บทที่ 5

เครื่องบันทึกเสียงพูดดิจิตอล

5.1 การพัฒนาโปรแกรมเข้ารหัสและถอดรหัส

5.1.1 การพัฒนาโปรแกรมเข้ารหัสและถอดรหัสโดยใช้การจำลองโปรแกรม CCS

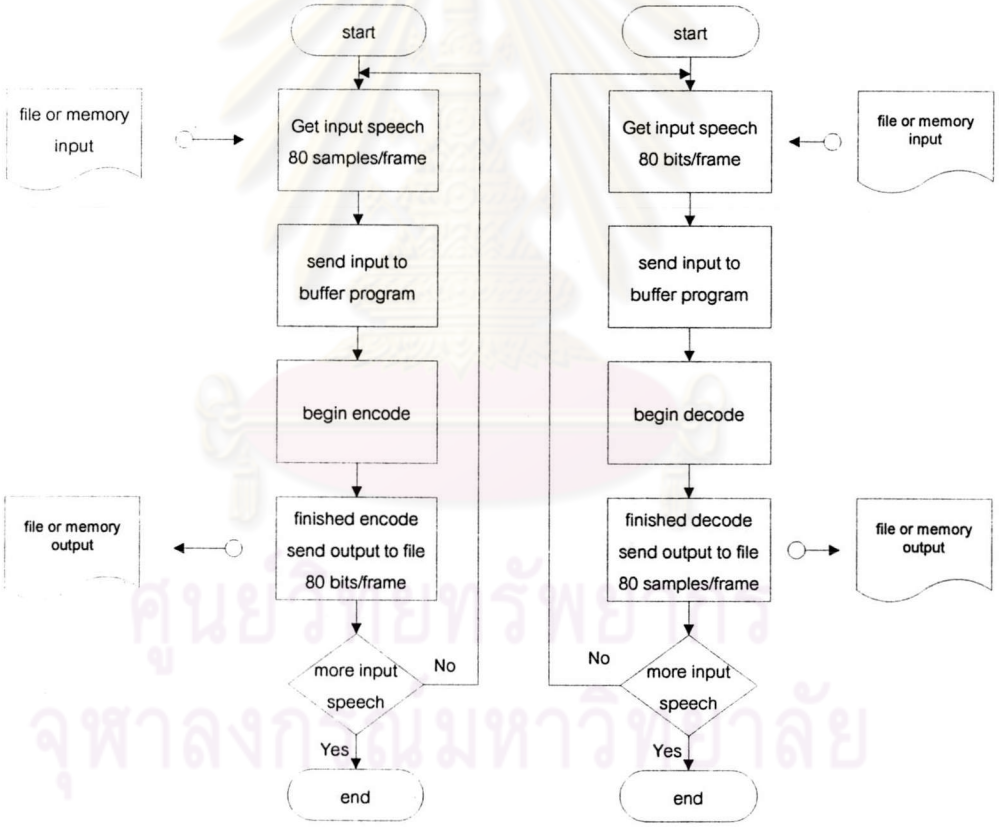
การเขียนโปรแกรมโดยใช้การจำลองโปรแกรม CCS ในขั้นนี้เป็นขั้นตอนแรกของการทำงานมีจุดประสงค์เพื่อศึกษาและทำความเข้าใจกับหลักการทำงานของการเข้ารหัสแบบ CS-ACELP ตามมาตรฐาน G.729 อย่างถูกต้องเสียก่อน การเขียนโปรแกรมโดยใช้การจำลองโปรแกรม CCS ไม่ใช่การทำงานแบบเวลาจริง ดังนั้นข้อมูลขาเข้าที่ใช้จึงต้องเก็บไว้เป็นไฟล์หรือกำหนดไว้ในหน่วยความจำ โดยข้อมูลเสียงที่ใช้ได้มาจากการบันทึกเสียงแล้วเก็บไว้ก่อนและหลังจากการเข้ารหัสเสร็จก็จะถูกเก็บไว้ในไฟล์หรือหน่วยความจำเช่นเดิม ในขั้นตอนนี้จะแบ่งการเขียนโปรแกรมออกเป็นสองส่วนคือส่วนโปรแกรมการเข้ารหัสและส่วนของโปรแกรมการถอดรหัส ดังแสดงโฟล์วชาร์ตการทำงานของการเข้ารหัสและถอดรหัสดังรูปที่ 5.1

การเขียนโปรแกรมเริ่มจากการเขียนโปรแกรมการเข้ารหัสก่อน ขั้นตอนการดำเนินงานเริ่มตั้งแต่การติดตั้งโปรแกรม Code Composer Studio V 2.0 จากนั้นเชื่อมต่อบอร์ดทดลอง TMS320C6711 DSK เข้ากับคอมพิวเตอร์ส่วนบุคคล ผ่านทางพอร์ตขนาน รันโปรแกรม Code Composer Studio และสร้างไฟล์โปรเจคท์ (coder.pjt) ทำการเพิ่มไฟล์ที่จำเป็นในการใช้งานบอร์ดทดลอง TMS320C6711 DSK ได้แก่ dskc6711.cmd (เป็นไฟล์กำหนดตำแหน่งในการใช้หน่วยความจำ) rts6701.lib เป็นไลบรารีไฟล์ของดีเอสพี TMS320C6711 ไฟล์ Vectors.asm เป็นไฟล์ สำหรับกำหนดตำแหน่งการบริการการอินเตอร์รัปต์ จากนั้นทำการสร้างไฟล์ภาษาซี (*.c) ที่ประกอบไปด้วยฟังก์ชันหลัก main() เพื่อทำหน้าที่การเข้ารหัส

การเขียนโปรแกรมการเข้ารหัสจะเริ่มจากการเขียนเป็นโปรแกรมย่อยในการทำงานของแต่ละบล็อกว่าสามารถทำงานได้ถูกต้องหรือไม่ จากนั้นจึงรวมฟังก์ชันย่อยที่ทำหน้าที่อยู่ในกลุ่มที่ต่อเนื่องกันไว้ในไฟล์เดียวกันโดยมีไฟล์หลักเรียกใช้ฟังก์ชันย่อยในแต่ละไฟล์ตามบล็อกการทำงานของตัวเข้ารหัสเพื่อให้สามารถแก้ไขและตรวจสอบได้ง่ายขั้นตอนการทำงานของโปแกรมการเข้ารหัสจะประกอบไปด้วยฟังก์ชันหลักที่ทำหน้าที่เรียกใช้ฟังก์ชันย่อยในการเข้ารหัสตามบล็อกไดอะแกรมของการเข้ารหัส ในการตรวจสอบการทำงานของโปรแกรมจะเปรียบเทียบผลลัพธ์ที่ได้จาก

การคำนวณโดยใช้ดีเอสพีกับผลลัพธ์ที่ได้จากมาตรฐานของ ITU-T ว่าผลลัพธ์ที่ได้มีค่าหรือมีแนวโน้มของพารามิเตอร์ต่างๆ ใกล้เคียงกันหรือไม่

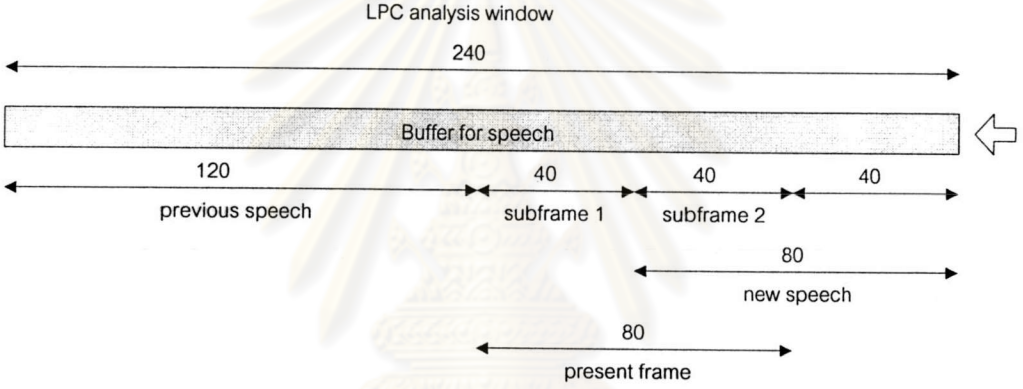
การเขียนโปแกรมในขั้นตอนนี้จะเลือกโหมดการทำงานของโปรแกรม CCS เป็นแบบตัวดีบัก (Debugger) เพราะว่าการเลือกการทำงานแบบดีบักสามารถที่จะสั่งให้ดีเอสพีทำงานแบบทีละคำสั่ง (Single step) หรือคำสั่งรัน (Run) เพื่อให้ทำงานจนจบโปรแกรมหรือสามารถที่จะหยุด ณ. ตำแหน่งที่ต้องการได้ทันทีโดยใช้เครื่องมือ Breakpoint การเลือกการทำงานแบบดีบักทำให้เราสามารถตรวจสอบค่าตัวแปรหรือการเปลี่ยนแปลงค่าตัวแปรต่างๆ ได้โดยใช้เครื่องมือ Watch windows หรือสามารถดูค่าที่เก็บอยู่ภายในหน่วยความจำได้ทุกตำแหน่งและยังสามารถดูโปรแกรมภาษาแอสเซมบลีที่ได้จากการคอมไพล์ภาษาซี และสามารถดูรูปสัญญาณต่างๆ ได้ทั้งในโดเมนเวลาและโดเมนความถี่ โดยใช้ Graph คุณสมบัติของเครื่องมือหลายๆ อย่างที่มีอยู่ในโปรแกรม CCS จึงทำให้ง่ายและสะดวกต่อการพัฒนา การตรวจสอบ การแก้ไขโปแกรม



รูปที่ 5.1 โฟล์วชาร์ตการทำงานของตัวเข้ารหัสและถอดรหัสโดยใช้การจำลองโปรแกรม CCS

ขั้นตอนการทำงานของโปรแกรมการเข้ารหัสจะมีฟังก์ชันหลักในการเรียกใช้ฟังก์ชันย่อยในการทำงาน สามารถอธิบายโดยรวมได้ดังนี้

1. กำหนดค่าตัวแปรแบบสแตติก(Static variable) และตัวแปรโกลบอล(Global variable) เพื่อใช้ในการคำนวณ
2. เรียกฟังก์ชันย่อยเพื่อกำหนดค่าเริ่มต้นของวงจรกรองสัญญาณ Pre-processing
3. เรียกฟังก์ชันย่อยเพื่อกำหนดค่าพารามิเตอร์และบัฟเฟอร์ที่ใช้สำหรับเก็บข้อมูลเสียงที่ใช้ในการวิเคราะห์และกำหนดค่าเริ่มต้นให้เป็นศูนย์ โดยในการเข้ารหัสเสียงแบบ CS-ACELP จะแบ่งหน้าต่างที่ใช้ในการวิเคราะห์ขนาด 240 ตัวอย่างสุ่มและแบ่งบัฟเฟอร์ในการเก็บค่าพารามิเตอร์ต่างๆ ดังแสดงในรูปที่ 5.2



รูปที่ 5.2 ชุดของสัญญาณสุ่มที่ใช้ในการวิเคราะห์ตัวทำนายเชิงเส้น

4. อ่านค่าอินพุตแบบจำนวนเต็มขนาด 16 บิต จำนวน 1 เฟรมหรือ 80 ตัวอย่างสุ่มเข้ามาเก็บไว้ในบัฟเฟอร์อินพุต
5. ทำการแปลงค่าในบัฟเฟอร์อินพุตแบบจำนวนเต็มให้เป็นแบบเลขทศนิยมและเก็บไว้ในบัฟเฟอร์ new speech
6. เรียกฟังก์ชันกรองสัญญาณแบบความถี่สูงผ่านที่มีความถี่ตัด 140 เฮิรตซ์ เพื่อทำหน้าที่กรองสัญญาณอินพุตขนาด 240 ตัวอย่างสุ่มแล้วเก็บไว้ที่ตำแหน่งเดิม
7. เรียกฟังก์ชันย่อยเพื่อคำนวณหาค่าอัตสหสัมพันธ์โดยใช้สัญญาณทั้งหมด 240 ตัวอย่างสุ่ม และค่าอัตสหสัมพันธ์ที่ได้จากการคำนวณจะนำมาใช้ในการคำนวณหาค่าสัมประสิทธิ์ของวงจรกรองตัวทำนายเชิงเส้น หลังจากที่ได้ค่าสัมประสิทธิ์ LP แล้วจะ

ทำการเปลี่ยนค่าสัมประสิทธิ์ LP ให้เป็นค่า Line Spectral Pairs (LSP) เพื่อนำไปทำการควอนไทซ์และทำการประมาณค่าในช่วง

8. หลังจากที่ได้ค่า สัมประสิทธ์ LSP จะเรียกฟังก์ชันย่อยเพื่อทำการควอนไทซ์สัมประสิทธิ์ LSP โดยการเปลี่ยนสัมประสิทธิ์ LSP ให้เป็นค่าสัมประสิทธิ์ LSF และใช้ตัวทำนายแบบ MA ที่มีอันดับเท่ากับ 4 ในการทำนายค่าสัมประสิทธิ์ LSF โดยใช้บิต L0 เป็นตัวเก็บค่าและ ค่าสัมประสิทธิ์ที่คำนวณได้กับค่าที่ทำนายได้จะถูกควอนไทซ์แบบเวกเตอร์แบบ 2 สเตจ โดยสเตจแรกจะใช้เวกเตอร์ขนาด 1x10 จำนวน 128 รูปแบบ (7บิต) และใช้ตัวชี้ L1 ส่วนสเตจที่สองเป็นเวกเตอร์ควอนไทซ์แบบ 10 บิตที่ถูกสร้างขึ้นจากชุดเวกเตอร์ที่มีขนาด 1x5 2 ตัว ใช้ตัวชี้เป็น L2 L3 สำหรับข้อมูล 32 รูปแบบ (5 บิต) ทำการเก็บค่าพารามิเตอร์เอาต์พุต L0 L1 L2 L3 ไว้ในบัฟเฟอร์เอาต์พุต
9. ทำการประมาณค่าในช่วงสัมประสิทธิ์ LSP ทั้งที่ควอนไทซ์และยังไม่ควอนไทซ์ จากนั้นทำการเปลี่ยนค่าสัมประสิทธิ์ LSP เป็นสัมประสิทธิ์ตัวทำนายเชิงเส้นเพื่อใช้ในการคำนวณหาสัญญาณเอ็กไซเทชัน
10. เรียกฟังก์ชันคำนวณสัญญาณเอ็กไซเทชัน โดยใช้การกรองสัญญาณความผิดพลาดด้วยวงจรกรองสัญญาณแบบ Perceptual weighting ที่มีสัมประสิทธิ์ได้มาจากสัมประสิทธิ์ของวงจรกรองสัญญาณ LP ที่ยังไม่ได้ควอนไทซ์
11. คำนวณหาค่า open-loop pitch การหาค่า open-loop pitch เพื่อที่จะลดความซับซ้อนในการค้นหาค่าประวิงเวลาใน adaptive-codebook โดยสามารถประมาณค่าได้โดยใช้สัญญาณเสียงที่ถูกถ่วงน้ำหนักแล้วทำการหาค่าสหสัมพันธ์กันมากที่สุด 3 ค่า
12. ในขั้นตอนที่ผ่านมาจะเป็นการคำนวณแบบเป็นเฟรม ในขั้นตอนต่อจากนี้ไปจะเป็นการคำนวณทีละเฟรมย่อย การทำงานของเฟรมย่อยเริ่มจากการคำนวณหาค่าการตอบสนองของอิมพัลส์ h(n) ของวงจรกรองสัญญาณ weighted synthesis เพราะจำเป็นต้องใช้ในการคำนวณ adaptive-codebook และ fixed-codebook
13. คำนวณหาค่าสัญญาณเป้าหมาย x(n) โดยคำนวณจากการลบการตอบสนองของวงจรกรอง weight synthesis ออกจากสัญญาณเสียงที่ถูกถ่วงน้ำหนัก
14. คำนวณหาค่าพารามิเตอร์ adaptive-codebook ประกอบด้วยค่าประวิงเวลา (delay) และอัตราการขยาย โดยใช้วงจรกรองสัญญาณพิตช์ซึ่งจะทำการค้นหาสัญญาณกระตุ้นที่มีการประวิงเวลาน้อยกว่าความยาวของเฟรมย่อย จากนั้นจะทำการเข้ารหัสค่าประวิงเวลาของ adaptive-codebook ซึ่งก็คือค่า P1 ขนาด 6 บิต

สำหรับเฟรมย่อยแรก ส่วนเฟรมย่อยที่ 2 คือ P2 มีขนาด 3 บิตและคำนวณค่าพาริตี P0 จากค่าของ P1 เพื่อใช้ในการตรวจสอบการผิดพลาดของข้อมูล จากนั้นทำการคำนวณหาค่าอัตราการขยายของ adaptive-codebook

15. คำนวณค่า fixed-codebook โดยการหาค่าต่ำสุดของ mean squared error ระหว่างสัญญาณเสียงที่ถูกถ่วงน้ำหนัก และสัญญาณเสียงที่สร้างกลับขึ้นมาที่ถูกถ่วงน้ำหนักแล้ว จากนั้นทำการเข้ารหัส fixed-codebook โดยที่ตำแหน่งของพัลส์สามพัลส์แรกจะเข้ารหัสชุดละ 3 บิต พัลส์ที่สี่ใช้จำนวน 4 บิตและแต่ละพัลส์จะมีค่าขนาดหรือเครื่องหมายอีกพัลส์ละบิตรวมทั้งหมดแล้ว 17 บิต
16. คำนวณหาค่า code-book ของการควอนไทซ์อัตราการขยายโดยในสเตจแรกจะประกอบด้วย code-book A จำนวน 4 บิต และสเตจที่สองประกอบด้วย codebook B ขนาด 4 บิต
17. ทำการปรับหน่วยความจำเพื่อใช้สำหรับการคำนวณของเฟรมย่อยถัดไป และทำการคำนวณเฟรมย่อยที่ 2 โดยที่ในส่วนของพาริตีบิตนั้นไม่ต้องคำนวณจากนั้นทำการปรับหน่วยความจำอีกครั้งก่อนกลับสู่ฟังก์ชันหลัก พารามิเตอร์ที่ได้จากการถอดรหัสแต่ละบิตแสดงดังตารางที่ 3.3
18. ทำการเก็บค่าพารามิเตอร์จำนวน 80 บิต ต่อเฟรมไว้ในหน่วยความจำ จากนั้นก็กลับสู่โปรแกรมหลักเพื่อเริ่มคำนวณสำหรับเฟรมต่อไป

ในส่วนของตัวถอดรหัสก็จะมีลักษณะของโปรแกรมคล้ายกับการเข้ารหัสคือเริ่มจากการสร้างไฟล์โปรเจคท์ (coder.pjt) ทำการเพิ่มไฟล์ที่จำเป็นในการใช้งานบอร์ดทดลอง TMS320C6711 DSK ได้แก่ ไฟล์ dskc6711.cmd ไฟล์ rts6701.lib ไฟล์ Vectors.asm จากนั้นทำการสร้างไฟล์ภาษาซี (*.c) ที่ประกอบไปด้วยฟังก์ชันหลัก main() เพื่อทำหน้าที่ถอดรหัสการเขียนโปรแกรมจะสร้างเป็นฟังก์ชันย่อยก่อนจากนั้นจึงรวมฟังก์ชันย่อยที่ทำหน้าที่ต่อเนื่องกันไว้ในไฟล์เดียวกันและมีฟังก์ชันหลักสำหรับการเรียกใช้งานในแต่ละไฟล์ตามบล็อกไดอะแกรมของการถอดรหัส ขั้นตอนการทำงานของโปรแกรมสามารถอธิบายโดยรวมได้ดังนี้

1. กำหนดค่าตัวแปรแบบสแตติกและตัวแปรแบบโกลบอลเพื่อใช้ในการคำนวณ
2. เรียกฟังก์ชันย่อยเพื่อกำหนดค่าเริ่มต้นของวงจรกรองสัญญาณ Post-processing
3. เรียกฟังก์ชันย่อยเพื่อกำหนดค่าพารามิเตอร์และบัฟเฟอร์ที่ใช้สำหรับเก็บสัญญาณเสียงที่ทำการสร้างกลับคืน
4. กำหนดตัวแปรเริ่มต้นและพารามิเตอร์สำหรับวงจรกรอง Post-filter
5. รับค่าพารามิเตอร์อินพุตที่ได้เข้ารหัสไว้มาทีละเฟรม (80 บิต)

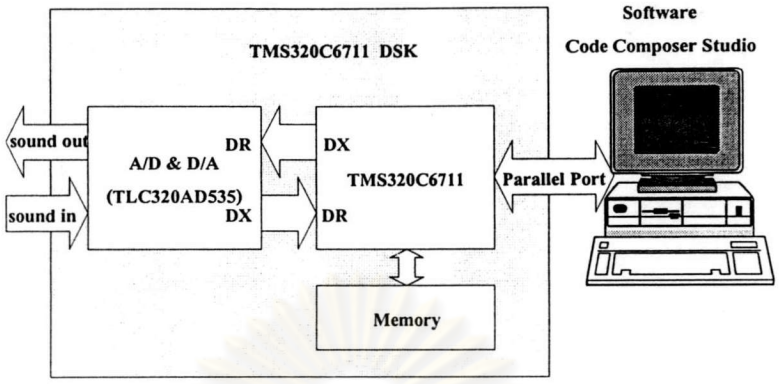
6. ทำการถอดรหัสพารามิเตอร์ L0 L1 L2 L3 เพื่อสร้างสัมประสิทธิ์ LSP ที่ถูกควอนไทซ์และทำการประมาณค่าในช่วง 2 ชุด (ของเฟรมย่อยแต่ละเฟรม) จากนั้นเปลี่ยนสัมประสิทธิ์ LSP ที่มาจากการประมาณค่าในช่วงจะเปลี่ยนเป็นสัมประสิทธิ์ตัวทำนายเชิงเส้นเพื่อใช้ในการสังเคราะห์เสียง
7. การคำนวณในลำดับต่อนี้ไปจะคำนวณทีละเฟรมย่อยโดยเริ่มคำนวณค่าพาริตีบิตจากค่าของ P1 ถ้าพาริตีบิตที่คำนวณขึ้นมาไม่ตรงกับพาริตีบิต P0 ค่าประวิงเวลา T1 จะถูกปรับให้มีค่าเป็นจำนวนเต็มของ T2 ของเฟรมหน้า
8. ทำการถอดรหัสเวกเตอร์ adaptive-codebook โดยที่เวกเตอร์ของ adaptive-codebook จะได้มาจากการประมาณค่าในช่วงของสัญญาณกระตุ้น
9. ถอดรหัสเวกเตอร์ของ fixed-codebook โดยพารามิเตอร์ของ fixed-codebook ที่ถอดรหัสออกมาใช้ในการหาตำแหน่งของของพัลซ์ของสัญญาณกระตุ้น ส่วนเครื่องหมายของพัลซ์จะได้มาจากพารามิเตอร์ S
10. คำนวณการถอดรหัสอัตราการขยายของ adaptive-codebook และ fixed-codebook
11. ทำการสังเคราะห์เสียงกลับคืนมาในเฟรมย่อยแต่ละเฟรมและสัญญาณเสียงที่ได้จากการสังเคราะห์ขึ้นมานั้นจะถูกป้อนให้กับวงจรกรอง Post-processing และวงจรกรองสัญญาณความถี่สูงผ่านและวงจรปรับขนาด
12. จากนั้นทำการเปลี่ยนค่าของสัญญาณเอาต์พุตที่ได้ที่อยู่ในรูปเลขทศนิยมให้อยู่ในรูปของเลขจำนวนเต็มขนาด 16 บิต จำนวน 80 ตัวอย่างสุ่มต่อเฟรมเพื่อส่งออกไปยังบัฟเฟอร์เอาต์พุตหรือเก็บไว้ในหน่วยความจำต่อไป
13. กลับสู่โปรแกรมหลักเพื่อทำการรับค่าพารามิเตอร์อินพุตเข้ามาเพื่อทำการถอดรหัสในเฟรมต่อไป

การพัฒนาโปรแกรมการถอดรหัสจะใช้สัญญาณอินพุตจากโปรแกรมการเข้ารหัสในขั้นตอนแรกเพื่อที่จะสามารถตรวจสอบผลลัพธ์ได้ว่าสามารถถอดที่รหัสได้มีสัญญาณเหมือนกับสัญญาณอินพุตที่ใช้ในการเข้ารหัสเพียงใด

5.1.2 การพัฒนาโปรแกรมเข้ารหัสและถอดรหัสแบบเวลาจริง

หลังจากการเขียนโปรแกรมในโหมดของการดีบัก จนได้ผลเป็นที่น่าพอใจแล้วก็จะมีการปรับปรุงโปรแกรมในส่วนของการรับค่าอินพุตสัญญาณเสียงจริง ผ่านวงจรการแปลงแอนาลอกเป็นดิจิตอลโดยใช้การบริการอินเตอร์รัปในการรับข้อมูลเข้ามาประมวลผลและส่งข้อมูลที่ได้จาก

การประมวลผลออกทางเอาต์พุตผ่านวงจรการแปลงสัญญาณดิจิตอลเป็นแอนาลอกเพื่อให้สามารถทำงานแบบเวลาจริงได้ทัน บล็อกไดอะแกรมการทำงานแบบเวลาจริงแสดงดังรูปที่ 5.3

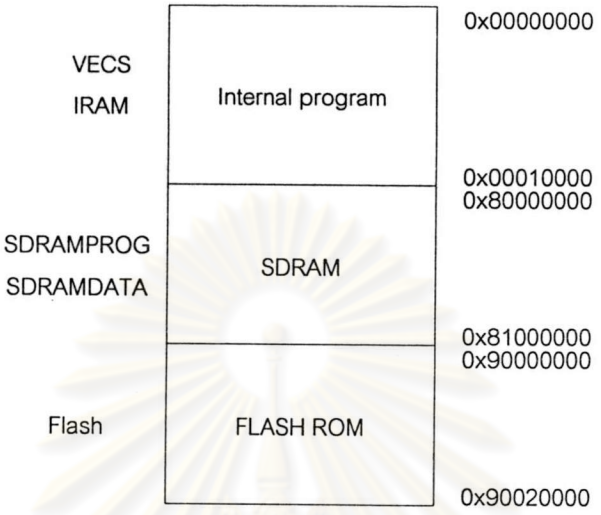


รูปที่ 5.3 บล็อกไดอะแกรมการรับส่งข้อมูลแบบเวลาจริง

ในขั้นตอนที่ผ่านมาการประมวลผลยังไม่เป็นแบบเวลาจริง ดังนั้นการจัดสรรหน่วยความจำในการใช้งานจึงยังไม่มีความสำคัญเท่าใดนัก แต่ในการเข้ารหัสและถอดรหัสแบบเวลาจริงนั้นการประมวลผลต้องมีความเร็ว ดังนั้นในการจัดการหน่วยความจำที่ใช้ในการเบโค้ดและตัวแปรต่างๆ ที่ได้จากการคอมไพล์ก็จะมีผลต่อความเร็วในการทำงานด้วย การกำหนดตำแหน่งในการใช้หน่วยความจำของระบบโดยกำหนดในไฟล์ c6711dsk.cmd ภายในไฟล์ c6711dsk.cmd จะประกอบไปด้วยสองส่วนหลักๆ คือ MEMORY ดังแสดงดังรูปที่ 5.4 และ SECTIONS แสดงดังรูปที่ 5.5

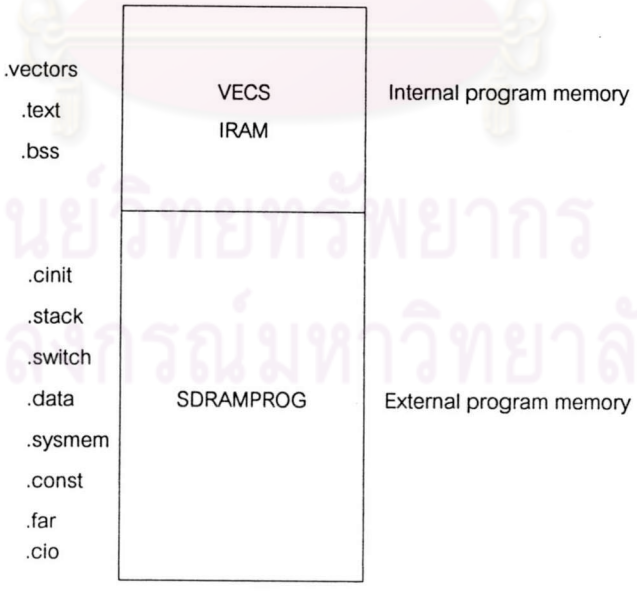
ส่วนที่ 1 MEMORY จะทำหน้าที่แบ่งขอบเขตของหน่วยความจำของระบบทั้งหน่วยความจำภายในชิปและหน่วยความจำภายนอก ออกเป็นส่วนๆ ได้แก่ หน่วยความจำภายในจะประกอบด้วย VECS และ IRAM ใช้สำหรับเก็บโค้ดของการอินเตอร์รัปต์และโค้ดการทำงานของโปรแกรม (Execute code) ส่วนหน่วยความจำภายนอกจะประกอบด้วย SDRAMPROG , SDRAMDATA ในส่วนของ SDRAMPROG จะใช้สำหรับเก็บโค้ดการทำงานของโปรแกรมที่เกินจากหน่วยความจำภายในชิป ส่วน SDRAMDATA จะใช้สำหรับเก็บข้อมูลเสียงที่ได้จากการบันทึกเสียงที่แบ่งออกเป็น 4 ช่อง ส่วนหน่วยความจำแบบ FLASH จะใช้สำหรับเก็บโค้ดโปรแกรมในการทำงานที่ไม่ต้องควบคุมการทำงานจากโปรแกรม CCS หรือในการนำไปใช้งานจริงโดยเมื่อเปิดแหล่งจากไฟระบบก็จะทำหน้าที่คัดลอกโค้ดโปรแกรมที่เก็บอยู่ในหน่วยความจำแบบแฟลชลงไปยังหน่วยความจำภายในขนาด 64 kbytes จากนั้นก็จะเริ่มทำงานตามโค้ดโปรแกรมที่ได้เขียนไว้

ในส่วนของการเขียนโปรแกรมลงในหน่วยความจำแบบแฟลชนั้นจะมีขั้นตอนการเขียนต่างหาก ซึ่งไม่สามารถที่จะเขียนโดยใช้การ Download จากโปรแกรม CCS ได้โดยตรง รายละเอียดการเขียนโปรแกรมแฟลชจะกล่าวในหัวข้อถัดไป



รูปที่ 5.4 การกำหนดขอบเขตของหน่วยความจำในส่วนของ MEMORY

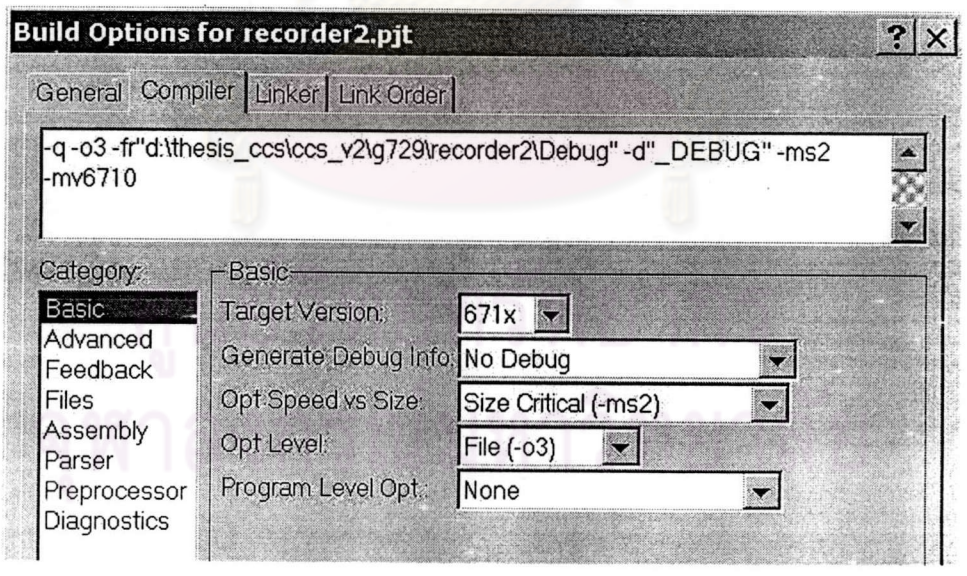
ส่วนที่ 2 SECTIONS จะทำหน้าที่เป็นตัวกำหนดรายละเอียดของโค้ดและตัวแปรต่างๆ ที่ได้จากการคอมไพล์ที่จะนำไปเก็บในส่วนของ MEMORY ซึ่งประกอบด้วย .vectors , .text , .bss , .cinit , .stack , .switch , .data , .sysmem , .const , .far , .cio



รูปที่ 5.5 การกำหนดขอบเขตของหน่วยความจำในส่วนของ SECTIONS

ในการใช้งานโดยทั่วไปจะกำหนด .vectors .text .cinit ให้อยู่ในส่วนของ IRAM และกำหนดให้ .bss อยู่ใน SDRAMPROG แต่ในวิทยานิพนธ์นี้เป็นการประมวลผลที่ต้องการความเร็วในการประมวลผลจึงต้องกำหนดให้ .text และ .bss อยู่ในหน่วยความจำภายในคือ IRAM และกำหนดให้ .cinit อยู่ในหน่วยความจำภายนอกคือ SDRAMPROG จึงจะสามารถประมวลผลได้ทันเวลาเพราะว่าในการประมวลผลนั้นโปรแกรมจะใช้พื้นที่ของ .bss ในการคำนวณผลดังนั้นถ้านำพื้นที่ของ .bss ไปไว้ในหน่วยความจำภายนอกจะทำให้การประมวลผลช้าและไม่สามารถประมวลผลได้แบบเวลาจริง

ในการประมวลผลแบบเวลาจริงนั้นจอกจากจะจัดสรรในส่วนของหน่วยความจำเพื่อเพิ่มความเร็วในการประมวลผลแล้ว ในโปรแกรม CCS จะมีตัวเลือกในการคอมไพล์เพื่อให้การประมวลผลมีความเร็วเพิ่มขึ้น (Optimization level) และตัวเลือกในการคอมไพล์ว่าจะต้องการความเร็วในการประมวลผลหรือต้องการขนาดของโค้ดมีขนาดเล็ก (Optimization speed vs size) โดยถ้าเลือกความเร็วก็จะทำให้จำนวนโค้ดของโปรแกรมมีขนาดใหญ่ขึ้นแต่ถ้าเลือกให้ขนาดโค้ดโปรแกรมมีขนาดเล็กลงก็จะทำให้การประมวลผลมีความเร็วที่ลดลงเช่นกัน ดังนั้นในการคอมไพล์จึงต้องเลือกให้เหมาะสมกับการใช้งาน โดยในวิทยานิพนธ์นี้เลือกการคอมไพล์สำหรับการทำงานแบบเวลาจริงดังแสดงในรูปที่ 5.6 มีรายละเอียดคือเลือก Opt speed vs size เป็น size critical (-ms2) และเลือก opt level เป็น File (-o3) และเลือกการทำงานแบบไม่มีการดีบัก (NO Debug) เพื่อให้การประมวลผลทำงานได้อย่างมีประสิทธิภาพ



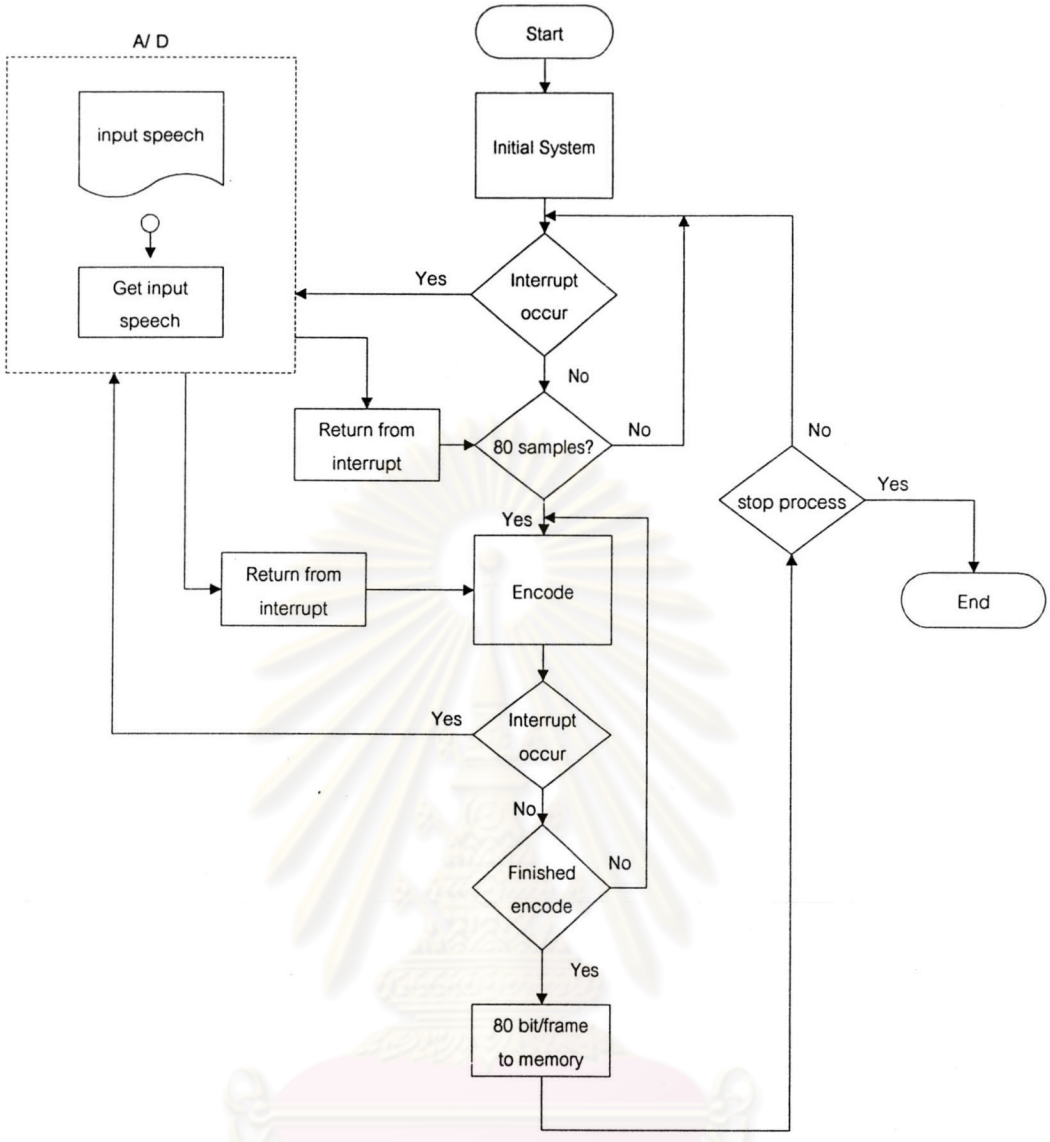
รูปที่ 5.6 การเลือกการคอมไพล์ในการทำงานแบบเวลาจริง

การรับสัญญาณเสียงเข้ามาผ่านทางวงจรแอนาลอกเป็นดิจิตอลจะอาศัยการเกิดอินเตอร์รัปต์อันเนื่องจากการรับข้อมูลผ่านทางพอร์ตอนุกรม โดยความถี่ที่ใช้ในการสุ่มเท่ากับ 8 kHz ซึ่งก็หมายถึงว่าจะเกิดการอินเตอร์รัปต์ในส่วนของการรับข้อมูล 8000 ครั้งต่อวินาทีและในส่วนของการส่งข้อมูลเสียงออกทางเอาต์พุตนั้นก็จะใช้การอินเตอร์รัปต์ที่เกิดจากการส่งข้อมูลผ่านทางพอร์ตอนุกรมซึ่งความถี่ที่ใช้ก็มีค่าเท่ากันกับการรับข้อมูลคือ 8 kHz ซึ่งก็หมายถึงว่าจะเกิดการอินเตอร์รัปต์ในส่วนของการส่งข้อมูล 8000 ครั้งต่อวินาทีเช่นกัน แต่ในการประมวลผลการเข้ารหัสแบบ CS-ACELP จะทำการประมวลผลเป็นเฟรมๆ ละ 80 ตัวอย่างสุ่มหรือ 10 มิลลิวินาที ดังนั้นในการประมวลผลเพื่อให้สามารถประมวลผลทันแบบเวลาจริงแสดงว่าต้องประมวลผลให้เสร็จภายใน 10 มิลลิวินาที

การประมวลผลแบบเวลาจริงในขั้นตอนนี้แบ่งออกเป็น 2 กรณีคือ

กรณีที่ 1 การประมวลผลที่ละส่วน ประกอบด้วยส่วนแรกคือการเข้ารหัส แสดงโฟล์วชาร์ตการทำงานดังรูปที่ 5.7 การทำงานของส่วนการเข้ารหัสสามารถอธิบายการทำงานโดยรวมได้ดังนี้

1. กำหนดค่าเริ่มต้นของตัวแปรต่างๆ วงจรการเชื่อมต่อสัญญาณแอนาลอก กำหนดการอินเตอร์รัปต์จากการรับข้อมูลผ่านทางพอร์ตอนุกรม
2. จากนั้นรอรับค่าสัญญาณเสียงผ่านทางวงจรการแปลงสัญญาณแอนาลอกเป็นวงจรดิจิตอล โดยใช้การอินเตอร์รัปต์ ให้ครบ 80 ตัวอย่างสุ่ม (10 มิลลิวินาที) จากนั้นจึงเริ่มทำการเข้ารหัส ในขณะทำการเข้ารหัสอยู่นั้นก็จะเกิดการอินเตอร์รัปต์ขึ้น จากนั้นตัวประมวลผลก็จะหยุดการทำงานของโปรแกรมหลักเพื่อไปรับสัญญาณอินพุตแล้วกลับมาทำงานที่โปรแกรมหลัก กระบวนการการเข้ารหัสก็จะดำเนินไปเรื่อยๆ จนกระทั่งการเข้ารหัสเสร็จสิ้น (การเข้ารหัสจะต้องเสร็จสิ้นภายในเวลา 10 มิลลิวินาที)
3. เมื่อเข้ารหัสก็จะได้เอาต์พุตจำนวน 80 บิตต่อเฟรม จากนั้นนำเอาต์พุตที่ได้ไปจัดเก็บในหน่วยความจำตามตำแหน่งที่ต้องการ
4. จากนั้นเริ่มกลับไปทำงานของการเข้ารหัสในเฟรมต่อไป การเข้ารหัสจะดำเนินไปเรื่อยๆ จนกว่าจะพบเงื่อนไขในการหยุด



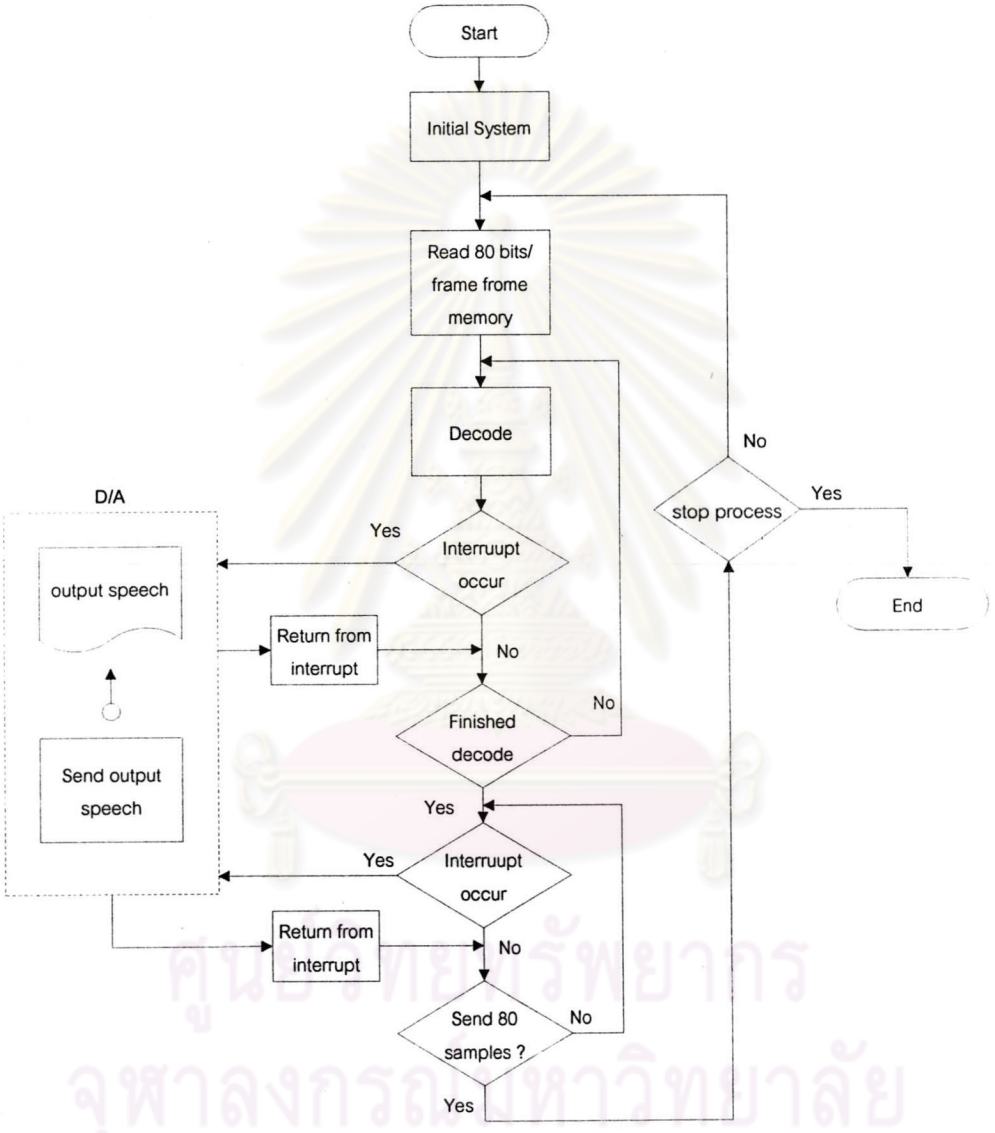
รูปที่ 5.7 โฟล์วชาร์ตการทำงานการเข้ารหัสแบบเวลาจริง

ส่วนที่สองคือการถอดรหัส หลังจากที่ได้ทำการเข้ารหัสและเก็บค่าพารามิเตอร์ที่ได้จากการเข้ารหัสไว้ในหน่วยความจำแล้วในขั้นตอนนี้ก็จะทำการถอดรหัสจากค่าพารามิเตอร์ที่ได้เก็บไว้ ดังแสดงโฟล์วชาร์ตการทำงานดังรูปที่ 5.8

การทำงานของส่วนการถอดรหัสสามารถอธิบายการทำงานโดยรวมได้ดังนี้

1. กำหนดค่าเริ่มต้นของตัวแปรต่างๆ วงจรการเชื่อมต่อแอนาลอก กำหนดการอินเตอร์รัปต์อันเกิดจากการส่งข้อมูลออกทางพอร์ตอนุกรม

2. อ่านค่าพารามิเตอร์ที่ได้เข้ารหัสไว้แล้วทีละเฟรม (80 บิต) เข้ามายังบัฟเฟอร์ จากนั้นทำการถอดรหัสเสียงจากพารามิเตอร์ที่ได้มา ในขณะเดียวกันก็จะส่งสัญญาณเอาต์พุตเสียงผ่านการอินเตอร์รัปต์ไปด้วย โดยในเฟรมแรกของการถอดรหัสนั้นจะส่งสัญญาณเอาต์พุตเป็นค่าศูนย์ออกไปก่อนเพราะว่าในเฟรมแรกที่เริ่มการถอดรหัสนั้นจะยังไม่มีสัญญาณเอาต์พุตที่จะส่งออกไป แต่หลังจากเฟรมแรกก็จะส่งสัญญาณเสียงที่ได้จากการถอดรหัสออกไป



รูปที่ 5.8 โฟล์วชาร์ตการทำงานการถอดรหัสแบบเวลาจริง

3. กระบวนการถอดรหัสจะต้องเสร็จสิ้นภายในเวลาไม่เกิน 10 มิลลิวินาที หลังจากกระบวนการถอดรหัสเสร็จสิ้นแล้วก็จะได้สัญญาณเอาต์พุตออกมา 80 ตัวอย่างสุ่ม

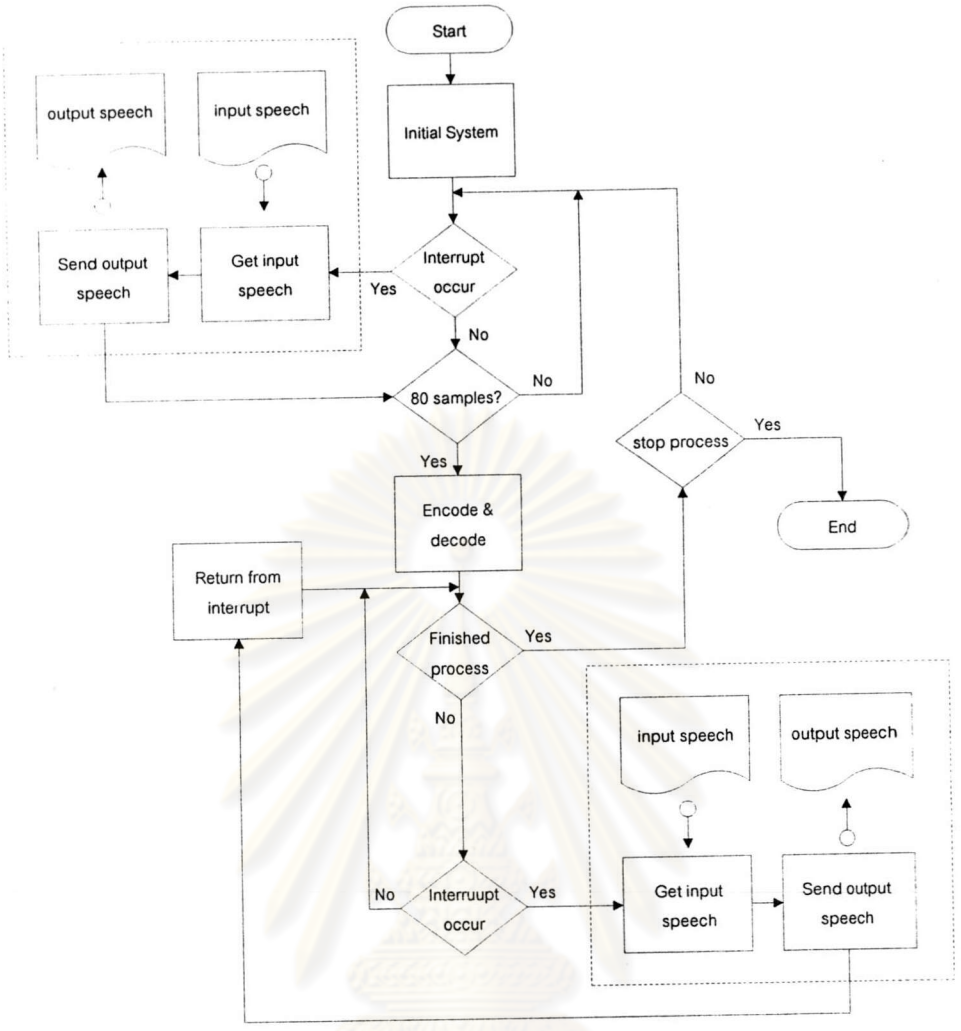
จากนั้นจะมีการตรวจสอบว่าได้ส่งสัญญาณเอาต์พุตออกไปครบ 80 ตัวอย่างสุ่มหรือยัง ถ้ายังก็จะรอส่งสัญญาณเอาต์พุตให้ครบ 80 ตัวอย่างสุ่มก่อนจึงจะเริ่มทำการถอดรหัสของเฟรมถัดไป

4. กระบวนการถอดรหัสก็จะดำเนินการไปเรื่อยๆ จนกว่าจะครบจำนวนเฟรมที่ต้องการถอดรหัสหรือพบเงื่อนไขในการหยุดซ

กรณีที่ 2 การเข้ารหัสถอดรหัสแบบเวลาจริง คือการรับข้อมูลเข้ามา 80 ตัวอย่างสุ่ม แล้วทำการเข้ารหัสให้เสร็จ จากนั้นทำการถอดรหัสแล้วส่งข้อมูลออกทางเอาต์พุตให้เสร็จภายในเวลา 10 มิลลิวินาที โฟล์วชาร์ตการทำงานแสดงดังรูปที่ 5.9

การทำงานของการเข้ารหัสและถอดรหัสแบบเวลาจริง เป็นการนำเอาโปรแกรมการเข้ารหัสและโปแกรมการถอดรหัสมารวมกันเพื่อให้สามารถทำงานต่อเนื่องกัน สามารถอธิบายการทำงานโดยรวมได้ดังนี้

1. เมื่อเริ่มรันโปรแกรม จะเริ่มทำการกำหนดค่าเริ่มต้นของระบบรวมทั้งการบริการอินเตอร์รัปต์ทั้งในส่วนของการอินเตอร์รัปต์ที่เกิดจากการรับข้อมูลและอินเตอร์รัปต์ที่เกิดจากการส่งข้อมูลผ่านทางพอร์ตอนุกรม
2. จากนั้นรอรับสัญญาณอินพุตผ่านทางการอินเตอร์รัปต์เพื่อจะรับสัญญาณอินพุตเข้ามาประมวลผล และในขณะเดียวกันนั้นก็จะส่งสัญญาณเอาต์พุตออกไปด้วยในขณะเดียวกัน
3. เมื่อรับสัญญาณอินพุตได้ครบจำนวน 1 เฟรม หรือ 80 ตัวอย่างสุ่มแล้วก็จะทำการเข้ารหัสจนเสร็จสิ้นจากนั้นก็ทำการถอดรหัสจากค่าพารามิเตอร์ที่ได้จากการเข้ารหัสอย่างต่อเนื่อง (ในขณะที่โปรแกรมหลักของการเข้ารหัสหรือการถอดรหัสทำงานอยู่นั้น กระบวนการรับสัญญาณอินพุตและการส่งสัญญาณเอาต์พุตก็จะดำเนินการไปเรื่อยๆ)
4. จากนั้นเมื่อทำการถอดรหัสเสร็จสิ้นแล้วก็จะมีการตรวจสอบว่าส่งข้อมูลออกทางเอาต์พุตครบ 80 ตัวอย่างสุ่มหรือยังถ้าครบก็จะเริ่มกลับไปทำงานในเฟรมต่อไป (กระบวนการการเข้ารหัสและถอดรหัสนั้นต้องประมวลผลให้ทันภายในเวลา 1 เฟรม หรือ 10 มิลลิวินาที จึงจะถือได้ว่าสามารถทำงานได้ในแบบเวลาจริง)



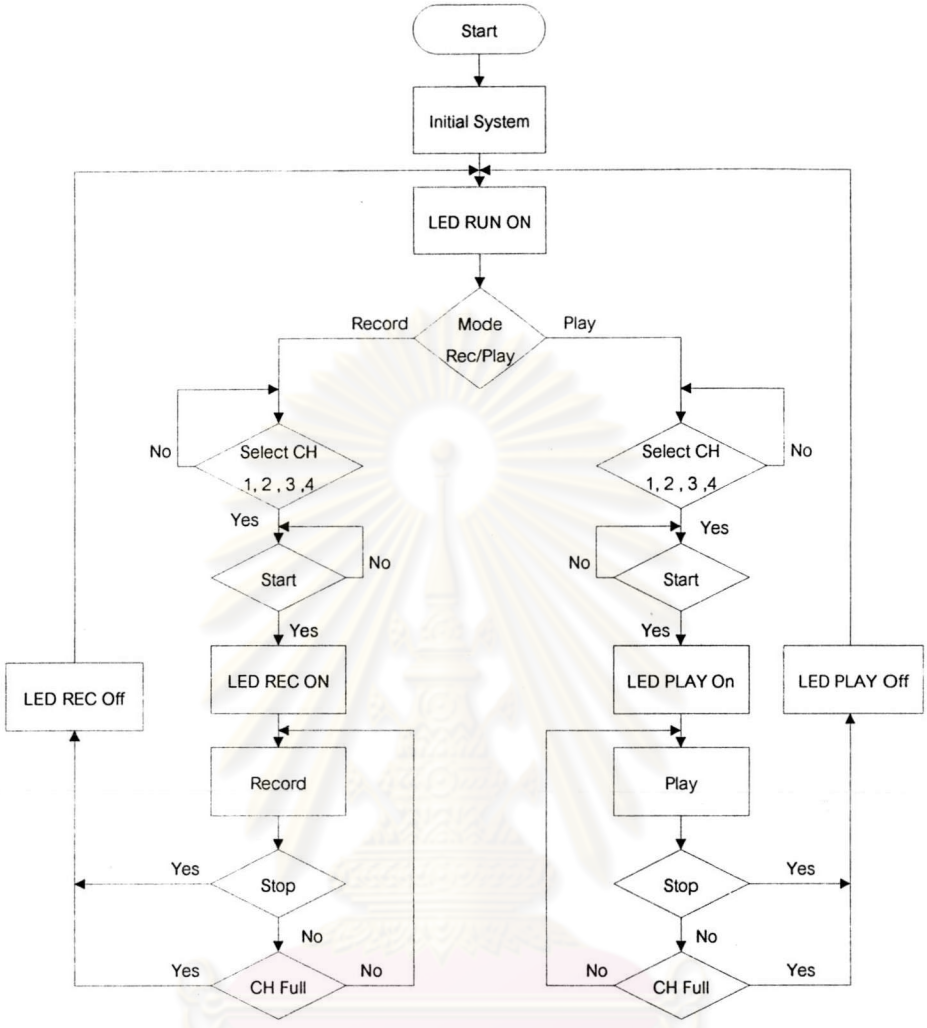
รูปที่ 5.9 โฟล์วชาร์ตการทำงานของการเข้ารหัสถอดรหัสแบบเวลาจริง

5.2 การสร้างเครื่องบันทึกเสียงพูดดิจิตอล

5.2.1 โครงสร้างทางซอฟต์แวร์

หลังจากที่ได้พัฒนาโปรแกรมการเข้ารหัสและถอดรหัสแบบเวลาจริงได้แล้วในขั้นตอนนี้ก็จะเป็นการสร้างเครื่องบันทึกเสียงโดยใช้โปรแกรมการเข้ารหัสและถอดรหัสที่ได้มาปรับปรุงเพิ่มเติมโปรแกรมในส่วนของการควบคุมการทำงานและการแสดงผลโดยโปรแกรมของเครื่องบันทึกเสียงจะต้องประกอบไปด้วยโปรแกรมการเข้ารหัสสำหรับใช้ในขั้นตอนการบันทึกและโปรแกรมการถอดรหัสใช้สำหรับการเล่นเพื่อฟังเสียงที่ได้ทำการบันทึกไว้ โฟล์วชาร์ตการทำงานแสดงดัง

รูปที่ 5.10 และบล็อกไดอะแกรมการทำงานของเครื่องบันทึกเสียงพูดดิจิตอลแสดงดังรูปที่ 5.11 และสามารถอธิบายการทำงานของโปรแกรมได้ดังนี้



รูปที่ 5.10 โฟล์วชาร์ตการทำงานของเครื่องบันทึกเสียงพูดดิจิตอล

1. เมื่อเริ่มรันโปรแกรม จะทำการเริ่มกำหนดค่าเริ่มต้นของระบบ
2. กำหนดให้ LED Run แสดงสถานะติดในโหมดของการรันโปรแกรม
3. รอให้เลือกโหมดการทำงานว่าจะเลือกโหมดบันทึกหรือโหมดของการเล่นเสียง
4. กรณีที่เลือกโหมดการทำงานเป็นแบบบันทึก การทำงานก็จะเข้าสู่โหมดของการบันทึก และรอให้กดเลือกช่องที่จะทำการบันทึกซึ่งมีทั้งหมด จำนวน 4 ช่องคือ 1 2 3 4
5. รอให้กดปุ่มเริ่มต้นการทำงาน LED Record จะแสดงสถานะติดและ LED Run ก็จะดับ หลังจากนั้นโปรแกรมก็เริ่มการทำงานในโหมดของการบันทึกโดยการรับ

สัญญาณเสียงจากช่องทางอินพุตโดยหลังจากที่การเข้ารหัสเสร็จแต่ละเฟรม จะนำข้อมูลที่ได้จำนวน 80 บิต จากการเข้ารหัสไปเก็บในตำแหน่งของแต่ละช่องที่ได้ทำการเลือก

6. กระบวนการบันทึกข้อมูลจะดำเนินการไปเรื่อยๆ และในขณะเดียวกันก็จะมีการตรวจสอบว่าจำนวนข้อมูลที่บันทึกนั้นมีขนาดเกินที่กำหนดหรือไม่ (ถ้าเกินก็จะหยุดการบันทึกและกำหนดให้ LED Record ก็จะดับและให้ LED Run ติดแทนและกลับไปเริ่มต้นที่การเลือกโหมดการทำงานอีกครั้ง) ถ้ายังไม่เกินก็จะทำการตรวจสอบว่ามีการกดปุ่มหยุดหรือไม่ (ถ้ามีก็จะหยุดการบันทึกและกำหนดให้ LED Record ดับและให้ LED RUN ติดแทนและกลับไปเริ่มต้นที่การเลือกโหมดการทำงานอีกครั้ง) ถ้ายังไม่กดก็จะทำการบันทึกเสียงไปเรื่อยๆ จนกว่าจะครบจำนวนเวลาคือ 1 ชั่วโมง
7. ส่วนในการทำงานของโหมดการเล่นก็จะมีการทำงานที่ลักษณะคล้ายกันคือ เมื่อเลือกการทำงานเป็นโหมดเล่นเสียง จากนั้นก็จะรอให้เลือกว่าจะทำการเล่นเสียงที่ช่องไหนโดยเลือกหมายเลขช่องที่ต้องการฟังเสียง
8. รอให้กดปุ่มเริ่มเล่นเสียง จากนั้น LED Play ก็จะติดและ LED Run ก็จะดับ จากนั้นโปรแกรมจะไปอ่านข้อมูลที่ได้ทำการเข้ารหัสและเก็บบันทึกไว้ในหน่วยความจำตรงตำแหน่งของแต่ละช่องออกมาที่ละ 80 บิต และทำการถอดรหัสเสียงพูดและส่งข้อมูลออกทางเอาต์พุต
9. กระบวนการเล่นเสียงจะดำเนินการไปเรื่อยๆ และในขณะเดียวกันก็จะมีการตรวจสอบว่าได้ทำการเล่นเสียงถึงตรงตำแหน่งที่กำหนดในแต่ละช่องทีได้ทำการเลือกหรือยัง (ถ้าถึงก็จะหยุดการบันทึกและกำหนดให้ LED Play ดับและให้ LED Run ติดแทนและกลับไปเริ่มต้นที่การเลือกโหมดการทำงานอีกครั้ง) ถ้ายังไม่ถึงก็จะทำการตรวจสอบว่ามีการกดปุ่มหยุดหรือไม่ (ถ้ามีก็จะหยุดการเล่นเสียงและกำหนดให้ LED Play ดับและให้ LED Run ติดแทนและกับไปเริ่มต้นที่การเลือกโหมดการทำงานอีกครั้ง) ถ้ายังไม่กดก็จะทำการเล่นเสียงไปเรื่อยๆ จนกว่าจะครบจำนวนเวลาคือ 1 ชั่วโมง

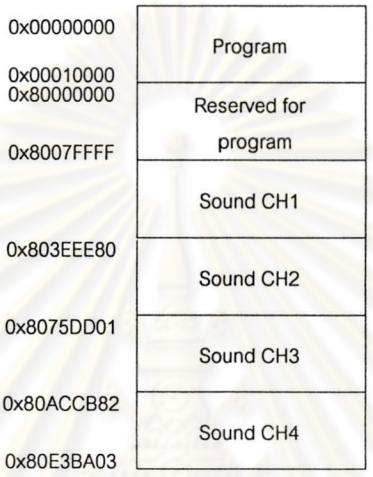
5.2.2 โครงสร้างทางฮาร์ดแวร์

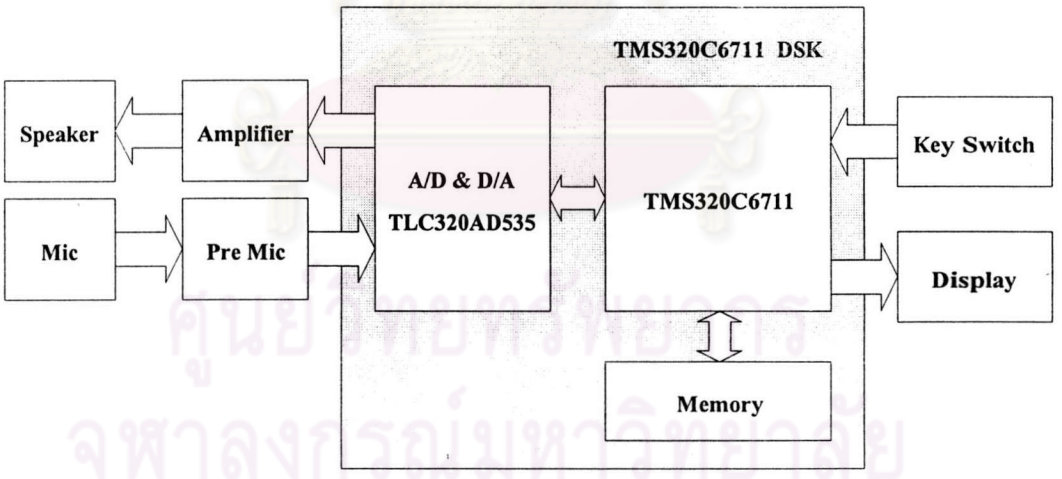
ส่วนประกอบทางด้านฮาร์ดแวร์ของเครื่องบันทึกเสียงพูดดิจิตอลแสดงดังรูปที่ 5.11 จะประกอบด้วยส่วนต่างๆ ดังนี้

1. ตัวประมวลผลสัญญาณดิจิตอลใช้เบอร์ TMS320C6711
2. หน่วยความจำประกอบด้วยสองส่วนคือหน่วยความจำสำหรับเก็บโปรแกรมและหน่วยความจำสำหรับเก็บข้อมูลเสียงที่ได้จากการบันทึกเสียง โดยในส่วนของหน่วย

ความจำที่ใช้ในการเก็บข้อมูลเสียงนั้นมี 4 ช่อง คือ ช่อง 1 ช่อง 2 ช่อง 3 ช่อง 4 แสดงดังรายละเอียดดังตารางที่ 5.1 โดยหน่วยความจำของระบบประกอบด้วยหน่วยความจำภายใน 64 Kbytes ใช้สำหรับเก็บโปรแกรมที่ใช้ในการทำงานและหน่วยความจำภายนอกแบบ SDRAM ขนาด 16 Mbytes นั้นใช้สำหรับเก็บโปรแกรมบางส่วนและที่เหลือจะใช้สำหรับเก็บข้อมูลเสียงที่ได้ทำการบันทึก

ตารางที่ 5.1 การกำหนดหน่วยความจำของเครื่องบันทึกเสียงพูดดิจิตอล

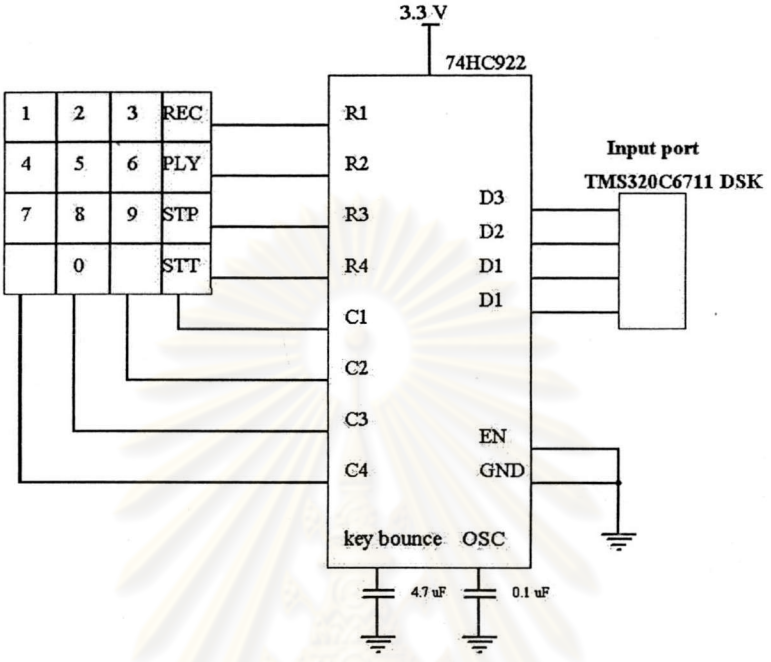




รูปที่ 5.11 บล็อกไดอะแกรมของเครื่องบันทึกเสียงพูดดิจิตอล

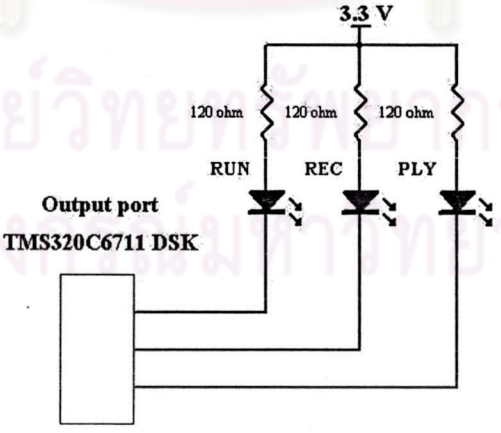
3. คีย์สวิตซ์สำหรับควบคุมการทำงานของเครื่องบันทึกเสียง คีย์สวิตซ์ที่ใช้จะเป็นแบบเมตริกซ์ หรือคีย์แพดขนาด 4x4 และใช้ชิปเข้ารหัส (Encode) เบอร์ 74HC922

เพื่อลดขนาดของจำนวนคีย์สวิตช์เนื่องจากจำนวนของอินพุตที่ใช้งานมีเพียง 4 ช่อง และต่อสัญญาณเอาต์พุตจาก 74HC922 เข้ากับส่วนของอินพุตพอร์ตของ TMS320C6711 DSK แสดงวงจรดังรูปที่ 5.12



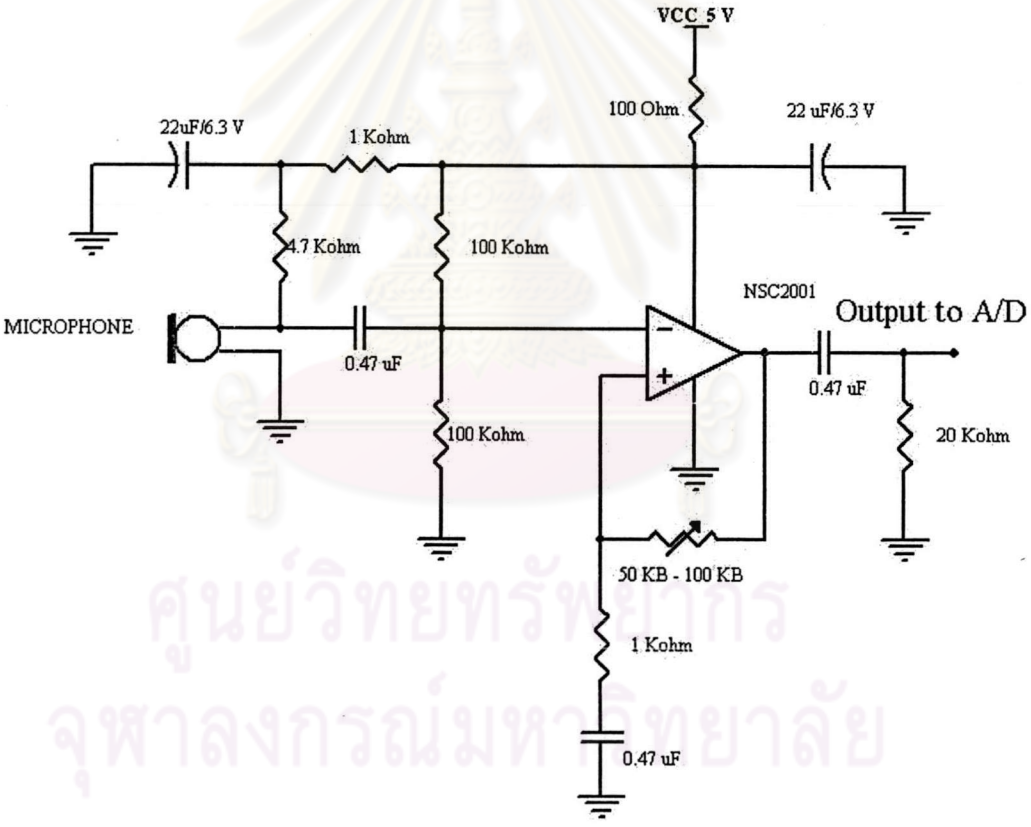
รูปที่ 5.12 วงจรสวิตช์ควบคุมการทำงาน

4. ส่วนของการแสดงผลการทำงานใช้ แอลอีดี 3 ดวงเพื่อแสดงสถานะการรันโปรแกรม (RUN) การบันทึก (REC) การเล่นเสียง (PLY) แสดงวงจรดังรูปที่ 5.13



รูปที่ 5.13 วงจรแอลอีดีแสดงสถานะการทำงาน

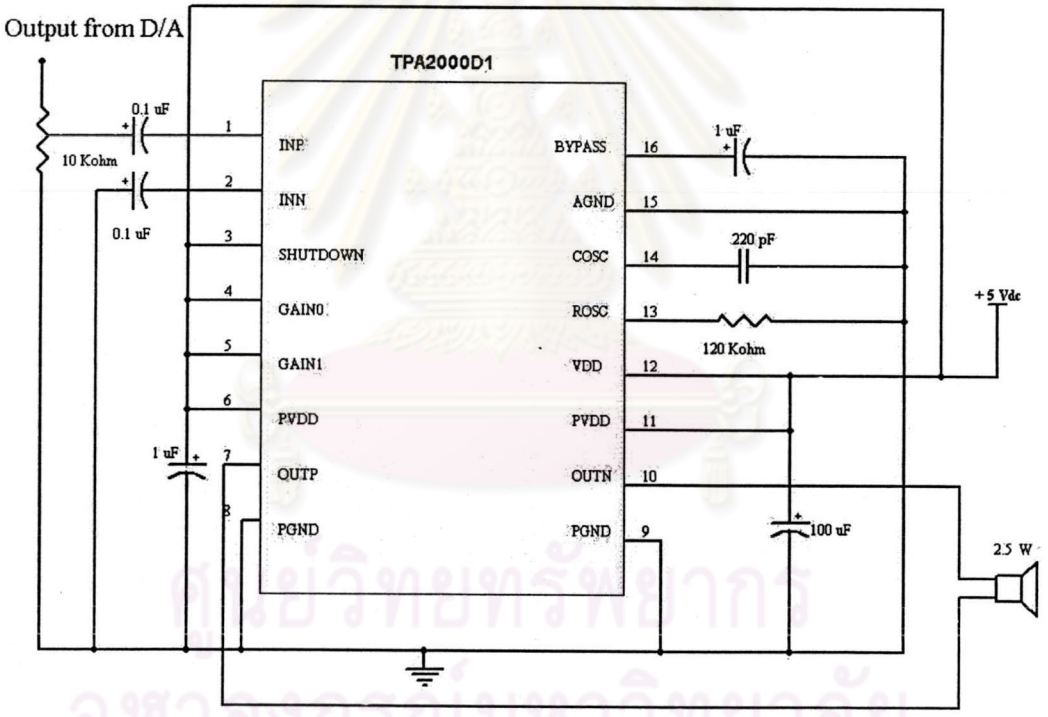
5. วงจรการเชื่อมต่อสัญญาณแอนาลอกใช้ชิปเบอร์ TLC320AD535
6. ส่วนของสัญญาณอินพุตประกอบด้วยสองส่วนคือไมโครโฟนสำหรับรับสัญญาณเสียง และวงจรปรีไมค์สำหรับขยายสัญญาณเสียง แสดงวงจรดังรูปที่ 5.14 ในการนำไปใช้งานจริงๆ โดยการรับสัญญาณจริงจากไมค์โครโฟนนั้นสัญญาณจะมีขนาดที่เบาจึงต้องมีการขยายสัญญาณให้เพียงพอก่อนที่จะส่งเข้าสู่วงจรแปลงสัญญาณแอนาลอกเป็นดิจิตอลต่อไป คุณสมบัติของวงจรขยายสัญญาณเอาต์พุตมีดังนี้
    - ใช้ไมโครโฟนแบบคอนเดนเซอร์
    - ใช้ออปแอมเบอร์ NCS2001
    - สามารถใช้แหล่งจ่ายได้ตั้งแต่ 0.9-6 Vdc
    - วงจรขยายแบบ Non Inverting Amplifier



รูปที่ 5.14 วงจรขยายสัญญาณไมโครโฟน

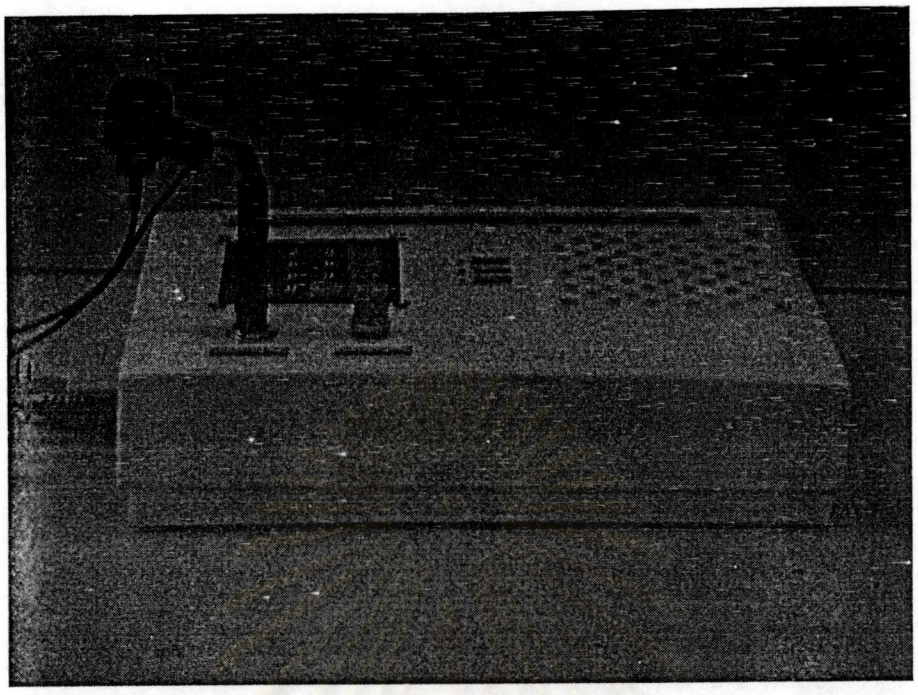
7. ส่วนของสัญญาณเอาต์พุตประกอบด้วยสองส่วนคือลำโพงและส่วนของวงจรขยายสัญญาณ ดังแสดงวงจรดังรูปที่ 5.15 หลังจากที่สัญญาณผ่านวงจรการแปลงดิจิตอลเป็นแอนาลอกแล้วต้องผ่านวงจรขยายสัญญาณเพื่อให้มีขนาดแรงพอที่จะขับลำโพง คุณสมบัติของวงจรขยายมีดังนี้

- ใช้ไอซีเบอร์ TPA2000D1
- สามารใช้กับแหล่งจ่ายไฟ ตั้งแต่ 3-5.5 V
- THD 0.15% ที่ 1.5 W
- อินพุตอิมพีแดนซ์ 24 Kohms
- อัตราการขยาย 23.5 dB
- กำลังขยาย 2 W
- ลำโพง ขนาด 2.5 W

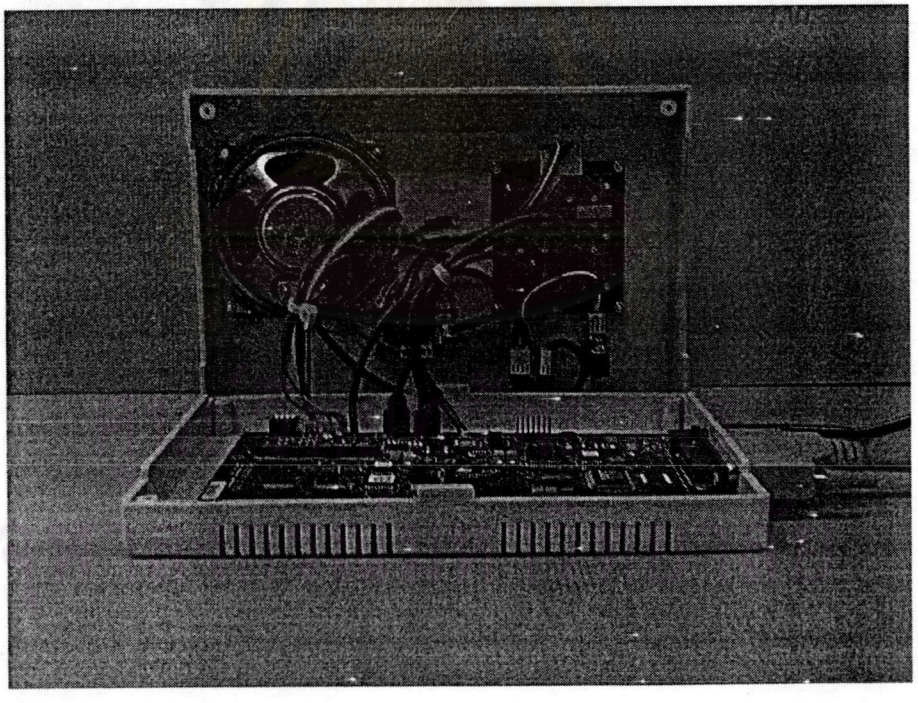


รูปที่ 5.15 วงจรขยายสัญญาณเสียงสำหรับขับลำโพง

นำส่วนประกอบต่างๆ มาสร้างเป็นเครื่องบันทึกเสียงพูดดิจิตอลต้นแบบเพื่อนำไปทดสอบการทำงานและใช้งาน รูปเครื่องบันทึกเสียงพูดดิจิตอลต้นแบบแสดงได้ดังรูปที่ 5.16 และรูปที่ 5.17



รูปที่ 5.16 เครื่องบันทึกเสียงพูดดิจิตอลต้นแบบ



รูปที่ 5.17 ลักษณะภายในของเครื่องบันทึกเสียงพูดต้นแบบ

5.2.3 การโปรแกรมหน่วยความจำแบบแฟลช

ในการใช้งานจริงๆ นั้นการเก็บโปรแกรมที่ทำงานจะต้องเก็บไว้ในหน่วยความจำถาวรแบบรอมหรือแฟลชรอม เพื่อให้สามารถทำงานได้เมื่อเวลาเปิดเครื่องใช้งาน ดังนั้นหลังจากที่ได้พัฒนาโปรแกรมการทำงานจนเสร็จเรียบร้อยแล้วจะต้องทำการเขียนโค้ดโปรแกรมไปเก็บไว้ในหน่วยความจำถาวร โดยมีขั้นตอนดังต่อไปนี้

1. หลังจากที่ได้พัฒนาโปรแกรมบนโปรแกรม CCS จนสามารถทำงานได้ถูกต้องแล้วก็ทำการคอมไพล์จะได้ไฟล์ *.out (เป็นไฟล์ที่ใช้สำหรับเขียนไปยังตัวดีเอสพีเพื่อให้ทำงานตามที่ต้องการ) โดยมีข้อกำหนดว่าในไฟล์ dskc6711.cmd (เป็นไฟล์ที่กำหนดตำแหน่งการใช้งานของหน่วยความจำ) นั้นจะต้องกำหนดไฟล์นามสกุล .text .switch .cinit จะต้องกำหนดตำแหน่งหน่วยความจำภายใน 0x00010000 เท่านั้น ในส่วนของไฟล์นามสกุลอื่นจะกำหนดลงตำแหน่งไหนก็ได้

2. จากนั้นทำการเปลี่ยนไฟล์ *.out เป็นไฟล์ *.hex เพื่อใช้ในการเขียนไปยังหน่วยความจำแบบแฟลชรอม โดยเริ่มสร้างไฟล์ hex.cmd ดังนี้

```
*.out    /* ชื่อไฟล์สำหรับที่ต้องการจะแปลงเป็น *.hex */
-  a
-  image
-  zero
-  memwidth  8ROMS
{
    FLASH :org = 0 ; len = 0x20000 , romwidth = 8, files = {*.hex}
}
```

3. จากนั้นทำการคอมไพล์จากไฟล์ *.out ไปเป็นไฟล์ *.hex โดยใช้โปรแกรม hex6x.exe ที่ทำงานบน Dos ในการแปลงดังนี้

```
C:\...\  hex6x  hex.cmd
```

หลังจากรันโปรแกรม hex6x.exe โดยไม่มีข้อผิดพลาดก็จะได้ไฟล์โปรเจคท์ที่จะต้องเขียนไปยังหน่วยความจำแบบแฟลชรอม (*.hex)

4. จากนั้นเริ่มทำการเขียนโปรแกรมไปยังหน่วยความจำแบบแฟลชโดยใช้โปรแกรม flash.exe ในขั้นตอนนี้จะต้องต่อบอร์ด TMS320C6711 DSK เข้ากับพอร์ตขนานของคอมพิวเตอร์ด้วย จากนั้นเปิดแหล่งจ่ายไฟและทำการโปรแกรมดังนี้

C:\...\ flash filename -s0x90000000

ถ้าคอมติดต่อกับบอร์ดได้ก็จะทำการเขียนข้อมูลไปยังหน่วยความจำแบบแฟลชโดยแสดงข้อความดังนี้ Programming the flash..... จนกว่าจะโปรแกรมเสร็จถ้าไม่มีข้อผิดพลาดใดๆ เกิดขึ้นโปรแกรมก็จะแสดงข้อความ Flash successfully programmed เป็นอันเสร็จสำหรับขั้นตอนการโปรแกรมหน่วยความจำแบบแฟลชรอม จากนั้นก็ทำการรีเซตบอร์ดโดยการปิดแหล่งจ่ายแล้ว และก็เปิดแหล่งจ่ายไฟให้กับบอร์ด ก็จะเริ่มทำงานตามโปรแกรมที่ได้เขียนไป

5.2.4 ฟังก์ชันการทำงาน

ฟังก์ชันการทำงานของเครื่องบันทึกเสียงจะควบคุมผ่านทางคีย์สวิตซ์ทางด้านหน้าของเครื่องแสดงดังรูปที่ 5.18

รูปที่ 5.18 ด้านหน้าเครื่องบันทึกเสียงพูดดิจิตอลต้นแบบ

โหมดการทำงานของเครื่องบันทึกเสียงพูดดิจิตอลประกอบด้วยโหมดการทำงาน 2 โหมดคือ

1. โหมดบันทึก (Record) การทำงานในโหมดบันทึกจะเริ่มจากเมื่อเริ่มรันโปรแกรมสังเกตุที่แอลอีดีตำแหน่ง RUN จะติดจากนั้นให้กดปุ่มบันทึก REC จากนั้นให้เลือกหมาย

เลขที่จะทำการบันทึก จำนวนช่องที่สามารถบันทึกได้มีทั้งหมด 4 ช่องคือคีย์ 1 , 2 , 3 , 4 หลังจากที่เลือกหมายเลขช่องที่จะทำการบันทึกเสร็จแล้วให้กดปุ่ม STT เพื่อเริ่มการบันทึกข้อมูล (LED ที่ตำแหน่งบันทึกจะติดแทน) การบันทึกข้อมูลจะบันทึกไปเรื่อยๆ จนครบเวลา 1 ชั่วโมงหรือจนกว่าจะมีการกดปุ่ม STP ถ้ามีการกดคีย์ STP ก็จะหยุดบันทึก (LED ที่ตำแหน่ง RUN จะติดแทน) หลังจากนั้นก็สามารถบันทึกช่องอื่นๆ ที่เหลือได้โดยมีขั้นตอนเหมือนกันทุกอย่าง

2. โหมดเล่นเสียง (Play) การทำงานในโหมดของการเล่นเสียงก็จะมีขั้นตอนคล้ายๆ กับการบันทึกเสียงคือ เริ่มจากการกดคีย์ PLY จากนั้นให้เลือกหมายเลขที่จะทำการเล่นเสียงซึ่งก็มี 4 ช่องเช่นกัน คือ 1 , 2 , 3 , 4 หลังจากที่เลือกหมายเลขเสร็จแล้วให้ทำการกดคีย์ STT การเล่นก็จะไปเรื่อยๆ จนกว่าจะครบ 1 ชั่วโมง หรือจนกว่าจะมีการกดปุ่ม STP ถ้ามีการกดปุ่ม STOP ก็จะหยุดเล่น (LED ที่ตำแหน่ง RUN จะติดแทน)