เอกสารอ้างอิง

ดร. กาญจนา นาคสกุล, "ระบบเสียงภาษาไทย" คณะอักษรศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย กรุงเทพมหานคร : โรงพิมพ์จุฬาลงกรณ์มหาวิทยาลัย, 2530.

ชัยยง ภักดีกิจเจริญ, "CU-TSB 32010" Senior Project คณะวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย, ปีการศึกษา 2531.

ผศ.ดร.วีระ ริ้วพิทักษ์, "ระบบตรวจรู้เสียงพูดด้วยคอมพิวเตอร์" วารสารวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ปีที่ 6 ฉบับที่ 1 กุมภาพันธ์ 2531.

Bung, E., "Speaker Recognition by Computer," Philips Tech., Rev 37, No.8, pp. 207-219.

Dubnowski, J.J., "Real-time hardware pitch detector, " IEEE Trans. ASSP, Vol. ASSP-24, No. 1, pp 2-8, Feb. 1976.

Flanagan, J.L., Speech Analysis, Synthesis and Perception, 2nd expand edition, Springer Verleg, Berlin, 1972.

Gupta, V., Lenning, M. and Mermelstein, P., "Decision rules for speaker independent isolated word recognition," Proc. IEEE Int. Conf., ASSP 9.2.1-4.

Holtzman, J., "Automatic speech recognition error/no decision tradeoff curves," IEEE Trans. ASSP, ASSP-32, pp. 1232-1235, 1984.

Lamel, L.F., Rabiner, L.R., Rosenberg, A.E. and Wilpon, J.G., "An improved detection for isolated word recognition," IEEE Trans. ASSP, ASSP-29, No. 4, Aug. 1989.

Markel, J.D., "The SIFT algorithm for fundamental frequency estimation," IEEE Trans., Vol. AU-20, No. 5, pp 367-377, Dec. 1972.

Markel, J.D. and Gray, A.H. Jr., Linear Prediction of Speech, Springer Verlag, New York, 1976.

Martin, T., "Applications of limited vacabulary recognition systems," in Rec. 1974 Symp. Speech Recognition D.R. Reddy, Ed., New York:Academic, pp. 55-71, 1975.

Noll, A.M., "Cepstrum pitch determination, " JASA, Vol. 41, No. 2, pp 293-309, Feb. 1967.

O'Shaughnessy, D., Speech Communication Human and Manchine, Addison-Weslay

Publishing Company, 1987.

Parson T.W., Voice and Speech processing, McGraw-Hill Book Co., New York, 1986.

Sakoe, H. and Chiba, S., "Dynamic Programming Algorithm OPtimization for Spoken Word Recognition," IEEE Trans. ASSP, Vol. ASSP-26, No. 1, Feb. 1978.

Schroeder, M.R., "Vocoder: Analysis and Synthesis of Speech, " Proc. IEEE, Vol. 54, No.5, pp. 720-734, May, 1966.

Shannon, C.E., "Communication in the presence of noise, " Proc. IRE, Vol. 37, pp. 10-31, Jan., 1949.

Sondhi, M.M., "New methods of pitch extraction," IEEE Trans., Vol. AU-16, No. 2, pp. 262-266, June 1968.

Rabiner, L.R. and Gold, B., Theory and Application of Digital Signal Processing, Prentice Hall, Englewood Cliffs, New Jersey, 1975

Rabiner, L.R., Levinson, S.E, Rosenberg A.E. and Wilpon J.G., "Speaker-independent recognition of insolated words using a clustering technique, " IEEE Trans. ASSP, Vol. ASSP-27, No. 4, pp. 336-349, Aug. 1972.

Rabiner, L.R., Rosenberg, A.E. and Levison S.E., "Considerations in dynamic time wraping algorithms for discrete word recognition," IEEE Trans. ASSP, Vol. ASSP-26, No. 6, Dec. 1978

Ronald, W.S., Russell M.M., Thomas P.B., TMS32010 User's Guides, Texas Instruments Incorporated, 1983.

Witten, I.H., Principle of Computer Speech, Academic Press Inc. (LONDON) Ltd. 1982.

ภาคผนวก

ภาคผนวก ก

หลักเกณฑ์คัดเลือกคำที่ใช้ในการทดสอบ

ในการคัดเลือกคำที่ใช้ในการทดสอบระบบการรับรู้เสียงพูดด้วยคอมพิวเตอร์ ได้แบ่งออกเป็น 2 กลุ่ม คือ กลุ่มแรก อาศัยหลักเกณฑ์การจำแนกลักษณะเสียงพูด ทางด้านภาษาศาสตร์ และกลุ่มที่สอง ได้คัดเลือกจากรายงานการวิจัยเรื่อง การประมวลผลข้อมูลภาษาไทยด้วยไมโครคอมพิวเตอร์ ของยืน ภู่วรวรรณและคณะ

หลักเกณฑ์คัดเลือกคำในกลุ่มแรก

เสียงพูดของมนุษย์สามารถแบ่งออกกว้าง ๆ ได้ 2 พวก คือ พวกที่สามารถแยกออกจากเสียงอื่นได้โดยเด็ดขาด มีลักษณะเฉพาะตัว กับพวกที่เป็นส่วนประกอบของเสียงอื่น ไม่อาจจะแยกเปล่งเสียงได้โดยลำพัง ในพวกแรกเรียกว่าเป็น Segmental Sounds เช่น เสียงสระ เสียงพยัญชนะ เป็นต้น ส่วนในพวกหลังเรียกว่าเป็น Supra-segmental Features เช่น วรรณยุกต์ ทำนองเสียง เป็นต้น

1. เสียงสระ

ลักษณะของเสียงสระที่สำคัญ คือ เป็นเสียงก้องที่เปล่งโดยให้ลมออกทางช่องปากอย่างที่อวัยวะในช่องปากไม่ปิดกั้นลม หรือทำให้เกิดเป็นช่องแคบ จนลมต้องออกอย่างมีเสียงเสียดแทรก หรืออย่างที่ทำให้อวัยวะในช่องปากส่วนใดส่วนหนึ่งสะบัด ดังนั้นในการออกเสียงสระ ลักษณะของปากจึงจำเป็นต้องเป็นโพรง การที่จะเปล่งเสียงสระให้แตกต่างกัน ทำได้โดยการแต่งรูปปากให้มีลักษณะต่าง ๆ กัน นั่นเอง

ในการอธิบายเสียงสระแต่ละเสียง สามารถทำได้โดยอาศัยหลักเกณฑ์ดังนี้

ก. ลักษณะของริมฝีปาก จะแบ่งเป็นริมฝีปากห่อกลม (Rounded) หรือ ริมฝีปากรี (Unrounded)

ข. ส่วนของลิ้น ในขณะที่ทำการออกเสียงสระ จะมีส่วนของลิ้นสองส่วนที่จะยกขึ้นใกล้เพดาน การเรียกเสียงสระจึงเรียกตามส่วนของของลิ้นที่ยกขึ้นนั้น ถ้าออกเสียงสระใดแล้วลิ้นส่วนหน้ายกขึ้นใกล้เพดานแข็ง จะเรียกสระนั้นว่า สระหน้า (Front Vowel) ถ้าออกเสียงสระใดแล้วลิ้นส่วนหลังยกขึ้นใกล้เพดานอ่อน จะเรียกสระนั้นว่า สระหลัง (Back Vowel)

ค. ระดับของลิ้น ในภาษาไทยระดับความสูงต่ำของลิ้น แบ่งออกเป็น 4 ระดับ คือ สูง กลางสูง กลางต่ำ ต่ำ ถ้ายกลิ้นขึ้นสูง ช่องว่างระหว่างลิ้นกับเพดานจะปิดแคบ เสียงสระที่เกิดเมื่อลิ้นยกขึ้นสูง เรียกว่า สระสูง หรือ สระปิด (Close Vowel) ถ้าออกเสียงสระใดแล้ว

ลิ้นวางราบอยู่ในระดับต่ำ จะเรียกว่า สระต่ำ หรือ สระเปิด (Open Vowel)

2. เสียงพยัญชนะ

เสียงพยัญชนะสามารถจำแนกได้เป็นหลายลักษณะ เช่น เสียงพยัญชนะอาจเป็นเสียงก้องหรือไม่ก้องก็ได้ ลมที่ใช้ในการเปล่งเสียงเมื่อผ่านเส้นเสียง จะถูกบีบให้ผ่านช่องแคบช่องใดช่องหนึ่ง จนเกิดเป็นเสียงเสียดแทรกก็ได้ ลมนั้นจะไม่ออกทางปาก หรือ ทางจมูก แต่จะกลับถูกกลืนลงไปใหม่ก็ได้

ในการจำแนกเสียงพยัญชนะ ทำโดยการสรุปลักษณะที่เหมือน ๆ กันได้เป็น

ก. ความก้องหรือไม่ก้อง เสียงพยัญชนะที่เปล่งออกมาในขณะที่เส้นเสียงปิดเรียกว่าเสียงก้อง และเสียงพยัญชนะที่เปล่งออกมาในขณะที่เส้นเสียงเปิด เรียกว่า เสียงไม่ก้อง เช่น ในภาษาไทย เสียงพยัญชนะ บ ด เป็นเสียงก้อง คู่กับเสียง ป ต ซึ่งเป็นเสียงไม่ก้อง เป็นต้น

ข. ลักษณะของลมที่ผ่านเส้นเสียงออกไป ลักษณะลมที่ผ่านเส้นเสียงขึ้นมาเกิดเป็นเสียงพยัญชนะนั้น สามารถแบ่งออกได้เป็นหลายลักษณะ คือ

1. กัก (Stop) หมายถึง การที่ลมซึ่งเปล่งออกมา มาถูกกัก ณ ที่ใดที่หนึ่งในช่องปากชั่วระยะหนึ่ง การกักลมอาจจะกักได้ที่ช่องว่างระหว่างเส้นเสียง หรือใช้ลิ้นกักไว้ที่เพดานอ่อน ที่เพดานแข็ง ที่ปุ่มเหงือก ที่ลิ้น หรือใช้ริมฝีปากทั้งสองปิดเข้าหากันก็ได้ เช่น เสียงพยัญชนะตัวสะกดในคำว่า นับ นัด นัก เป็นต้น

2. ระเบิด (Plosive) หมายถึง ลมที่ถูกปล่อยออกมาอย่างแรง และโดยรวดเร็วหลังจากที่มาถูกกักอยู่ ณ ที่ใดที่หนึ่งในปาก การกักลมเพื่อทำเสียงระเบิดนี้ กระทำได้โดยการปิดอวัยวะในช่องปาก เช่น ปิดริมฝีปากทั้งสอง เอาลิ้นไปกดไว้ที่หลังฟัน ที่ปุ่มเหงือก ที่เพดานแข็ง หรือที่เพดานอ่อน เป็นต้น ตัวอย่างของเสียงพยัญชนะระเบิดเช่น เสียง ป ต พ ท เป็นต้น

3. นาสิก (Nasal) หมายถึง ลมที่ถูกกักในช่องปาก แล้วผ่านออกไปทางจมูก การที่ลมออกทางจมูกได้ เราต้องบังคับอวัยวะส่วนที่เรียกว่า เพดานอ่อน และลิ้นไก่ ให้อยู่ในลักษณะพักเพื่อเปิดช่องด้านหลังให้ลมขึ้นไปทางจมูกได้ เช่นเสียง ม น ง เป็นต้น

4. ข้าง (Lateral) หมายถึง ลมที่ออกจากปากทางด้านข้างของลิ้น เมื่อลมที่ผ่านเส้นเสียงขึ้นมาถึงช่องปาก ถ้าลิ้นกดเพดานตรงแนวช่องกลางลิ้นไว้ แล้วปล่อยให้ลมออกมาทางด้านข้าง ๆ ได้ ก็จะทำให้เกิดเสียงขึ้น เรียกว่า เสียงข้าง (Lateral Sound) เช่น เสียง ล เป็นต้น

5. เสียงเสียดแทรก (Fricative) หมายถึง ลมที่ผ่านออกมาอย่างไม่สะดวก ต้องผ่านช่องแคบในช่องปาก ซึ่งทำให้เกิดเสียงเสียดแทรกดังขึ้น ช่องแคบในปากจะเกิดได้ก็โดยที่อวัยวะต่าง ๆ เช่น ริมฝีปาก ฟัน หรือ ลิ้น วางอยู่ใกล้กันมากจนลมผ่านออกไปไม่สะดวก เช่น เสียง ฟ ซ ส เป็นต้น

ค. ตำแหน่งที่เกิดของเสียง คือ ตำแหน่งที่ลมทำอาการนั้น ๆ ตำแหน่งที่เกิดของเสียงจะมีที่สำคัญอยู่ 6 แห่งด้วยกัน คือ ริมฝีปาก ฟัน ปุ่มเหงือก เพดานแข็ง เพดานอ่อน และช่องว่างระหว่างเส้นเสียง

ในการคัดเลือกคำที่ใช้ในการทดสอบกลุ่มที่ 1 ได้อาศัยหลักเกณฑ์ดังกล่าว. โดยเลือกเสียงสระ 3 เสียง คือ สระอา สระอิ และสระอี มีเสียงพยัญชนะ 5 เสียง คือ ป พ บ ฟ และ ม ส่วนพยัญชนะท้าย และเสียงวรรณยุกต์ไม่มีการกำหนด

ภาคผนวก ข

รายละเอียดและการใช้งานบอร์ด TSB

บอร์ด TSB (TMS32010 Single Borad) เป็นบอร์ดที่พัฒนาขึ้นมาเพื่อให้สามารถใช้งานในหลายด้าน โดยจะอาศัยประสิทธิภาพที่สูงของไมโครโปรเซสเซอร์เบอร์ TMS32010 บอร์ด TSB ประกอบด้วยส่วนสำคัญหลักต่าง ๆ ดังนี้

1. ส่วนของอินพุท เอาท์พุท

เป็นส่วนของบอร์ดที่ใช้ในการส่งผ่านข้อมูลระหว่างไมโครโปรเซสเซอร์ กับอุปกรณ์รอบตัว ซึ่งสามารถทำได้จากคำสั่งที่เป็นภาษาเครื่องคือ IN และ OUT โดยคำสั่งนี้จะทำการนำข้อมูลจากหน่วยความจำข้อมูล ส่งออกไปทางพอร์ท

1.1 ระบบการรับส่งข้อมูลของ A/D และ D/A

1.1.1 วงจรส่วน D/A ใช้ไอซีในการแปลงสัญญาณจากดิจิตอลเป็นอนาลอก เบอร์ AD565A โดยใช้แหล่งจ่ายไฟ +12 ถึง -12 โวลท์ สัญญาณอนาลอกมีช่วงตั้งแต่ +10 ถึง -10 โวลท์ ใช้เวลาในการเปลี่ยนข้อมูลประมาณ 250 นาโนวินาที ส่วนข้อมูลดิจิตอลมีขนาด 12 บิท เนื่องจากข้อมูลดิจิตอลที่ได้มีขนาดเพียง 12 บิท ข้อมูล 16 บิทที่ได้จากบัสสัญญาณ (Data Bus) จะตัด 4 บิทต่ำออก

1.1.2 วงจรส่วน A/D ใช้ไอซีในการแปลงสัญญาณจากอนาลอกเป็นดิจิตอล เบอร์ ADC80-Z ใช้เวลาในการแปลงสัญญาณประมาณ 25 มิลลิวินาที ทำงานโดยใช้แหล่งจ่ายไฟ +12 ถึง -12 โวลท์ รับสัญญาณอนาลอกที่ +10 ถึง -10 โวลท์ ข้อมูลดิจิตอลมีขนาด 12 บิท

1.2 ระบบการรับข้อมูล

ทำได้โดยการใช้คำสั่ง IN (IN port) สามารถแบ่งได้เป็น 2 แบบ คือ

1.2.1 แบบรับตามรายคาบ มีจุดสำคัญที่ต้องกำหนดไว้ คือ ต้องกำหนดให้มีการใช้สัญญาณนาฬิกาสำหรับการชักตัวอย่าง (Sampling rate clock) โดยการตั้งค่าในการชักตัวอย่างให้เหมาะสม สัญญาณอนาลอกจะถูกแปลงทุก ๆ คาบของเวลา ตามค่าที่ตั้งไว้ให้ เมื่อทำการแปลงสัญญาณสำเร็จในแต่ละครั้ง จะมีสัญญาณ IORDY (IO Ready) ออกมา ซึ่งสัญญาณนี้จะเข้าทางขา -INT หรือ -BIO ตามที่ได้ทำการตั้งสวิทช์ไว้

1.2.2 แบบรับเมื่อเรียกขอ ระบบการรับข้อมูลแบบนี้ เหมาะกับงานบางชนิดที่ไม่มีความจำเป็นที่จะรับสัญญาณเป็นคาบของเวลา แต่มีการรับข้อมูลเมื่อต้องการเท่านั้น การใช้งานในระบบนี้เมื่อกำหนดค่าบิทต่าง ๆ ถูกต้องแล้ว จะต้องอ่านค่าของข้อมูลทิ้งไปหนึ่งครั้งก่อน ในขณะที่อ่านค่าครั้งแรกนั้นตัวไอซีจะทำการแปลงสัญญาณอนาลอกทันที เมื่อการแปลงเสร็จสิ้น จะมีสัญญาณ IORDY (IO Ready) ออกมา ซึ่งสัญญาณนี้จะเข้าทางขา -INT หรือ -BIO ตามที่ได้ทำการตั้งสวิทซ์ไว้

1.3 ระบบการส่งข้อมูล

ทำได้โดยการใช้คำสั่ง OUT (OUT port) สามารถแบ่งได้เป็น 2 แบบ คือ

1.3.1 แบบส่งตามรายคาบ มีจุดสำคัญที่ต้องกำหนดไว้ คือ ต้องกำหนดให้มีการใช้สัญญาณนาฬิกาสำหรับการชักตัวอย่าง (Sampling rate clock) โดยการตั้งค่าในการชักตัวอย่างให้เหมาะสม การใช้งานในระบบนี้เหมาะสำหรับงานที่ต้องการส่งสัญญาณออกเป็นคาบของเวลาที่กำหนด ค่าที่ถูกแปลงออกไปนั้นจะได้จากบัฟเฟอร์ (Buffer) ซึ่งถูกส่งออกมาจากพอร์ท ถ้าไม่มีการส่งข้อมูลจากพอร์ท ข้อมูลที่จะได้ไปคือค่าสุดท้ายของการส่งนั่นเอง

1.3.2 แบบส่งทันที การส่งข้อมูลในแบบนี้ ข้อมูลที่ถูกส่งผ่านออกทางพอร์ทจะถูกส่งตรงให้แก่ตัวไอซีทันที และค่านั้นจะไม่มีการเปลี่ยนแปลง จนกระทั่งมีการส่งผ่านออกอีกครั้ง

2. หน่วยความจำโปรแกรม

ในส่วนนี้ประกอบด้วยหน่วยความจำ 4 กิโลเวิร์ด โดยจะสามารถให้โปรเซสเซอร์สามารถอ่านโปรแกรมได้ และสามารถที่จะให้ไมโครคอมพิวเตอร์สามารถย้ายโปรแกรมที่จะทำงานบนบอร์ด TSB เข้า และออกได้

3. หน่วยความจำข้อมูล

เนื่องจากไมโครโปรเซสเซอร์เบอร์ TMS32010 มีหน่วยความจำข้อมูลภายในน้อยมาก คือ 144 เวิร์ด ซึ่งไม่เพียงพอต่อการทำงานบางประเภท ดังนั้นบอร์ด TSB จึงได้มีส่วนขยายหน่วยความจำสำหรับข้อมูลมีขนาด 64 กิโลเวิร์ด เนื่องจากหน่วยความจำส่วนนี้ไม่ได้อยู่ในส่วนออกแบบของโปรเซสเซอร์ ดังนั้นการใช้งานหน่วยความจำส่วนนี้จะต้องอาศัยการส่งข้อมูลตามพอร์ทต่าง ๆ ที่กำหนดไว้ หน่วยความจำที่ใช้เป็นหน่วยความจำสแตติก (Static RAM) เบอร์ 6264-10 ซึ่งมีขนาดของหน่วยความจำ 8 กิโลเวิร์ด จึงต้องใช้ทั้งหมด 16 ตัว

4. ส่วนของภาคอนาลอก

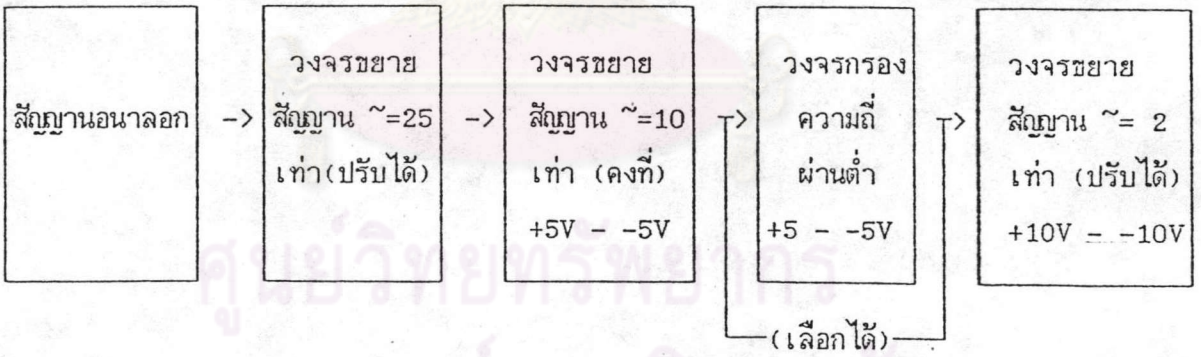
ในส่วนของภาคอนาลอกได้แบ่งออกเป็น 2 ส่วน คือ ภาคเข้า และภาคออก ดังมีรายละเอียด ดังนี้

4.1 ภาคอนาลอกเข้า

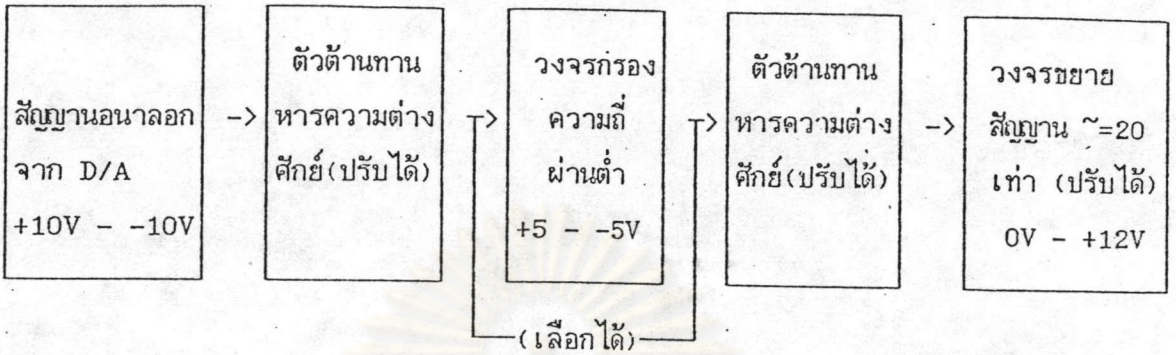
ภาคนี้จะมีสัญญาณเข้าเป็นไมโครโฟน ต่อเข้ากับวงจรขยายสัญญาณให้ใหญ่ขึ้น สัญญาณที่ถูกขยายนี้จะมีความต่างศักย์ในช่วง -5 ถึง +5 โวลท์ เมื่อสัญญาณมาถึงจุดนี้แล้วก็จะถูกผ่านให้วงจรกรองความถี่ต่ำ (Low Pass Filter) โดยจะกรองความถี่ที่สูงกว่าที่กำหนดออกแล้วเข้าสู่วงจรขยายสัญญาณ เพื่อให้สัญญาณมีช่วงความต่างศักย์ -10 ถึง +10 โวลท์ ดังรูปที่ ข.1

4.2 ภาคอนาลอกออก

สัญญาณที่ออกจากไอซีมา จะผ่านวงจรขยายให้สัญญาณออกในช่วง -10 ถึง +10 โวลท์ แล้วใช้ตัวต้านทานปรับค่าได้เพื่อหารความต่างศักย์ให้เหลือ +5 ถึง -5 โวลท์ แล้วจึงผ่านวงจรกรองความถี่ต่ำผ่าน สัญญาณที่ออกมาจะนำไปผ่านตัวต้านทานเพื่อหารความต่างศักย์ แล้วนำไปผ่านวงจรขยาย และปรีแอมป์ เพื่อให้สามารถต่อเข้ากับลำโพง หรือภาคขยายเสียงได้ โดยมีอัตราการขยายประมาณ 20 เท่าที่ 0 โวลท์ถึง +12 โวลท์



รูปที่ ข.1 แสดงขั้นตอนของวงจรการขยายสัญญาณอนาลอกภาคเข้า



รูปที่ ข.2 แสดงขั้นตอนของวงจรการขยายสัญญาณอนาลอกภาคออก

5. ส่วนที่ติดต่อกับไมโครคอมพิวเตอร์

บอร์ด TSB ได้ออกแบบให้ใช้งานร่วมกับไมโครคอมพิวเตอร์โดยเฉพาะ โดยไมโครคอมพิวเตอร์สามารถเข้าถึงบอร์ด TSB ได้หลายส่วนตามที่ออกแบบไว้ ดังนี้

5.1 การเข้าถึงส่วนของหน่วยความจำของบอร์ด TSB ซึ่งแบ่งออกเป็น

5.1.1 ส่วนของหน่วยความจำโปรแกรม 4 กิโลเวิร์ด

5.1.2 ส่วนของหน่วยความจำข้อมูล 64 กิโลเวิร์ด

5.2 สามารถสร้างสัญญาณต่าง ๆ เพื่อควบคุมบอร์ด TSB ได้

การติดต่อของไมโครคอมพิวเตอร์จะผ่านทางช่องส่งผ่านของตัวไมโครคอมพิวเตอร์เอง ปัญหาที่เกิดขึ้นประการหนึ่ง คือเนื่องจากไมโครคอมพิวเตอร์เป็นระบบบัสข้อมูล 8 บิท การเลื่อนของข้อมูลนั้นจะเลื่อนเมื่อทำการส่งผ่านเข้า และออก ทางช่องที่เป็นการส่งสัญญาณไบท์สูงเท่านั้น ในการกำหนดระบบการทำงานร่วมกับไมโครคอมพิวเตอร์นั้น จะใช้บิท LINK/STAND ALONE เป็นตัวกำหนด ถ้าบิทนี้มีค่าเป็น 1 การทำงานทุกส่วนของบอร์ด TSB จะถูกงดทันที บิทนี้ใช้เพื่อ

หยุดการทำงานของบอร์ด TSB และกำหนดค่าเริ่มต้นต่าง ๆ รวมถึงการบรรจุโปรแกรมลงในส่วนของหน่วยความจำโปรแกรม การทำงานขนานกันของไมโครคอมพิวเตอร์กับบอร์ด TSB ทำได้โดยการกำหนดให้บิทนี้มีค่าเป็น 0 ในสภาวะนี้ ไมโครคอมพิวเตอร์จะไม่สามารถเข้าถึงส่วนของหน่วยความจำทั้ง 2 ส่วนได้เลย นอกจากส่งสัญญาณควบคุมบางส่วนให้แก่บอร์ด TSB เท่านั้น

6. การใช้งาน

โปรแกรมที่ใช้ และเกี่ยวข้องกับบอร์ด TSB แบ่งออกได้เป็น 3 ส่วนคือ

6.1 ส่วนของโปรแกรมเพื่อการพัฒนา (TMS32010 Development Tools) ประกอบด้วยโปรแกรมสำคัญ คือ

6.1.1 โปรแกรมแปลภาษาแอสเซมบลีของ TMS32010

6.1.2 โปรแกรมลิงก์เกอร์ของ TMS32010

6.1.3 โปรแกรมจำลองการทำงานของ TMS32010

ในส่วนนี้จะเป็นส่วนที่ช่วยพัฒนาในการเขียนโปรแกรมภาษแอสเซมบลีของ TMS32010 ให้ได้ออปเจคเพื่อนำไปใช้บนบอร์ด TSB แต่อย่างไรก็ตามรูปแบบของออปเจคที่ได้จากโปรแกรมลิงก์เกอร์ยังไม่สามารถนำไปใช้บนบอร์ด TSB ได้ จึงได้มีการพัฒนาโปรแกรม REFORM เพื่อเปลี่ยนรูปของออปเจคใหม่ให้สามารถนำไปใช้บนบอร์ด TSB ได้

6.2 ส่วนที่จัดการเกี่ยวกับการรับส่งข้อมูลกับหน่วยความจำต่าง ๆ บนบอร์ด ในส่วนนี้จะทำโปรแกรมในลักษณะที่เป็นโมดูล (Module) เพื่อให้สามารถเรียกใช้โดยโปรแกรมประยุกต์อื่น ๆ ซึ่งจะประกอบด้วยโมดูลที่สามารถติดต่อได้ทั้งหน่วยความจำโปรแกรม และหน่วยความจำข้อมูล

6.3 ส่วนที่ติดต่อระหว่างไมโครคอมพิวเตอร์ กับบอร์ด TSB ในส่วนนี้จะทำโปรแกรมในลักษณะที่เป็นโมดูลเช่นกัน โดยจะประกอบด้วยโมดูลที่จะให้ไมโครคอมพิวเตอร์สามารถส่งสัญญาณไปควบคุมการทำงานบนบอร์ด TSB ได้

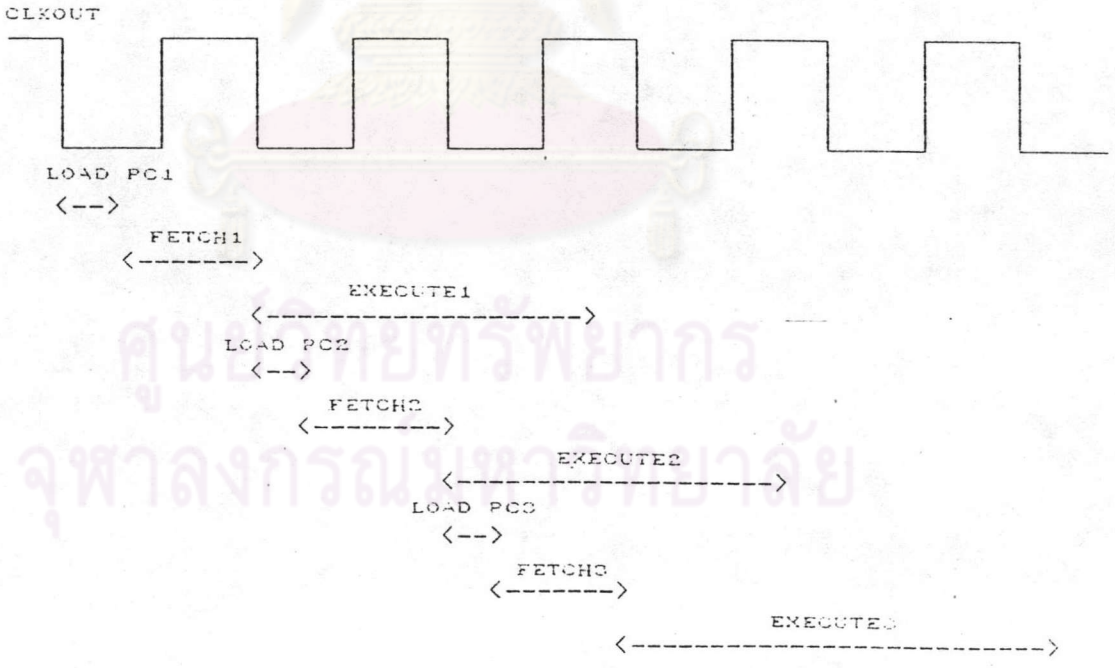
ภาคผนวก ค.

คุณสมบัติของไมโครโปรเซสเซอร์เบอร์ TMS32010

1. สถาปัตยกรรมของ TMS320 (TMS320 Architecture)

สถาปัตยกรรมของไอซีในตระกูล TMS320 นั้นได้ทำการปรับปรุงมาจากสถาปัตยกรรมของฮารวาร์ด (Harvard Architecture) ซึ่งเดิมหน่วยความจำโปรแกรมและข้อมูลจะแยกกันอยู่ และอนุญาตให้คำสั่งสามารถดึงและปฏิบัติงานในทั้งสองส่วนได้ สำหรับ TMS320 นั้นยังอนุญาตให้สามารถเคลื่อนย้ายข้อมูลไปมาระหว่างหน่วยความจำโปรแกรมกับหน่วยความจำข้อมูลได้ โดยในการใช้งาน เราสามารถเก็บค่าคงที่ไว้ในหน่วยความจำโปรแกรม และเรียกเข้าไปในหน่วยความจำข้อมูลเมื่อต้องการใช้งานได้ ทำให้ความต้องการสำหรับรอมภายนอกหมดไป ซึ่งจะเป็นการเพิ่มความยืดหยุ่นในการใช้งานมากขึ้น

ลักษณะของสถาปัตยกรรมของฮารวาร์ด สามารถแสดงการทำงานได้ดังรูปที่ ค.1 คือ ขณะที่สัญญาณ CLKOUT ตกลง PC จะถูกบรรจุด้วยคำสั่ง (LOAD PC2) ในขณะที่คำสั่งปัจจุบัน (EXECUTE 1) ถูกแปลและปฏิบัติงาน คำสั่งถัดไปจะถูกเรียก (FETCH 2) ขณะที่คำสั่งปัจจุบันยังคงปฏิบัติงานอยู่ (EXECUTE 1) และแม้แต่การดึงคำสั่งถัดไป (FETCH 3) อาจเกิดขึ้นได้ขณะที่คำสั่งปัจจุบัน (EXECUTE 2) และคำสั่งก่อนหน้านี้ ยังคงปฏิบัติงานอยู่ได้



รูปที่ ค.1 แสดงการทำงานของสถาปัตยกรรมของฮารวาร์ด

2. หน่วยคำนวณ (Arithmatic Elements)

2.1 หน่วยคำนวณและตรรก (Arithmatic and Logical Unit)

หน่วยนี้เป็นหน่วยที่ใช้ในการทำงานทางด้านคำนวณและตรรกทั่วไป มีขนาด 32 บิต โดยมีตัวสะสม (Accummulator) เป็นที่เก็บผลลัพธ์ และเป็นโอเปอแรนด์ตัวแรก

2.2 ตัวสะสม (Accumulator)

ตัวสะสมจะเป็นที่เก็บผลลัพธ์จากหน่วยคำนวณและตรรก และส่วนมากมักจะเป็นอินพุตสำหรับหน่วยคำนวณและตรรก ตัวสะสมมีขนาด 32 บิต ตัวสะสมจะถูกแบ่งเป็น 2 ส่วน คือ ส่วนลำดับสูง (high-order word) คือบิตที่ 31 ถึง 16 และส่วนลำดับต่ำ (low-order word) คือบิตที่ 15 ถึง 0

สถานะของตัวสะสม (Accumulator Status) สถานะล้น (overflow) ของตัวสะสมสามารถทราบได้จากเรจิสเตอร์ล้นของตัวสะสม (Accumulator Overflow Flag Register: OV) เรจิสเตอร์นี้จะถูกกำหนดถ้ามีการล้นเกิดขึ้นในตัวสะสม เนื่องจากเรจิสเตอร์นี้เป็นส่วนหนึ่งของเรจิสเตอร์สถานะ (Status Register) ดังนั้นสถานะล้น จึงสามารถนำไปเก็บในหน่วยความจำข้อมูลได้ เมื่อเรจิสเตอร์นี้ถูกกำหนด จะมีเพียงคำสั่ง BV (Branch on Overflow) และการแก้ไขเรจิสเตอร์สถานะเท่านั้นที่สามารถเปลี่ยนแปลงได้

การทดสอบตัวสะสมในลักษณะอื่น ๆ สามารถใช้คำสั่งเกี่ยวกับการเคลื่อนย้าย (Branch) ได้โดยคำสั่งเหล่านี้จะทำการเคลื่อนย้ายต่อเมื่อการทดสอบเป็นจริง ดังแสดงในตารางที่ ค.1

2.3 หน่วยคูณ (Multiplier)

หน่วยคูณแบบขนาน 16 x 16 บิต ประกอบด้วย 3 หน่วย คือ เรจิสเตอร์ที (T Register) เรจิสเตอร์พี (P Register) และอะเรย์ของหน่วยคูณ เรจิสเตอร์ที่มีขนาด 16 บิต ใช้เป็นที่เก็บตัวตั้ง และเรจิสเตอร์พีมีขนาด 32 บิต ใช้เป็นที่เก็บผลคูณ

ในการที่จะใช้หน่วยคูณ เราต้องนำข้อมูลจากหน่วยความจำข้อมูล ไปไว้ในเรจิสเตอร์ที โดยใช้คำสั่ง LT (Load T Register) LTA หรือ LTD แล้วใช้คำสั่ง MPY (Multiply) หรือ MPYK (Multiply Immediate) เพื่อทำการคูณ ผลลัพธ์ที่ได้จะนำมาเก็บในเรจิสเตอร์พี ผลคูณที่ได้สามารถนำไปบวก ลบ หรือบรรจุลงในตัวสะสมได้โดยใช้คำสั่งต่าง ๆ เช่น APAC SPAC LTA LTD หรือ PAC

คำสั่ง	เงื่อนไขการทดสอบตัวสะสม
BLZ	< 0
BLEZ	<= 0
BGZ	> 0
BGEZ	>= 0
BNZ	<> 0
BNEZ	= 0

ตารางที่ ค.1 แสดงเงื่อนไขการทดสอบตัวสะสม

2.4 ตัวทำการเลื่อน (Shifters)

ตัวทำการเลื่อนมีอยู่ 2 แบบ แบบแรก คือตัวทำการเลื่อนแบบบาร์เรล (Barrel Shifter) ใช้สำหรับเลื่อนข้อมูลจากหน่วยความจำไปไว้ในหน่วยคำนวณและตรรก และแบบที่สอง คือตัวทำการเลื่อนแบบขนาน (Parallel Shifter) ใช้สำหรับเลื่อนข้อมูลจากหน่วยสะสมไปไว้ในหน่วยความจำข้อมูล

2.4.1 ตัวทำการเลื่อนแบบบาร์เรล (Barrel Shifter)

ตัวทำการเลื่อนแบบบาร์เรล จะทำการเติมศูนย์ที่บิตลำดับต่ำ (low-order bit) และเติมเครื่องหมายที่บิตลำดับสูง (high-order bit) โดยวิธี Arithmatic left-shift ซึ่งหมายความว่า ถ้าบิตที่มีนัยสูงสุด (Most Significant bit) ของข้อมูลเป็นศูนย์ ค่าที่ทำการเติมทางด้านซ้ายของบิตนี้คือศูนย์ และถ้าบิตที่มีนัยสูงสุดของข้อมูลเป็นหนึ่ง ค่าที่ทำการเติมคือหนึ่ง เช่นกัน ซึ่งแตกต่างจาก Logical left-shift ซึ่งจะเติมค่าศูนย์เสมอ

คำสั่งของ TMS320 บางคำสั่งสามารถจัดการเฉพาะส่วนล่างของตัวสะสม และหน่วยความจำข้อมูล โดยไม่คำนึงถึงเครื่องหมาย คือจัดการข้อมูลแบบ 16 บิต เช่น ADDS (Add to Accumulator with Sign-Extended Suppressed) SUBS เป็นต้น

2.4.2 ตัวทำการเลื่อนแบบขนาน (Parallel Shifter)

ตัวทำการเลื่อนแบบขนาน จะใช้เฉพาะกับคำสั่ง SACH (Store Accumulator High with Shift) เท่านั้น ตัวทำการเลื่อนนี้จะเลื่อนตัวสะสมไปทางซ้าย และจะนำค่า 16 บิตสูงไปบรรจุในหน่วยความจำข้อมูล ผลที่ได้จะทำให้บิตสูงของตัวสะสมหายไป

ค่าเลื่อนที่ใช้ได้มีเพียง 0, 1 หรือ 4 ประโยชน์สำหรับตัวทำการเลื่อนประเภทนี้คือ จะใช้เพื่อทำการเก็บผลลัพธ์ที่ได้จากการคูณ

3. หน่วยความจำข้อมูล (Data Memory)

หน่วยความจำข้อมูลมีจำนวน 144 เวิร์ด ขนาด 16 บิตอยู่บนชิพ (Chip) เราอาจเก็บข้อมูลไว้นอกชิพ และสามารถอ่านเข้ามาในชิพได้ โดยใช้คำสั่ง TBLR (Table Read) คำสั่งนี้จะทำการเคลื่อนย้ายข้อมูลจากหน่วยความจำโปรแกรม ไปไว้ยังหน่วยความจำข้อมูล คำสั่ง TBLW (Table Write) สามารถใช้ได้ในทางกลับกัน คำสั่งทั้งสองนี้ใช้เวลา 3 รอบในการปฏิบัติงาน นอกจากนี้ เราอาจใช้คำสั่ง IN และ OUT ได้โดย คำสั่ง IN จะอ่านข้อมูลจากอุปกรณ์และส่งมาเก็บไว้ในหน่วยความจำข้อมูล ทั้งคำสั่ง IN และ OUT สามารถใช้อ่านและเขียนจากหน่วยความจำข้อมูลกับหน่วยเก็บข้อมูลภายนอกขนาดใหญ่ได้ ในฐานะที่เป็น อุปกรณ์ วิธีนี้จะใช้เวลาน้อยกว่า เนื่องจากใช้เวลาเพียง 2 รอบในการปฏิบัติงานเท่านั้น

การกำหนดที่อยู่ของหน่วยความจำข้อมูล (Data Memory Addresing) มีอยู่ 3 แบบ คือ

3.1 การกำหนดที่อยู่แบบอ้อม (Indirect Addressing)

การกำหนดที่อยู่แบบอ้อมนั้นจะใช้ 8 บิตล่างของเรจิสเตอร์เสริม (Auxiliary Register) เป็นที่อยู่ของหน่วยความจำข้อมูล ซึ่งเพียงพอที่จะอ้างได้ถึง 144 เวิร์ด ไม่จำเป็นต้องมีเพจ (Page) เรจิสเตอร์เสริมปัจจุบันสามารถเลือกได้โดยใช้ตัวชี้เรจิสเตอร์เสริม (Auxiliary Register Pointer :ARP) เราสามารถเพิ่มหรือลดค่าของเรจิสเตอร์เสริมได้โดยอัตโนมัติ

3.2 การกำหนดที่อยู่แบบตรง (Direct Addressing)

การกำหนดที่อยู่แบบตรงจะใช้ 7 บิตของคำสั่งร่วมกับตัวชี้เพจ (Data-Page Pointer :DP) เพื่อกำหนดที่อยู่บนหน่วยความจำข้อมูล โดยใช้

ตัวชี้เพจ	ที่อยู่บนหน่วยความจำข้อมูล
0	0-127
1	128-144

ตัวชี้เพจนี้เป็นส่วนหนึ่งของเรจิสเตอร์สถานะ ดังนั้นจึงสามารถเก็บในหน่วยความจำข้อมูล

3.3 การกำหนดที่อยู่แบบอิมมีเดียท (Immediate Addressing)

TMS32010 มีคำสั่งพิเศษแบบอิมมีเดียท เช่น MPYK (Multiply Immediate), LACK, LDPK, LARK และ LARP คำสั่งเหล่านี้จะมีข้อมูลอยู่เป็นส่วนหนึ่งของคำสั่งเลย ไม่ได้อยู่ในหน่วยความจำข้อมูล

4. เรจิสเตอร์ (Register)

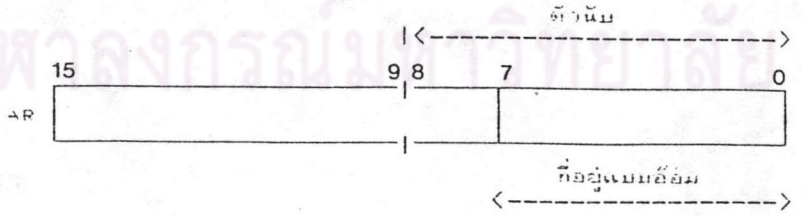
4.1 เรจิสเตอร์เสริม (Auxiliary Registers: AR)

เรจิสเตอร์เสริมมีอยู่ 2 ตัวขนาด 16 บิต ซึ่งไม่รวมอยู่ในหน่วยความจำข้อมูล หน้าที่ของเรจิสเตอร์เสริมสามารถทำได้ 3 อย่าง คือ เป็นที่เก็บข้อมูลชั่วคราว ใช้สำหรับกำหนดที่อยู่แบบอ้อม และใช้ควบคุมการวน (Loop Control)

การควบคุมการวนสามารถทำได้โดยใช้คำสั่ง BANZ (Branch on Auxiliary Register Not Zero) คำสั่งนี้จะตรวจว่าเรจิสเตอร์ปัจจุบันเท่ากับศูนย์หรือไม่ ถ้าไม่ค่าของเรจิสเตอร์จะลดลงหนึ่ง และเคลื่อนย้ายไป ดังตัวอย่าง

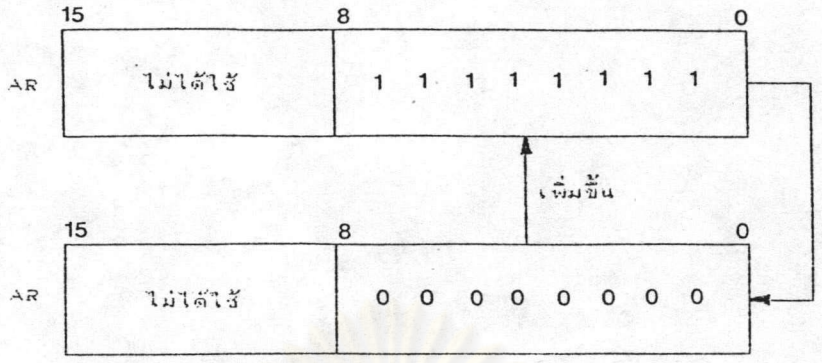
```
        LARP    AR0         เลือก AR0 เป็นเรจิสเตอร์เสริมปัจจุบัน
        LARK    AR0,5       บรรจุค่า 5 ที่ AR0
LOOP    ADD     *           บวกค่าหน่วยความจำข้อมูลที่ตำแหน่ง 5
        BANZ    LOOP        กับตัวสะสม
```

เมื่อเรจิสเตอร์เสริมถูกเพิ่ม หรือลดค่าโดยการกำหนดที่อยู่แบบอ้อม หรือโดยคำสั่ง BANZ ค่าของเรจิสเตอร์เสริม 9 บิตล่าง จะเปลี่ยนแปลง ซึ่งมากกว่าการใช้กำหนดที่อยู่แบบอ้อมอยู่ 1 บิตดังแสดงในรูปที่ ค.2

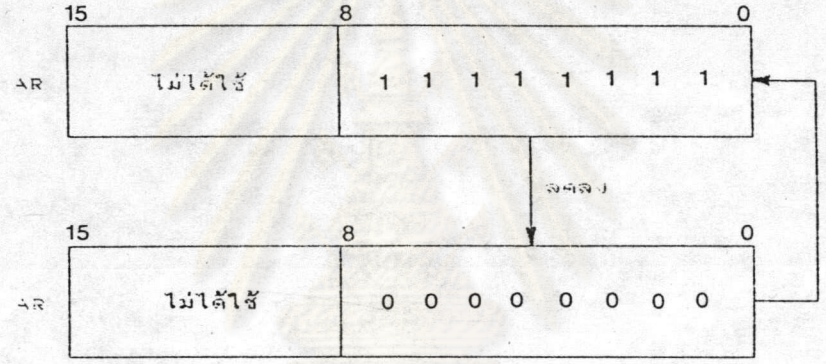


รูปที่ ค.2 แสดงถึงตัวนับของรีจิสเตอร์เสริม

ลักษณะของตัวนับ (Counter) ของเรจิสเตอร์เสริมจะเป็นแบบวนรอบ (Circular Counter) ดังแสดงในรูปที่ ค.3.1 และ ค.3.2



รูปที่ ค.3.1 แสดงถึงการเพิ่มของตัวนับโดยอัตโนมัติ



รูปที่ ค.3.2 แสดงถึงการลดของตัวนับโดยอัตโนมัติ

ค่าของเรจิสเตอร์เสริมสามารถเก็บ และเรียกจากหน่วยความจำข้อมูลได้ด้วยคำสั่ง SAR (Save Auxiliary Register) และ LAR (Load Auxiliary Register) ซึ่งมีประโยชน์ในการจัดการข้อมูลเพื่อนำไปใช้ต่อไป

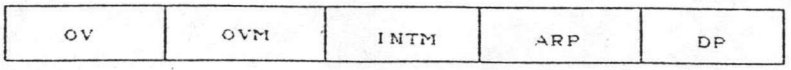
4.2 <u>ตัวชี้เรจิสเตอร์เสริม</u> (Auxiliary Register Pointer: ARP)

ตัวชี้เรจิสเตอร์มีอยู่ 1 ตัวขนาด 1 บิต ซึ่งอยู่เป็นส่วนหนึ่งของเรจิสเตอร์สถานะ ใช้เป็นตัวบ่งถึงเรจิสเตอร์เสริมปัจจุบัน โดย

<u>ตัวชี้เรจิสเตอร์เสริม</u>	<u>เรจิสเตอร์เสริมปัจจุบัน</u>
0	AR0
1	AR1

5. <u>เรจิสเตอร์สถานะ</u> (Status Register)

รูปที่ ค.4 จะแสดงให้เห็นถึงบิตทั้ง 5 ของเรจิสเตอร์สถานะ บิตสถานะเหล่านี้สามารถตรวจรู้แยกได้โดยใช้คำสั่งต่าง ๆ เรจิสเตอร์สถานะสามารถนำไปเก็บในหน่วยความจำข้อมูลได้โดยใช้คำสั่ง SST (Save Status Register) และเรียกกลับได้ด้วยคำสั่ง LST (Load Status Register) ยกเว้นบิต INTM



รูปที่ ค.4 แสดงถึงองค์ประกอบของเรจิสเตอร์สถานะ

บิต OV – มีค่าเป็น 0 เมื่อตัวสะสมไม่อยู่ในสถาวะล้น
มีค่าเป็น 1 เมื่อตัวสะสมอยู่ในสภาวะล้น และสามารถใช้คำสั่ง BV เพื่อเปลี่ยนค่าบิตนี้

บิต OVM – มีค่าเป็น 0 เมื่อยกเลิกสภาวะล้น (Overflow Disable)
มีค่าเป็น 1 เมื่อกำหนดสภาวะล้น (Overflow Enable)
คำสั่ง SOVM (Set Overflow Mode Register) จะทำการบรรจุค่า 1 ให้บิต OVM และคำสั่ง ROVM (Reset Overflow Mode Register) จะทำการบรรจุค่า 0 ให้บิต OVM

บิต INTM – มีค่าเป็น 0 เมื่อกำหนดการขัดจังหวะ (Interrupt Enable)
มีค่าเป็น 1 เมื่อยกเลิกการขัดจังหวะ (Interrupt Disable)
คำสั่ง EINT (Enable Interrupt) จะทำการบรรจุค่า 1 ให้บิต INTM และคำสั่ง DINT (Disable Interrupt) จะทำการบรรจุค่า 0 ให้บิต INTM เมื่อมีการขัดจังหวะเกิดขึ้น บิต INTM จะถูกเปลี่ยนค่าเป็น 1 โดยอัตโนมัติก่อนที่จะไปทำงานที่รูทีนบริการการขัดจังหวะ (Interupt Service Routine)

บิต ARP – มีค่าเป็น 0 เมื่อเลือก AR0
มีค่าเป็น 1 เมื่อเลือก AR1
คำสั่ง MAR (Modify Auxiliary Register) หรือ LARP (Load Auxiliary Register) หรือ คำสั่งที่อื่น ๆ ที่สามารถใช้กำหนดที่อยู่แบบอ้อมได้ จะสามารถเปลี่ยนแปลง ค่าของบิต ARP ได้

บิต DP - มีค่าเป็น 0 เมื่อต้องการใช้หน่วยความจำข้อมูล 128 เวิร์ดแรกมีค่าเป็น 1 เมื่อต้องการใช้หน่วยความจำข้อมูล 16 เวิร์ด ท้ายคำสั่ง LDP (Load Data Memory Page Pointer) หรือ LDP (Load Data Page Pointer Immediate) สามารถเปลี่ยนแปลงค่าของบิต DP ได้

5.1 การบันทึกค่าของเรจิสเตอร์สถานะ

ค่าของเรจิสเตอร์สถานะ สามารถบันทึกในหน่วยความจำข้อมูลได้ โดยใช้คำสั่ง SST ถ้าคำสั่ง SST ปฏิบัติโดยใช้การกำหนดที่อยู่แบบตรง ค่าของเรจิสเตอร์สถานะจะถูกนำไปเก็บที่เพจหนึ่ง ณ ตำแหน่งที่กำหนดในคำสั่งโดยอัตโนมัติ ดังนั้นคำสั่ง SST เมื่อใช้กับการกำหนดที่อยู่แบบตรง จะสามารถกำหนดที่อยู่ได้เพียง 16 ที่เท่านั้น เนื่องจากในเพจนี้มีหน่วยความจำเพียง 16 เวิร์ด เท่านั้น แต่ถ้าใช้การกำหนดที่อยู่แบบอ้อม ค่าของเรจิสเตอร์สถานะสามารถเก็บ ณ ตำแหน่งใดของหน่วยความจำก็ได้

รูปที่ ค.5 จะแสดงให้เห็นถึงตำแหน่งของบิตสถานะ ที่บันทึกในหน่วยความจำข้อมูลหลังจากปฏิบัติคำสั่ง SST

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OV	OVM	INTM	1	1	1	1	ARP	1	1	1	1	1	1	///	DP

/// = ไม่ใช้

รูปที่ ค.5 แสดงถึงเรจิสเตอร์สถานะที่บันทึกโดยคำสั่ง SST

คำสั่ง LST จะใช้ในการเรียกเรจิสเตอร์สถานะจากหน่วยความจำข้อมูล ในการใช้คำสั่งนี้ เราจะต้องกำหนดให้ตัวชี้เพจมีค่าเป็นหนึ่ง ก่อนที่เราจะเรียกเรจิสเตอร์กลับคืนมา เรจิสเตอร์สถานะทุกบิตจะเปลี่ยนแปลงตามค่าที่เรียกขึ้นมา ยกเว้นบิต INTM

6. ฟังก์ชันอินพุต เอาท์พุต (Input/Output Function)

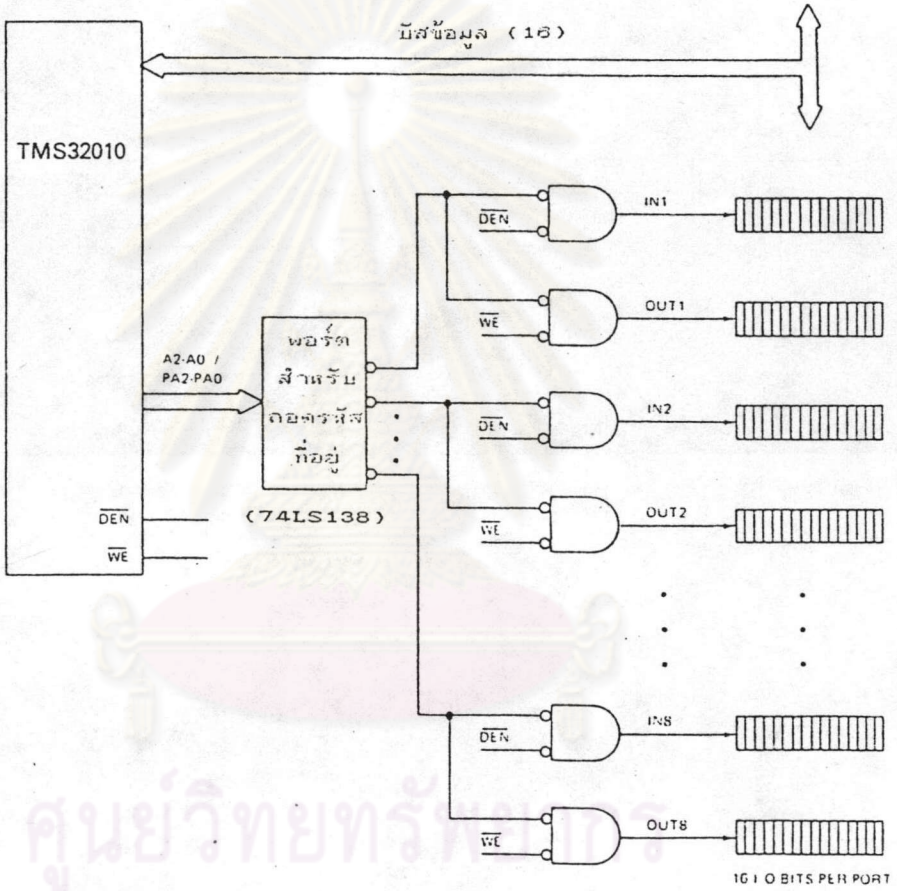
6.1 คำสั่ง IN และ OUT

การส่งถ่ายข้อมูลไปมากับอุปกรณ์ต่าง ๆ รอบข้าง (Peripheral) สามารถทำได้โดยใช้คำสั่ง IN หรือ OUT ข้อมูลจะถูกส่งถ่ายไปตามสายบัสข้อมูล (Data Bus) ขนาด 16 บิตทั้งส่งออกหรือส่งเข้าหน่วยความจำข้อมูลก็ตาม จะขึ้นอยู่กับสัญญาณควบคุม 2 ตัว คือ

สัญญาณ $\overline{DEN}$ (Data Enable) กับสัญญาณ $\overline{WE}$ (Write Enable)

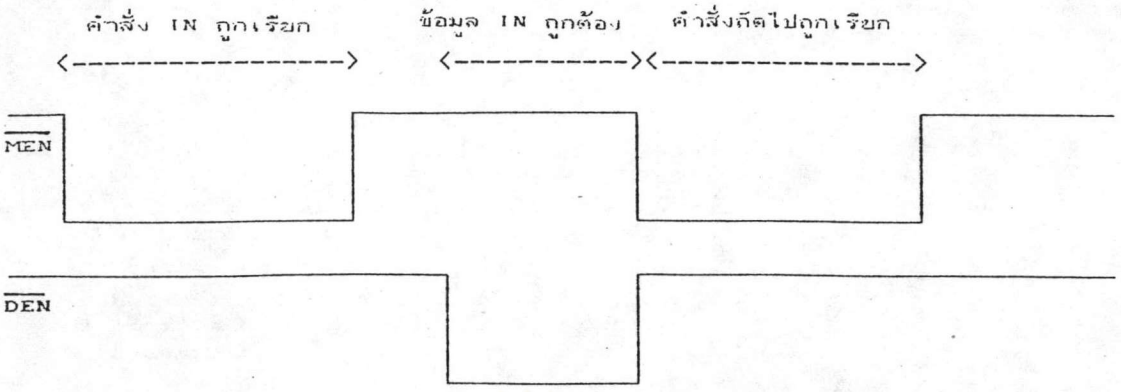
สายบัสข้อมูลที่ติดต่อออกสู่ภายนอก จะมีลักษณะเป็นแบบส่งถ่ายข้อมูลได้ทั้ง 2 ทิศทาง คือทั้งไปและกลับนั้น ตามปกติจะมีสภาวะลอยตัว (High-Impedance) คือไม่เป็นทั้งลอจิก (Logic) "1" และ "0" นอกจากเมื่อสัญญาณ $\overline{WE}$ เปลี่ยนสถานะเป็นลอจิก "0" สัญญาณ $\overline{WE}$ นั้นจะมี สภาวะเป็น "0" ในช่วงระหว่างรอบนาฬิกา (Clock Cycle) แรกของคำสั่ง OUT และในช่วงรอบนาฬิกาที่สองสำหรับคำสั่ง TBLW

จากรูปที่ ค.6 แสดงถึงการติดต่อกับอุปกรณ์ภายนอก สำหรับ 128 อินพุต/เอาท์พุตบิต โดยใช้การมัลติเพล็กซ์ (Multiplexed) ข้อมูล 16 บิต สำหรับพอร์ตอินพุต 8 พอร์ต และสำหรับพอร์ตเอาท์พุต 8 พอร์ต

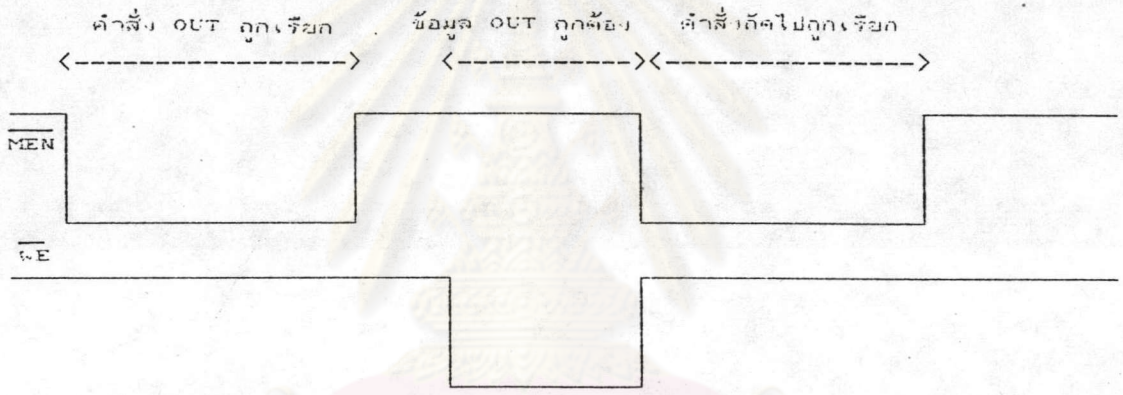


รูปที่ ค.6 แสดงถึงวิธีการติดต่อกับอุปกรณ์ภายนอก

การปฏิบัติคำสั่ง IN จะทำให้เกิดการสร้าง สัญญาณ $\overline{DEN}$ ขึ้นมา สำหรับใช้ในการส่งถ่ายข้อมูลจากอุปกรณ์รอบข้างเข้าสู่แรมข้อมูล (ดังรูปที่ ค.7.1) คำสั่ง IN นี้เป็นคำสั่งเดียวเท่านั้นที่จะทำให้สัญญาณ $\overline{DEN}$ เกิดขึ้นมา การปฏิบัติคำสั่ง OUT ก็จะสร้างสัญญาณ $\overline{WE}$ ขึ้นมาสำหรับการส่งถ่ายข้อมูลจากแรมข้อมูล ไปยังอุปกรณ์รอบข้าง (ดังรูปที่ ค.7.2) สัญญาณ $\overline{WE}$ จะเกิดขึ้นเฉพาะการใช้คำสั่ง OUT และ TBLW เท่านั้น



รูปที่ ค.7.1 แสดงถึงช่วงเวลาของคำสั่ง IN



รูปที่ ค.7.2 แสดงถึงช่วงเวลาของคำสั่ง OUT

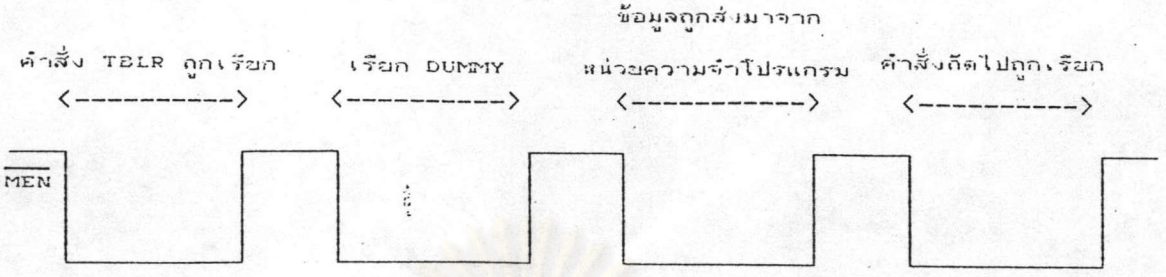
เราสามารถใช้บิตที่มีนัยต่ำสุด (Least Significant Bit: LSB) 3 บิต ล่าง คือ PA2 ถึง PA0 มาใช้ในการมัลติเพล็กซ์ เพื่อทำการกำหนดเป็นที่อยู่ของพอร์ต สำหรับคำสั่ง IN และ OUT ได้ ส่วนบิตอื่น ๆ ที่เหลือสำหรับบัสข้อมูลนั้น คือ A11 ถึง A3 ก็จะคงสภาวะเป็นลอจิก "0" ระหว่างที่มีการปฏิบัติคำสั่งเหล่านี้

6.2 คำสั่ง TBLR และ TBLW

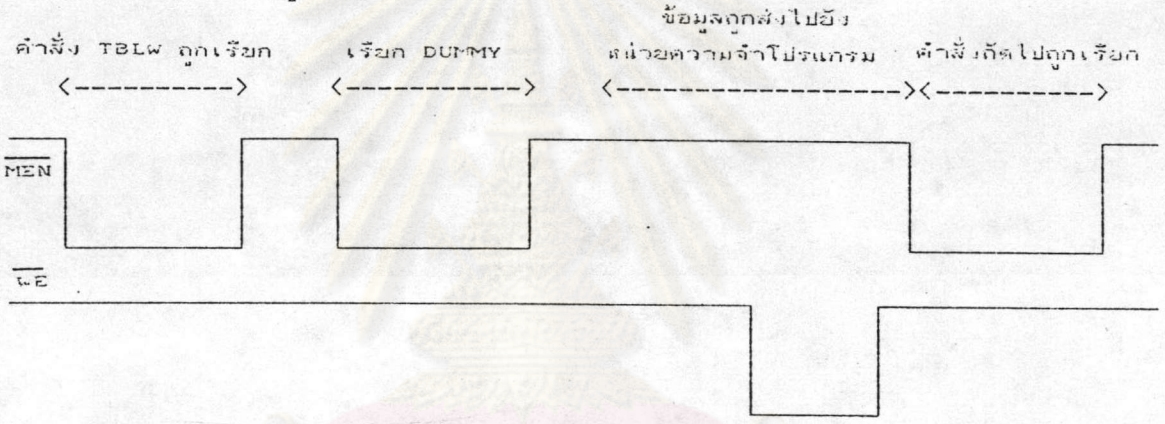
คำสั่ง TBLR กับ TBLW ใช้ในการส่งถ่าย ข้อมูลเป็นเวิร์ด ระหว่างเนื้อที่ของโปรแกรมและข้อมูล คำสั่ง TBLR จะใช้ในการอ่านข้อมูลเป็นเวิร์ดจากรอมโปรแกรมที่อยู่ในชิพ หรือจากรอมหรือแรมโปรแกรมที่อยู่ภายนอกชิพ แล้วแต่ชนิดของชิพที่ใช้เข้าไปยังแรมข้อมูล ส่วนคำ

สั่ง TBLW ใช้ในการเขียนข้อมูลเป็นเวิร์ดจากแรมข้อมูลที่อยู่ในชิพไปยังแรมโปรแกรมที่อยู่นอกชิพ

การปฏิบัติคำสั่ง TBLR จะสร้างสัญญาณ $\overline{\text{MEN}}$ (Memory Enable) ขึ้นมาเพื่อใช้ในการอ่านข้อมูลจากหน่วยความจำโปรแกรม ส่วนการปฏิบัติคำสั่ง TBLW จะสร้างสัญญาณ $\overline{\text{WE}}$ ขึ้นมา



รูปที่ ค.8.1 แสดงถึงช่วงเวลาของคำสั่ง TBLR



รูปที่ ค.8.2 แสดงถึงช่วงเวลาของคำสั่ง TBLW

จากรูปที่ ค.8.1 และ ค.8.2 ช่วงการเรียก Dummy จะทำการเรียกคำสั่งที่อยู่ถัดจากคำสั่ง TBLR หรือ TBLW เข้ามาแล้วละทิ้งไป หลังจากนั้นคำสั่งที่อยู่ถัดจากคำสั่ง TBLR หรือ TBLW นี้จะถูกเรียกอีกครั้งหลังจากทำการปฏิบัติคำสั่ง TBLR หรือ TBLW เสร็จสิ้นแล้ว

6.3 การเข้ารหัสที่อยู่ (Address Bus Decoding)

เนื่องจากสัญญาณในการติดต่อทั้งสามนี้ คือ สัญญาณ $\overline{\text{MEN}}$, $\overline{\text{WE}}$ และ $\overline{\text{DEN}}$ ต่างมีความเกี่ยวข้องซึ่งกันและกันอยู่เป็นลักษณะเฉพาะ ดังนั้นจึงเป็นสิ่งสำคัญในการที่จะนำมาใช้ในการออกแบบ เมื่อต้องการใช้หน่วยความจำโปรแกรมที่อยู่ภายนอกชิ้นมา ด้วยเหตุที่คำสั่ง OUT กับคำสั่ง TBLW นั้นใช้สัญญาณ $\overline{\text{WE}}$ เพียงอย่างเดียวในการบ่งบอกถึงช่วงที่ข้อมูลถูกต้อง สามารถใช้ข้อมูล

ในช่วงนั้นได้ นั่นคือคำสั่งทั้งสองนี้จะไม่สามารถแยกความแตกต่างจากกัน ในด้านการดูสัญญาณควบคุมในการติดต่อได้เลย นอกเสียจากว่าการใช้บัสที่อยู่ (Address Bus) มาทำการเข้ารหัส (Decode) การปฏิบัติคำสั่ง TBLW จะเขียนข้อมูลไปยัง อุปกรณ์รอบข้าง ส่วนการปฏิบัติคำสั่ง OUT จะทำการเขียนทับหน่วยความจำโปรแกรมที่ตำแหน่ง 0 ถึง 7

ไม่ว่าจะมีการใช้ลอจิกในการเข้ารหัส อย่างไรก็ตามจะไม่มีทางที่จะใช้คำสั่ง TBLW ในการเขียนข้อมูลลงไปในหน่วยความจำโปรแกรมที่ตำแหน่ง 0 ถึง 7 อย่างเป็นเอกเทศ ทั้งนี้เพราะว่าบัสที่อยู่นั้นเป็นอันเดียวกัน สำหรับทั้งคำสั่ง OUT และ TBLW

7. ขา $\overline{\text{BIO}}$

เป็นขาสัญญาณที่ต่อออกสู่ภายนอก คอยทำการตรวจสอบสัญญาณที่ขานี้ ถ้าเมื่อใดสัญญาณที่ขานี้เป็นลอจิก "0" การปฏิบัติคำสั่ง BIOZ จะทำการเคลื่อนย้ายไปทำในตำแหน่งที่กำหนดทันที โดยจะมีการสุ่มดูสัญญาณที่ขานี้ทุก ๆ รอบนาฬิกาโดยไม่มีการค้างค่า (Latch) เอาไว้

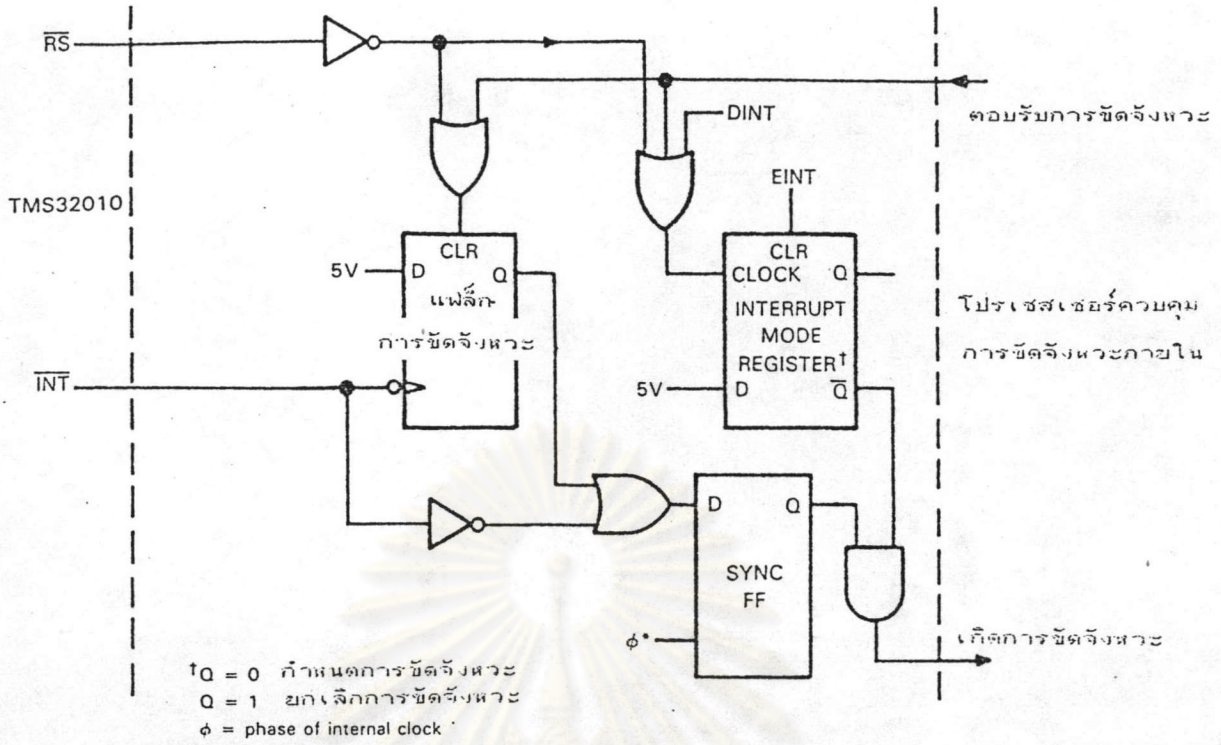
ขาสัญญาณ $\overline{\text{BIO}}$ นี้มีประโยชน์มาก สำหรับใช้ในการตรวจตราดูสถานะของอุปกรณ์รอบข้างภายนอก และเป็นทางเลือกที่มีประโยชน์อีกทางหนึ่ง นอกจากการใช้สัญญาณขัดจังหวะ ในกรณีที่ไม่มีความจำเป็นที่จะต้องไปขัดจังหวะของ Time Critical Loops

8. สัญญาณขัดจังหวะ (Interrupt)

สัญญาณขัดจังหวะจะถูกสร้างขึ้นมาได้จาก 2 กรณี คือ จากการป้อนสัญญาณช่วงขอบขาลงจากลอจิก "1" เป็น "0" เข้าที่ขา $\overline{\text{INT}}$ (Interrupt) หรือโดยการทำให้สัญญาณที่ขา $\overline{\text{INT}}$ คงค่าเป็นลอจิก "0" อยู่ก็ได้ แผนผังแสดงถึงวงจรภายในที่ทำหน้าที่ในการขัดจังหวะของ TMS32010 ดังแสดงในรูปที่ ค.9

โดย SYNC FF เป็นตัวฟลิบฟลอบ (Flip-Flop) ที่ใช้ในการสมกาล (Synchronize) สัญญาณขัดจังหวะภายนอก กับวงจรสัญญาณขัดจังหวะภายในตัว TMS32010 เมื่อมีการยินยอมให้มีการรับสัญญาณขัดจังหวะแล้ว การขัดจังหวะจะเกิดขึ้นเมื่อมีสัญญาณลอจิก "0" ปรากฏที่ขา $\overline{\text{INT}}$ หรือเมื่อมีสัญญาณขอบขาลงถูกค้างค่าเอาไว้ในอินเทอร์รัพท์แฟล็ก (Interrupt Flag) อย่างใดอย่างหนึ่ง

ถ้าเรจิสเตอร์ INTM ถูกกำหนด สัญญาณขัดจังหวะที่เกิดขึ้นจะสามารถผ่านไปยังโปรเซสเซอร์ควบคุมการขัดจังหวะภายใน (Internal Interrupt Processor: IIP) ได้ IIP จะเริ่มทำการให้บริการสัญญาณขัดจังหวะที่เกิดขึ้น โดยการเคลื่อนย้ายไปยังตำแหน่งที่ 2 ในหน่วยความจำโปรแกรม โดยจะมีการหน่วงเวลาของการบริการสัญญาณขัดจังหวะที่เกิดขึ้น ในกรณีดังต่อไปนี้



รูปที่ ค.9 แสดงถึงแผนผังลอจิกในการขัดจังหวะอย่างง่าย

ก. จนกระทั่งจบรอบทั้งหลายของคำสั่งแบบมัลติไซเคิล (Multicycle)

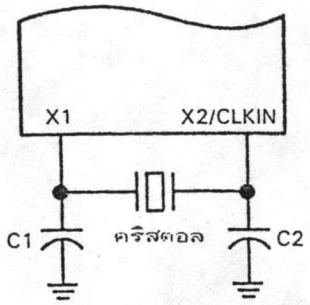
ข. จนกระทั่งคำสั่งที่ตามหลังคำสั่ง MPY หรือ MPYK ถูกปฏิบัติจนหมดแล้ว

ค. จนกระทั่งคำสั่งที่ตามหลังคำสั่ง EINT ถูกปฏิบัติ เมื่อมีการยกเลิกสัญญาณขัดจังหวะมาก่อนหน้านี้แล้วในแบบนี้จะทำให้คำสั่ง RET (Return) ที่ถูกปฏิบัติหลังจากมีการขัดจังหวะ และเป็นการกำหนดสัญญาณขัดจังหวะ ที่ช่วงท้ายของรูทีนขัดจังหวะ

เมื่อเริ่มเข้าสู่รูทีนบริการการขัดจังหวะแล้ว IIP จะส่งสัญญาณตอบรับออกมา ซึ่งจะเป็นการไปยกเลิกบิต INTM หรือเป็นการยกเลิกสัญญาณขัดจังหวะนั่นเอง และก็จะเป็นการไปเปลี่ยนอินเทอร์รัพท์แฟล็กกลับด้วย

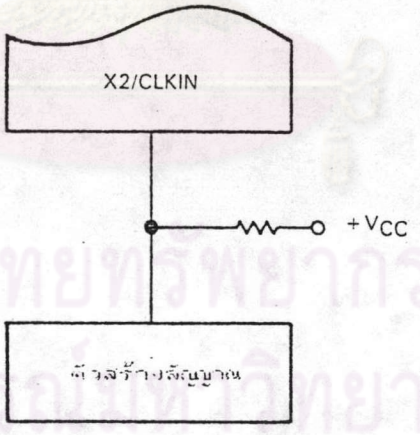
ในรูปที่ ค.9 จะแสดงถึงคำสั่ง DINT หรือสัญญาณที่เป็นการรีเซทโดยฮาร์ดแวร์ จะไปกำหนดเรจิสเตอร์ INTM และทำการห้ามการขัดจังหวะ ในขณะที่คำสั่ง EINT จะเป็นตัวเปลี่ยนเรจิสเตอร์ INTM สัญญาณขัดจังหวะจะยังคงถูกค้างค่าอยู่ในขณะที่ยังมีการห้ามสัญญาณขัดจังหวะไว้ อย่างไรก็ตามคำสั่ง DINT และ EINT จะไม่ส่งผลถึงอินเทอร์รัพท์แฟล็กเลย

ในรูปที่ ค.10 จะแสดงถึงลำดับของคำสั่งที่ปรากฏขึ้นมา เมื่อมีการขัดจังหวะเกิดขึ้น ส่วนของการเรียก Dummy คือคำสั่งที่จะถูกทำการเรียกเข้ามา แต่ไม่มีการปฏิบัติและคำสั่งนี้จะถูกดึง และทำการปฏิบัติอีกครั้ง หลังจากการทำรูทีนการขัดจังหวะเสร็จสิ้นแล้ว



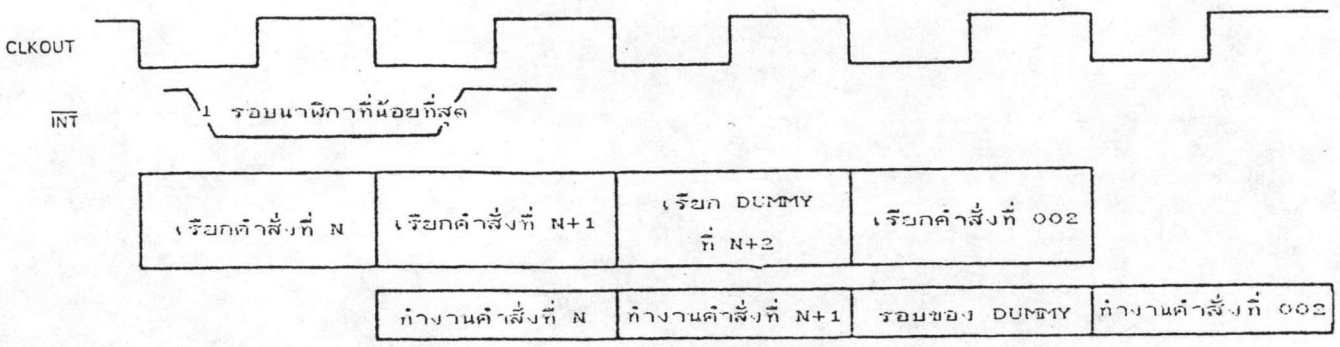
รูปที่ ค.12 แสดงถึงการต่อนาฬิกาภายใน

การใช้แหล่งความถี่จากภายนอก ทำได้โดยการส่งสัญญาณเข้ามาโดยตรงที่ขา X2/CLKIN แล้วปล่อยขา X1 ลอยไว้เฉย ๆ แต่ในกรณีที่ใช้แหล่งความถี่จากภายนอกนี้ ควรจะใช้ตัวความต้านทานมาต่อพูลอัพ (Pull-up) เข้าดังรูปที่ ค.13 เพราะว่าระดับแรงดันของสัญญาณลอจิก "1" จะต้องมีค่าอย่างต่ำ 2.8 โวลต์ ในขณะที่ลอจิกเกทแบบ ทีทีแอล (TTL) มาตรฐานทั่ว ๆ ไปสามารถมีลอจิกเป็น "1" ได้ เมื่อมีแรงดันเข้ามาประมาณ 2.4 โวลต์ เท่านั้น ค่าของตัวต้านทานที่มาต่อพูลอัพนี้ขึ้นกับสิ่งต่าง ๆ เช่น ระดับแรงดันของลอจิก "1" ของแหล่งสัญญาณภายนอก กระแสและจำนวนของอุปกรณ์ที่แหล่งสัญญาณภายนอกนั้นต้องรับภาระ ตัวความต้านทานควรจะใช้ค่าที่สูงที่สุดเท่าที่จะทำได้ และยังคงมีสัญญาณ CLKIN เข้ามาถูกต้องตามความต้องการ



รูปที่ ค.13 แสดงถึงการต่อแหล่งกำเนิดความถี่ภายนอก

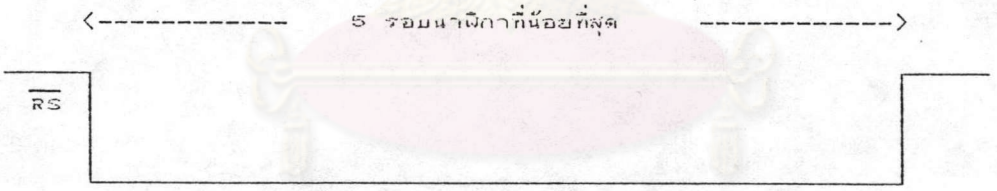
การล่าช้าทางเวลาของสัญญาณ CLKIN และ CLKOUT จะไม่มีค่าคงที่ แต่จะแปรไปอาจจะถึง 1 คาบของสัญญาณและยังขึ้นกับอุณหภูมิด้วย ดังนั้นการออกแบบทางฮาร์ดแวร์ ที่ต้องขึ้นอยู่กับความล่าช้าของเวลานี้ด้วยไม่ควรจะนำมาใช้



รูปที่ ค.10 แสดงถึงช่วงเวลาในขณะเกิดการขัดจังหวะ

9. สัญญาณรีเซท (Reset)

สัญญาณรีเซทจะเกิดขึ้นมา เมื่อมีการทำให้สัญญาณที่ขา $\overline{RS}$ เป็นลอจิก "0" ไม่ต่ำกว่า 5 รอบนาฬิกา (ดังรูปที่ ค.11) สัญญาณที่ขา $\overline{DEN}$, $\overline{WE}$ และ $\overline{MEN}$ จะถูกบังคับให้เป็นลอจิก "1" ส่วนบัสข้อมูลจะอยู่ในสถานะลอยตัว บัสที่อยู่ A11 ถึง A0 จะถูกลบในช่วงรอบต่อไป หลังจากขอบขาลงของสัญญาณ $\overline{RS}$ แล้ว ขา $\overline{RS}$ จะทำการห้ามการขัดขังหวะ ลบอินเทอร์รัพท์แฟล็กเรจิสเตอร์และออกจากสภาวะล้น โดยเรจิสเตอร์ไม่เปลี่ยนแปลง



รูปที่ ค.11 แสดงถึงช่วงเวลาของสัญญาณรีเซท

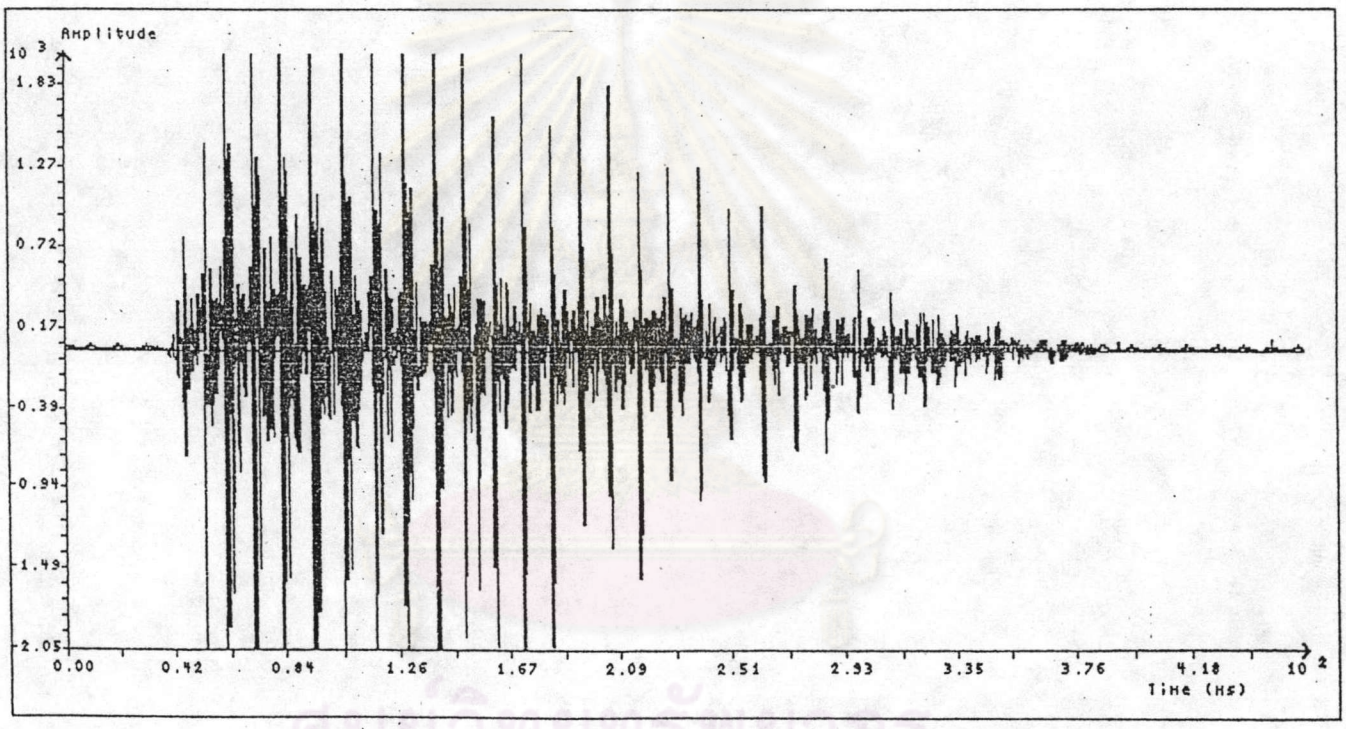
10. นาฬิกาและตัวแกว่ง (Clock/Oscillator)

TMS32010 สามารถใช้ได้ทั้งจากการแกว่ง (Oscillate) จากภายใน หรือจะรับสัญญาณนาฬิกาจากแหล่งความถี่ภายนอกก็ได้

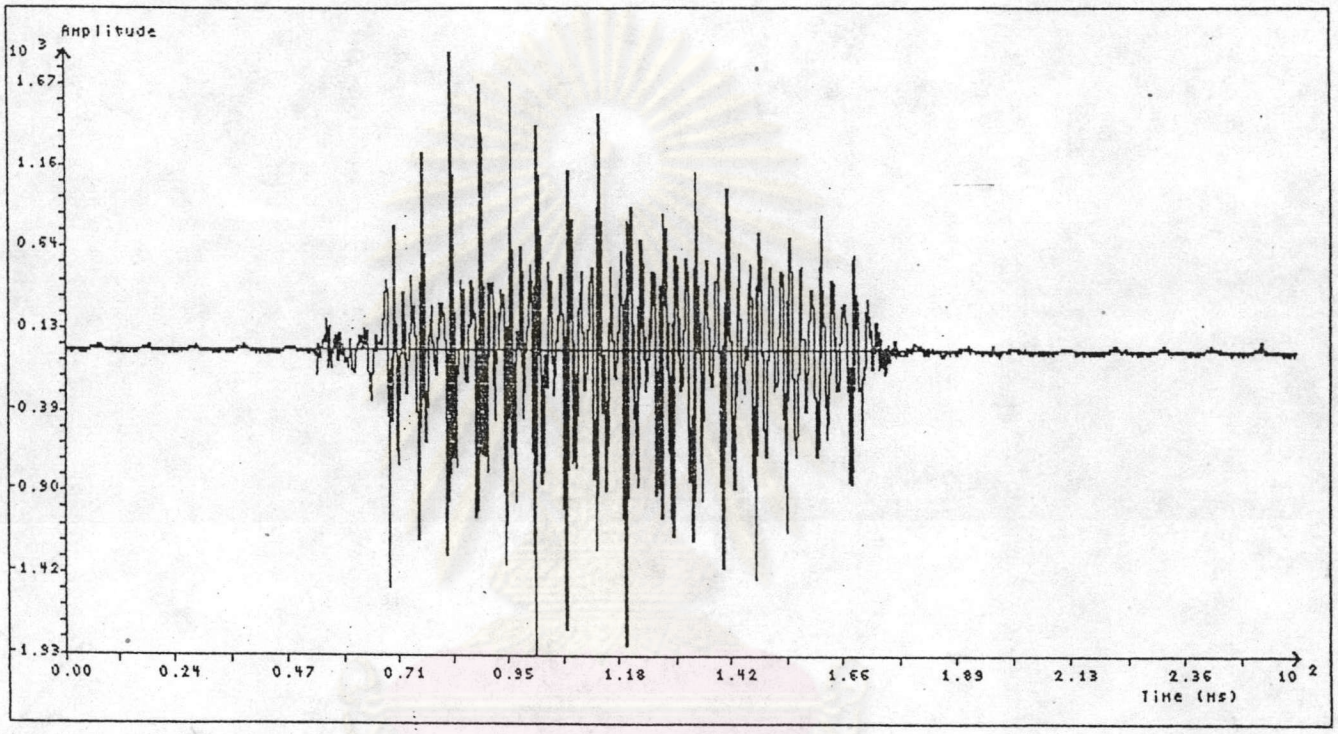
การใช้ตัวแกว่งภายใน (Internal Oscillator) ทำได้โดยการต่อคริสตอล (Crystal) เข้าไประหว่างขา X1 และ X2/CLKIN ความถี่ CLKOUT และช่วงเวลาของสัญญาณนาฬิกาจะเท่ากับ 1/4 ของความถี่พื้นฐานของตัวคริสตอล ดังรูปที่ ค.12

ภาคผนวก ง

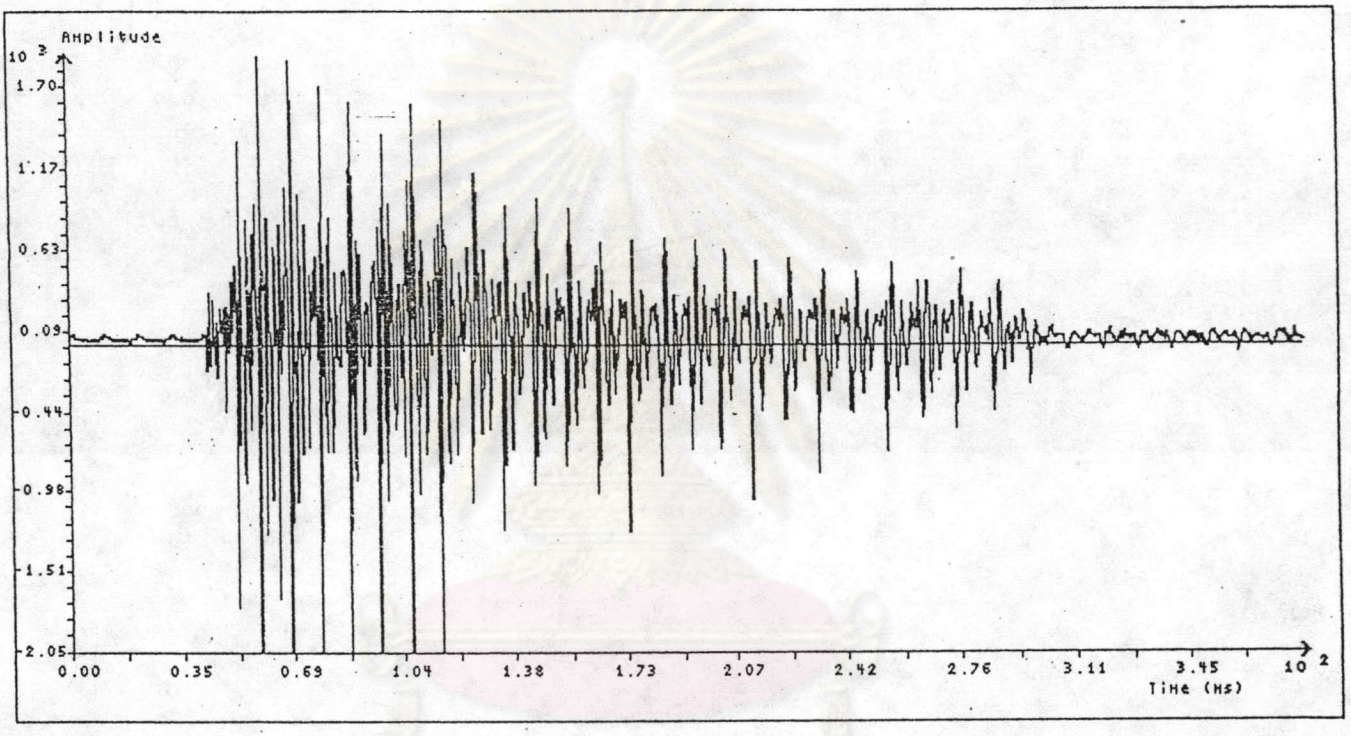
แสดงตัวอย่างของภาพสัญญาณเสียงที่ใช้ในการทดสอบ



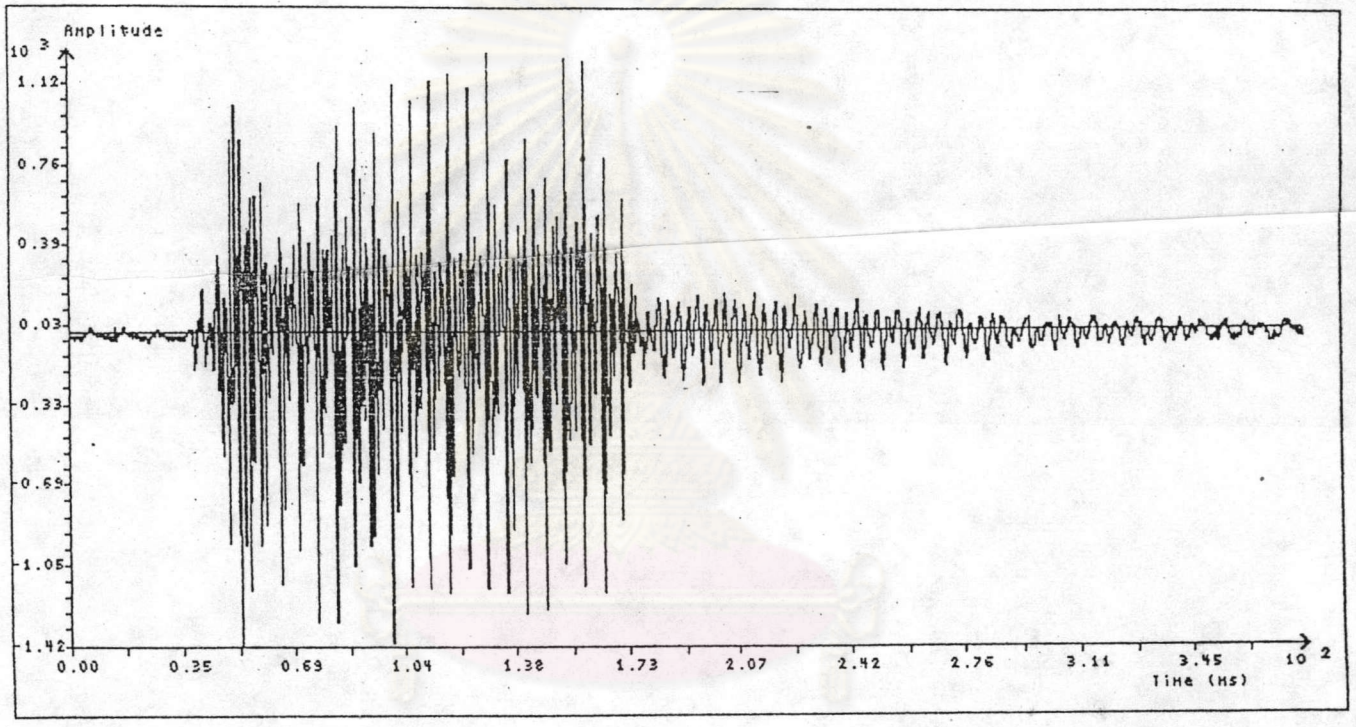
รูปที่ ง.1 แสดงภาพสัญญาณเสียงของคำว่า "ปาก"



รูปที่ ง.2 แสดงภาพสัญญาณเสียงของคำว่า "ปิด"



รูปที่ ง.3 แสดงภาพสัญญาณเสียงของคำว่า "ปิน"



รูปที่ ง.4 แสดงภาพสัญญาณเสียงของคำว่า "เป็น"

ประวัติผู้เขียน

นายไพศาล ธรรมโชติทอง เกิดปี พ.ศ. 2508 ที่กรุงเทพมหานคร สำเร็จการศึกษาวิทยาศาสตร์บัณฑิต ภาควิชาเคมีวิศวกรรม คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปี พ.ศ. 2526 ปัจจุบันเป็น Programmer Analyst ที่ซิตี้แบงก์ (ประเทศไทย)