรือ "Mark" หรือ "Off" หรือ -5 V ถึง -15 V ที่ด้านส่ง (DTE) หรือ -3 V ถึง -15 V ที่ด้านรับ (DCE) การส่งข้อมูลจะเกิดขึ้นได้ต้องมีสัญญาณควบคุมที่เกี่ยวข้อง "On" ก่อน คือสัญญาณ RTS, CTS, (D)CD, DTR และ DSR

Receive data เป็นสัญญาณข้อมูลจาก DCE มายัง DTE ในขณะที่ไม่ได้ส่งข้อมูลมันจะมีสถานะเป็น Logic "1" หรือ "Mark" หรือ "Off" หรือ -5 V ถึง -15 V ที่ด้านส่งหรือ -3 V ถึง -15 V ที่ด้านรับ ในขณะที่ไม่ได้ส่งข้อมูลมันจะมีสถานะเป็น Logic "1" หรือ "Mark" หรือ "Off" หรือ -5 V ถึง -15 V ที่ด้านส่ง (DCE) หรือ -3 V ถึง -15 V ที่ด้านรับ (DTE) กรณีที่เป็นการสื่อสารแบบ Half-duplex สัญญาณ RD จะอยู่สถานะ "Off" ขณะที่สัญญาณ RTS อยู่ในสถานะ "On" และสัญญาณ RD จะยังคงอยู่ในสถานะ "Off" อีกชั่วระยะหนึ่งหลังจากที่สัญญาณ RTS เปลี่ยนจากสถานะ "On" มาเป็น "Off" แล้วเพื่อให้การรับ-ส่งข้อมูลเสร็จสมบูรณ์

Request to send เป็นสัญญาณจากDTE ส่งไปให้DCEเพื่อขอส่งข้อมูลไปปรกติจะอยู่ในสถานะ "Off" เมื่อต้องการส่งข้อมูลจะเปลี่ยนเป็นสถานะ "On" จนกว่าการส่งเสร็จสิ้นจึงเปลี่ยนกลับมาที่สถานะ "Off" ตามเดิม ทั้งนี้ทาง DTE ต้องได้รับสัญญาณ CTS จึงจะสามารถส่งข้อมูล TD ไปยัง DCE ได้ และสัญญาณ RTS ที่กลับสู่สถานะ "Off" จะไม่สามารถเปลี่ยนเป็น "On" ใหม่ขณะที่สัญญาณ CTS อยู่ในสถานะ "On" ต้องรอจนกว่าสัญญาณ CTS เปลี่ยนมาอยู่ในสถานะ "Off" ก่อน เพื่อป้องกันการเกิด Overrun

Clear to send เป็นสัญญาณที่ DCE ส่งให้ DTE เพื่อแจ้งว่าพร้อมรับการส่งข้อมูลจาก DTE

Data set ready เป็นสัญญาณที่ DCE ส่งให้ DTE เพื่อแจ้งว่า DCE สามารถเชื่อมโยงไปยังปลายทางและพร้อมที่จะติดต่อแล้ว

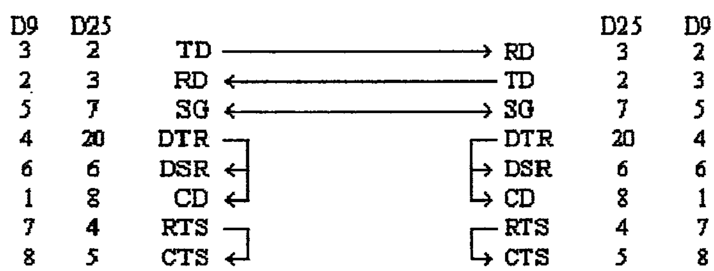
Data Terminal ready เป็นสัญญาณที่ DTE ส่งให้ DCE เพื่อแจ้งว่า DTE พร้อมหรือต้องการจะติดต่อสื่อสาร ซึ่งสัญญาณ DTR นี้ต้องเกิดก่อนทาง DCE จึงจะทำการติดต่อไปยังปลายทางและเมื่อติดต่อได้แล้วจึงส่งสัญญาณ DSR มายัง DTE เพื่อแจ้งให้รู้ว่าพร้อมรับการสื่อสารแล้ว และถ้า DTR เปลี่ยนเป็น Off แปลว่า DTE ไม่ต้องการติดต่อสื่อสารแล้วทาง DCE ก็จะปิดช่องสื่อ-สารและเปลี่ยนสัญญาณ DSR เป็น Off ทันที คู่สัญญาณระหว่าง RTS กับ CTS เป็นเรื่องของความพร้อมเกี่ยวกับช่องสื่อสารระหว่าง DTE กับ DCE ส่วนคู่สัญญาณ DTR กับ DSR เป็นเรื่องของความพร้อมเกี่ยวกับตัวอุปกรณ์ DTE กับ DCE

Data carrier detect เป็นสัญญาณที่ DCE ส่งให้ DTE เพื่อแจ้งว่าได้รับสัญญาณพาหะจาก DCE ที่อยู่อีกด้านหนึ่งของการสื่อสาร ซึ่งหมายความว่า ช่องการสื่อสารระหว่าง DCE ทั้ง 2 ไม่ขาดตอน พร้อมที่จะทำการสื่อสารได้ ซึ่งอุปกรณ์ DTE หรือโปรแกรมที่ควบคุมการสื่อสารมักจะตรวจ สัญญาณนี้ ถ้าไม่อยู่ที่ on แสดงว่าช่องการสื่อสารขาด ก็จะไม่ทำการรับหรือส่งข้อมูล

Ring Indicator เป็นสัญญาณจาก DCE แจ้งให้ DTE รู้ว่ามีการเรียกจาก DCE ที่อยู่อีกด้านหนึ่งของการสื่อสาร ซึ่งมักจะใช้ในระบบ automatic answering

2.5.2 Null Modems

Null Modem ใช้สำหรับเชื่อมโยงระหว่าง DTE 2 ตัวเข้าด้วยกันโดยตรง ซึ่งโดยมากใช้ในการถ่ายข้อมูลระหว่างคอมพิวเตอร์ หรือใช้ในการพัฒนาระบบไมโครโปรเซสเซอร์



รูปที่ 2.23 แสดงการเชื่อมต่อของสายสัญญาณ

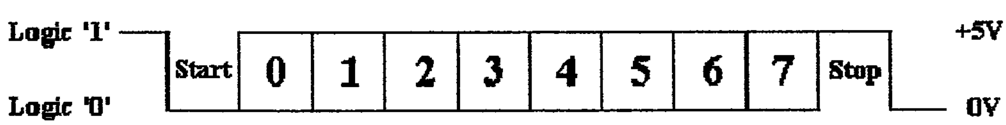
2.5.3 Flow Control

การติดต่อระหว่าง DTE และ DCE จะมีการควบคุมการไหลของข้อมูล เพื่อไม่ให้เกิดกา over flow ขึ้นได้ ซึ่งมีอยู่ 2 แบบคือ Hardware flow control และ Software flow control
สำหรับ Software flow control มักจะเรียกว่า Xon/Xoff flow control ซึ่งใช้รหัส ASCII 17 เป็นสัญญาณ Xon และใช้รหัส ASCII 19 เป็นสัญญาณ Xoff หลักการทำงานก็ง่าย ๆ คือ Modem จะมี Buffer อยู่เมื่อ Modem รับข้อมูลจาก Computer จน Bufferใกล้จะเต็มมันก็จะส่งสัญญาณ Xoff ไปให้ Computer เพื่อให้ Computer หยุดส่งข้อมูลให้มันชั่วคราว และเมื่อ Buffer มีที่ว่างถึงระดับหนึ่ง Modem ก็จะส่งสัญญาณ Xon ไปให้ Computer เพื่อให้ Computer ส่งข้อมูลให้มันต่อการควบคุมโดยวิธีนี้ประหยัดสาย สัญญาณ เพราะรับ-ส่งผ่าน TD และ RD แต่อาจทำให้การสื่อสารช้าลงอย่างเห็นได้ชัดในกรณีที่ใช้กับการสื่อสารที่มี speed ต่ำ เพราะแต่ละตัวอักษร ASCII ที่รับ-ส่งจะมีขนาด 10 Bit

ส่วน Hardware flow control มักจะเรียกว่า RTS/CTS flow control จะใช้สายสัญญาณของ Serial Port ในการควบคุม ทำให้ไม่บั่นทอนความเร็วของข้อมูล หลักการทำงานคือ เมื่อ Modem มีที่ว่างเพื่อรับข้อมูล มันก็จะส่งสัญญาณ CTS ไปให้ Computer และเมื่อมันใกล้จะเต็ม มันก็จะหยุดส่งสัญญาณ CTS ไปให้ Computer

2.5.4 รูปคลื่น สัญญาณ RS-232

การสื่อสารโดย RS-232 เป็นการสื่อสารแบบ asynchronous หมายความว่าสัญญาณ clock ที่ใช้ควบคุมจังหวะไม่ได้ส่งไปพร้อมกับ Data แต่จะใช้ start bit เป็นตัว sync.ในแต่ละ word ของการสื่อสารและใช้สัญญาณ clock ภายในของแต่ละด้านเป็นตัวให้จังหวะเอง



รูปที่ 2.24 รูปคลื่นของสัญญาณที่ส่ง

แสดงลักษณะของสัญญาณจาก UART เมื่อใช้ format แบบ 8N1 คือ 8 data bits ไม่มี parity bit และมี 1 stop bit ขณะที่ idle จะอยู่ในสถานะ "Mark" หรือ logic "1" การส่งจะเริ่มจากการส่ง start bit คือ logic "0" และตามด้วย LSB bit จนหมด data bits และถ้ามี parity bit ก็จะส่งที่จุดนี้แล้วลงท้ายด้วย stop bit ซึ่งมีค่าเป็น logic "1" ในรูปได้แสดง bit ที่ต่อถัดจาก stop bit ซึ่งมีค่าเป็น logic "0" หมายความว่าเป็น start bit ของ การส่ง word ถัดไป แต่ถ้ายังไม่มีการส่ง word ถัดไป ก็ต้องอยู่ในสถานะของ logic "1" ซึ่งเป็นสถานะของ idle และถ้าสายอยู่ในสถานะของ logic "0" นาน

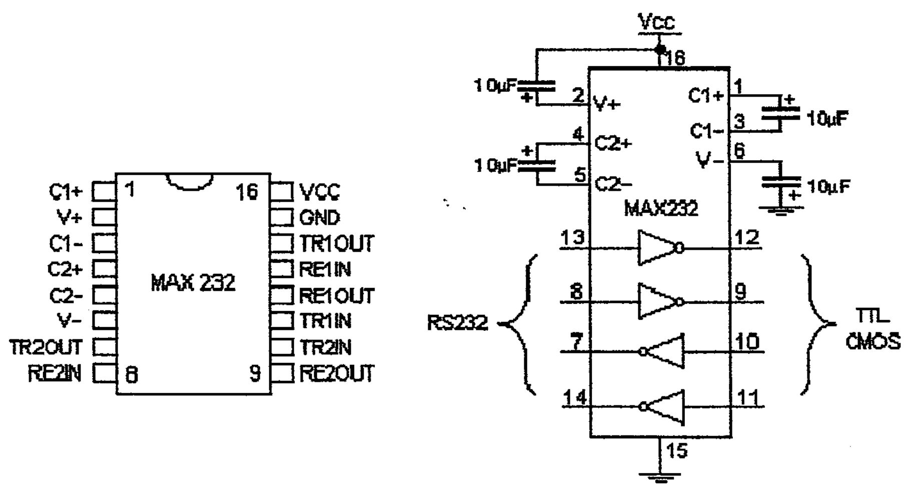
กว่าเวลาของการส่ง 1 full word ระบบจะถือว่าเป็นสัญญาณ "Break" เพื่อหยุดการสื่อสาร ดังนั้นต้องไม่ลืมที่จะสั่งในสายกลับสู่สถานะ idle เมื่อสิ้นสุดการส่ง การรับ-ส่งข้อมูลในลักษณะนี้เรียกว่าแบบ frame คือมีกรอบปิดล้อมข้อมูลไว้ด้วย start bit และ stop bit

2.5.5 ตัวแปลงสัญญาณ RS-232

สัญญาณ RS-232 มีค่าแรงไฟต่างจากที่ใช้ใน UART ดังนั้นจึงต้องมีตัวแปลงสัญญาณเพื่อแปลงระดับสัญญาณให้เหมาะสมก่อนที่ จะเชื่อมต่อกับพอร์ตอนุกรม หรือ RS-232 port ของคอมพิวเตอร์

สัญญาณ RS-232 นั้น logic "0" จะมีค่า +3 V ถึง +25 V และ logic "1" จะมีค่า –3 V ถึง –25 V ส่วนค่าระหว่าง –3 V ถึง +3 V เป็นค่า undefined ระดับสัญญาณนี้ใช้กับทุกสัญญาณไม่ใช่เฉพาะสัญญาณรับ-ส่งข้อมูลเท่านั้นแต่ยังรวมถึงสัญญาณควบคุมต่าง ๆ เช่น DTR, RTS, CTS เป็นต้น

IC ที่ใช้มักจะเป็นเบอร์ 1488 (RS-232 Driver) และ 1489 (RS-232 Receiver) โดยภายในแต่ละตัวจะประกอบด้วย inverter 4 ตัวและต้องการไฟเลี้ยง 2 ชุดคือ +7.5 V ถึง +15 V และ –7.5 V ถึง –15 V ซึ่งอาจจะมีปัญหาในเครื่องที่มีไฟเลี้ยง +5 V เพียงชุดเดียว แต่ก็ยังมี IC อีกตัวหนึ่งคือเบอร์ MAX-232 ซึ่งมีวงจร charge pump สามารถสร้างไฟ +10 V และ –10 V จากไฟ +5 V ได้พร้อมทั้งมี 2 Tx และ 2 Rx อยู่ใน package เดียวกัน และรองรับ baud rate ได้ถึง 120 Kbps จึงสะดวกมากเพราะใช้ IC เพียงตัวเดียว รูปข้างล่างคือ MAX-232



รูปที่ 2.25 แสดงโครสร้างภายในและตำแหน่งขาต่างๆของ Max232

ส่วนการที่เราจะนำข้อมูลมาใช้งานก็ต้องแปลงเป็น Parallel ก่อนซึ่งเป็นหน้าที่ของ UART ซึ่งปัจจุบันไมโครคอนโทรลเลอร์มักจะมี serial communication interface (SCI) อยู่ในตัว แต่อาจจะมีงานบางอย่างที่ไม่ได้ใช้ไมโครคอนโทรลเลอร์และต้องการประมวลผลข้อมูลกับพอร์ตอนุกรม เช่น ต่อ ADC เข้ากับ UART หรือต่อ LCD display เข้ากับ Serial comm. ก็ต้องใช้ UART ช่วย เช่น เบอร์ 8250 หรือ 16550A หรือเบอร์อื่น ๆ ที่ได้กล่าวมาแล้ว แต่มี UART อีกพวกหนึ่งที่แยก Tx bus กับ Rx bus ออกจากกันทำให้มีความคล่องตัวมากขึ้น

2.6 โปรแกรม Visual Basic (VB)

เป็นโปรแกรมสำหรับพัฒนาโปรแกรมประยุกต์ที่กำลังเป็นที่นิยมใช้อยู่ในปัจจุบัน โปรแกรม Visual Basic เป็นโปรแกรมที่ได้เปลี่ยนรูปแบบการเขียนโปรแกรมใหม่ โดยมีชุดคำสั่งมาสนับสนุนการทำงาน มีเครื่องมือต่าง ๆ ที่เรียกกันว่า คอนโทรล(Controls) ไว้สำหรับช่วยในการออกแบบโปรแกรม โดยเน้นการออกแบบหน้าจอแบบกราฟฟิก หรือที่เรียกว่า Graphic User Interface (GUI) ทำให้การจัดรูปแบบหน้าจอเป็นไปได้ง่าย และในการเขียนโปรแกรมนั้นจะเขียนแบบ Event - Driven Programming คือ โปรแกรมจะทำงานก็ต่อเมื่อเหตุการณ์ (Event) เกิดขึ้น ตัวอย่างของเหตุการณ์ได้แก่ ผู้ใช้เลื่อนเมาส์ ผู้ใช้กดปุ่มบนคีย์บอร์ด ผู้ใช้กดปุ่มเมาส์ เป็นต้น เครื่องมือ หรือ คอนโทรล ต่าง ๆ ที่ Visual Basic ได้เตรียมไว้ให้ ไม่ว่าจะเป็น Form TextBox Label ฯลฯ ถือว่าเป็นวัตถุ (Object ในที่นี้ขอใช้คำว่า ออบเจ็กต์) นั่นหมายความว่า ไม่ว่าจะเป็นเครื่องมือใด ๆ ใน Visual Basic จะเป็นออบเจ็กต์ทั้งสิ้น สามารถที่จะควบคุมการทำงาน แก้ไขคุณสมบัติของออบเจ็กต์นั้นได้โดยตรง ในทุกๆ ออบเจ็กต์จะมีคุณสมบัติ (properties) และเมธอด (Methods) ประจำตัว ซึ่งในแต่ละออบเจ็กต์ อาจจะมีคุณสมบัติและเมธอดที่เหมือน หรือต่างกันก็ได้ ขึ้นอยู่กับชนิดของออบเจ็กต ในการพัฒนาโปรแกรมประยุกต์ด้วย Visual Basic การเขียนโค้ดจะถูกแบ่งออกเป็นส่วนๆ เรียกว่า โพรซีเดอร์ (procedure) แต่ละโพรซีเดอร์จะประกอบไปด้วย ชุดคำสั่งที่พิมพ์เข้าไปแล้ว ทำให้คอนโทรลหรือออบเจ็กต์นั้น ๆ ตอบสนองการกระทำของผู้ใช้ ซึ่งเรียกว่าการเขียนโปรแกรมเชิงวัตถุ (Object Oriented Programming-OOP) แต่ตัวภาษา Visual Basic ยังไม่ถือว่าเป็นการเขียนโปรแกรมแบบ OOP อย่างแท้จริง เนื่องจากข้อจำกัดหลายๆ อย่างที่ Visual Basic ไม่สามารถทำได้

บทที่ 3
การออกแบบและการทำงาน

3.1 แนวทางการออกแบบ

3.1.1 จุดประสงค์ก่อนออกแบบ

1. วงจรภาครับสัญญาณจากการเคาะสามารถทำงานได้

2. ควบคุมให้สามารถวัดระยะทางระหว่างตัวรับกับจุดที่เคาะได้

3. นำผลต่างของเวลาที่รับได้ส่งเข้าไปที่ FPGA ไปเปรียบเทียบกับสัญญาณนาฬิกา เพื่อที่จะนำความผลต่างของสัญญาณนาฬิกานี้ไปหาพิกัด X, Y โดยใช้ MCS-51

3.1.2 ขั้นตอนการออกแบบ

ใช้หลักการของการเคลื่อนที่ของคลื่นที่ได้จากการเคาะเคลื่อนที่ไปบนผิวของวัตถุตัวกลาง โดยที่ปลายทางมีตัวรับคอยรับสัญญาณอยู่ โดยมีความสัมพันธ์กันทางหลักฟิสิกส์ที่ว่า ระยะเวลาที่คลื่นใช้เคลื่อนที่จากจุดหนึ่งไปอีกจุดหนึ่งสัมพันธ์กับระยะทาง $s = vt$

ดังนั้นหลักการการบอกตำแหน่งของการเคาะ บนจอโปรเจคเตอร์ มีหลักการทำงานดังนี้ เริ่มด้วย การเคาะบนจอที่สร้างขึ้นโดยแผ่นกระจก สัญญาณจากการเคาะจะถูกส่งไปที่ตัวรับที่ตำแหน่งระยะทางที่เรากำหนดไว้นั่นคือที่มุมทั้ง4ของบอร์ด โดยตัวรับจะมีอยู่ 4 ตำแหน่ง คือ P0,P1,P2,P3 ตามลำดับ ที่ตำแหน่ง $(X_0, Y_0,),(X_1, Y_1,),(X_2, Y_2)$ และ (X_3, Y_3) ตามลำดับ การหาพิกัด (X,Y) ทำได้โดยการวัดระยะทางที่อยู่ในรูปของเวลาระหว่างจุดที่เคาะยังจุดอ้างอิงทั้ง 4 คือ R+R0 , R+R1, R+R2 , และ R+R3 โดยใช้ FPGA เป็นตัวควบคุมการนับโดยใช้วงจรเคาน์เตอร์ หากตำแหน่งที่เคาะอยู่ใกล้ตัวรับใดระยะทางที่คลื่นเคลื่อนที่ไปยังตัวรับนั้นจะน้อยมีผลให้จำนวนที่นับของเคาน์เตอร์นั้นจะน้อย ค่าจำนวนข้อมูลดังกล่าวที่ได้จะส่งให้กับ MCS-51 เพื่อทำหน้าที่คำนวณค่าพิกัด X,Y จากนั้นก็จะส่งข้อมูล ไปยังคอมพิวเตอร์ นำไปควบคุมการทำงานของเมาส์ต่อไป

ภาครับสัญญาณจากการเคาะ

วงจรในส่วนของภาครับสัญญาณจากการเคาะ จะต้องสามารถรับสัญญาณการเคาะที่ได้จากเปียโซที่มีขนาดแรงดันแอมพลิจูดประมาณมากกว่า 200 mV เพื่อต้องการเวลาที่ถูกต้องจึงรับสัญญาณที่มีการเปลี่ยนแปลงลูกแรกเท่านั้น โดยวงจรภาครับ ทำหน้าขยายสัญญาณให้แรงขึ้นแล้ว

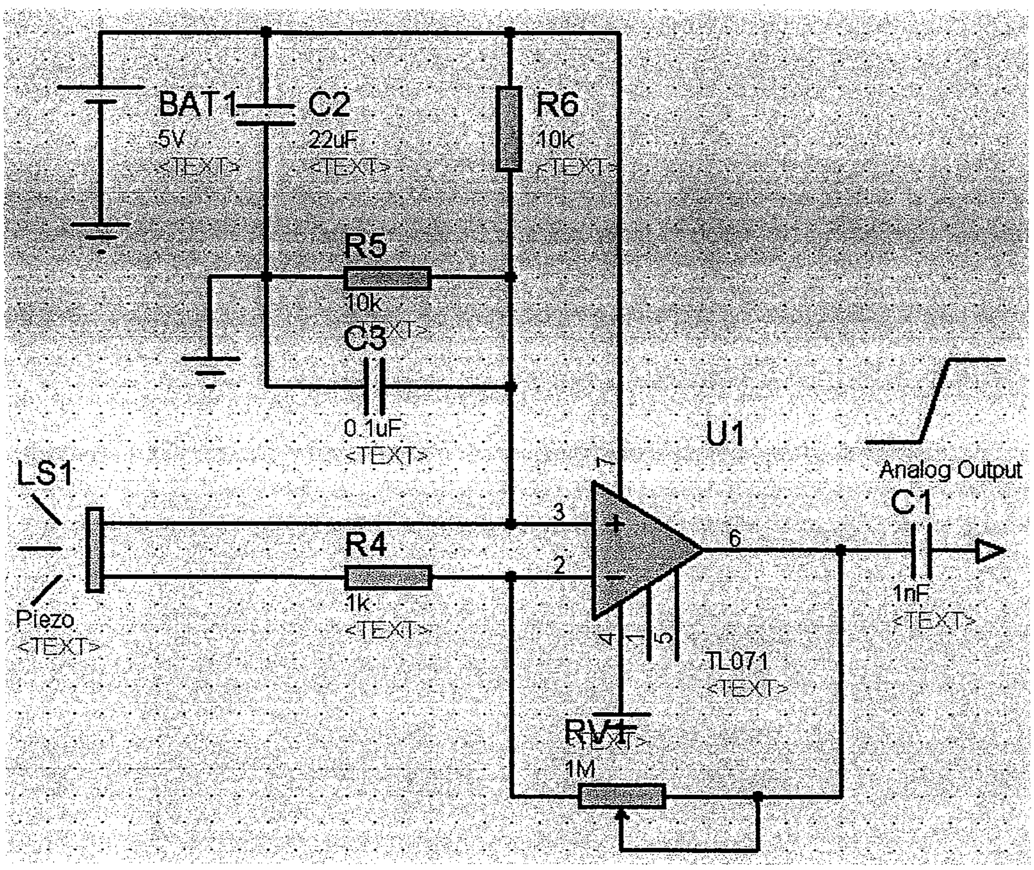
เปลี่ยนจากสัญญาณอนาล็อกให้เป็นสัญญาณดิจิตอล เพื่อส่งให้กับวงจรเกทใน FPGA เมื่อวงจรภาครับตัวใดตัวหนึ่งได้รับสัญญาณเป็นตัวแรก เคาน์เตอร์ตัวอื่นจะเริ่มนับและหยุดเมื่อวงจรภาครับชุดนั้นได้รับสัญญาณครบทั้ง 4 จุด หรือ รอจนถึงเวลาที่ได้คำนวณไว้ว่าถ้าเวลาผ่านไปเท่านี้ภาครับยังรับสัญญาณได้ไม่ครบให้ตัวรับเริ่มรับค่าใหม่ ค่าที่ได้จะนำไปคำนวณหาตำแหน่งของตัวชี้ต่อไป

ภาคประมวลผล

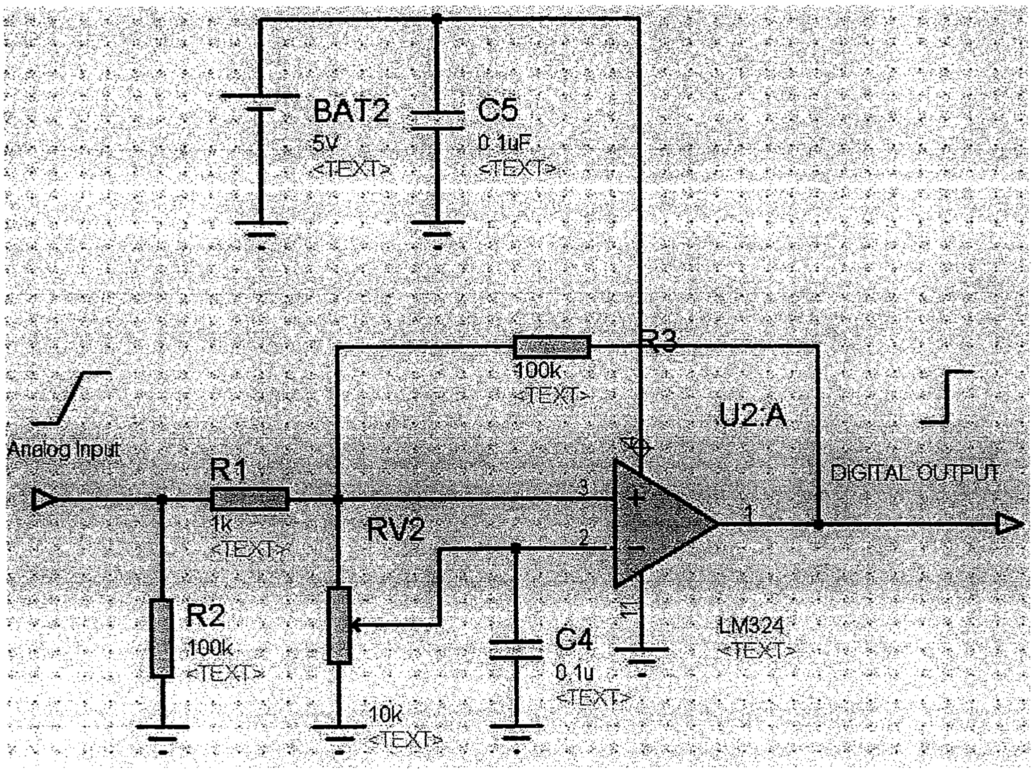
วงจรในส่วนของ FPGA ที่ทำหน้าที่รับสัญญาณจากวงจรภาครับ นำสัญญาณที่เป็นผลต่างของเวลามาเปรียบเทียบและนับระยะเวลาที่เกิดขึ้นระหว่างตัวส่งกับตัวรับแต่ละจุดโดยใช้วงจรเคาว์นเตอร์ทำการนับ ค่าเวลาที่ได้จะส่งให้กับ MCS-51 เพื่อทำหน้าที่ติดต่อกับ Visual Basic ผ่านทางพอร์ตสื่อสารแบบ RS-232 จากนั้นให้โปรแกรม Visual Basic ทำการคำนวณทางสมการคณิตศาสตร์ แล้วนำพิกัดระนาบ X, Y ที่ได้จากการคำนวณไปควบคุมการเคลื่อนที่ของเมาส์เพื่อสั่งงานหน้าจอวินโดว์ ต่อไป

3.2 ภาครับสัญญาณจากการเคาะ

หน้าที่ของวงจรภาครับสัญญาณจากการเคาะคือขยายสัญญาณให้มีความแรงขึ้นหลายๆเท่า จากนั้นก็ทำการเปรียบเทียบแรงดัน และให้สัญญาณดังกล่าวเปลี่ยนเป็นพัลส์รูปสี่เหลี่ยม แล้วหลังจากนั้นก็จะทำการส่งสัญญาณไปในส่วนของ FPGA ต่อไป



รูปที่ 3.1 แสดงวงจร Piezo Amplifier รับสัญญาณการเคาะ



รูปที่ 3.2 แสดงวงจร Comparator

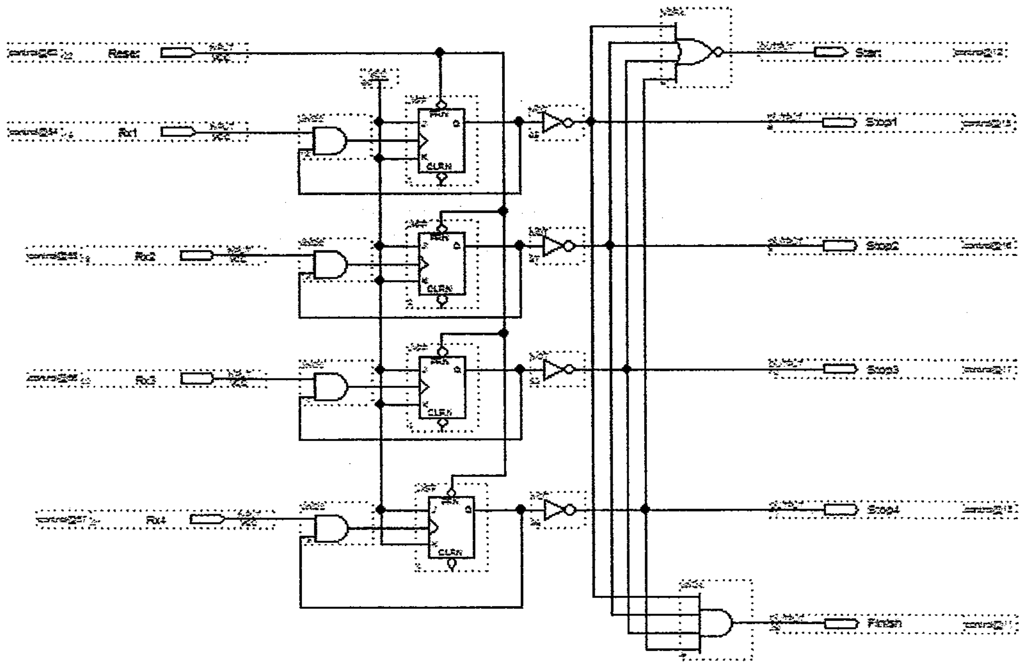
การทำงานในส่วนนี้สามารถแบ่งออกได้เป็น 2 ชุดด้วยกันคือ ชุดแรกได้แก่ภาครับสัญญาณเข้ามาแล้วทำการขยายสัญญาณ โดยในที่นี้ใช้ TL701 เป็นตัวขยายสัญญาณ input ที่รับเข้ามาเพื่อขยายสัญญาณ ส่วนชุดที่สองจะเป็นวงจร COMPARATOR LM324 ทำการเปรียบเทียบแรงดันจากภาคขยายดังกล่าว แล้วสัญญาณที่ออกจาก COMPARATOR จะนำไปรวมกันในส่วนของวงจร FPGA

3.3 หลักการทำงานของส่วน FPGA

ในส่วนของ FPGA นี้จะมีอยู่ 2 ส่วนที่สำคัญคือส่วนของภาค Control และ ภาค Counter แล้วนำทั้งสองส่วนนี้มาประกอบเข้าด้วยกัน โดยหลักการแล้วเราต้องการให้ส่วนของ FPGA นี้ทำหน้าที่นับผลต่างและแสดงผลต่างที่นับได้ในแต่ละภาครับที่รับจากการเคาะโปรเจคเตอร์ได้ที่เวลาต่างกัน ก็เพื่อนำส่วนของผลต่างทางเวลานี้ไปเป็นข้อมูลในการคำนวณหาพิกัด X, Y ต่อไปในส่วนของการคำนวณ

ภาค Control

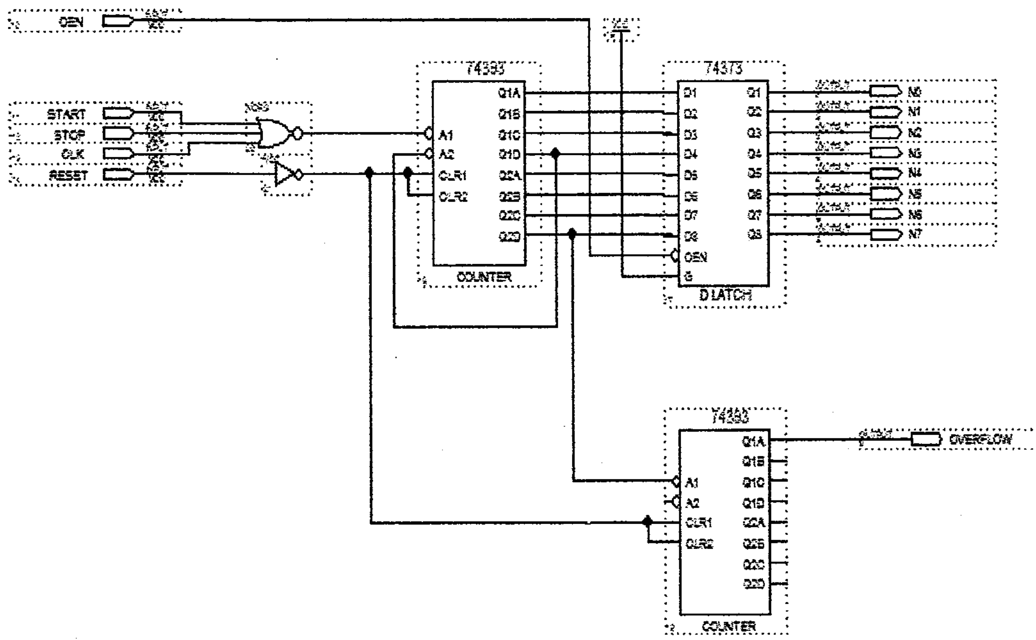
ภาคนี้จะทำหน้าที่ในการควบคุมการเริ่มนับและการหยุดนับของภาคCounter ในแต่ละส่วนของภาครับ ดังนั้นภาคนี้จึงต้องทำหน้าผลิตสัญญาณการเริ่มต้นการนับ และสัญญาณการหยุดนับ โดยจะต้องสัมพันธ์กับสัญญาณ Rx ในแต่ละตัวแต่ละภาครับที่จะรับคลื่นได้จากภาคส่ง



รูปที่ 3.3 แสดงภาพวงจรการทำงานภายใน Block Control

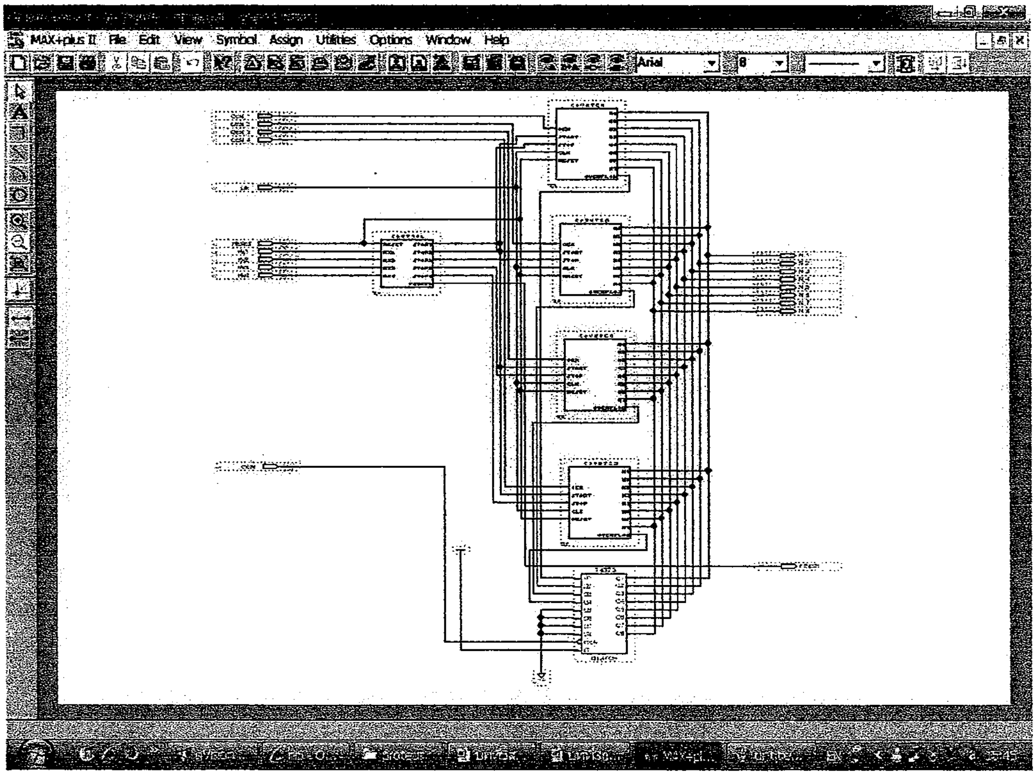
ภาค Counter

ภาคนี้จะทำหน้าที่หลักคือ นับผลต่างทางเวลาของสัญญาณ Rx ในแต่ละตัวในรูปของสัญญาณนาฬิกา ภาคนี้จะประกอบด้วย IC สองชนิดด้วยกันคือ 74393 คือ IC Counter ชนิดนับฐาน 16 8 บิต และ IC 74373 คือ IC ดีแลตช์ ดังนั้นภาคนี้จะนับได้ 0-255 ถ้าเกินนี้ก็จะส่งสัญญาณ Overflow ออกมาเสมือนว่าภาครับคลื่นภาคนั้นไม่สามารถที่จะรับคลื่นที่ส่งมาจากภาคส่งได้



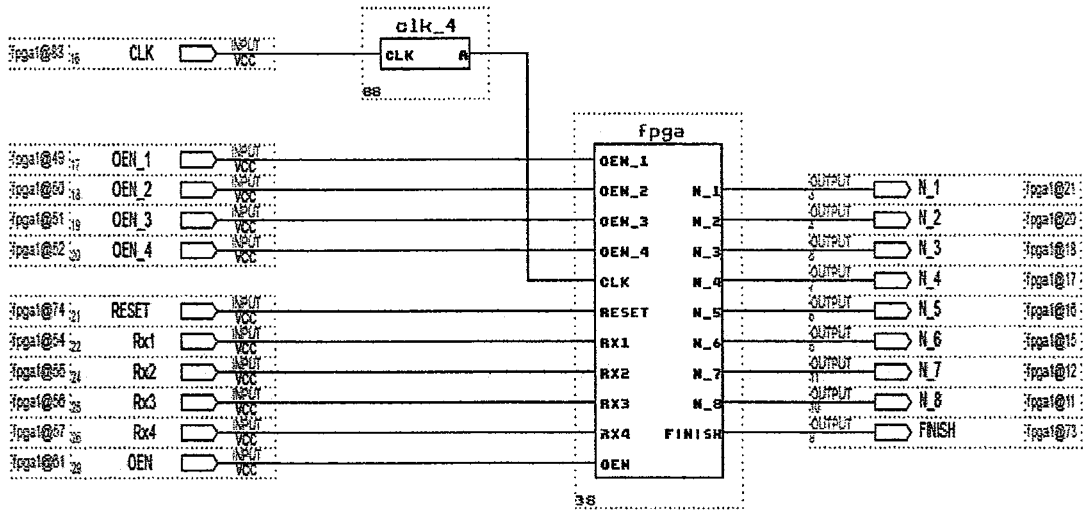
รูปที่ 3.4 แสดงภาพวงจรการทำงานภายใน Block Counter

วงจร FPGA



รูปที่ 3.5 แสดงภาพวงจรในส่วนของ FPGA

วงจรรวม FPGA

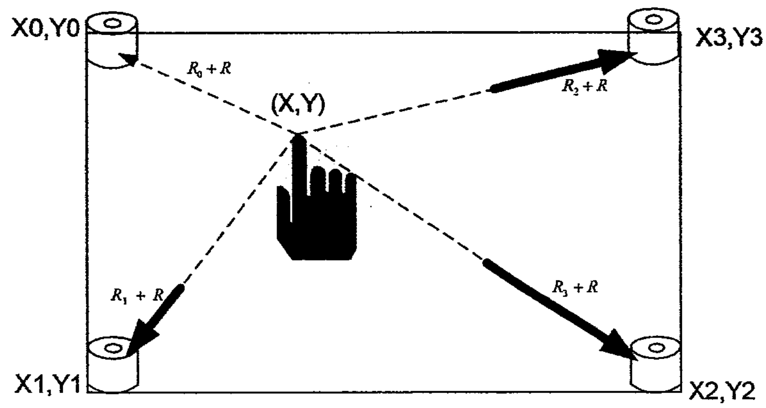


รูปที่ 3.6 แสดงภาพวงจรทั้งหมดในส่วนของ FPGA

3. 4 วงจรภาคประมวลผล

3.4.1 การคำนวณหาพิกัดตำแหน่ง

การคำนวณหาพิกัดตำแหน่ง อาศัยหลักการทางคณิตศาสตร์ โดยมีวิธีการคำนวณดังนี้



รูปที่ 3.7 แสดงการวางตำแหน่งของตัวรับ

สมมติว่าจุดที่เคาะอยู่ที่ตำแหน่ง (x,y) และเราทำการวางตำแหน่งของตัวรับทั้ง 4 ตัว ตามจุดที่กำหนดที่ทราบค่าแน่นอนคือจุด (X_0 ,Y_0), (X_1 , Y_1), (X_2 , Y_2), (X_3 ,Y_3) ดังรูป เมื่อมีการเคาะโปรเจคเตอร์สัญญาณคลื่นจะถูกส่งไปยังตัวรับทั้ง 4 ตัว เปรียบเสมือนการโยนก้อนหินลงในน้ำ ระยะเวลาที่คลื่นเดินทางจากตำแหน่งเคาะไปยังตัวรับแต่ละตัวไม่เท่ากันขึ้นอยู่กับขนาดระยะทาง ระยะเวลาของสัญญาณจะถูกส่งไปที่บอร์ด FPGA เพื่อนับค่าเวลาที่ต่างกัน และส่งไปที่ MCS-51 เพื่อใช้คำนวณ

เมื่อตัวรับตัวแรกได้รับสัญญาณดังกล่าวแล้วจะทำให้เคาน์เตอร์เริ่มทำงาน โดยรับค่าไปเรื่อยๆ จนครบทุกตัว แล้วนำค่าที่ได้ไปคำนวณหาค่า R_0, R_1, R2, R3 ออกมาแล้วนำค่า R_0, R_1, R2, R3 ไปคำนวณหาค่าพิกัด X, Y ต่อไป

การคำนวณหาพิกัด

1. การคำนวณเพื่อให้ได้ค่า R_0 , R_1 , R_2 , R_3

   เมื่อเราได้ค่าที่เคาน์เตอร์แต่ละตัวนับได้ (n) เราสามารถหาเวลาที่ใช้ได้ (t) จากสูตรต่อไปนี้

$$T = N/f \qquad \text{--------------- (3.1)}$$

โดย T คือ เวลาที่แตกต่างกันระหว่างตัวรับตัวแรกกับตัวรับตัวนั้น

N คือ จำนวนที่เคาว์นเตอร์นับได้

f คือ ความถี่ที่เคาว์นเตอร์ใช้นับ

และเราสามารถหาค่า R_0, R_1 , R_2 , R_3 ได้จากสูตรต่อไปนี้

จาก $R = VT$ --------------- (3.2)

จะได้

$$R_0 = VN_0/f$$

$$R_1 = VN_1/f$$

$$R_2 = VN_2/f$$

$$R_3 = VN_3/f$$

โดย V คือ ความเร็วของคลื่นในวัตถุที่เป็นตัวกลางในการเคาะ

R_0 , R_1 , R_2 , R_3 คือรัศมีที่แตกต่างกันระหว่างตัวรับตัวนั้นกับตัวรับที่อยู่ใกล้ตัวส่งมากที่สุด หลังจากนั้นเราจะนำค่า R_0 , R_1 , R_2 , R_3 ไปคำนวณหาค่า X , Y ต่อไป

2. การคำนวณหาค่า X , Y

   โดยจุด (X_0, Y_0), (X_1, Y_1), (X_2, Y_2), (X_3, Y_3) เป็นตำแหน่งที่เราทราบค่าแน่นอน จากรูปที่ 3.6 เราสามารถคำนวณได้ดังนี้

$$(X - X_0)^2 + (Y - Y_0)^2 = (R + R_0)^2 \quad \text{..................(1)}$$

$$(X - X_1)^2 + (Y - Y_1)^2 = (R + R_1)^2 \quad \text{..................(2)}$$

$$(X - X_2)^2 + (Y - Y_2)^2 = (R + R_2)^2 \quad \text{..................(3)}$$

$$(X - X_3)^2 + (Y - Y_3)^2 = (R + R_3)^2 \quad \text{..................(4)}$$

จัดรูปสมการด้านบน จะได้

$$X^2 - 2X_0X + X_0^2 + Y^2 - 2Y_0Y + Y_0^2 = R^2 + 2R_0R + R_0^2$$
$$X^2 - 2X_1X + X_1^2 + Y^2 - 2Y_1Y + Y_1^2 = R^2 + 2R_1R + R_1^2$$
$$X^2 - 2X_2X + X_2^2 + Y^2 - 2Y_2Y + Y_2^2 = R^2 + 2R_2R + R_2^2$$
$$X^2 - 2X_3X + X_3^2 + Y^2 - 2Y_3Y + Y_3^2 = R^2 + 2R_3R + R_3^2$$

$$X^2 + Y^2 - R^2 = 2X_0X + 2Y_0Y + 2R_0R - X_0^2 - Y_0^2 + R_0^2 \quad \text{...........} (5)$$
$$X^2 + Y^2 - R^2 = 2X_1X + 2Y_1Y + 2R_1R - X_1^2 - Y_1^2 + R_1^2 \quad \text{...........} (6)$$
$$X^2 + Y^2 - R^2 = 2X_2X + 2Y_2Y + 2R_2R - X_2^2 - Y_2^2 + R_2^2 \quad \text{...........} (7)$$
$$X^2 + Y^2 - R^2 = 2X_3X + 2Y_3Y + 2R_3R - X_3^2 - Y_3^2 + R_3^2 \quad \text{...........} (8)$$

นำสมการที่ (5) = (6) ; (5) = (7); (5) = (8)

$$2(X_0 - X_1)X + 2(Y_0 - Y_1)Y + 2(R_0 - R_1)R = X_0^2 - X_1^2 + Y_0^2 - Y_1^2 + R_1^2 - R_0^2$$
$$2(X_0 - X_2)X + 2(Y_0 - Y_2)Y + 2(R_0 - R_2)R = X_0^2 - X_2^2 + Y_0^2 - Y_2^2 + R_2^2 - R_0^2$$
$$2(X_0 - X_3)X + 2(Y_0 - Y_3)Y + 2(R_0 - R_3)R = X_0^2 - X_3^2 + Y_0^2 - Y_3^2 + R_3^2 - R_0^2$$

กำหนดให้

$A_1 = 2(X_0 - X_1)$ $\quad$ $A_2 = 2(X_0 - X_2)$ $\quad$ $A_3 = 2(X_0 - X_3)$

$B_1 = 2(Y_0 - Y_1)$ $\quad$ $B_2 = 2(Y_0 - Y_2)$ $\quad$ $B_3 = 2(Y_0 - Y_3)$

$C_1 = 2(R_0 - R_1)$ $\quad$ $C_2 = 2(R_0 - R_2)$ $\quad$ $C_3 = 2(R_0 - R_3)$

$D_1 = X_0^2 - X_1^2 + Y_0^2 - Y_1^2 + R_1^2 - R_0^2$ $\quad$ $D_2 = X_0^2 - X_2^2 + Y_0^2 - Y_2^2 + R_2^2 - R_0^2$

$D_3 = X_0^2 - X_3^2 + Y_0^2 - Y_3^2 + R_3^2 - R_0^2$

จะได้ 3 สมการที่ไม่ทราบค่าตัวแปร 3 ตัวคือ X, Y, R

$$A_1X + B_1Y + C_1R = D_1$$
$$A_2X + B_2Y + C_2R = D_2$$
$$A_3X + B_3Y + C_3R = D_3$$

$$\begin{vmatrix} A_1 & B_1 & C_1 \\ A_2 & B_2 & C_2 \\ A_3 & B_3 & C_3 \end{vmatrix} \begin{vmatrix} X \\ Y \\ R \end{vmatrix} \qquad \begin{vmatrix} D_1 \\ D_2 \\ D_3 \end{vmatrix}$$

กำหนดให้

$$U = \begin{vmatrix} A_1 & B_1 & C_1 \\ A_2 & B_2 & C_2 \\ A_3 & B_3 & C_3 \end{vmatrix}$$

จะได้

$$X = \frac{\begin{vmatrix} D_1 & B_1 & C_1 \\ D_2 & B_2 & C_2 \\ D_3 & B_3 & C_3 \end{vmatrix}}{U}$$

$$Y = \frac{\begin{vmatrix} A_1 & D_1 & C_1 \\ A_2 & D_2 & C_2 \\ A_3 & D_3 & C_3 \end{vmatrix}}{U}$$

$$R = \frac{\begin{vmatrix} A_1 & B_1 & D_1 \\ A_2 & B_2 & D_2 \\ A_3 & B_3 & D_3 \end{vmatrix}}{U}$$

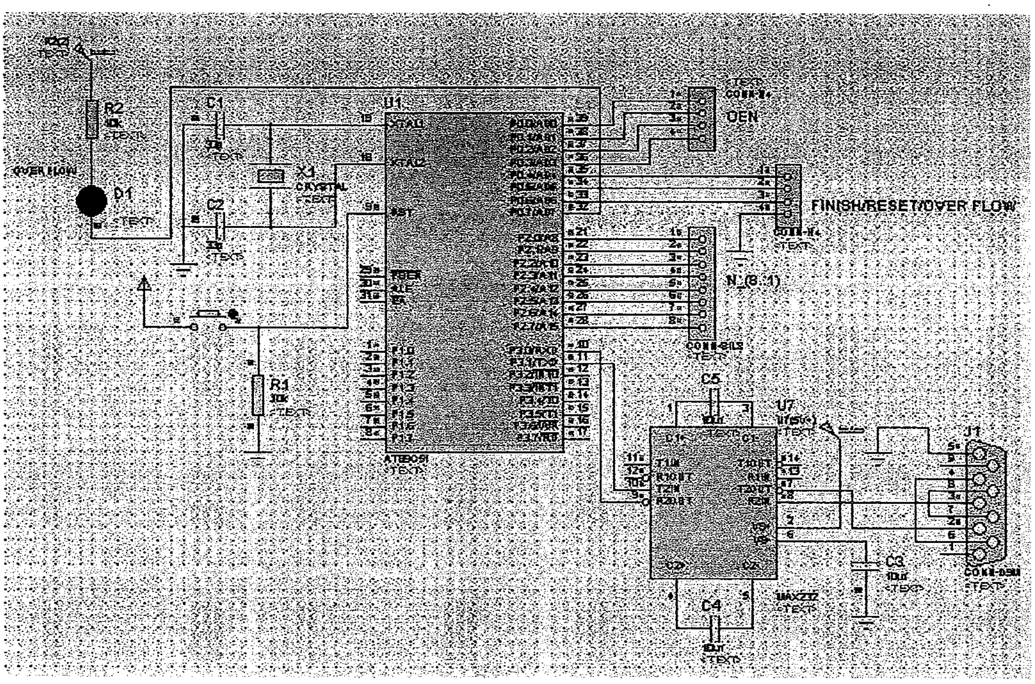
จากสมการสามารถคำนวณค่าที่ได้ในระนาบ 2 มิติคือ

X, Y: พิกัดของตำแหน่งที่เคาะ

R : เป็นระยะทางที่คลื่นใช้เคลื่อนที่ถึงตัวรับตัวแรกซึ่งเป็นระยะที่สั้นที่สุด

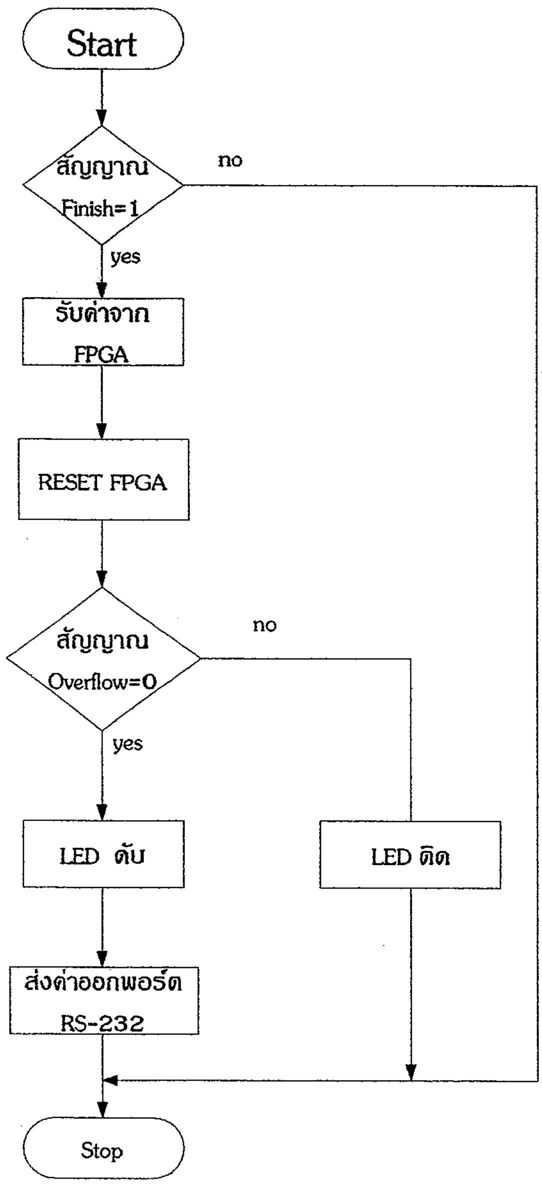
3.5 หลักการทำงานของ MCS-51

MCS-51 จะคอยเช็คสัญญาณ Finish อยู่ตลอดว่าเป็น High หรือเป็น Low ถ้าเป็น Low ให้สิ้นสุดการทำงาน แต่ถ้าเป็น High ให้ทำการรับค่า N1, N2, N3, N4, Overflow จาก FPGA แล้วจากนั้นทำการ Reset FPGA และเช็คสัญญาณ Overflow ว่าเป็น High หรือเป็น Low ถ้าเป็น High ให้ LED ติด แล้วสิ้นสุดการทำงาน แต่ถ้าเป็น Low ให้ LED ดับ และส่งค่า ค่า N1, N2, N3, N4 ออกทางพล็อต RS-232



รูปที่3.8 แสดงการต่อใช้งาน MCS-51

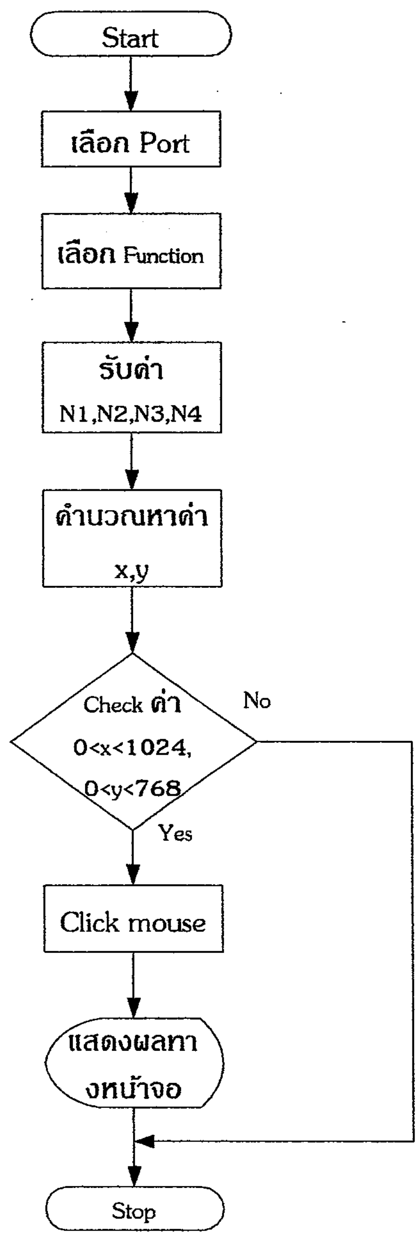
Flow chart การทำงานของ MCS-51



รูปที่3.9 แสดง Flow chart การทำงานของ MCS-51

3.6 หลักการทำงานของ Visual Basic

Visual Basic จะทำการรับค่าที่ส่งมาจาก MCS-51 เข้ามาทาง com port แล้วทำการนำค่า N1, N2, N3, N4 มาคำนวณหาค่า X,Y แล้วทำการเช็คค่า X และ Y ว่า 0 < X < 1024 , 0 < Y < 768 ถ้า X และY ตรงตามเงื่อนไขให้ทำการคลิ๊กเมาท์ และแสดงผลทางหน้าจอ



รูปที่3.10แสดง Flow chart การทำงานของ Visual Basic

บทที่4

การทดลองและผลการทดลอง

ในบทที่ 4 เป็นการทดลอง และแสดงผลการทดลองในรูปแบบต่าง ๆ โดยแยกออกมาเป็นส่วนๆในการทดลองและบันทึกผล ซึ่งสามารถแยกออกเป็นข้อหัวดังนี้

1. ส่วนของตัวรับ

1.1. เปรียบเทียบขนาดสัญญาณที่รับได้จากตัวรับชนิดต่างๆ

1.2. เปรียบเทียบสัญญาณที่มุมตำแหน่งต่างๆ รอบ เปียโซ

1.3. เปรียบเทียบระยะทางที่ใช้เคาะ

2. ส่วนของวงจรภาครับสัญญาณเคาะ

2.1 สัญญาณที่จุดต่างๆของวงจรภาครับสัญญาณจากการเคาะ

2.2 สัญญาณเอ้าท์พุตของวงจรภาครับเมื่อป้อนแรงดันอินพุทที่ระดับต่างๆ

2.3 เปรียบเทียบค่าหน่วงเวลาของสัญญาณเอ้าท์พุตของวงจรทั้ง 4 ชุด

3. ส่วนของ FPGA ในภาคต่างๆ

3.1 ผลการ Simulate ส่วนของ Block Control

3. 2 ผลการ Simulate ส่วนของ Block Counter

3. 3 ผลการ Simulate วงจรทั้งหมดของ FPGA

4. ส่วนของ MCS-51

4.1 ทดลองส่งค่าจาก mcs-51 เข้าสู่คอมพิวเตอร์โดยแสดงผลด้วยโปรแกรม HyperTerminal

4.2 ทดลองรับค่าจาก FPGA เข้าสู่ mcs-51 แล้วนำมาแสดงผลด้วยโปรแกรม HyperTerminal

5. ส่วนของโปรแกรม Visual Basic

5.1 ทดลองแสดงพิกัด X, Y ที่ตำแหน่งต่าง ๆ ของจอคอมพิวเตอร์

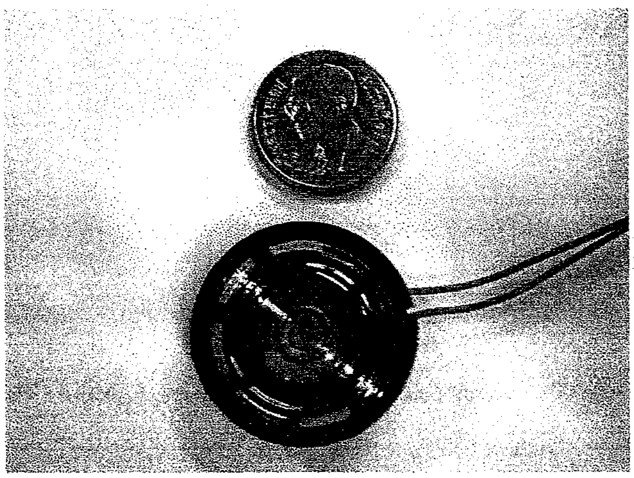
5.2 ทดลองการรับค่าเวลาจาก Mcs-51 แล้วแสดงค่าออกมา

5.3 ทดลองกำหนดพิกัด แล้วสั่งเม้าส์ให้ทำงานตามพิกัดที่กำหนด

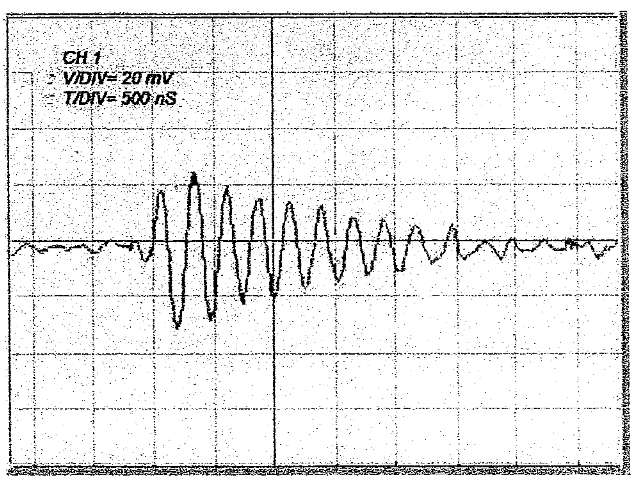
4.1 ส่วนของตัวรับ

4.1.1 เปรียบเทียบขนาดสัญญาณที่รับได้จากตัวรับสัญญาณที่ได้จากการเคาะชนิดต่างๆ

4.1.1.1 ลำโพงเล็ก

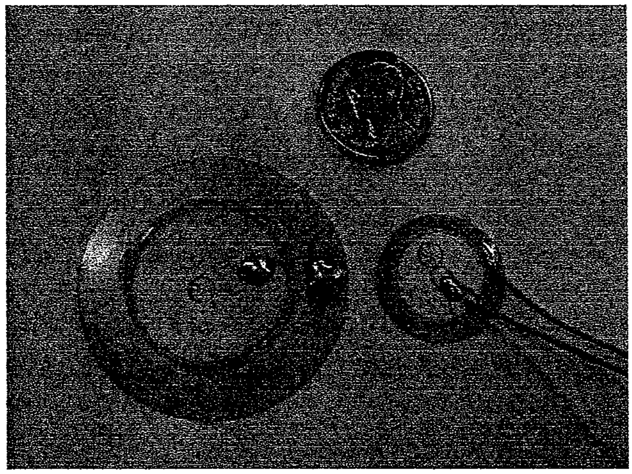


รูปที่ 4.1 ลำโพงเล็ก

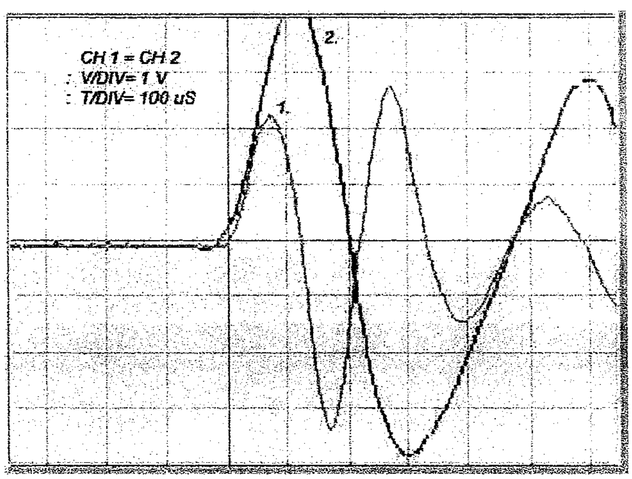


รูปที่ 4.2 สัญญาณที่ได้จากลำโพงเล็ก

4.1.1.2 เปียโซขนาดเล็กและเปียโซขนาดใหญ่แบบเปลือย

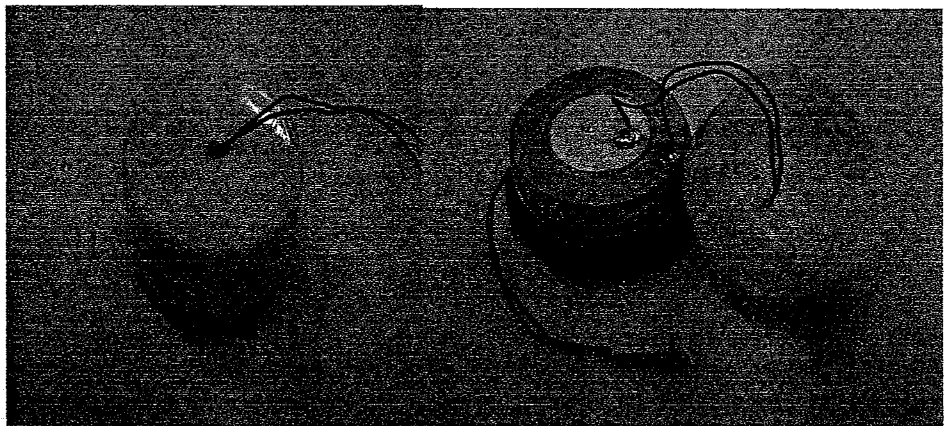


รูปที่ 4.3 เปียโซขนาดเล็กและเปียโซขนาดใหญ่แบบเปลือย

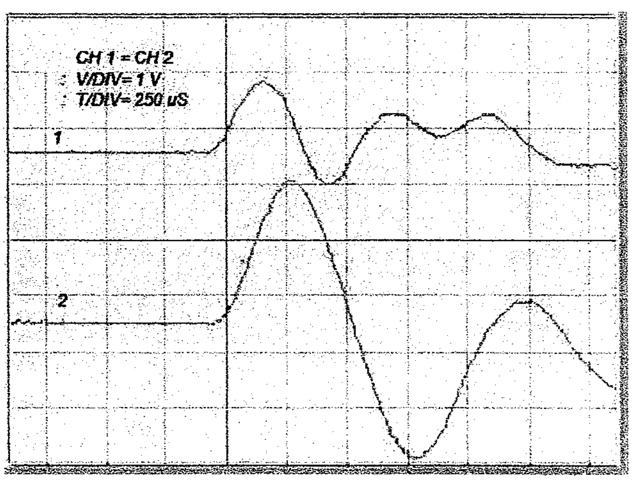


รูปที่ 4.4 สัญญาณที่ได้ CH1. เปียโซขนาดเล็ก CH2. เปียโซขนาดใหญ่

4.1.1.3 เปียโซขนาดใหญ่แบบมีฝากล่อง



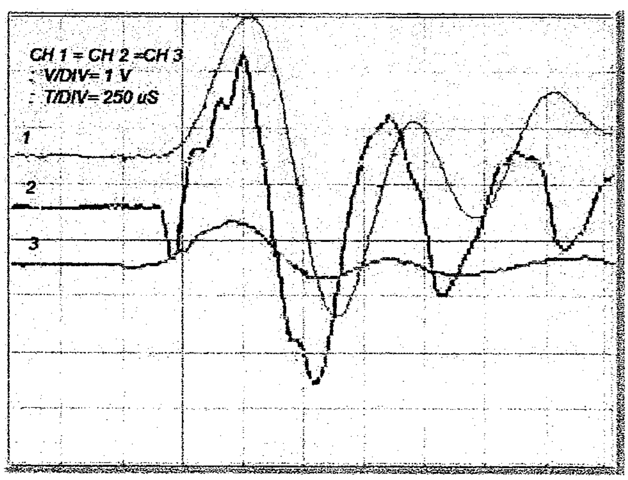
รูปที่ 4.5 เปียโซขนาดใหญ่แบบมีฝากล่อง



รูปที่ 4.6 สัญญาณที่ได้ CH1. เปียโซแบบเปลือย CH2.เปียโซ แบบมีฝากล่อง ใช้นิ้วเคาะ

4.2 เปรียบเทียบวัตถุที่ใช้เคาะ

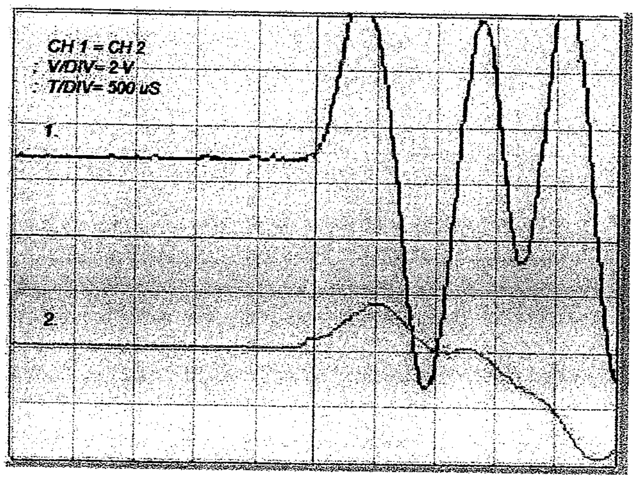
รูปที่ 4.7 - เหล็กปลายแหลม
- เหล็กปลายหุ้มเทปกาว
- ปลายนิ้ว



รูปที่ 4.8 สัญญาณจาก CH1. เหล็กปลายหุ้มเทปกาว
CH2. เหล็กปลายแหลม
CH3. ปลายนิ้ว

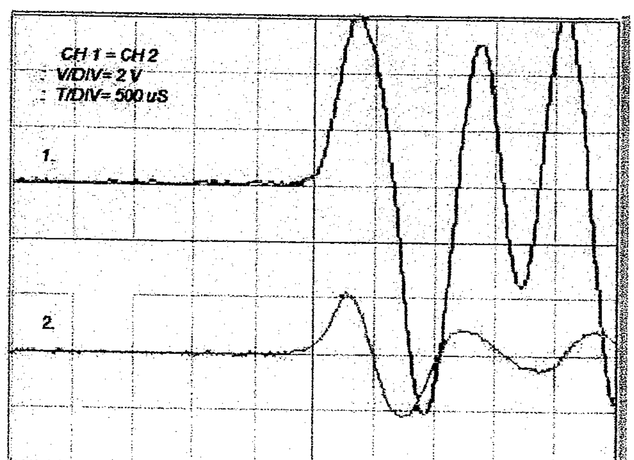
4.3 เปรียบเทียบตัวกลางที่ใช้เคาะ

4.3.1 แผ่นกระจกกับแผ่นฟิวเจอร์บอร์ด



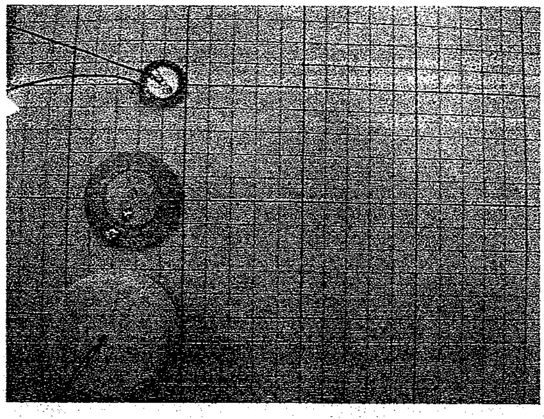
รูปที่ 4.9สัญญาณจาก CH1. แผ่นกระจก CH2. แผ่นฟิวเจอร์บอร์ด

4.3.2 แผ่นกระจกกับแผ่นไม้ (โต๊ะ)



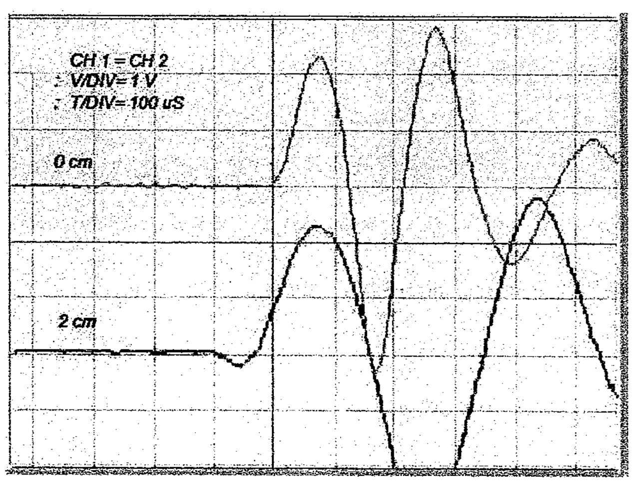
รูปที่ 4.10 สัญญาณจาก CH1.แผ่นกระจก CH2.แผ่นไม้ (โต๊ะ)

4.4 เปรียบเทียบระยะทางที่ใช้เคาะ

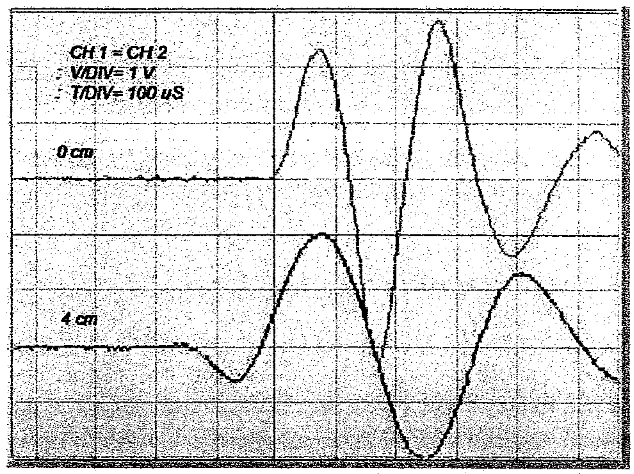


รูปที่ 4.11 ลักษณะระยะทางที่ใช้เคาะ

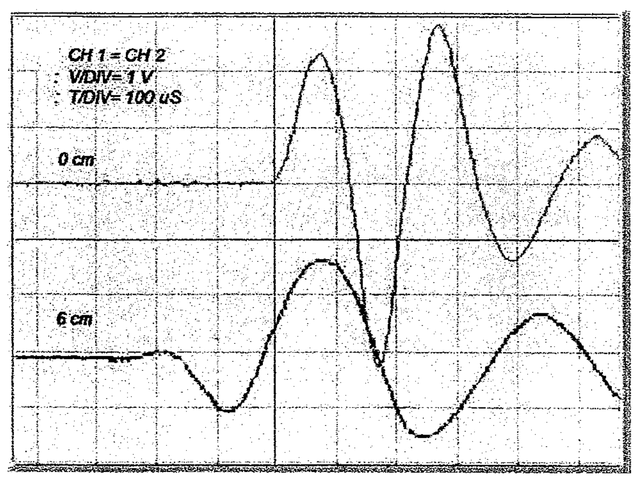
4.4.1 เปียโซอิเลคทริคขนาดเล็กแบบเปลือยที่ระยะต่างๆ



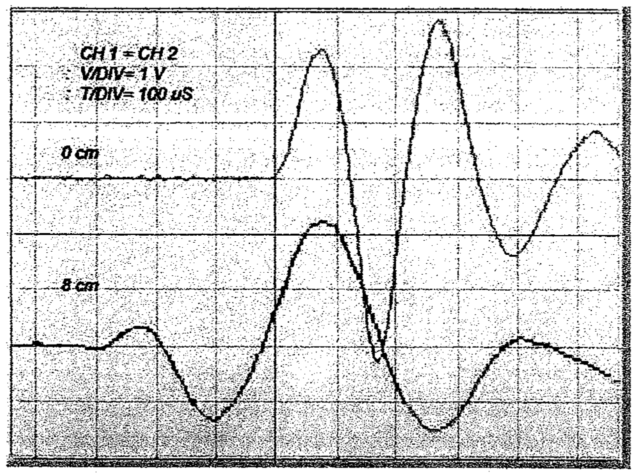
รูปที่ 4.12 สัญญาณที่ระยะ 0 กับ 2 cm



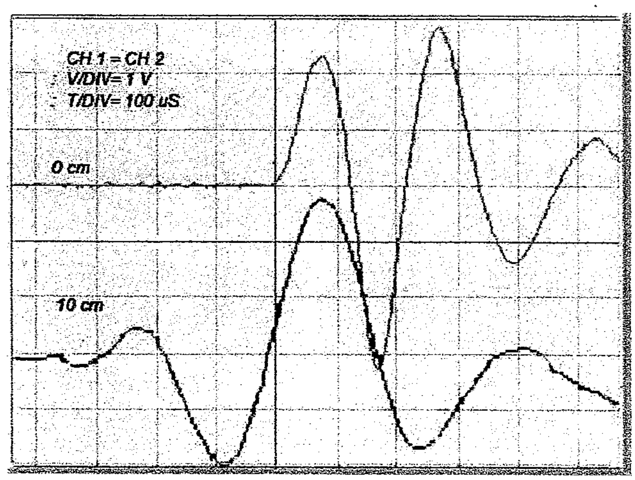
รูปที่ 4.13 สัญญาณที่ระยะ 0 กับ 4 cm



รูปที่ 4.14 สัญญาณที่ระยะ 0 กับ 6 cm

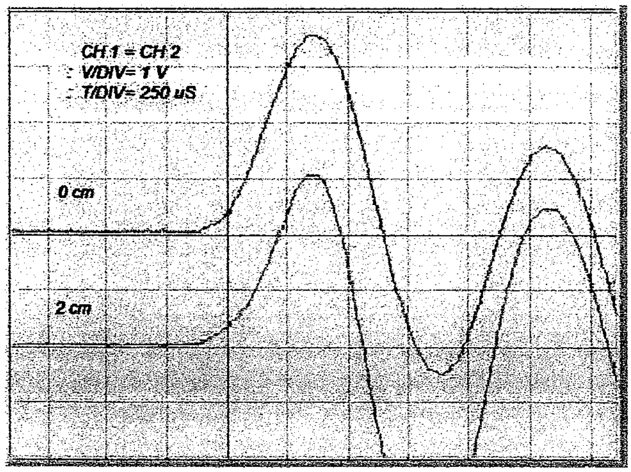


รูปที่ 4.15 สัญญาณที่ระยะ 0 กับ 8 cm

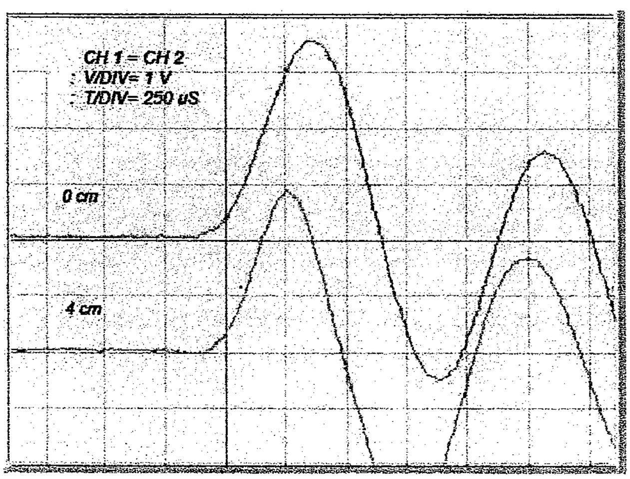


รูปที่ 4.16 สัญญาณที่ระยะ 0 กับ 10 cm

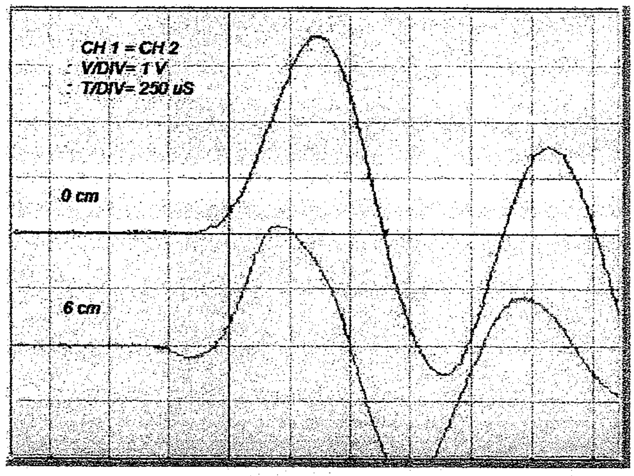
4.4.2 เปียโซอิเลคทริคขนาดใหญ่แบบมีฝากล่องที่ระยะต่างๆ



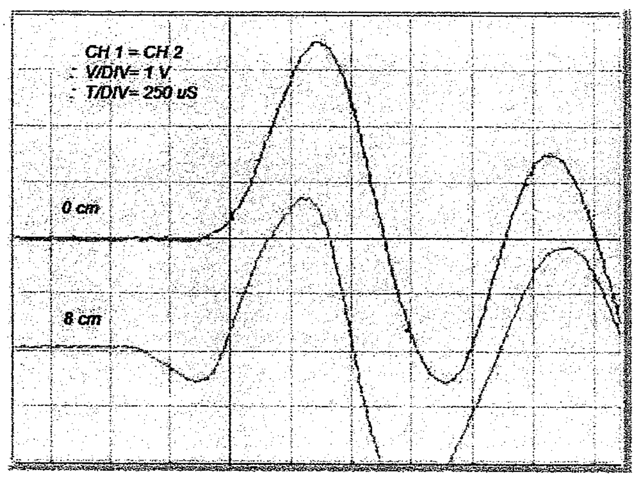
รูปที่ 4.17 สัญญาณที่ระยะ 0 กับ 2 cm



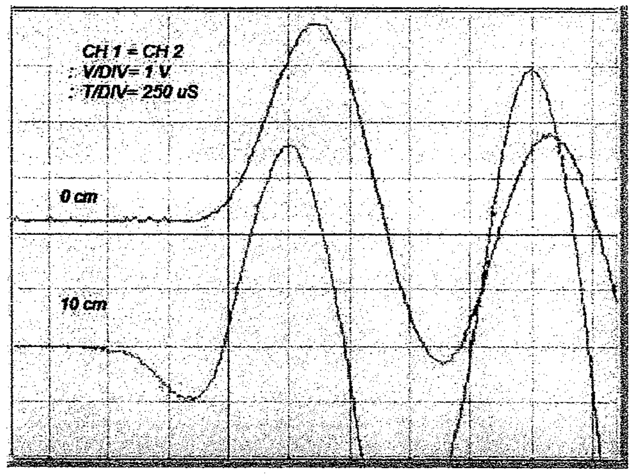
รูปที่ 4.18 สัญญาณที่ระยะ 0 กับ 4 cm



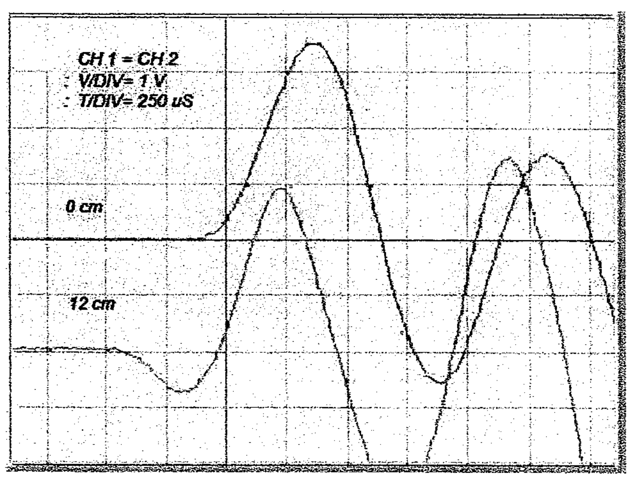
รูปที่ 4.19 สัญญาณที่ระยะ 0 กับ 6 cm



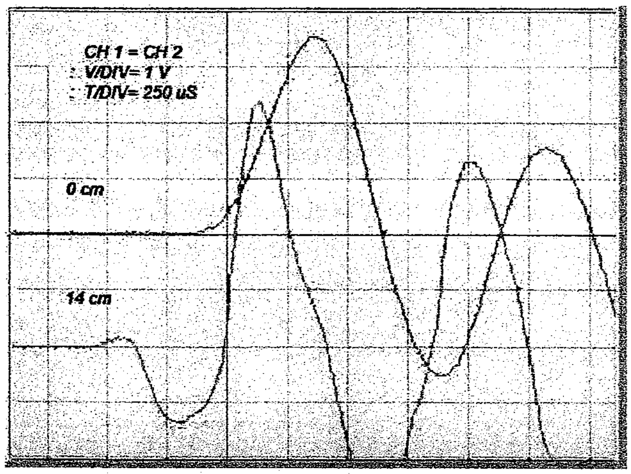
รูปที่ 4.20 สัญญาณที่ระยะ 0 กับ 8 cm



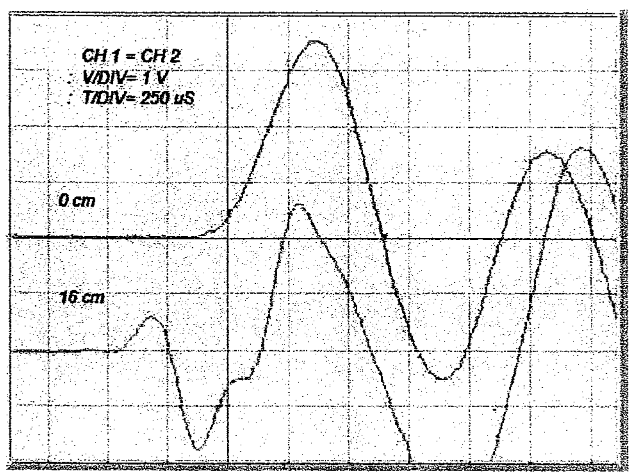
รูปที่ 4.21 สัญญาณที่ระยะ 0 กับ 10 cm



รูปที่ 4.22 สัญญาณที่ระยะ 0 กับ 12 cm

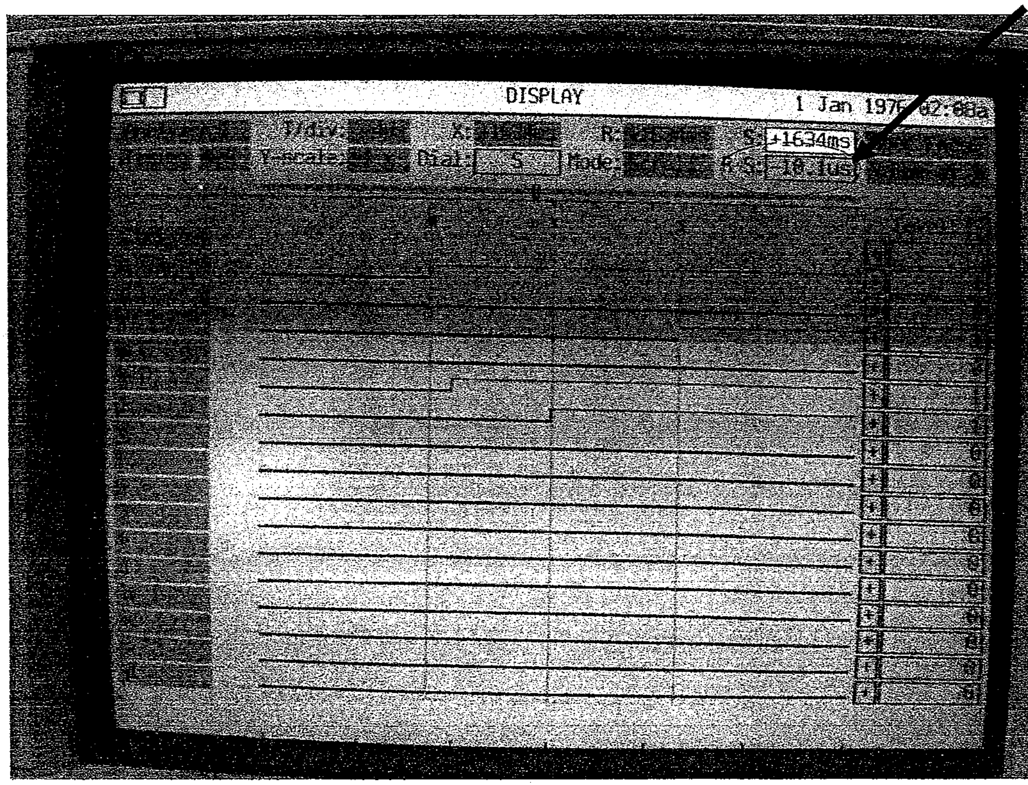


รูปที่ 4.23 สัญญาณที่ระยะ 0 กับ 14 cm



รูปที่ 4.24 สัญญาณที่ระยะ 0 กับ 16 cm

เปรียบเทียบค่าหน่วงเวลาของสัญญาณเอาต์พุตของวงจรทั้ง 4 ชุด เมื่อป้อนสัญญาณอินพุทที่มีขนาดเท่ากัน



รูปที่4.25 แสดงสัญญาณที่วัดได้จาก Logic Analyzer

โดย

T/Div = 2 us

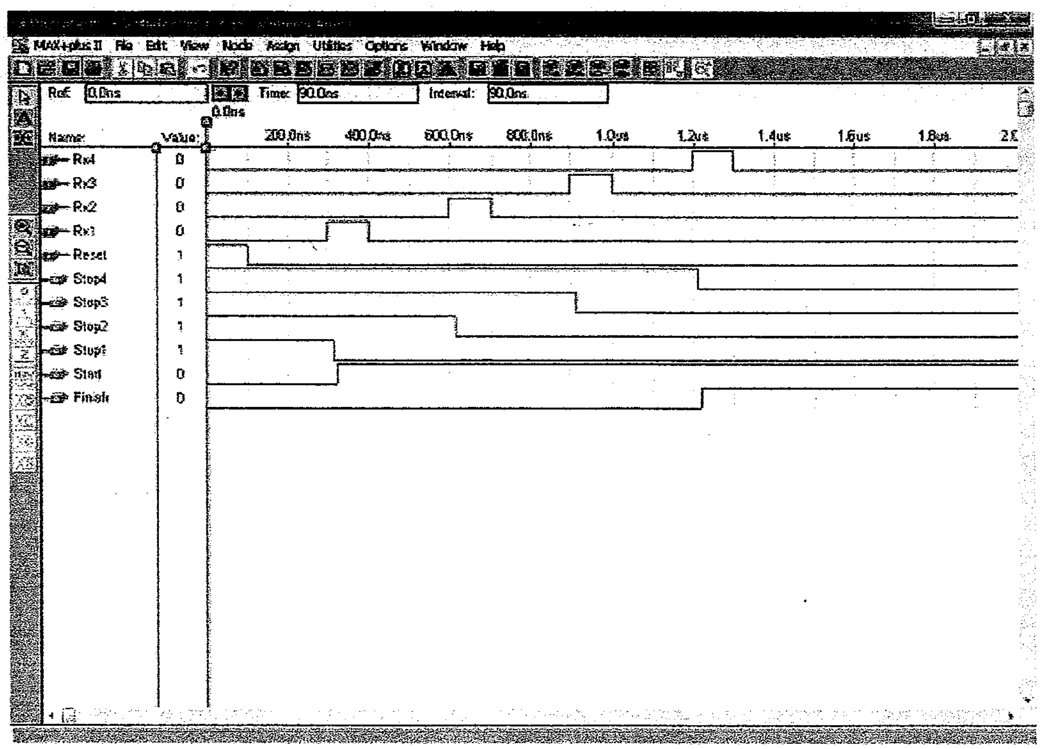
R – S = 10.1uS ซึ่งเป็นเวลาที่ต่างกันมากสุดของวงจรแต่ละชุด

4.5 แสดงผลการ Simulate ของ FPGA

ในหัวข้อนี้ได้แสดงผลการ Simulate วงจรที่ได้ออกแบบไว้ในแต่ละส่วนการทำงาน ซึ่งส่วนหลักๆ ได้แก่ ส่วนของ Block Control และ Block Counter

4.5.1 ผลการ Simulate ส่วนของ Block Control

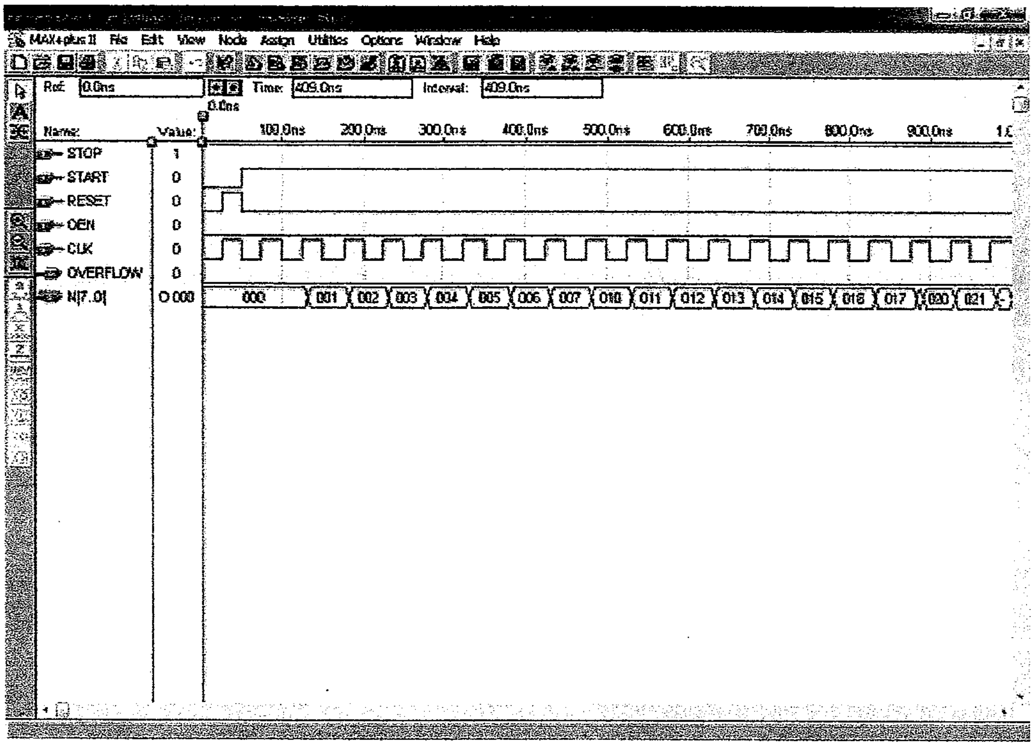
ในส่วนของ Block Control นี้เป็นส่วนที่เริ่มรับสัญญาณจากการเคาะ และไปควบคุมการทำงานในส่วน Block Counter ให้เริ่มนับและหยุดนับ



รูปที่ 4.26 แสดง Timing Diagram ของ Block Control ที่ได้จากการ Simulate

4.5.2 ผลการ Simulate ส่วนของ Block Counter

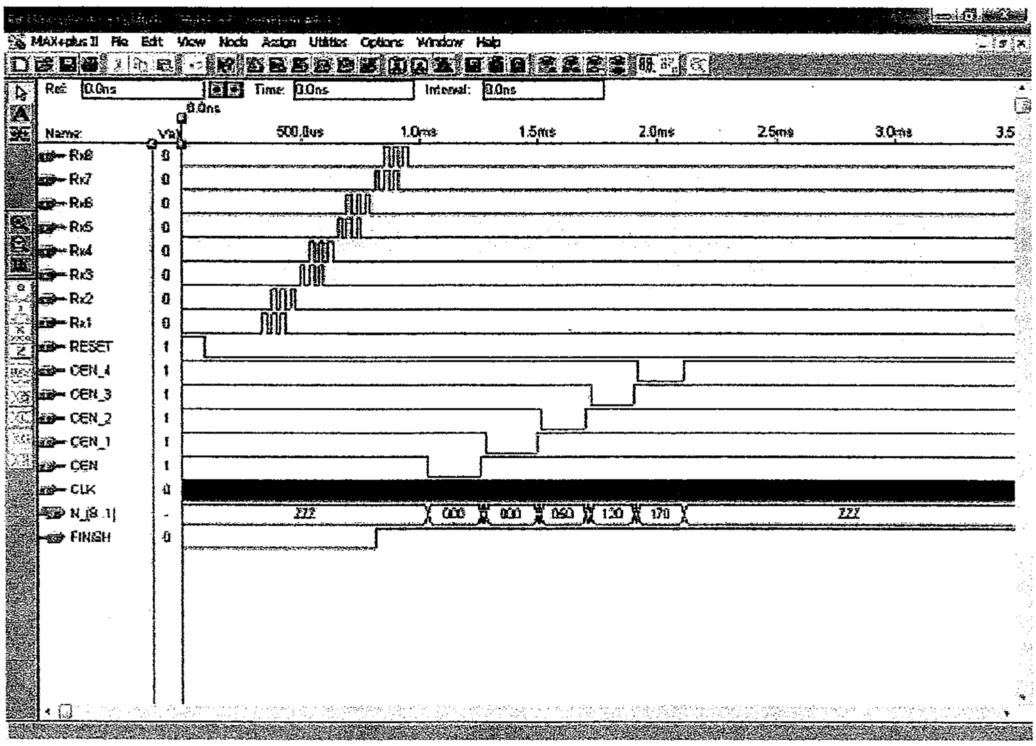
ในส่วนของ Block Counter นี้ได้ทำหน้าที่เป็นตัวจับเวลาของสัญญาณที่ส่งจากการเคาะจนกระทั่งตัวรับรับสัญญาณได้



รูปที่ 4.27 แสดง Timing Diagram ของ Block Counter ที่ได้จากการ Simulate

4.5.3 ผลการ Simulate วงจรทั้งหมดของ FPGA

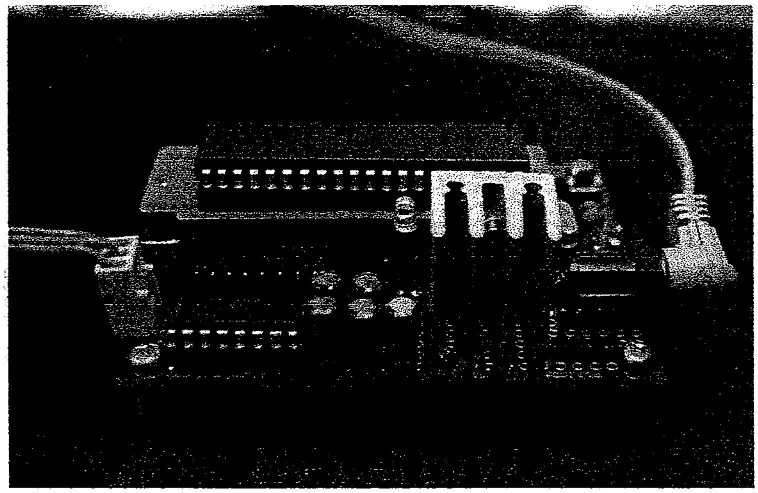
ในหัวข้อนี้ได้แสดงภาพวงจรการทำงานทุกๆ ส่วนของ FPGA ที่ได้นำมาประกอบกันอย่างสมบูรณ์และผลการ Simulate วงจรที่ได้ประกอบแล้วดังนี้



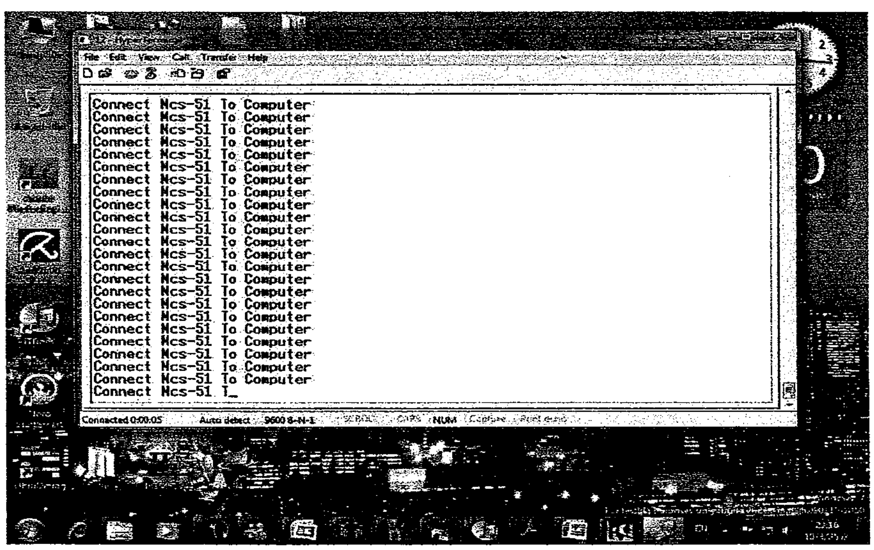
รูปที่ 4.28 แสดง Timing Diagram ของวงจรทั้งหมดของ FPGA

4.6 ส่วนของ MCS-51

4.6.1. ทดลองส่งค่าจาก mcs-51 เข้าสู่คอมพิวเตอร์โดยแสดงผลด้วยโปรแกรม HyperTerminal

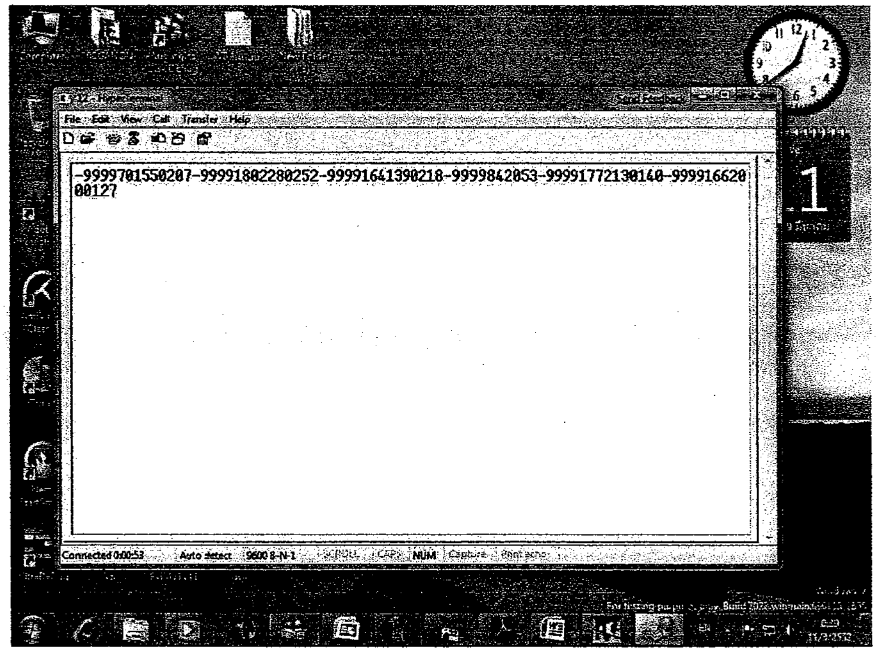


รูปที่4.29 แสดงการต่อชุดอุปกรณ์ในการทดลอง



รูปที่ 4.30 ผลการส่งข้อมูลจาก Mcs-51 เข้าสู่คอมพิวเตอร์โดยแสดงผลด้วยโปรแกรม HyperTerminal

4.6.2 ทดลองรับค่าจาก FPGA เข้าสู่ mcs-51 แล้วนำมาแสดงผลด้วยโปรแกรม HyperTerminal



รูปที่ 4.31 แสดงการรับค่าจาก FPGA เข้าสู่ mcs-51 แล้วนำมาแสดงผลด้วยโปรแกรม HyperTerminal

4.7 ส่วนของโปรแกรม Visual Basic

4.7.1 ทดลองแสดงพิกัด X, Y ที่ตำแหน่งต่าง ๆ ของจอคอมพิวเตอร์

รูปที่ 4.32 รูปแสดงพิกัดที่ตำแหน่ง X = 37, Y = 33

รูปที่ 4.33 รูปแสดงพิกัดที่ตำแหน่ง X = 196, Y = 351

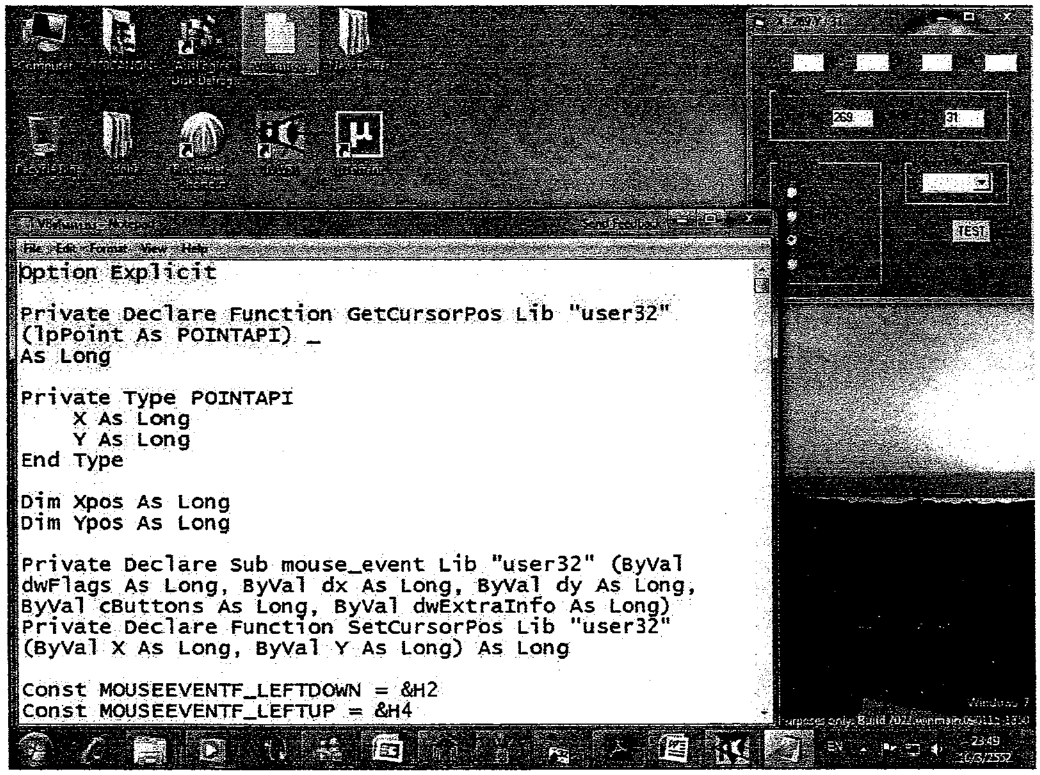
4.7.2 ทดลองการรับค่าเวลาจาก Mcs-51 แล้วแสดงค่าออกมา

การทดลองรับค่าเวลาจาก Mcs-51 แล้วแสดงค่าเวลาออกมาเพื่อแสดงว่า Mcs-51 สามารถเชื่อมต่อกับ Visual Basic ได้สมบูรณ์

รูปที่ 4.34 แสดงการรับค่าเวลาจาก Mcs-51

4.7.3 ทดลองกำหนดพิกัด แล้วสั่งเม้าส์ให้ทำงานตามพิกัดที่กำหนด

เป็นการทดลองกำหนดพิกัดแล้วเลือก Function การทำงานของเม้าส์เพื่อทดสอบการทำงานว่าเม้าส์สามารถทำได้ดังแสดงไว้ดังรูป



รูปที่ 4.35 แสดงการกำหนดค่าพิกัด แล้วให้เม้าส์ทำการดับเบิลคลิก
ที่พิกัดที่กำหนด

บทที่ 5
สรุปผลการทดลอง

จากการทดลองคุณสมบัติต่างๆ ของขนาดของอุปกรณ์ที่ใช้เป็นเซ็นเซอร์ สัญญาณการเคาะที่มุมต่างๆ ตลอดจนศึกษาปัจจัยต่างๆที่มีผลต่อขนาดและเฟสของรูปคลื่นของสัญญาณ เพื่อที่จะนำไปวิเคราะห์ในการสร้างและออกแบบชิ้นงานให้ตรงกับความต้องการและมีประสิทธิภาพสูงสุด ซึ่งผลที่ได้สรุปได้ดังนี้

5.1 สรุปเปรียบเทียบความไวและขนาดสัญญาณที่รับได้จากตัวรับชนิดต่างๆ

ลำโพง มีความไวต่อการเคาะน้อยมาก โดยขนาดสัญญาณมีเพียง 50 mVp-p เมื่อขยับแผ่นไดอะแฟรมจากการใช้ลมเป่าแรงๆ เท่านั้น

เปียโซขนาดเล็กแบบเปลือย มีความไวต่อการเคาะปานกลาง 5 Vp-p แต่ต้องเคาะด้วยวัตถุปลายแหลมแรงๆ เท่านั้น

เปียโซขนาดใหญ่แบบเปลือย มีความไวต่อการเคาะปานกลาง 8 Vp-p โดยเคาะแรงๆด้วยวัตถุของแข็งทั่วไป เช่น แท่งไม้ ดินสอไม้

เปียโซขนาดใหญ่แบบมีฝากล่อง มีความไวต่อการเคาะสูง 5 Vp-p โดยเคาะเบา ๆ ด้วยปลายนิ้ว

ดังนั้น ชนิดตัวรับที่มีความไวและขนาดสัญญาณที่ได้เหมาะแก่การนำไปใช้งานคือ เปียโซขนาดใหญ่แบบมีฝากล่อง

5.2 สรุปเปรียบเทียบวัตถุที่ใช้เคาะ

จากผลการทดลองสังเกตได้ว่าขนาดแอมพลิจูดที่ได้จากการเคาะโดยใช้เหล็กปลายแหลมและเหล็กหุ้มเทปกาวจะมีขนาดแอมพลิจูดมากถึง 6 Vp-p ซึ่งต่างจากการเคาะด้วยปลายนิ้วที่ได้เพียง 1.5 Vp-p สังเกตว่าเหล็กปลายแหลมจะให้รูปสัญญาณที่ไม่สมดุลสมมาตรกัน ทั้งนี้เกิดขึ้นเนื่องจากการกระทบกันโดยตรงระหว่างของแข็งที่มีความหนาแน่นสูงนั่นคือเหล็กกับกระจกที่เป็นบอร์ดรองรับเสียง ต่างจากเหล็กหุ้มเทปกาวและนิ้วซึ่งมีการลักษณะที่ไม่แข็งจนเกินไปทำให้สัญญาณที่ได้มีลักษณะสมมาตร แต่สิ่งที่เราสนใจในการทดลองนี้คือ เคาะด้วยปลายนิ้วมือจะให้ขนาดแรงดันเพียงพอต่อการนำไปใช้งานเท่านั้นซึ่งจากการทดลองนั้นก็ได้แรงดัน 1.5 Vp-p ซึ่งหากนำไปผ่านวงจรขยายก็เพียงพอต่อการนำไปใช้งานได้

5.3 สรุปเปรียบเทียบตัวกลางที่ใช้เคาะ

จากผลการทดลองเคาะที่ตำแหน่งระยะห่างเดียวกันบนวัตถุตัวกลางชนิดต่างๆ สังเกตได้ว่า

แผ่นฟิวเจอร์บอร์ด มีขนาดแอมพลิจูด 6 Vp-p รูปสัญญาณไม่แน่นอน

แผ่นไม้(โต๊ะ) มีขนาดแอมพลิจูด 4 Vp-p รูปสัญญาณแน่นอน

แผ่นกระจก มีขนาดแอมพลิจูดมากถึง 14 Vp-p รูปสัญญาณแน่นอนสมมาตรกัน

สรุปได้ว่าวัตถุที่นำมาใช้รองรับเป็นบอร์ดสำหรับการเคาะ หากมีความหนาแน่นของเนื้อวัตถุสูง จะให้ขนาดของแอมพลิจูดที่สูงและแน่นอน ดังนั้นเราควรนำกระจกมาใช้เป็นวัตถุรองรับการเคาะในงานนี้จึงเหมาะสมที่สุด

5.4 สรุปเปรียบเทียบระยะทางที่ใช้เคาะ

จากผลการทดลองสังเกตได้ว่าที่ระยะเคาะที่ตำแหน่ง 0 cm ซึ่งใกล้ตัวรับมากสุด สัญญาณจะมีแอมพลิจูดมากสุดและลูกคลื่นลูกแรกมีเฟสเป็นซีกบวกก่อน จากนั้นเมื่อระยะทางห่างออกไปจากตัวรับขนาดแอมพลิจูดของสัญญาณจะเล็กลงไปเรื่อยๆ สัมพันธ์กับระยะทางที่ห่างออกไปดังนั้นจากข้อมูลดังกล่าวเราจึงต้องออกแบบวงจรให้สัญญาณลูกแรกที่เข้ามาจะนำสัญญาณเริ่มต้นดังกล่าวไปหาผลต่างทางเวลาใช้คำนวณต่อไป

5.5 สรุปผลจากการทดลอง วงจรภาครับสัญญาณเคาะ

สามารถขยายสัญญาณที่ได้จากการเคาะโดยขยายอินพุท เพื่อนำสัญญาณส่งไป SUM กันใน FPGA โดยใช้ OR Gate จากการทดลองป้อนอินพุทที่ระดับแรงดันต่างๆ สังเกตได้ว่าอินพุตต่ำสุดที่วงจรขยายได้แล้วได้เอ้าท์พุทที่เป็นลอจิก 1 คือ 600 mV ซึ่งมีผลทำให้ไม่สามารถขยายสัญญาณที่เคาะตำแหน่งไกลๆ จากเปียโซได้ และจากการเปรียบเทียบค่าหน่วงเวลาของสัญญาณเอ้าท์พุตของวงจรทั้ง 4 ชุด เมื่อป้อนสัญญาณอินพุทที่มีขนาดเท่ากัน ผลปรากฏว่าวงจรทั้ง 4 ชุด มีค่าหน่วงเวลาที่ต่างกันมากสุด คือ 10.1 uS

5.6 สรุปผลจากการทดลอง Simulate การทำงานในส่วนของ FPGA ในภาคต่างๆ ได้ผลดังนี้

ในส่วนของFPGA มีอยู่ 2 ส่วนที่สำคัญ คือภาค control และภาค counter

ภาค Control ทำหน้าที่ในการเริ่มนับและหยุดนับเมื่อมีสัญญาณเริ่มต้นเข้ามา โดยจะรับค่าได้เพียงครั้งเดียวเท่านั้น จะรับได้อีกครั้งก็ต่อเมื่อมีการรีเซ็ต

ภาค Counter ทำหน้าที่รับผลต่างทางเวลาสัญญาณที่เข้ามาในแต่ละตัวซึ่งจากการออกแบบนี้ค่าที่นับได้จะมีขนาด 8 บิต นับได้จำนวน 0-255 ซึ่งหากเราต้องการความละเอียดของข้อมูลก็เพียงเพิ่มจำนวนบิตในการนับ

5.7 สรุปรวม

จากการทดลองทั้งหมดในโปรเจคชิ้นนี้สามารถสรุปเป็นคุณสมบัติความสามารถได้ คือ

5.7.1สร้างวงจรที่ใช้รับสัญญาณอินพุตจาก เปียโซขนาดใหญ่ ทั้ง 4 จุดเพื่อใช้เป็นเซ็นเซอร์เพื่อส่งสัญญาณให้วงจรภาครับซึ่งมีวงจรภาค Amplifier ภาค Comparator แล้วส่งสัญญาณให้วงจรรวมในชิพ FPGA

5.7.2 ได้ทดสอบสร้างวงจร MCS-51 ดังรูป 4.38 เพื่อใช้ส่งสัญญาณในการติดต่อกับ Hyper Terminal ผ่านทางพอร์ต RS-232 ซึ่งสามารถสื่อสารได้

5.5.3 ทดสอบวงจร MCS-51 ที่สร้างขึ้นในการติดต่อกับโปรแกรม Visual Basic ซึ่งวงจร MCS-51 สามารถควบคุมให้แสดงผลที่ Form ของโปรแกรม Visual Basic ได้ตามที่เราโปรแกรมไว้

5.7.4 เมื่อเราทำการเคาะในตำแหน่งที่ไกลออกไป (ระยะไกลสุดของการเคาะ) สัญญาณที่ได้มีขนาดเบาเกินไปทำให้เกิดการ delay ทางเวลาขึ้นในวงจรขยาย

5.7.5 เมื่อทำการทดสอบเคาะในพิกัดต่าง ๆ แล้วจำนวนที่เคาร์เตอร์นับได้ในแต่ละครั้งจะมีค่าเปลี่ยนแปลงไปตาม วัสดุที่ใช้เคาะ แรงที่ใช้เคาะ ทำให้การคำนวณพิกัดได้ไม่ตรง ณ ตำแหน่งเดิมตลอดทุกครั้ง

5.7.6 จากค่าของการเคาะที่ตำแหน่งต่าง ๆ ที่เป็นพิกัดที่แน่นอนแล้วเมื่อนำมาคำนวณหาค่าความเร็วของคลื่นปรากฏว่าค่าของความเร็วเปลี่ยนไป ณ ทุกตำแหน่งส่งผลให้การคำนวณพิกัดมีความคลาดเคลื่อนไป ทำให้การแสดงผลไม่ถูกต้อง

5.7.7 จากการทดลองจะมีการ Error เกิดขึ้นเนื่องมาจากการเคาะ เพราะเมื่อเราทำการเคาะด้วยวัตถุที่ต่างกันคลื่นเสียงที่เดินทางในกระจกก็จะมีลักษณะของคลื่นเสียงที่ต่างกันกล่าวคือความยาวของคลื่นเสียงมีค่าต่างกัน เหตุผลของการ Error อีกอย่างหนึ่งคือน้ำหนักของการเคาะที่ไม่เท่ากันทำให้กว่าที่คลื่นจะเดินทางมาถึงตัว Sensor อาจทำให้ผลของการเคาเตอร์มีค่าเปลี่ยนไปส่งผลให้สมการที่ใช้มาคำนวณไม่สามารถคำนวณค่าที่แท้จริงออกมาได้ ดังนั้นในแนวทางการแก้ไข ทางที่ดีที่สุดคือต้องหาวัสดุที่นำมาใช้ในการเคาะเป็นวัสดุเดียวกัน และการเคาะควรที่จะมีน้ำหนักการเคาะที่เท่ากันด้วย เหตุผลของการ Error อีกประการหนึ่งคืออุปกรณ์ที่นำมาใช้งาน อย่างเช่น Sensor ควรหา Sensor ที่มีคุณภาพสูงมาใช้ หรือแม้แต่วงจรก็ควรมีคุณสมบัติที่เหมือนกันเพื่อที่จะทำให้ค่าการ Delay ของการทำงานลดลง

กิตติกรรมประกาศ

โปรเจคเรื่องนี้เป็นระบบสั่งงานหน้าจอโปรเจกเตอร์ด้วยปลายนิ้วเพื่อควบคุมหน้าจอวินโดว์แทนเมาส์ ได้รับการช่วยเหลือจากอาจารย์ที่ปรึกษา อาจารย์ ชินภัทร นันทจิวากรชัย รวมถึงอาจารย์หลายท่านที่ไม่ได้กล่าวถึง ณ ที่นี้ ที่ให้คำแนะนำตลอดจนช่วยอบรมให้ความรู้ทางวิชาการและการประกอบอาชีพทางวิศวกรรมขอบคุณเพื่อนๆภาคอิเล็กทรอนิกส์ทุกคน ที่ได้ให้ความช่วยเหลือทางด้านแนวความคิดและการดำเนินความสะดวกในการทำโปรเจค ทั้งทางด้านอุปกรณ์อิเล็กทรอนิกส์ เครื่องคอมพิวเตอร์ และหนังสือต่างๆ คณะผู้จัดทำมีความประทับใจในความอบอุ่นและความมีน้ำใจเหล่านี้ และนำความรู้สึกเหล่านี้เป็นกำลังใจในการประกอบอาชีพและการดำรงชีวิต ตลอดจนการถ่ายทอดให้แก่น้องๆรุ่นต่อไป

คณะผู้จัดทำ

เอกสารอ้างอิง

1. Astron Logic Research &Development, เปิดโลก FPGA กับ WIZARD PLD-A01, กรุงเทพฯ. สำนักพิมพ์แอสตรอนส์
2. กิติศักดิ์ นวลศรี, ภูริวัจน์ รอดน้อย, อักษร เพ็ชรผจญ, "ระบบสั่งงานหน้าจอโปรเจกเตอร์ด้วยปลายนิ้วเพื่อควบคุมหน้าจอวินโดร์แทนเมาส์" , ปพ. ก677ร 2550, วิทยานิพนธ์ของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมอิเล็กทรอนิกส์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
3. เกรียงวิทย์ ศรีนรดิษฐ์เลิศ, "ฟิสิกส์ KEY ENTRANCE" , หน้า 55-58.
4. เฉลิมชัย จีรพันธุ์, "การสร้างและวิเคราะห์วงจรไฟฟ้าสมมูล BVD ของสารเปียโซอิเล็กทริก", วพ ฉ417ก 2549, หน้า 4-8, วิทยานิพนธ์ของการศึกษาตามหลักสูตรปริญญา วิศวกรรมศาสตร์ บัณฑิต สาขาวิศวกรรมการวัดคุม สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร ลาดกระบัง
5. ศิริชัย เรืองเดช, สุรเดช สนงาม, "เม้าส์สามมิติ", ปพ ศ452ม 2546, ปริญญานิพนธ์ของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต ภาควิชาอิเล็กทรอนิกส์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
6. อภิชาต ภู่พลับ, เริ่มต้นเขียนโปรแกรมติดต่อและควบคุมฮาร์ดแวร์ด้วย Visual Basic, กรุงเทพฯ InFopress Developer Book, 2546

ภาคผนวก

โปรแกรมในส่วนของไมโครคอนโทรลเลอร์

```
#include<stdio.h>
#include<reg51.h>
sbit finish=P1^0;
sbit oe1=P1^1;
sbit oe2=P1^2;
sbit oe3=P1^3;
sbit oe4=P1^4;
sbit reset=P1^5;

void serial_inter()
{
        SCON=0x50;
        TMOD=0x20;
        TH1=0xFD;
        EA=1;
        ES=1;
        TR1=1;
        TI=1;
}
main()
{
        int N1=0,N2=0,N3=0,N4=0;
        unsigned int i,j;
        serial_inter();
        for(;;)
```

```
{
        P0=0x00;
        P1=0xFE;

        serial_inter();
        if(finish==1)
        {
                oe1=0;
            for(i=0;i<5000;i++);
                N1=P0;
                oe1=1;

                oe2=0;
                for(i=0;i<5000;i++)
                N2=P0
                oe2=1;

                oe3=0;
                for(i=0;i<5000;i++)
                N3=P0;
                oe3=1;

                oe4=0;
                for(i=0;i<5000;i++)
                N4=P0;
                oe4=1;

        printf("-9999");
```

```
            for(i=0;i<5000;i++);
      printf("%d",N1);
            for(i=0;i<5000;i++);
      printf("%d",N2);
            for(i=0;i<5000;i++);
       printf("%d",N3);
            for(i=0;i<5000;i++);
        printf("%d",N4);


            for(j=0;j<15000;j++);
            P1=0xDE;
            for(i=0;i<5000;i++);
            P1=0xFE;
      }
   }
}
```

โปรแกรมในส่วนของ Visual basic

```
Option Explicit

Private Declare Function GetCursorPos Lib "user32" (lpPoint As POINTAPI) _
As Long

Private Type POINTAPI
   X As Long
   Y As Long
End Type

Dim Xpos As Long
Dim Ypos As Long

Private Declare Sub mouse_event Lib "user32" (ByVal dwFlags As Long, ByVal dx As Long,
ByVal dy As Long, ByVal cButtons As Long, ByVal dwExtraInfo As Long)
Private Declare Function SetCursorPos Lib "user32" (ByVal X As Long, ByVal Y As Long) As Long

Const MOUSEEVENTF_LEFTDOWN = &H2
Const MOUSEEVENTF_LEFTUP = &H4
Const MOUSEEVENTF_RIGHTDOWN = &H8
Const MOUSEEVENTF_RIGHTUP = &H10

Dim data(0 To 4) As String
Dim i As Byte

Private Sub Form_Load()
```

```
Dim List As Variant

AddPort.AddItem 1, 0
AddPort.AddItem 2, 1
AddPort.AddItem 3, 2
AddPort.AddItem 4, 3
AddPort.AddItem 5, 4

MSComm1.CommPort = 1
MSComm1.InBufferSize = 1024
MSComm1.OutBufferSize = 512
MSComm1.RThreshold = 1
MSComm1.Settings = "9600,n,8,1"
MSComm1.PortOpen = True

End Sub

Private Sub MSComm1_OnComm()

On Error Resume Next
i = i + 1
If MSComm1.CommEvent = comEvReceive Then
data(i) = MSComm1.Input
End If
If data(i) = -9999 Then
i = 0
End If
If i = 4 Then
```

```
Aray1.Text = data(1)

Aray2.Text = data(2)

Aray3.Text = data(3)

Aray4.Text = data(4)

i = -1

End If

'-----------------------------------------------------------------------------------------------------------------------
-----

Dim Delta, X, x0, x1, x2, x3, Y, y0, y1, y2, y3, xt, yt, r, r0, r1, r2, r3, r4, a1, a2, a3, a4, b1, b2, b3,
b4, c1, c2, c3, c4, d1, d2, d3, d4 As Variant

r0 = (Aray1 * 1.410917454)

r1 = (Aray1 * 1.410917454)

r2 = (Aray3 * 1.410917454)

r3 = (Aray4 * 1.410917454)


x0 = 0

y0 = 0

x1 = 0

y1 = 90

x2 = 120

y2 = 90

x3 = 120

y3 = 0


a1 = (2 * x0) - (2 * x1)

a2 = (2 * x0) - (2 * x2)

a3 = (2 * x0) - (2 * x3)

b1 = (2 * y0) - (2 * y1)

b2 = (2 * y0) - (2 * y2)
```

```
b3 = (2 * y0) - (2 * y3)

c1 = (2 * r0) - (2 * r1)

c2 = (2 * r0) - (2 * r2)

c3 = (2 * r0) - (2 * r3)

d1 = 0 - 8100 - 0 + (r1 * r1) - (r0 * r0)

d2 = 0 - 22500 - 0 + (r2 * r2) - 0 - (r0 * r0)

d3 = 0 - 14400 - 0 + (r3 * r3) - 0 - (r0 * r0)

Delta = ((a1 * b2 * c3) + (b1 * c2 * a3) + (c1 * a2 * b3) - (a3 * b2 * c1) - (b3 * c2 * a1) - (c3 * a2
* b1))


X = ((d1 * b2 * c3) + (b1 * c2 * d3) - 0 + (c1 * d2 * b3)  - (d3 * b2 * c1) - 0 - (b3 * c2 * d1) - (c3 *
d2 * b1)) / Delta


Y = ((a1 * d2 * c3) + (d1 * c2 * a3) + (c1 * a2 * d3) - 0 - (a3 * d2 * c1) - 0 - (d3 * c2 * a1 - (c3 *
a2 * d1)) / Delta


r = ((a1 * b2 * d3) - 0 + (b1 * d2 * a3) + (d1 * a2 * b3) - 0 - (a3 * b2 * d1) - 0 - (b3 * d2 * a1)  -
(d3 * a2 * b1)) / Delta

X = (X) * 8.53333

Y = (Y) * 8.53333

'----------------------------------------------------------------------------------------------------------------
-----


If X > 1023 Then

      X = 1023

   If Y > 767 Then

      Y = 767

   ElseIf Y < 0 Then

      Y = 0

   End If
```

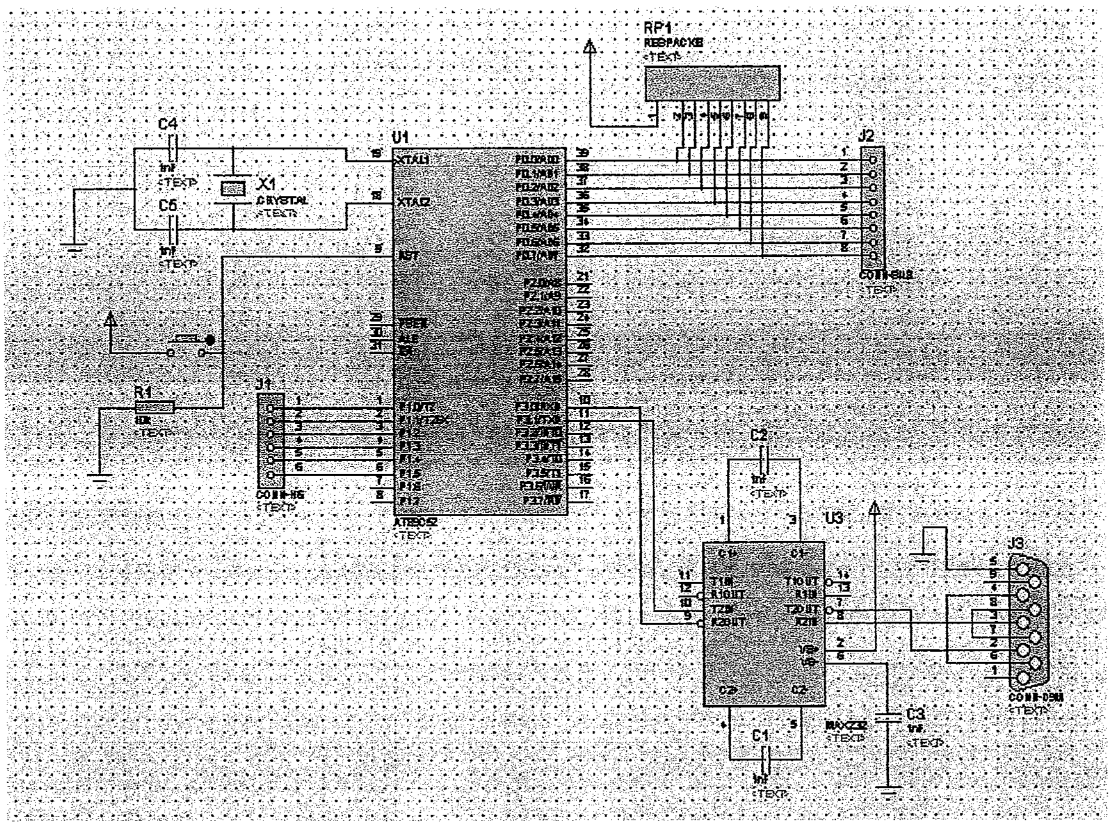
```
ElseIf X < 0 Then
    X = 0
  If Y > 768 Then
    Y = 767
  ElseIf Y < 0 Then
    Y = 0
  End If
End If


If LClick.Value = True Then
   Call MouseLClick(X, Y)
   lXlpos.Text = X
   lYlpos.Text = Y
ElseIf RClick.Value = True Then
   Call MouseRClick(X, Y)
   lXlpos.Text = X
   lYlpos.Text = Y
ElseIf DbClick.Value = True Then
   Call MouseDbClick(X, Y)
   lXlpos.Text = X
   lYlpos.Text = Y
ElseIf Noclick.Value = True Then
   Call MouseNoClick(X, Y)
   lXlpos.Text = X
   lYlpos.Text = Y
End If
```

```
End Sub


Sub MouseLClick(ByVal PosX As Long, ByVal PosY As Long)
  SetCursorPos PosX, PosY
  mouse_event MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0
  mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
End Sub

Sub MouseRClick(ByVal PosX As Long, ByVal PosY As Long)
  SetCursorPos PosX, PosY
  mouse_event MOUSEEVENTF_RIGHTDOWN, 0, 0, 0, 0
  mouse_event MOUSEEVENTF_RIGHTUP, 0, 0, 0, 0
End Sub

Sub MouseDbClick(ByVal PosX As Long, ByVal PosY As Long)
  SetCursorPos PosX, PosY
  mouse_event MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0
  mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
  mouse_event MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0
  mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
End Sub
Sub MouseNoClick(ByVal PosX As Long, ByVal PosY As Long)
  SetCursorPos PosX, PosY
  mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0
End Sub
```

```
Private Sub Test_Click()

Dim X, Y As Variant
X = lXlpos.Text
Y = lYlpos.Text

If LClick.Value = True Then
   Call MouseLClick(X, Y)
   lXlpos.Text = X
   lYlpos.Text = Y
ElseIf RClick.Value = True Then
   Call MouseRClick(X, Y)
   lXlpos.Text = X
   lYlpos.Text = Y
ElseIf DbClick.Value = True Then
   Call MouseDbClick(X, Y)
   lXlpos.Text = X
   lYlpos.Text = Y
ElseIf Noclick.Value = True Then
   Call MouseNoClick(X, Y)
   lXlpos.Text = X
   lYlpos.Text = Y
End If
End Sub

Private Sub Timer1_Timer()
Dim pt As POINTAPI
```

```
GetCursorPos pt
Xpos = pt.X
Ypos = pt.Y
Me.Caption = " X : " & Xpos & " Y : " & Ypos
End Sub
```

วงจร MCS-51



'eatures

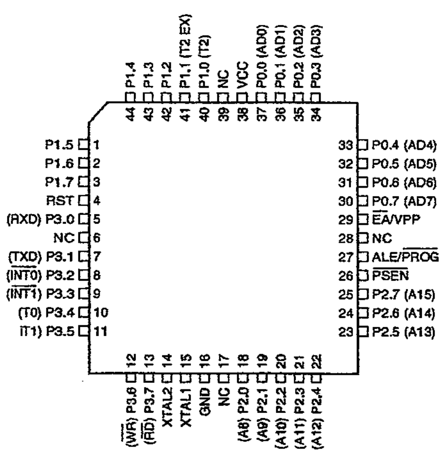
- Compatible with MCS-51™ Products
- 8K Bytes of In-System Reprogrammable Flash Memory
- Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three-level Program Memory Lock
- 256 x 8-bit Internal RAM
- 32 Programmable I/O Lines
- Three 16-bit Timer/Counters
- Eight Interrupt Sources
- Programmable Serial Channel
- Low-power Idle and Power-down Modes

)escription

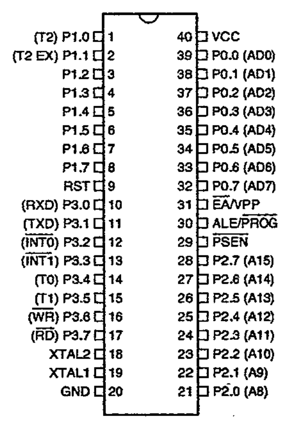
'he AT89C52 is a low-power, high-performance CMOS 8-bit microcomputer with 8K ytes of Flash programmable and erasable read only memory (PEROM). The device ; manufactured using Atmel's high-density nonvolatile memory technology and is ompatible with the industry-standard 80C51 and 80C52 instruction set and pinout. 'he on-chip Flash allows the program memory to be reprogrammed in-system or by a onventional nonvolatile memory programmer. By combining a versatile 8-bit CPU /ith Flash on a monolithic chip, the Atmel AT89C52 is a powerful microcomputer /hich provides a highly-flexible and cost-effective solution to many embedded control pplications.

'in Configurations

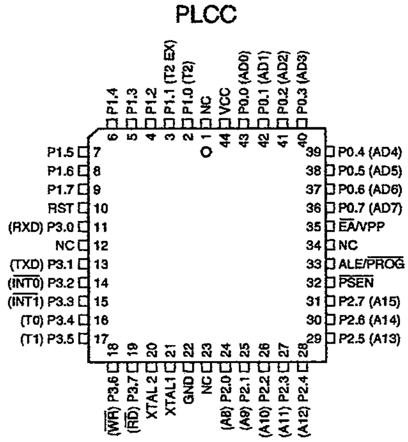
PQFP/TQFP



PDIP



PLCC

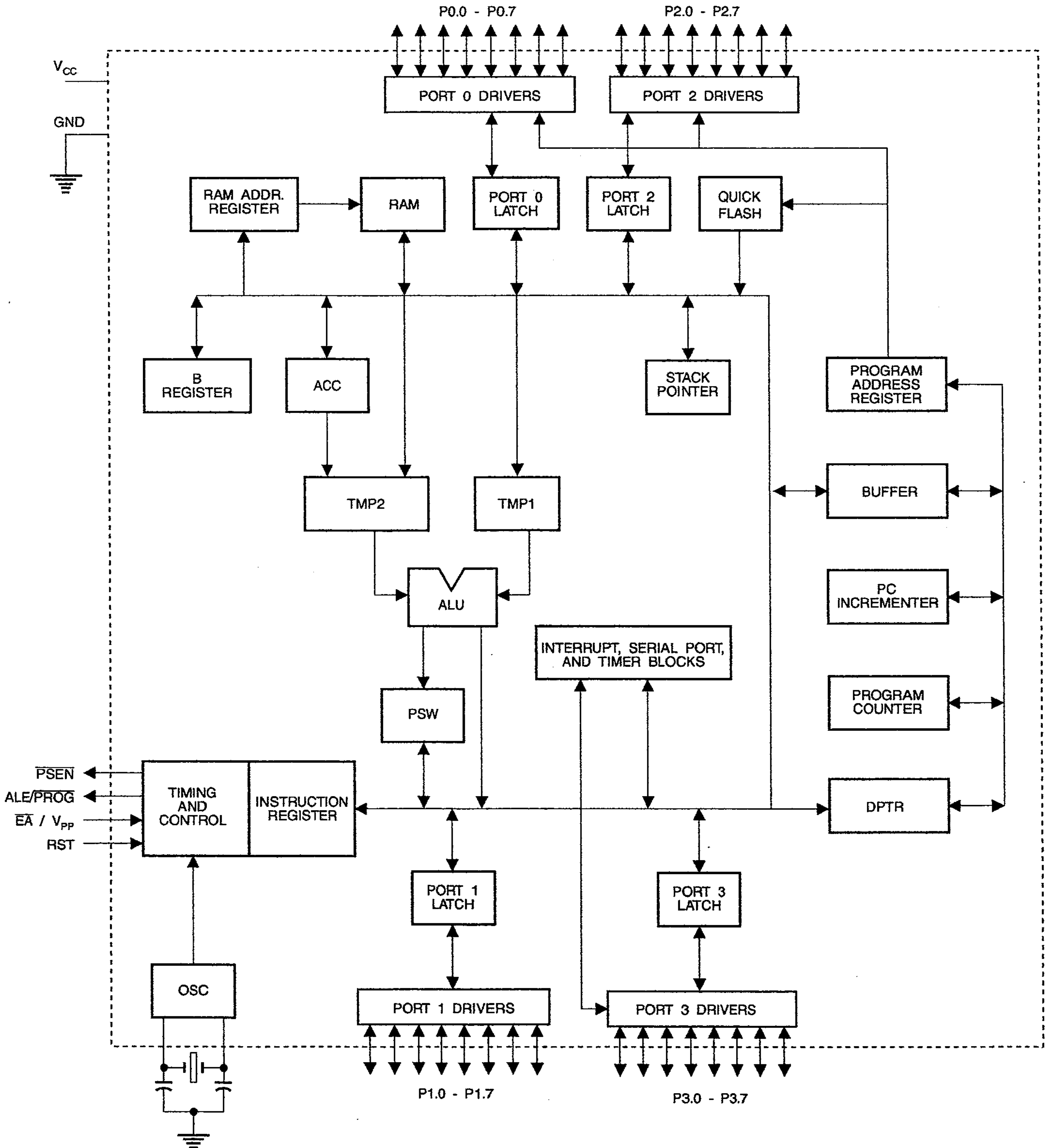


8-bit Microcontroller with 8K Bytes Flash

AT89C52

Rev. 0313H–02/00

Block Diagram



he AT89C52 provides the following standard features: 8K ytes of Flash, 256 bytes of RAM, 32 I/O lines, three 16-bit mer/counters, a six-vector two-level interrupt architecture, full-duplex serial port, on-chip oscillator, and clock cir- uitry. In addition, the AT89C52 is designed with static logic ɔr operation down to zero frequency and supports two oftware selectable power saving modes. The Idle Mode tops the CPU while allowing the RAM, timer/counters, erial port, and interrupt system to continue functioning. 'he Power-down mode saves the RAM contents but eezes the oscillator, disabling all other chip functions until ıe next hardware reset.

›in Description

'CC

ùpply voltage.

;ND

;round.

'ort 0

'ort 0 is an 8-bit open drain bi-directional I/O port. As an utput port, each pin can sink eight TTL inputs. When 1s re written to port 0 pins, the pins can be used as high- npedance inputs.

'ort 0 can also be configured to be the multiplexed low- rder address/data bus during accesses to external pro- ıram and data memory. In this mode, P0 has internal ullups.

'ort 0 also receives the code bytes during Flash program- ning and outputs the code bytes during program erification. External pullups are required during program erification.

'ort 1

'ort 1 is an 8-bit bi-directional I/O port with internal pullups. 'he Port 1 output buffers can sink/source four TTL inputs. Vhen 1s are written to Port 1 pins, they are pulled high by he internal pullups and can be used as inputs. As inputs, 'ort 1 pins that are externally being pulled low will source urrent (I_{IL}) because of the internal pullups.

n addition, P1.0 and P1.1 can be configured to be the imer/counter 2 external count input (P1.0/T2) and the imer/counter 2 trigger input (P1.1/T2EX), respectively, as hown in the following table.

'ort 1 also receives the low-order address bytes during lash programming and verification.

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)

Port 2

Port 2 is an 8-bit bi-directional I/O port with internal pullups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pul- lups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3

Port 3 is an 8-bit bi-directional I/O port with internal pullups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89C51, as shown in the following table.

Port 3 also receives some control signals for Flash pro- gramming and verification.

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{\text{INT0}}$ (external interrupt 0)
P3.3	$\overline{\text{INT1}}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{\text{WR}}$ (external data memory write strobe)
P3.7	$\overline{\text{RD}}$ (external data memory read strobe)

RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device.

ALE/$\overline{\text{PROG}}$

Address Latch Enable is an output pulse for latching the low byte of the address during accesses to external mem- ory. This pin is also the program pulse input ($\overline{\text{PROG}}$) during Flash programming.

In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external ming or clocking purposes. Note, however, that one ALE ulse is skipped during each access to external data ıemory.

 desired, ALE operation can be disabled by setting bit 0 of FR location 8EH. With the bit set, ALE is active only durıg a MOVX or MOVC instruction. Otherwise, the pin is 'eakly pulled high. Setting the ALE-disable bit has no ffect if the microcontroller is in external execution mode.

'SEN

'rogram Store Enable is the read strobe to external proram memory.

Vhen the AT89C52 is executing code from external proram memory, PSEN is activated twice each machine ycle, except that two PSEN activations are skipped during ach access to external data memory.

EA/VPP

External Access Enable. EA must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, EA will be internally latched on reset.

EA should be strapped to V_{CC} for internal program executions.

This pin also receives the 12-volt programming enable voltage (V_{PP}) during Flash programming when 12-volt programming is selected.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier.

'able 1. AT89C52 SFR Map and Reset Values

0F8H									0FFH
0F0H	B 00000000								0F7H
0E8H									0EFH
0E0H	ACC 00000000								0E7H
0D8H									0DFH
0D0H	PSW 00000000								0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000			0CFH
0C0H									0C7H
0B8H	IP XX000000								0BFH
0B0H	P3 11111111								0B7H
0A8H	IE 0X000000								0AFH
0A0H	P2 11111111								0A7H
98H	SCON 00000000	SBUF XXXXXXXX							9FH
90H	P1 11111111								97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000			8FH
80H	P0 11111111	SP 00000111	DPL 00000000	DPH 00000000				PCON 0XXX0000	87H

Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in Table 1.

Note that not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

User software should not write 1s to these unlisted locations, since they may be used in future products to invoke new features. In that case, the reset or inactive values of the new bits will always be 0.

Timer 2 Registers Control and status bits are contained in registers T2CON (shown in Table 2) and T2MOD (shown in Table 4) for Timer 2. The register pair (RCAP2H, RCAP2L) are the Capture/Reload registers for Timer 2 in 16-bit capture mode or 16-bit auto-reload mode.

Interrupt Registers The individual interrupt enable bits are in the IE register. Two priorities can be set for each of the six interrupt sources in the IP register.r

Table 2. T2CON – Timer/Counter 2 Control Registe

T2CON Address = 0C8H — Reset Value = 0000 0000B

Bit Addressable

Bit	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$
	7	6	5	4	3	2	1	0

Symbol	Function
TF2	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.
EXF2	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).
RCLK	Receive clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in serial port Modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.
TCLK	Transmit clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in serial port Modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
EXEN2	Timer 2 external enable. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
TR2	Start/Stop control for Timer 2. TR2 = 1 starts the timer.
C/$\overline{T2}$	Timer or counter select for Timer 2. C/$\overline{T2}$ = 0 for timer function. C/$\overline{T2}$ = 1 for external event counter (falling edge triggered).
CP/$\overline{RL2}$	Capture/Reload select. CP/$\overline{RL2}$ = 1 causes captures to occur on negative transitions at T2EX if EXEN2 = 1. CP/$\overline{RL2}$ = 0 causes automatic reloads to occur when Timer 2 overflows or negative transitions occur at T2EX when EXEN2 = 1. When either RCLK or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

Data Memory

The AT89C52 implements 256 bytes of on-chip RAM. The upper 128 bytes occupy a parallel address space to the Special Function Registers. That means the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space.

When an instruction accesses an internal location above address 7FH, the address mode used in the instruction specifies whether the CPU accesses the upper 128 bytes of RAM or the SFR space. Instructions that use direct addressing access SFR space.

For example, the following direct addressing instruction accesses the SFR at location 0A0H (which is P2).

```
MOV 0A0H, #data
```

ıstructions that use indirect addressing access the upper 28 bytes of RAM. For example, the following indirect ddressing instruction, where R0 contains 0A0H, accesses ıe data byte at address 0A0H, rather than P2 (whose ddress is 0A0H).

```
MOV @R0, #data
```

lote that stack operations are examples of indirect ddressing, so the upper 128 bytes of data RAM are available as stack space.

Timer 0 and 1

'imer 0 and Timer 1 in the AT89C52 operate the same way s Timer 0 and Timer 1 in the AT89C51.

Timer 2

'imer 2 is a 16-bit Timer/Counter that can operate as either timer or an event counter. The type of operation is elected by bit C/$\overline{T2}$ in the SFR T2CON (shown in Table 2). 'imer 2 has three operating modes: capture, auto-reload up or down counting), and baud rate generator. The nodes are selected by bits in T2CON, as shown in Table 3.

'imer 2 consists of two 8-bit registers, TH2 and TL2. In the 'imer function, the TL2 register is incremented every nachine cycle. Since a machine cycle consists of 12 oscillator periods, the count rate is 1/12 of the oscillator requency.

able 3. Timer 2 Operating Modes

RCLK +TCLK	CP/$\overline{RL2}$	TR2	MODE
0	0	1	16-bit Auto-reload
0	1	1	16-bit Capture
1	X	1	Baud Rate Generator
X	X	0	(Off)

n the Counter function, the register is incremented in esponse to a 1-to-0 transition at its corresponding external input pin, T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during S3P1 of the cycle following the one in which the transition was detected. Since two machine cycles (24 oscillator periods) are required to recognize a 1-to-0 transition, the maximum count rate is 1/24 of the oscillator frequency. To ensure that a given level is sampled at least once before it changes, the level should be held for at least one full machine cycle.

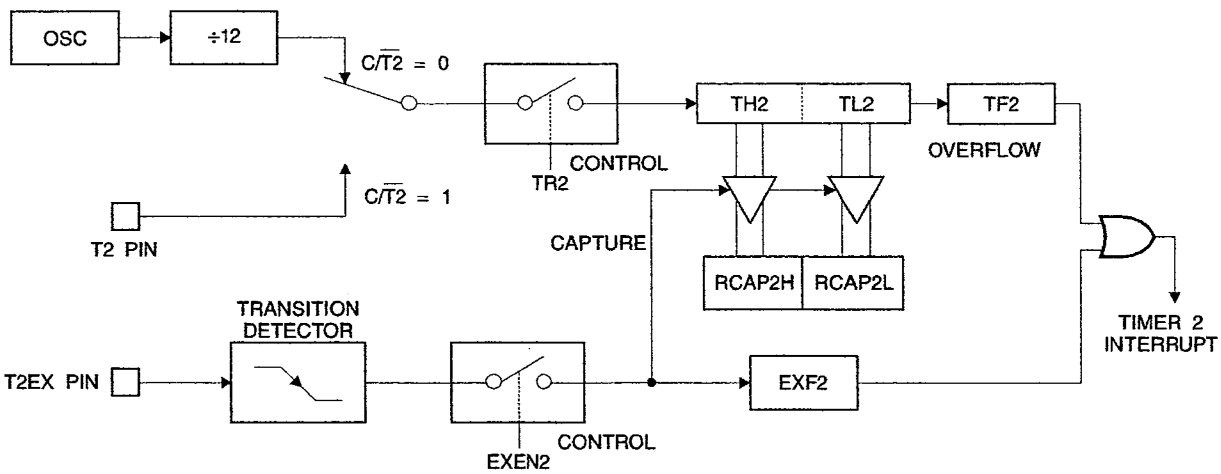
Capture Mode

In the capture mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 is a 16-bit timer or counter which upon overflow sets bit TF2 in T2CON. This bit can then be used to generate an interrupt. If EXEN2 = 1, Timer 2 performs the same operation, but a 1-to-0 transition at external input T2EX also causes the current value in TH2 and TL2 to be captured into RCAP2H and RCAP2L, respectively. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set. The EXF2 bit, like TF2, can generate an interrupt. The capture mode is illustrated in Figure 1.

Auto-reload (Up or Down Counter)

Timer 2 can be programmed to count up or down when configured in its 16-bit auto-reload mode. This feature is invoked by the DCEN (Down Counter Enable) bit located in the SFR T2MOD (see Table 4). Upon reset, the DCEN bit is set to 0 so that timer 2 will default to count up. When DCEN is set, Timer 2 can count up or down, depending on the value of the T2EX pin.

igure 1. Timer in Capture Mode



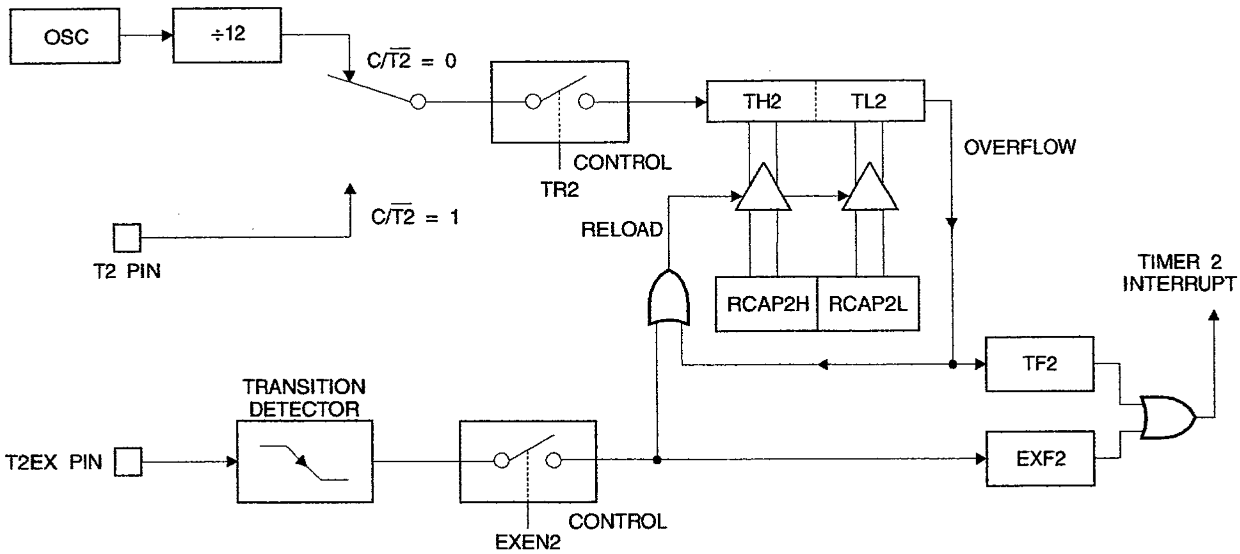
igure 2 shows Timer 2 automatically counting up when)CEN = 0. In this mode, two options are selected by bit :XEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to FFFFH and then sets the TF2 bit upon overflow. The verflow also causes the timer registers to be reloaded with ie 16-bit value in RCAP2H and RCAP2L. The values in 'imer in Capture ModeRCAP2H and RCAP2L are preset y software. If EXEN2 = 1, a 16-bit reload can be triggered ither by an overflow or by a 1-to-0 transition at external iput T2EX. This transition also sets the EXF2 bit. Both the 'F2 and EXF2 bits can generate an interrupt if enabled.

ietting the DCEN bit enables Timer 2 to count up or down, s shown in Figure 3. In this mode, the T2EX pin controls the direction of the count. A logic 1 at T2EX makes Timer 2 count up. The timer will overflow at 0FFFFH and set the TF2 bit. This overflow also causes the 16-bit value in RCAP2H and RCAP2L to be reloaded into the timer registers, TH2 and TL2, respectively.

A logic 0 at T2EX makes Timer 2 count down. The timer underflows when TH2 and TL2 equal the values stored in RCAP2H and RCAP2L. The underflow sets the TF2 bit and causes 0FFFFH to be reloaded into the timer registers.

The EXF2 bit toggles whenever Timer 2 overflows or underflows and can be used as a 17th bit of resolution. In this operating mode, EXF2 does not flag an interrupt.

igure 2. Timer 2 Auto Reload Mode (DCEN = 0)



'able 4. T2MOD – Timer 2 Mode Control Register

T2MOD Address = 0C9H | Reset Value = XXXX XX00B

Not Bit Addressable

	–	–	–	–	–	–	T2OE	DCEN
Bit	7	6	5	4	3	2	1	0

Symbol	Function
–	Not implemented, reserved for future
T2OE	Timer 2 Output Enable bit.
DCEN	When set, this bit allows Timer 2 to be configured as an up/down counter.

Figure 3. Timer 2 Auto Reload Mode (DCEN = 1)

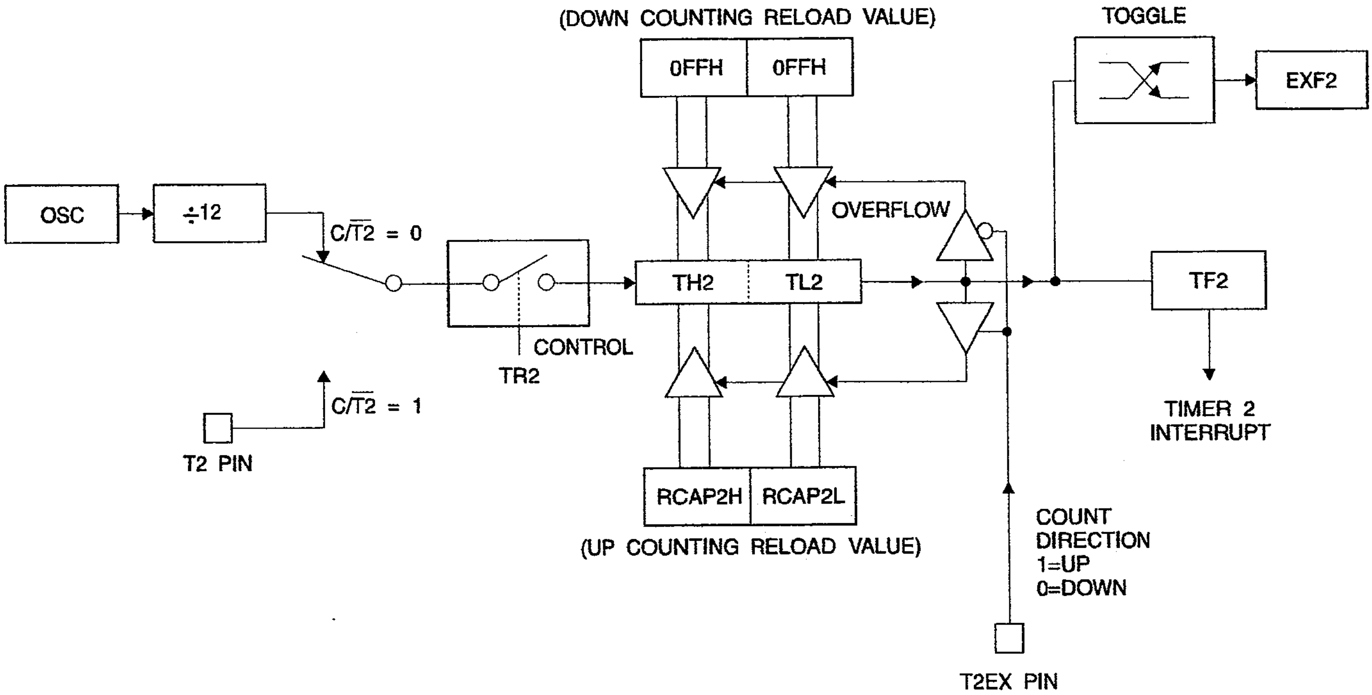
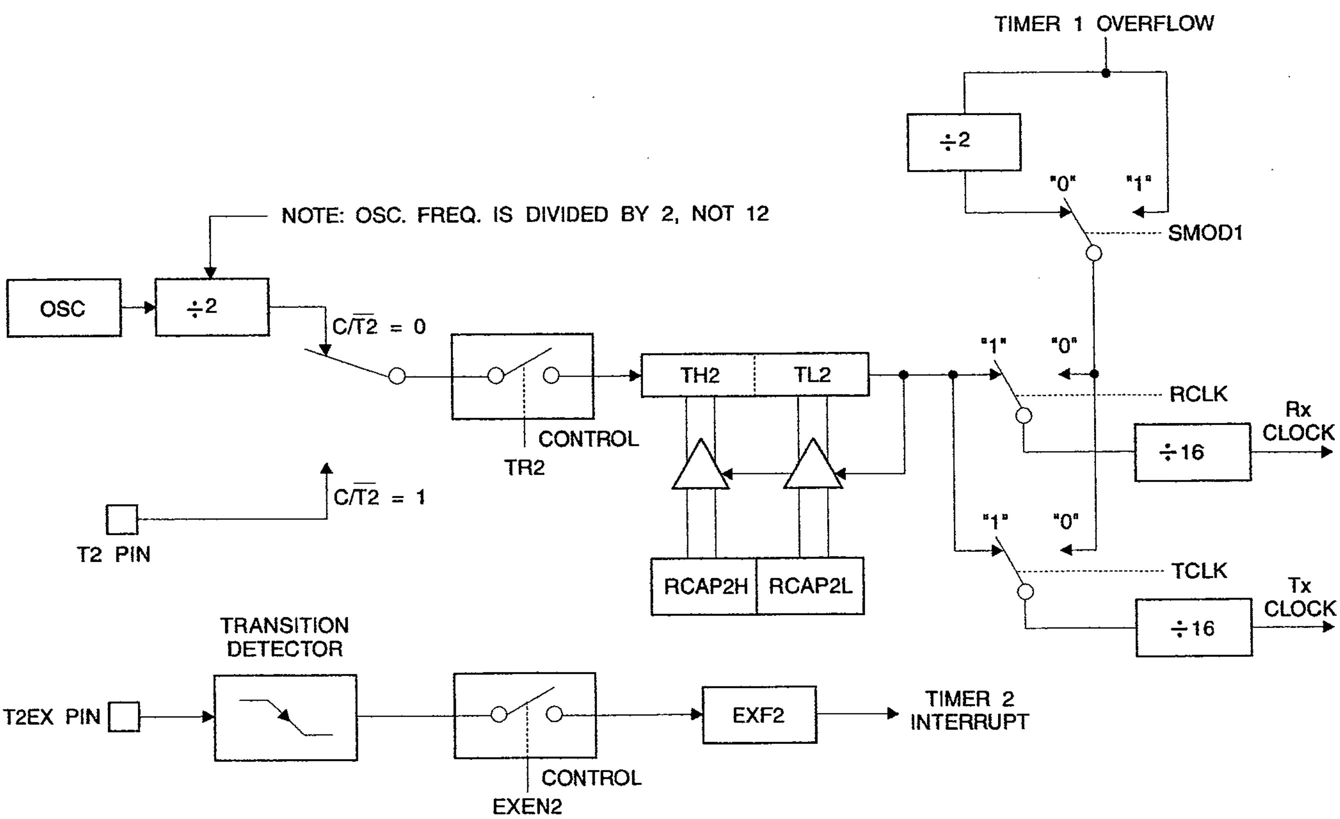


Figure 4. Timer 2 in Baud Rate Generator Mode



Baud Rate Generator

Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (Table 2). Note that the baud rates for transmit and receive can be different if Timer 2 is used for the receiver or transmitter and Timer 1 is used for the other function. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode, as shown in Figure 4.

The baud rate generator mode is similar to the auto-reload mode, in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

The baud rates in Modes 1 and 3 are determined by Timer 2's overflow rate according to the following equation.

$$\text{Modes 1 and 3 Baud Rates} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

The Timer can be configured for either timer or counter operation. In most applications, it is configured for timer operation ($C/\overline{T2} = 0$). The timer operation is different for Timer 2 when it is used as a baud rate generator. Normally, as a timer, it increments every machine cycle (at 1/12 the oscillator frequency). As a baud rate generator, however, it increments every state time (at 1/2 the oscillator frequency). The baud rate formula is given below.

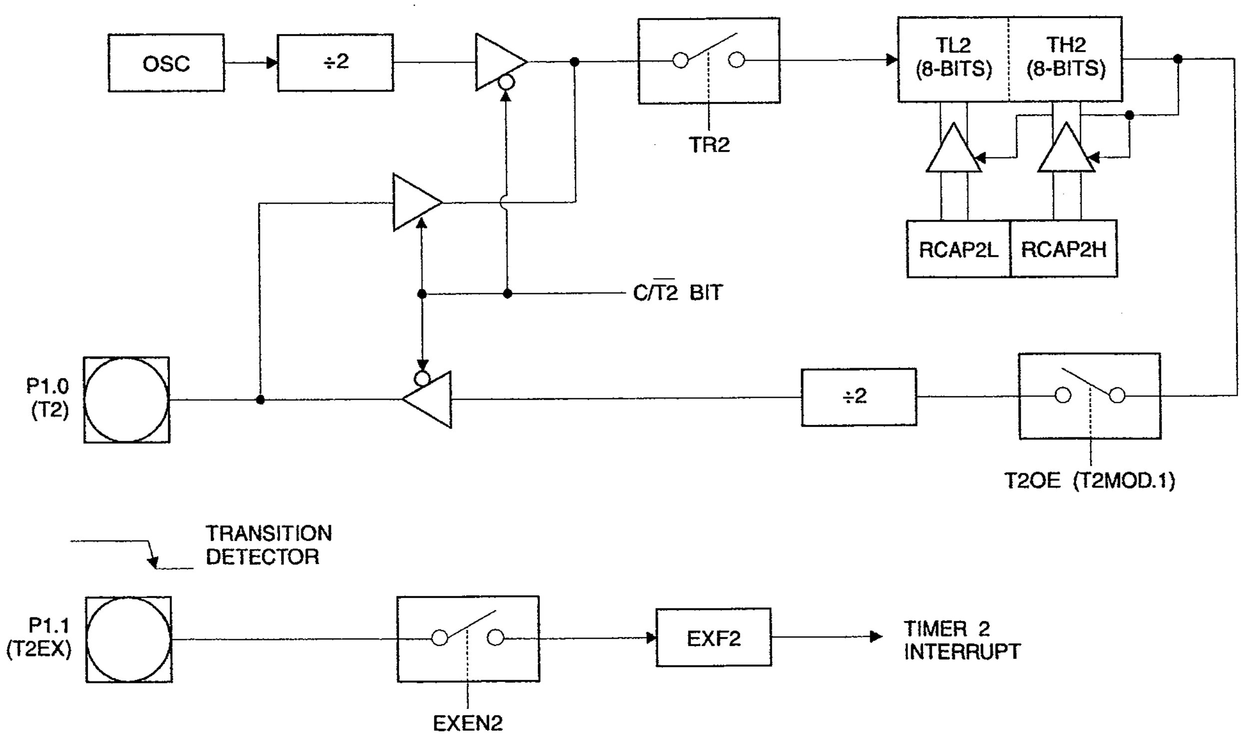
$$\frac{\text{Modes 1 and 3}}{\text{Baud Rate}} = \frac{\text{Oscillator Frequency}}{32 \times [65536 - (\text{RCAP2H,RCAP2L})]}$$

where (RCAP2H, RCAP2L) is the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

Timer 2 as a baud rate generator is shown in Figure 4. This figure is valid only if RCLK or TCLK = 1 in T2CON. Note that a rollover in TH2 does not set TF2 and will not generate an interrupt. Note too, that if EXEN2 is set, a 1-to-0 transition in T2EX will set EXF2 but will not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2). Thus when Timer 2 is in use as a baud rate generator, T2EX can be used as an extra external interrupt.

Note that when Timer 2 is running (TR2 = 1) as a timer in the baud rate generator mode, TH2 or TL2 should not be read from or written to. Under these conditions, the Timer is incremented every state time, and the results of a read or write may not be accurate. The RCAP2 registers may be read but should not be written to, because a write might overlap a reload and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing the Timer 2 or RCAP2 registers.

Figure 5. Timer 2 in Clock-out Mode



Programmable Clock Out

A 50% duty cycle clock can be programmed to come out on P1.0, as shown in Figure 5. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed to input the external clock for Timer/Counter 2 or to output a 50% duty cycle clock ranging from 61 Hz to 4 MHz at a 16 MHz operating frequency.

To configure the Timer/Counter 2 as a clock generator, bit $C/\overline{T2}$ (T2CON.1) must be cleared and bit T2OE (T2MOD.1) must be set. Bit TR2 (T2CON.2) starts and stops the timer.

The clock-out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L), as shown in the following equation.

$$\text{Clock-Out Frequency} = \frac{\text{Oscillator Fequency}}{4 \times [65536 - (\text{RCAP2H,RCAP2L})]}$$

In the clock-out mode, Timer 2 roll-overs will not generate an interrupt. This behavior is similar to when Timer 2 is used as a baud-rate generator. It is possible to use Timer 2 as a baud-rate generator and a clock generator simultaneously. Note, however, that the baud-rate and clock-out frequencies cannot be determined independently from one another since they both use RCAP2H and RCAP2L.

UART

The UART in the AT89C52 operates the same way as the UART in the AT89C51.

Interrupts

The AT89C52 has a total of six interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt. These interrupts are all shown in Figure 6.

Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once.

Note that Table shows that bit position IE.6 is unimplemented. In the AT89C51, bit position IE.5 is also unimplemented. User software should not write 1s to these bit positions, since they may be used in future AT89 products.

Timer 2 interrupt is generated by the logical OR of bits TF2 and EXF2 in register T2CON. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine may have to determine whether it was TF2 or EXF2 that generated the interrupt, and that bit will have to be cleared in software.

The Timer 0 and Timer 1 flags, TF0 and TF1, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle. However, the Timer 2 flag, TF2, is set at S2P2 and is polled in the same cycle in which the timer overflows.

Table 5. Interrupt Enable (IE) Register

(MSB) (LSB)

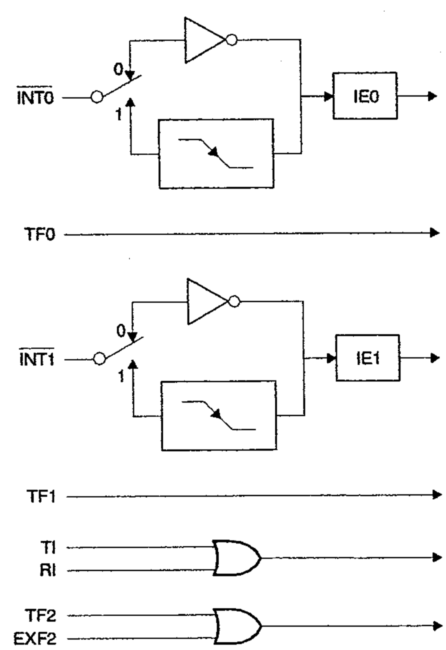
EA	–	ET2	ES	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.

Enable Bit = 0 disables the interrupt.

Symbol	Position	Function
EA	IE.7	Disables all interrupts. If EA = 0, no interrupt is acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
–	IE.6	Reserved.
ET2	IE.5	Timer 2 interrupt enable bit.
ES	IE.4	Serial Port interrupt enable bit.
ET1	IE.3	Timer 1 interrupt enable bit.
EX1	IE.2	External interrupt 1 enable bit.
ET0	IE.1	Timer 0 interrupt enable bit.
EX0	IE.0	External interrupt 0 enable bit.
User software should never write 1s to unimplemented bits, because they may be used in future AT89 products.		

Figure 6. Interrupt Sources



Oscillator Characteristics

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 7. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 8. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Idle Mode

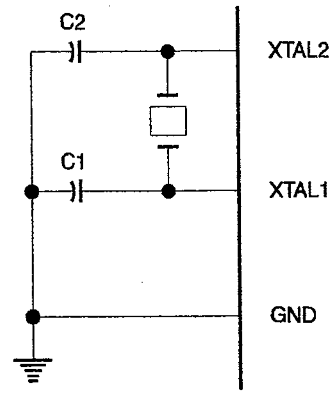
In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

Note that when idle mode is terminated by a hardware reset, the device normally resumes program execution from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when idle mode is terminated by a reset, the instruction following the one that invokes idle mode should not write to a port pin or to external memory.

Power-down Mode

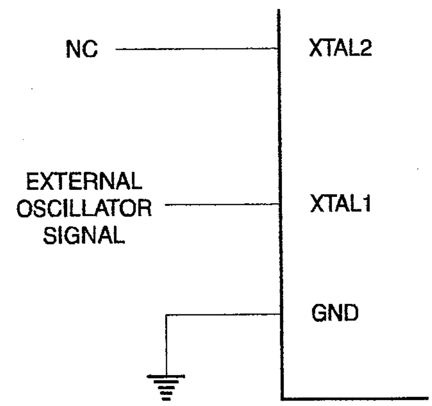
In the power-down mode, the oscillator is stopped, and the instruction that invokes power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the power-down mode is terminated. The only exit from power-down is a hardware reset. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

Figure 7. Oscillator Connections



Note: C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 8. External Clock Drive Configuration



Status of External Pins During Idle and Powe-down Modes

Mode	Program Memory	ALE	$\overline{PSEN}$	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-down	Internal	0	0	Data	Data	Data	Data
Power-down	External	0	0	Float	Data	Data	Data

Program Memory Lock Bits

The AT89C52 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the following table.

Lock Bit Protection Modes

Program Lock Bits				
	LB1	LB2	LB3	Protection Type
1	U	U	U	No program lock features.
2	P	U	U	MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory, $\overline{EA}$ is sampled and latched on reset, and further programming of the Flash memory is disabled.
3	P	P	U	Same as mode 2, but verify is also disabled.
4	P	P	P	Same as mode 3, but external execution is also disabled.

When lock bit 1 is programmed, the logic level at the $\overline{EA}$ pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value and holds that value until reset is activated. The latched value of $\overline{EA}$ must agree with the current logic level at that pin in order for the device to function properly.

Programming the Flash

The AT89C52 is normally shipped with the on-chip Flash memory array in the erased state (that is, contents = FFH) and ready to be programmed. The programming interface accepts either a high-voltage (12-volt) or a low-voltage (V_{CC}) program enable signal. The Low-voltage programming mode provides a convenient way to program the AT89C52 inside the user's system, while the high-voltage programming mode is compatible with conventional third-party Flash or EPROM programmers.

The AT89C52 is shipped with either the high-voltage or low-voltage programming mode enabled. The respective top-side marking and device signature codes are listed in the following table.

	V_{PP} = 12V	V_{PP} = 5V
Top-side Mark	AT89C52 xxxx yyww	AT89C52 xxxx - 5 yyww
Signature	(030H) = 1EH (031H) = 52H (032H) = FFH	(030H) = 1EH (031H) = 52H (032H) = 05H

The AT89C52 code memory array is programmed byte-by-byte in either programming mode. *To program any non-blank byte in the on-chip Flash Memory, the entire memory must be erased using the Chip Erase Mode.*

Programming Algorithm Before programming the AT89C52, the address, data and control signals should be set up according to the Flash programming mode table and Figure 9 and Figure 10. To program the AT89C52, take the following steps.

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.
3. Activate the correct combination of control signals.
4. Raise $\overline{EA}/V_{PP}$ to 12V for the high-voltage programming mode.
5. Pulse ALE/$\overline{PROG}$ once to program a byte in the Flash array or the lock bits. The byte-write cycle is self-timed and typically takes no more than 1.5 ms. Repeat steps 1 through 5, changing the address and data for the entire array or until the end of the object file is reached.

$\overline{\text{Data}}$ Polling The AT89C52 features $\overline{\text{Data}}$ Polling to indicate the end of a write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written data on PO.7. Once the write cycle has been completed, true data is valid on all outputs, and the next cycle may begin. $\overline{\text{Data}}$ Polling may begin any time after a write cycle has been initiated.

Ready/$\overline{\text{Busy}}$ The progress of byte programming can also be monitored by the RDY/$\overline{BSY}$ output signal. P3.4 is pulled low after ALE goes high during programming to indicate $\overline{BUSY}$. P3.4 is pulled high again when programming is done to indicate READY.

Program Verify If lock bits LB1 and LB2 have not been programmed, the programmed code data can be read back via the address and data lines for verification. The lock bits cannot be verified directly. Verification of the lock bits is achieved by observing that their features are enabled.

Chip Erase The entire Flash array is erased electrically by using the proper combination of control signals and by holding ALE/$\overline{PROG}$ low for 10 ms. The code array is written with all 1s. The chip erase operation must be executed before the code memory can be reprogrammed.

Reading the Signature Bytes The signature bytes are read by the same procedure as a normal verification of locations 030H, 031H, and 032H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows.

(030H) = 1EH indicates manufactured by Atmel
(031H) = 52H indicates 89C52
(032H) = FFH indicates 12V programming
(032H) = 05H indicates 5V programming

Programming Interface

Every code byte in the Flash array can be written, and the entire array can be erased, by using the appropriate combination of control signals. The write operation cycle is self-timed and once initiated, will automatically time itself to completion.

All major programming vendors offer worldwide support for the Atmel microcontroller series. Please contact your local programming vendor for the appropriate software revision.

Flash Programming Modes

Mode		RST	$\overline{PSEN}$	ALE/$\overline{PROG}$	$\overline{EA}/V_{PP}$	P2.6	P2.7	P3.6	P3.7
Write Code Data		H	L		H/12V	L	H	H	H
Read Code Data		H	L	H	H	L	L	H	H
Write Lock	Bit - 1	H	L		H/12V	H	H	H	H
	Bit - 2	H	L		H/12V	H	H	L	L
	Bit - 3	H	L		H/12V	H	L	H	L
Chip Erase		H	L	(1)	H/12V	H	L	L	L
Read Signature Byte		H	L	H	H	L	L	L	L

Note: 1. Chip Erase requires a 10 ms $\overline{PROG}$ pulse.

Figure 9. Programming the Flash Memory

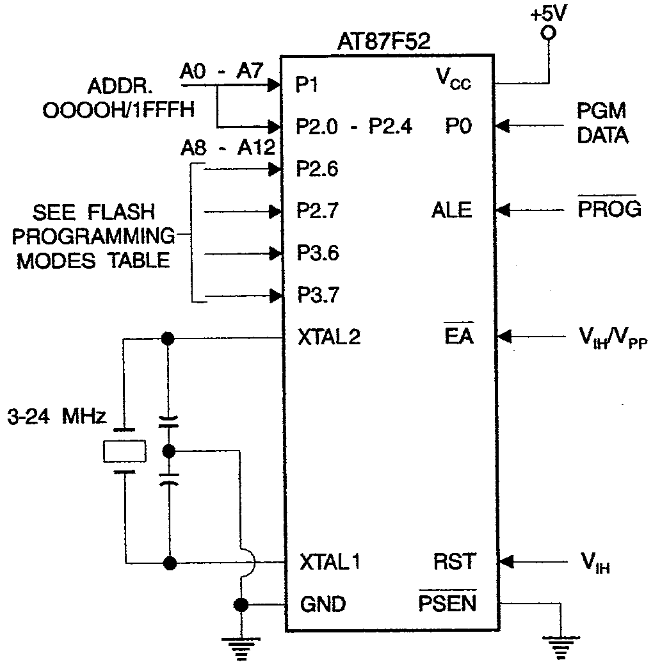
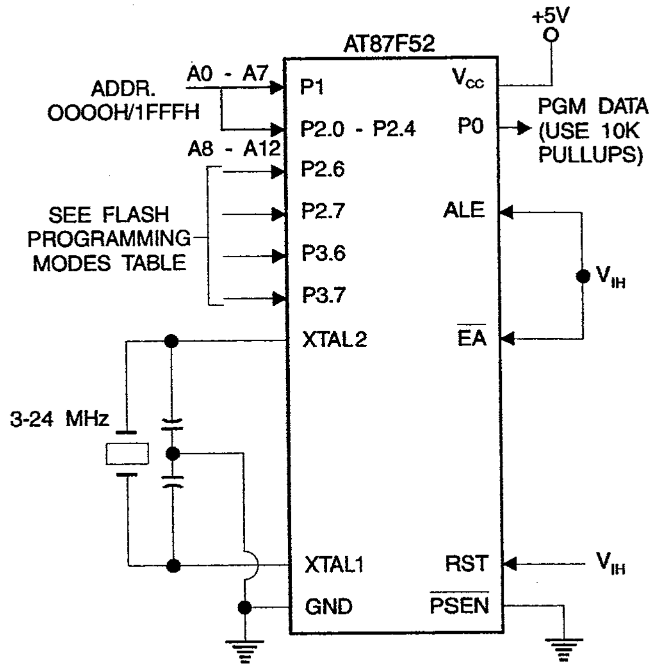


Figure 10. Verifying the Flash Memory



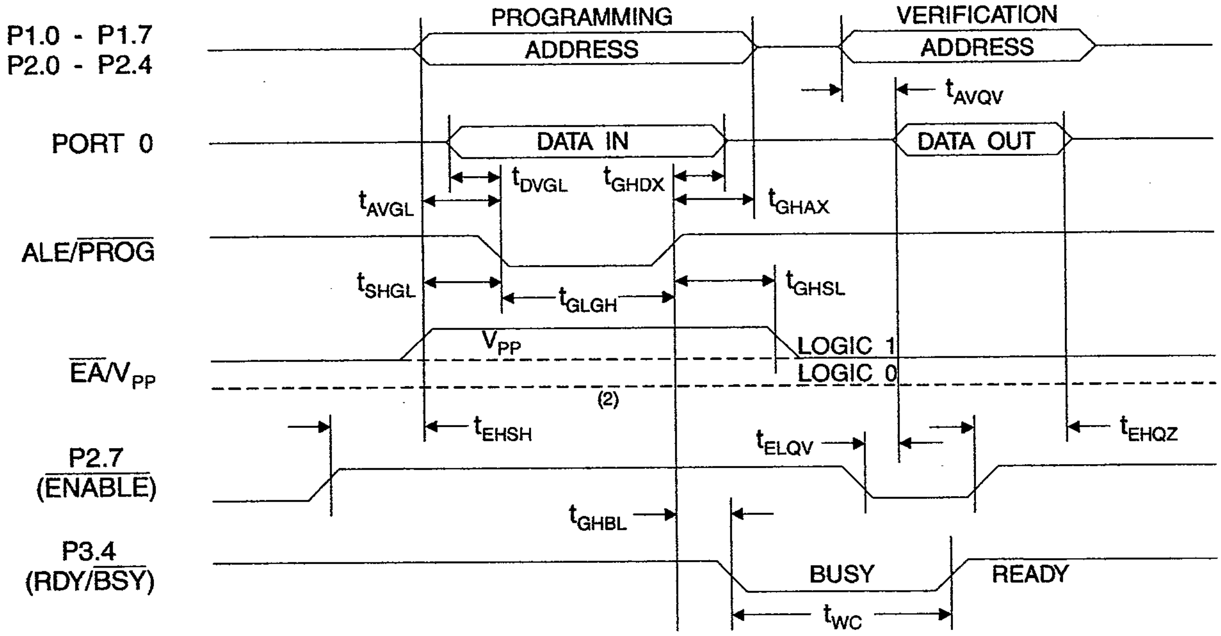
Flash Programming and Verification Characteristics

$T_A = 0°C$ to $70°C$, $V_{CC} = 5.0 \pm 10\%$

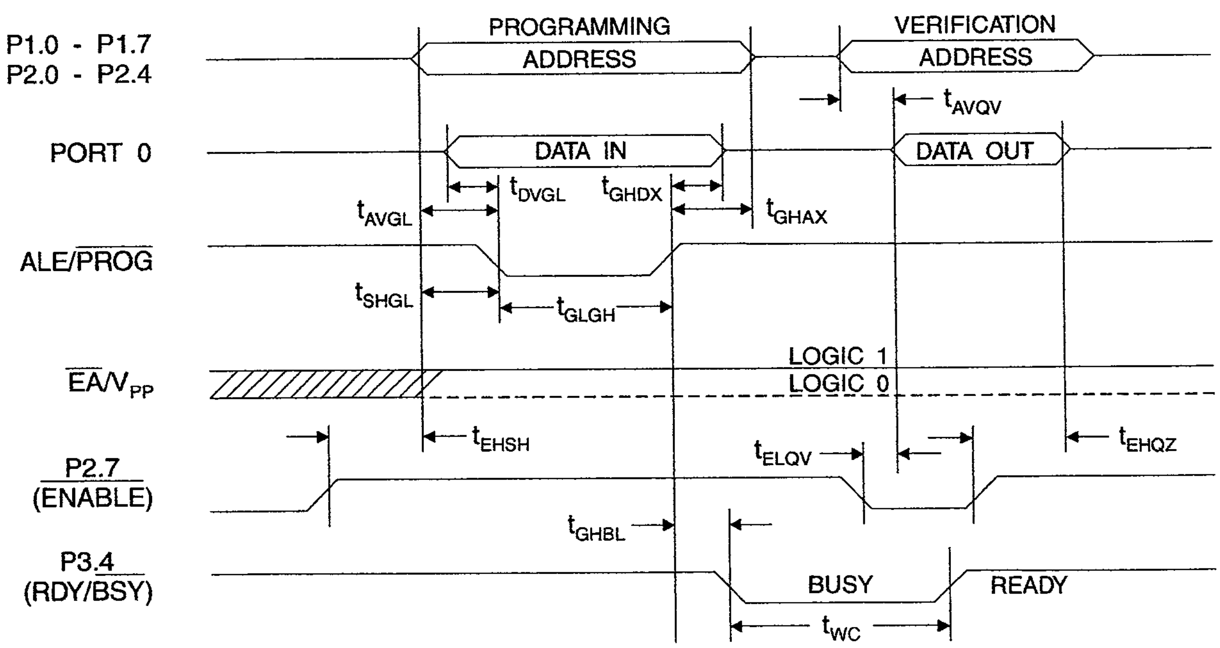
Symbol	Parameter	Min	Max	Units
V_{PP}(1)	Programming Enable Voltage	11.5	12.5	V
I_{PP}(1)	Programming Enable Current		1.0	mA
$1/t_{CLCL}$	Oscillator Frequency	3	24	MHz
t_{AVGL}	Address Setup to $\overline{\text{PROG}}$ Low	$48t_{CLCL}$		
t_{GHAX}	Address Hold after $\overline{\text{PROG}}$	$48t_{CLCL}$		
t_{DVGL}	Data Setup to $\overline{\text{PROG}}$ Low	$48t_{CLCL}$		
t_{GHDX}	Data Hold After $\overline{\text{PROG}}$	$48t_{CLCL}$		
t_{EHSH}	P2.7 ($\overline{\text{ENABLE}}$) High to V_{PP}	$48t_{CLCL}$		
t_{SHGL}	V_{PP} Setup to $\overline{\text{PROG}}$ Low	10		μs
t_{GHSL}(1)	V_{PP} Hold after $\overline{\text{PROG}}$	10		μs
t_{GLGH}	$\overline{\text{PROG}}$ Width	1	110	μs
t_{AVQV}	Address to Data Valid		$48t_{CLCL}$	
t_{ELQV}	$\overline{\text{ENABLE}}$ Low to Data Valid		$48t_{CLCL}$	
t_{EHQZ}	Data Float after $\overline{\text{ENABLE}}$	0	$48t_{CLCL}$	
t_{GHBL}	$\overline{\text{PROG}}$ High to $\overline{\text{BUSY}}$ Low		1.0	μs
t_{WC}	Byte Write Cycle Time		2.0	ms

Note: 1. Only used in 12-volt programming mode.

Flash Programming and Verification Waveforms - High-voltage Mode (V_{PP}=12V)



Flash Programming and Verification Waveforms - Low-voltage Mode (V_{PP}=5V)



Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin with Respect to Ground	-1.0V to +7.0V
Maximum Operating Voltage	6.6V
DC Output Current	15.0 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics

The values shown in this table are valid for T_A = -40°C to 85°C and V_{CC} = 5.0V ± 20%, unless otherwise noted.

Symbol	Parameter	Condition	Min	Max	Units
V_{IL}	Input Low-voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC}-0.1	V
V_{IL1}	Input Low-voltage ($\overline{EA}$)		-0.5	0.2 V_{CC}-0.3	V
V_{IH}	Input High-voltage	(Except XTAL1, RST)	0.2 V_{CC}+0.9	V_{CC}+0.5	V
V_{IH1}	Input High-voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC}+0.5	V
V_{OL}	Output Low-voltage(1) (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
V_{OL1}	Output Low-voltage(1) (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
V_{OH}	Output High-voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 μA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -25 μA	0.75 V_{CC}		V
		I_{OH} = -10 μA	0.9 V_{CC}		V
V_{OH1}	Output High-voltage (Port 0 in External Bus Mode)	I_{OH} = -800 μA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -300 μA	0.75 V_{CC}		V
		I_{OH} = -80 μA	0.9 V_{CC}		V
I_{IL}	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	μA
I_{TL}	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, V_{CC} = 5V ± 10%		-650	μA
I_{LI}	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	μA
RRST	Reset Pulldown Resistor		50	300	KΩ
C_{IO}	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
I_{CC}	Power Supply Current	Active Mode, 12 MHz		25	mA
		Idle Mode, 12 MHz		6.5	mA
	Power-down Mode(1)	V_{CC} = 6V		100	μA
		V_{CC} = 3V		40	μA

Notes: 1. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
Maximum I_{OL} per port pin: 10 mA
Maximum I_{OL} per 8-bit port:
Port 0: 26 mA Ports 1, 2, 3: 15 mA
Maximum total I_{OL} for all output pins: 71 mA
If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.
2. Minimum V_{CC} for Power-down is 2V.

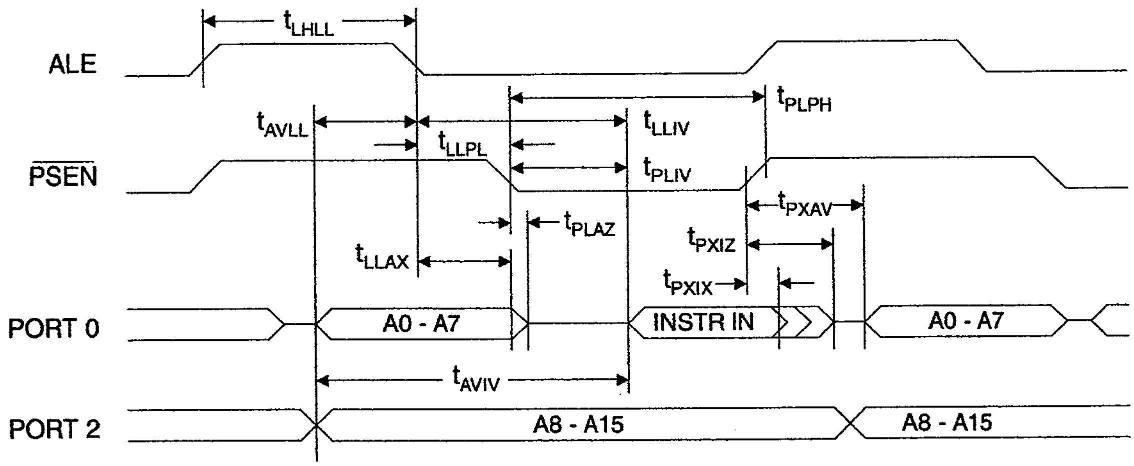
AC Characteristics

Under operating conditions, load capacitance for Port 0, ALE/$\overline{PROG}$, and $\overline{PSEN}$ = 100 pF; load capacitance for all other outputs = 80 pF.

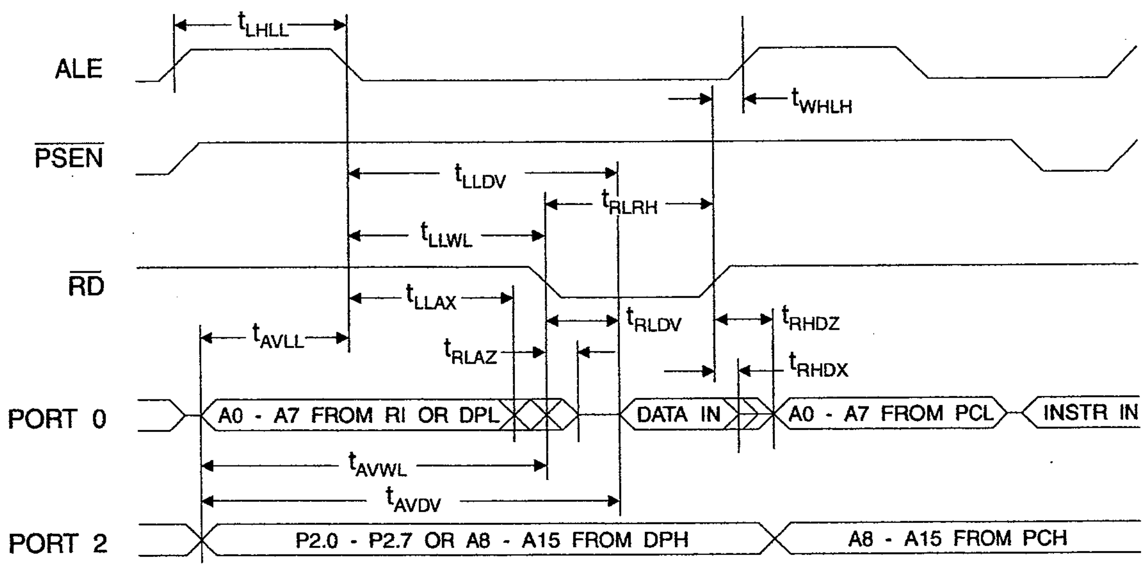
External Program and Data Memory Characteristics

Symbol	Parameter	12 MHz Oscillator		Variable Oscillator		Units
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator Frequency			0	24	MHz
t_{LHLL}	ALE Pulse Width	127		$2t_{CLCL}$-40		ns
t_{AVLL}	Address Valid to ALE Low	43		t_{CLCL}-13		ns
t_{LLAX}	Address Hold After ALE Low	48		t_{CLCL}-20		ns
t_{LLIV}	ALE Low to Valid Instruction In		233		$4t_{CLCL}$-65	ns
t_{LLPL}	ALE Low to $\overline{PSEN}$ Low	43		t_{CLCL}-13		ns
t_{PLPH}	$\overline{PSEN}$ Pulse Width	205		$3t_{CLCL}$-20		ns
t_{PLIV}	$\overline{PSEN}$ Low to Valid Instruction In		145		$3t_{CLCL}$-45	ns
t_{PXIX}	Input Instruction Hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	Input Instruction Float after $\overline{PSEN}$		59		t_{CLCL}-10	ns
t_{PXAV}	$\overline{PSEN}$ to Address Valid	75		t_{CLCL}-8		ns
t_{AVIV}	Address to Valid Instruction In		312		$5t_{CLCL}$-55	ns
t_{PLAZ}	$\overline{PSEN}$ Low to Address Float		10		10	ns
t_{RLRH}	$\overline{RD}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{WLWH}	$\overline{WR}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{RLDV}	$\overline{RD}$ Low to Valid Data In		252		$5t_{CLCL}$-90	ns
t_{RHDX}	Data Hold After $\overline{RD}$	0		0		ns
t_{RHDZ}	Data Float After $\overline{RD}$		97		$2t_{CLCL}$-28	ns
t_{LLDV}	ALE Low to Valid Data In		517		$8t_{CLCL}$-150	ns
t_{AVDV}	Address to Valid Data In		585		$9t_{CLCL}$-165	ns
t_{LLWL}	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	200	300	$3t_{CLCL}$-50	$3t_{CLCL}$+50	ns
t_{AVWL}	Address to $\overline{RD}$ or $\overline{WR}$ Low	203		$4t_{CLCL}$-75		ns
t_{QVWX}	Data Valid to $\overline{WR}$ Transition	23		t_{CLCL}-20		ns
t_{QVWH}	Data Valid to $\overline{WR}$ High	433		$7t_{CLCL}$-120		ns
t_{WHQX}	Data Hold After $\overline{WR}$	33		t_{CLCL}-20		ns
t_{RLAZ}	$\overline{RD}$ Low to Address Float		0		0	ns
t_{WHLH}	$\overline{RD}$ or $\overline{WR}$ High to ALE High	43	123	t_{CLCL}-20	t_{CLCL}+25	ns

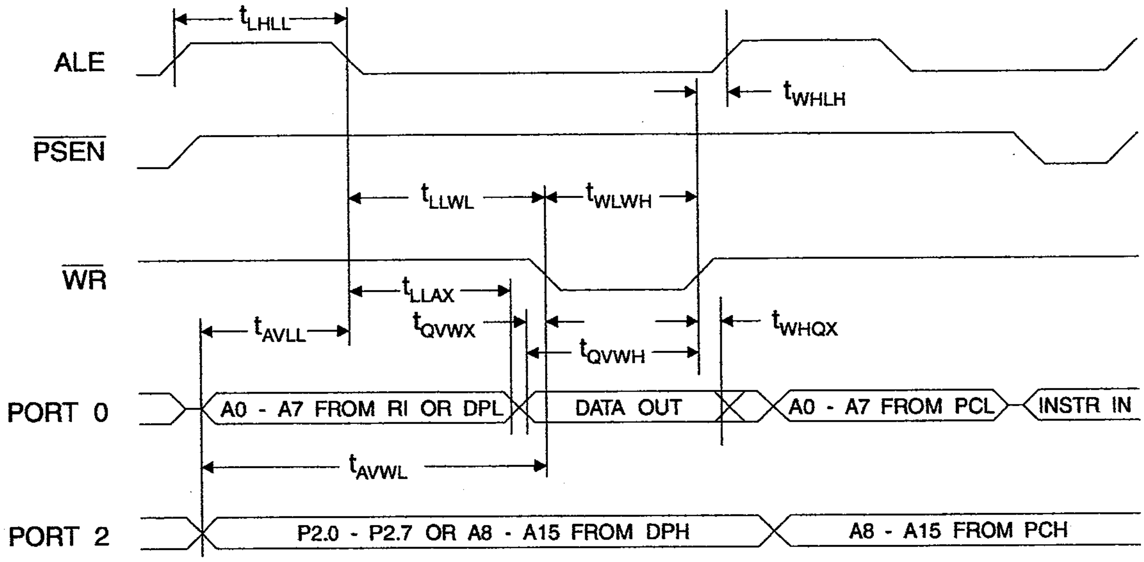
External Program Memory Read Cycle



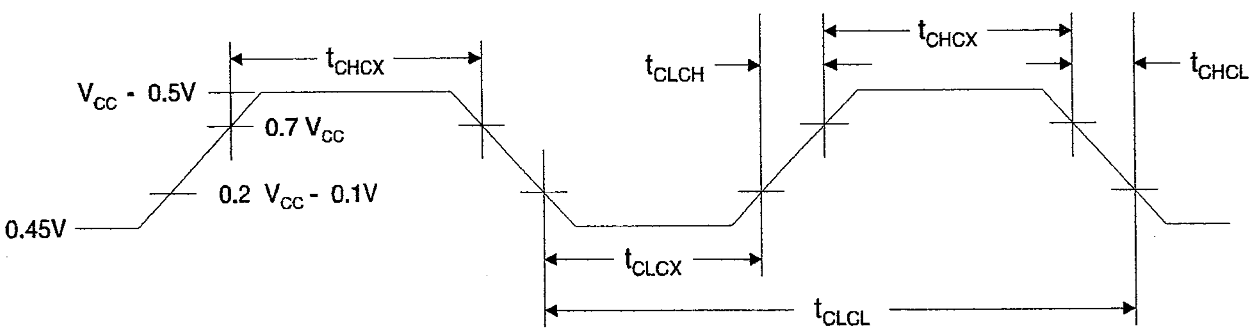
External Data Memory Read Cycle



External Data Memory Write Cycle



External Clock Drive Waveforms



External Clock Drive

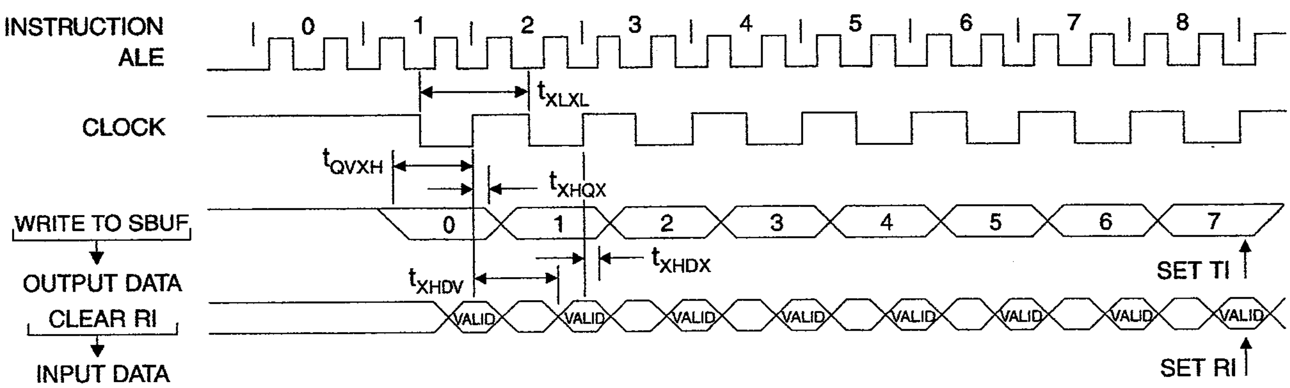
Symbol	Parameter	Min	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{CLCL}	Clock Period	41.6		ns
t_{CHCX}	High Time	15		ns
t_{CLCX}	Low Time	15		ns
t_{CLCH}	Rise Time		20	ns
t_{CHCL}	Fall Time		20	ns

Serial Port Timing: Shift Register Mode Test Conditions

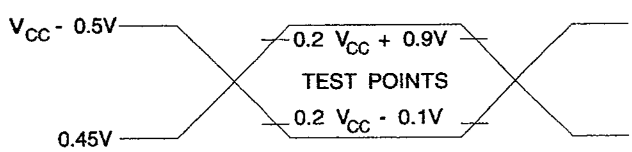
The values in this table are valid for V_{CC} = 5.0V ± 20% and Load Capacitance = 80 pF.

Symbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
t_{XLXL}	Serial Port Clock Cycle Time	1.0		$12t_{CLCL}$		μs
t_{QVXH}	Output Data Setup to Clock Rising Edge	700		$10t_{CLCL}$-133		ns
t_{XHQX}	Output Data Hold After Clock Rising Edge	50		$2t_{CLCL}$-117		ns
t_{XHDX}	Input Data Hold After Clock Rising Edge	0		0		ns
t_{XHDV}	Clock Rising Edge to Input Data Valid		700		$10t_{CLCL}$-133	ns

Shift Register Mode Timing Waveforms

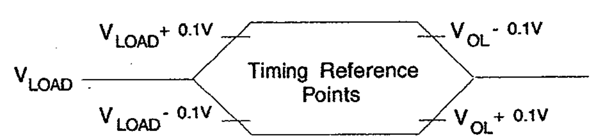


AC Testing Input/Output Waveforms(1)



Note: 1. AC Inputs during testing are driven at V_{CC} - 0.5V for a logic 1 and 0.45V for a logic 0. Timing measurements are made at V_{IH} min. for a logic 1 and V_{IL} max. for a logic 0.

Float Waveforms(1)



Note: 1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs.

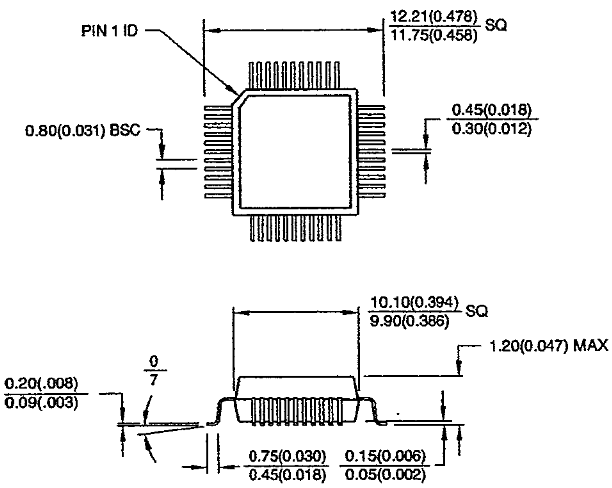
Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
12	5V ± 20%	AT89C52-12AC	44A	Commercial (0°C to 70°C)
		AT89C52-12JC	44J	
		AT89C52-12PC	40P6	
		AT89C52-12QC	44Q	
		AT89C52-12AI	44A	Industrial (-40°C to 85°C)
		AT89C52-12JI	44J	
		AT89C52-12PI	40P6	
		AT89C52-12QI	44Q	
16	5V ± 20%	AT89C52-16AC	44A	Commercial (0°C to 70°C)
		AT89C52-16JC	44J	
		AT89C52-16PC	40P6	
		AT89C52-16QC	44Q	
		AT89C52-16AI	44A	Industrial (-40°C to 85°C)
		AT89C52-16JI	44J	
		AT89C52-16PI	40P6	
		AT89C52-16QI	44Q	
20	5V ± 20%	AT89C52-20AC	44A	Commercial (0°C to 70°C)
		AT89C52-20JC	44J	
		AT89C52-20PC	40P6	
		AT89C52-20QC	44Q	
		AT89C52-20AI	44A	Industrial (-40°C to 85°C)
		AT89C52-20JI	44J	
		AT89C52-20PI	40P6	
		AT89C52-20QI	44Q	
24	5V ± 20%	AT89C52-24AC	44A	Commercial (0°C to 70°C)
		AT89C52-24JC	44J	
		AT89C52-24PC	40P6	
		AT89C52-24QC	44Q	
		AT89C52-24AI	44A	Industrial (-40°C to 85°C)
		AT89C52-24JI	44J	
		AT89C52-24PI	40P6	
		AT89C52-24QI	44Q	

Package Type	
44A	44-lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
44J	44-lead, Plastic J-leaded Chip Carrier (PLCC)
40P6	40-lead, 0.600" Wide, Plastic Dual Inline Package (PDIP)
44Q	44-lead, Plastic Gull Wing Quad Flatpack (PQFP)

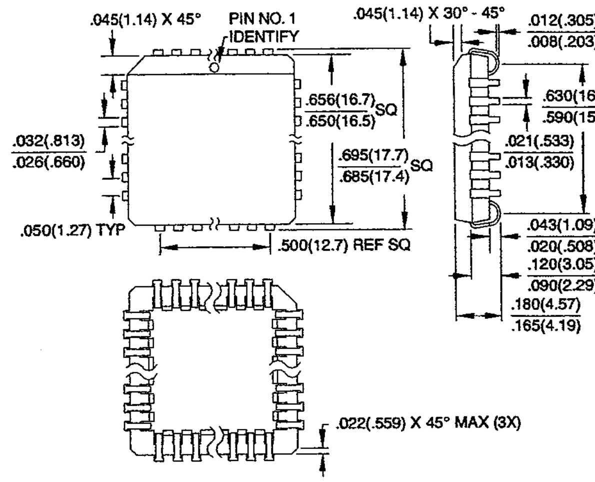
Packaging Information

44A, 44-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flatpack (TQFP)
Dimensions in Millimeters and (Inches)*
JEDEC STANDARD MS-026 ACB

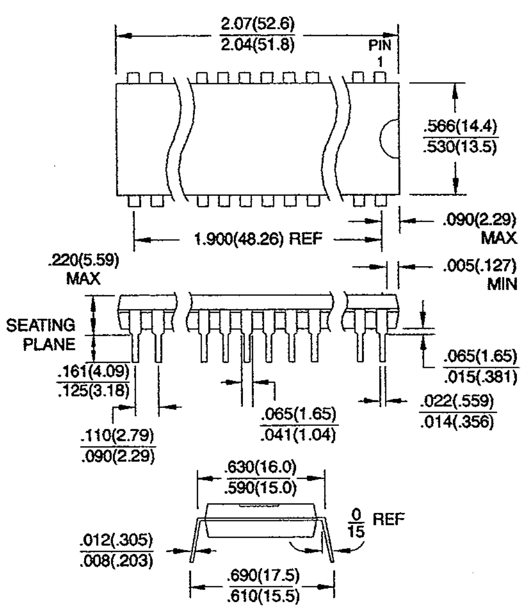


Controlling dimension: millimeters

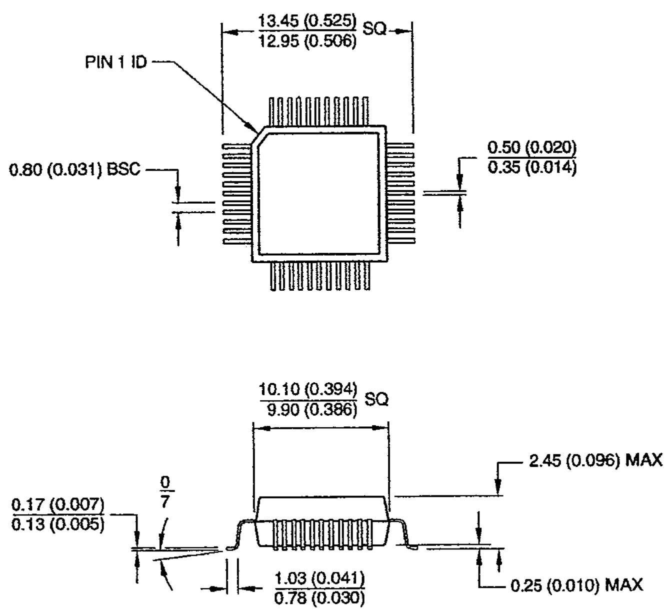
44J, 44-lead, Plastic J-leaded Chip Carrier (PLCC)
Dimensions in Inches and (Millimeters)
JEDEC STANDARD MS-018 AC



40P6, 40-lead, 0.600" Wide, Plastic Dual Inline Package (PDIP)
Dimensions in Inches and (Millimeters)



44Q, 44-lead, Plastic Quad Flat Package (PQFP)
Dimensions in Millimeters and (Inches)*
JEDEC STANDARD MS-022 AB



Controlling dimension: millimeters

Atmel Headquarters

Corporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL (408) 441-0311
FAX (408) 487-2600

Europe
Atmel U.K., Ltd.
Coliseum Business Centre
Riverside Way
Camberley, Surrey GU15 3YL
England
TEL (44) 1276-686-677
FAX (44) 1276-686-697

Asia
Atmel Asia, Ltd.
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

Japan
Atmel Japan K.K.
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Operations

Atmel Colorado Springs
1150 E. Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL (719) 576-3300
FAX (719) 540-1759

Atmel Rousset
Zone Industrielle
13106 Rousset Cedex
France
TEL (33) 4-4253-6000
FAX (33) 4-4253-6001

Fax-on-Demand
North America:
1-(800) 292-8635
International:
1-(408) 441-0732

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

BBS
1-(408) 436-4309

0313H–02/00/xM