สำนักหอสมุดกลาง_พระจอมเกล้าลาดกระบัง

การเข้ารหัสสัญญาณเสียงแบบ Real-time สำหรับมาตรฐาน ITU-T G.729 บน DSP chip TMS320C6713

Real-time Implementation of ITU-T G.729 Standard Voice Codec on DSP chip TMS320C6713

ธนพล ตั้งชูพงศ์
ธนสิทธิ์ ชวลิตธารา

2547
เลขหมู่.....
เลขทะเบียน.....61521
วัน,เดือน,ปี.....18 ก.ค. 2549
b.....11594707
i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเข้ารหัสสัญญาณเสียงแบบ Real-time สำหรับมาตรฐาน ITU-T G.729 บน DSP chip TMS320C6713

Real-time Implementation of ITU-T G.729 Standard Voice Codec on DSP chip TMS320C6713

โดย

ธนพล ตั้งชูพงศ์

ธนสิทธิ์ ชวลิตธารา

อาจารย์ที่ปรึกษา

ผศ.ดร.อรฉัตร จิตต์โสภักตร์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2547

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การเข้ารหัสสัญญาณเสียงแบบ Real-time สำหรับมาตรฐาน ITU-T G.729 บน DSP chip TMS320C6713

Real-time Implementation of ITU-T G.729 Standard Voice Codec on DSP chip TMS320C6713

ผู้จัดทำ

1. นายธนพล ตั้งชูพงศ์ รหัสประจำตัว 44010191
2. นายธนสิทธิ์ ชวลิตธารา รหัสประจำตัว 44010199

อรณัตร จิตต์โสภักตร์ อาจารย์ที่ปรึกษา

(ผศ.ดร.อรณัตร จิตต์โสภักตร์)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเข้ารหัสสัญญาณเสียงแบบ Real-time สำหรับมาตรฐาน ITU-T G.729 บน DSP chip TMS320C6713

นายธนพล ตั้งชูพงศ์ รหัสประจำตัว 44010191
นายธนสิทธิ์ ชวลิตธารา รหัสประจำตัว 44010199
ผศ.ดร.อรฉัตร จิตต์โสภักตร์ อาจารย์ที่ปรึกษา
ปีการศึกษา 2547

บทคัดย่อ

โครงงานนี้ได้ทำการพัฒนาการเข้ารหัสสัญญาณเสียงแบบ Real-time สำหรับมาตรฐาน ITU-T G.729 บน DSP chip TMS320C6713 (Real-time Implementation of ITU-T G.729 Standard Voice Codec on DSP chip TMS320C6713) โดยทำการพัฒนาไลบรารี สำหรับเข้ารหัสและถอดรหัสสัญญาณเสียงตามมาตรฐาน ITU-T G.729 บน DSP chip ตระกูล TMS320C6000 ซึ่งงานวิจัยนี้สามารถนำไปประยุกต์ใช้กับโครงการ Voice Over IP สำหรับการส่งสัญญาณเสียงผ่านเครือข่ายอินเตอร์เน็ต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

REAL-TIME IMPLEMENTATION OF ITU-T G.729 STANDARD VOICE CODEC ON DSP CHIP TMS320C6713

Mr.Thanaphon Tangchoopong
Mr.Thanasit Chawalittara
Asst Prof. Dr. Orachat Chitsobhuk Advisor

ABSTRACT

This thesis presents a development of an encoding and decoding speech signal as specified in ITU-T G.729 standard. It is useful for Voice over IP project since its capacity can improve the performance of communication systems like the internet. However, we try to search for algorithm which can decrease the complexity of the system while enhance this standard. We have implemented the proposed technique on DSP chip TMS320C6713

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้คงไม่สามารถเสร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และความร่วมมือจากหลาย ๆ ฝ่ายด้วยกัน บุคคลแรกที่ต้องกล่าวถึงเพราะมีส่วนสำคัญที่ทำให้วิทยานิพนธ์นี้สำเร็จลงได้ก็คือ อาจารย์ อรฉัตร์ จิตต์โสภักตร์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ และพี่ ๆ ที่ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ ที่ให้ความเอาใจใส่ แนะนำ และช่วยเหลือเสมอมา ซึ่งต้องขอบพระคุณเป็นอย่างยิ่ง

ท้ายที่สุดต้องขอขอบพระคุณบุคคลสำคัญที่สุดที่ทำให้ข้าพเจ้ามีวันนี้ ก็คือ บิดา มารดา อันเป็นที่เคารพรักยิ่ง ที่ให้กำลังใจและให้โอกาสในการศึกษาอย่างเต็มที่ ข้าพเจ้าขอระลึกพระคุณของท่านทั้งสอง และขอกราบขอบพระคุณมา ณ. ที่นี้

ธนพล ตั้งชูพงศ์

ธนสิทธิ์ ชวลิตธารา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

บทคัดย่อ.....I
ABSTRACT.....II
กิตติกรรมประกาศ.....III
สารบัญ.....IV
สารบัญรูปภาพ.....VI
สารบัญตาราง.....IX
บทที่ 1 บทนำ.....1
1.1 หลักการและเหตุผล.....1
1.2 วัตถุประสงค์.....1
1.3 ผลที่คาดว่าจะได้รับ.....1
1.4 ขอบเขตของการพัฒนา.....2
1.5 ขั้นตอนการดำเนินงาน.....2
บทที่ 2 ทฤษฎีการสังเคราะห์เสียง.....5
2.1 บทนำ.....5
2.2 การสังเคราะห์เสียงเชิงกายภาพ (Physical Model).....5
2.3 การสังเคราะห์เสียงเชิงคณิตศาสตร์ (Mathematical Model).....6
2.4 Code Excited Linear Prediction Model (CELP).....10
บทที่ 3 มาตรฐานการเข้ารหัสสัญญาณ G.729.....12
3.1 สัมประสิทธิ์ของวงจรกรอง (Filter coefficient).....12
3.2 สัญญาณกระตุ้น (Excitation).....22
3.3 การถอดรหัส.....38
3.4 การตรวจสอบว่ามีการสนทนาหรือไม่ (Voiced Acti..ity Detection (VAD)).....39
บทที่ 4 การปรับปรุงความเร็วในการเข้ารหัส (OPTIMIZATION).....41
4.1 แนวทางในการพัฒนา.....41
4.2 การปรับปรุงความเร็วโดยการแก้ไขที่มาตรฐานการเข้ารหัส.....41
4.3 การปรับปรุงความเร็วโดยการแก้ไขที่เครื่องมือ (การจัดการหน่วยความจำ & DSP BIOS Tool).....48
บทที่ 5 การออกแบบและสร้างไลบรารี.....51
5.1 อัลกอริทึมแบบมัลติแชนเนิล.....51
5.2 Abstract algorithm interface.....62

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.3 ลิงก์เกอร์คอมมานด์ (Linker Command) 72
บทที่ 6 การทดลอง 76
6.1 การทดลองตรวจสอบความถูกต้องของ ชุดเข้า/ถอดรหัส G.729 ที่โปรแกรมลงบน ดี เอส พี บอร์ด 76
6.2 การทดลองปรับปรุงความเร็วของชุดเข้ารหัสและถอดรหัส G.729 โดยอาศัย Build ออพชันของ Code Composer Studio V2.0 78
6.3 การทดลองเพื่อทดสอบความถูกต้องของโปรแกรม หลังแก้ไขให้โปรแกรมมีคุณสมบัติ Reentrant 81
6.4 การทดลองเพื่อทดสอบความถูกต้องของโปรแกรมหลังแก้ไขให้โปรแกรมเป็น Relocatable 82
6.5 การทดลองเพื่อทดสอบความถูกต้องของโปรแกรมหลังแก้ไขให้โปรแกรมเป็น มาตราฐานการเขียน อัลกอริทึมของผู้ผลิตไลบราลี 83
6.6 การทดลองเพื่อทดสอบความถูกต้องของโปรแกรมหลังแก้ไขให้โปรแกรมเป็น มาตราฐานการเขียน อัลกอริทึมของผู้ผลิตไลบราลี และเปลี่ยนอินเตอร์เฟสของฟังก์ชัน ของ Abstract algorithm interface .. 84
6.7 การทดลองเปรียบเทียบลักษณะสัญญาณคุณภาพเสียงที่ได้ระหว่างชุดสังเคราะห์เสียงG.729 กับ G.723.1 85
6.8 การทดลองทำการวัดเวลาสำหรับโมดูล ต่าง ๆ สำหรับการเข้ารหัส 97
6.9 การทดลองทำการวัดเวลาหลังจากทำการออพติไมซ์ตามหัวข้อวิจัยของ F.-K CHEN, J.-F YANG และ Y. –L YAN เรื่อง Candidate scheme for fast ACELP search 98
6.10 การทดลองทำการวัดเวลาหลังจากทำการออพติไมซ์โดยการบังคับโค้ดเซคชันให้อยู่ใน หน่วยความจำภายในของ ดี เอส พี บอร์ด 104
6.11 การทดลองทำการวัดเวลาหลังจากทำการออพติไมซ์โดยการแทนฟังก์ชันบางฟังก์ชั่นใน math.h ด้วยไลบราลี fastrts67x.h ของ ดี เอส พี 106
บทที่ 7บทวิจารณ์และสรุป 108
7.1 สรุปผลที่ได้จากการทำโครงงาน 108
7.2 แนวทางในการพัฒนาต่อ 109
บรรณานุกรม 110

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

รูปที่ 1-1 แสดงการดำเนินงานในภาคการศึกษา 1/2547 3
รูปที่ 1-2 แสดงการดำเนินงานในภาคการศึกษา 2/2547 4
รูปที่ 2-1 แสดงกลไกการเกิดเสียงของมนุษย์ 5
รูปที่ 2-2 แสดงการสังเคราะห์สัญญาณเสียง 6
รูปที่ 2-3 แสดง linear prediction mode of speech 8
รูปที่ 2-4 แสดงตัวอย่างลักษณะของสัญญาณเสียงที่เป็นคาบ 9
รูปที่ 2-5 แสดงตัวอย่างลักษณะของสัญญาณ เสียงแบบสุ่ม 9
รูปที่ 2-6 แสดงแบบจำลองการสังเคราะห์เสียงแบบ CELP 10
รูปที่ 2-7 แสดงสมการที่ใช้แสดงความสัมพันธ์ในการประมาณค่าของสัญญาณ โดยใช้วิธีการของการประมาณพันธะเชิงเส้น 11
รูปที่ 3-1 แสดงโฟลโปรแกรมของการหาสัมประสิทธิ์ของวงจรกรอง 14
รูปที่ 3-2 แสดงเฟรม ของสัญญาณอินพุทที่ใช้ในการคำนวณ 16
รูปที่ 3-3 แสดงวิธีการหาสัญญาณกระตุ้นที่มีลักษณะของสัญญาณเป็นคาบ (Voice sound) 24
รูปที่ 3-4 แสดงการหาองค์ประกอบสัญญาณเสียงสุ่ม (Unvoiced sound) 31
รูปที่ 3-5 แสดงโฟลโปรแกรมของการถอดรหัส 38
รูปที่ 3-6 แสดง Speech communication system with VAD 39
รูปที่ 4-1 แสดงเส้นโค้งของ $E[h[n]]^2$ 43
รูปที่ 4-2 แสดง ACELP STAGE SEARCH 44
รูปที่ 4-3 แสดงฟังก์ชัน Pilot, target และสัญญาณ synthesis ใน ตัวเข้ารหัสแบบ ACELP 46
รูปที่ 5-1 แสดงการทำ Data relocatable 52
รูปที่ 5-2 แสดงลำดับการทำงานของวัตถุ 63
รูปที่ 5-3 แสดงลิงก์เกอร์คอมมานด์ไฟล์makelib.bat 73
รูปที่ 5-4 แสดงตัวอย่างของลิงก์เกอร์คอมมานด์ไฟล์ g729enc.cmd 75
รูปที่ 6-1 แสดงสัญญาณเสียงต้นฉบับ 77
รูปที่ 6-2 แสดงสัญญาณเสียงสังเคราะห์โดยโมดูล G.729 77
รูปที่ 6-3 แสดงลักษณะของสัญญาณเสียงต้นฉบับ ในโดเมนของเวลา 86
รูปที่6-4 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของเวลา 86
ในการสังเคราะห์เสียงรอบที่ 1 86

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 6-5 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.723.1 ในโดเมนของเวล.....87
ในการสังเคราะห์เสียงรอบที่ 1.....87
รูปที่6-6 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของเวลา.....87
ในการสังเคราะห์เสียงรอบที่ 5.....87
รูปที่ 6-7 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.723.1 ในโดเมนของเวล.....88
ในการสังเคราะห์เสียงรอบที่ 5.....88
รูปที่ 6-8 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของเวล.....88
ในการสังเคราะห์เสียงรอบที่ 10.....88
รูปที่ 6-9 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.723.1 ในโดเมนของเวล.....89
ในการสังเคราะห์เสียงรอบที่10.....89
รูปที่ 6-10 แสดงการเปรียบเทียบสัญญาณต้นฉบับ กับสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 และ G.723.1 ในการสังเคราะห์เสียงรอบที่ 1.....91
รูปที่ 6-11 แสดงการเปรียบเทียบสัญญาณต้นฉบับ กับสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 และ G.723.1 ในการสังเคราะห์เสียงรอบที่ 5.....91
รูปที่ 6-12 แสดงการเปรียบเทียบสัญญาณต้นฉบับ กับสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 และ G.723.1 ในการสังเคราะห์เสียงรอบที่ 7.....92
รูปที่ 6-13 แสดงการเปรียบเทียบสัญญาณต้นฉบับ กับสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 และ G.723.1 ในการสังเคราะห์เสียงรอบที่ 10.....92
รูปที่ 6-14 แสดงสัญญาณต้นฉบับในโดเมนของ Spectral.....93
รูปที่ 6-15 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของ Spectral ในการสังเคราะห์เสียงรอบที่ 1.....93
รูปที่ 6-16 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส ฌ.723.1 ในโดเมนของ Spectral.....94
ในการสังเคราะห์เสียงรอบที่ 1.....94
รูปที่ 6-17 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของ Spectral.....94
ในการสังเคราะห์เสียงรอบที่ 5.....94
รูปที่ 6-18 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส ฌ.723.1 ในโดเมนของ Spectral.....95
ในการสังเคราะห์เสียงรอบที่ 5.....95
รูปที่ 6-19 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของ Spectral.....95
ในการสังเคราะห์เสียงรอบที่ 10.....95
รูปที่ 6-20 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.723.1 ในโดเมนของ Spectral.....96
ในการสังเคราะห์เสียงรอบที่ 10.....96

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 6-21 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสตามมาตรฐาน 99
รูปที่ 6-22 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 7 100
รูปที่ 6-23 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 6 100
รูปที่ 6-24 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 5 101
รูปที่ 6-25 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 4 101
รูปที่ 6-26 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 3 102
รูปที่ 6-27 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 2 102
รูปที่ 6-28 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 1 103
รูปที่ 6-29 แสดงการกำหนด Memory Map ให้กับ Segment ต่าง ๆ 104

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

ตารางที่ 1-1 แสดงแผนการดำเนินงานในภาคการศึกษาที่ 1/2547 3
ตารางที่ 1-2 แสดงการวางแผนการดำเนินงานในภาคการศึกษาที่ 2/2547 4
ตารางที่ 2-1 ความสัมพันธ์ระหว่างรูปแบบทางกายภาพกับรูปแบบทางคณิตศาสตร์ 7
ตารางที่ 3-1 แสดงโครงสร้างของ fixed codebook 32
ตารางที่ 3-2 แสดงบิทฟิลด์ที่ส่งให้ฝั่งถอดรหัส 38
ตารางที่ 4-1 แสดงเวลาที่ใช้ในแต่ละโมดูล 41
ตารางที่ 4-2 แสดงตำแหน่งพัลซ์ และพื้นที่ ของการค้นหาอินเด็ก 45
ตารางที่ 4-3 แสดงจำนวนของความเป็นไปได้ ในวิธีการค้นหาแบบ ACELP สำหรับ G.729 coders 47
ตารางที่ 4.4 MOS degradation for the proposed candidate 47
ตารางที่ 4-5 แสดงคุณสมบัติของบอร์ดที่ใช้ 48
ตารางที่ 5-1 แสดงขนาดของ Memtabแต่ละตารางหน่วยความจำของตัวเข้ารหัส 53
ตารางที่ 5.2 แสดงขนาดของ Memtabแต่ละตารางหน่วยความจำของตัวถอดรหัส 54
ตารางที่ 5.3 แสดงขนาดของตารางค่าคงที่ 54
ตารางที่ 5-4 แสดงออพชันของคำสั่ง lnk6x 74
ตารางที่ 6-1 แสดงผลการทดลองเมื่อเลือกระดับของการออพติไมซ์แบบต่าง ๆ 78
สำหรับโมดูล การเข้ารหัส G.729 78
ตารางที่ 6-1 แสดงผลการทดลองเมื่อเลือกระดับของการออพติไมซ์แบบต่าง ๆ 79
สำหรับโมดูล การเข้ารหัส G.729 (ต่อ) 79
ตารางที่ 6-2 แสดงผลการทดลองเมื่อเลือกระดับของการออพติไมซ์แบบต่าง ๆ 79
สำหรับโมดูล การถอดรหัส G.729 79
ตารางที่ 6-3 แสดงการเปรียบเทียบค่า RMSe และ Snr ของทั้งสอง Algorithm 89
ตารางที่ 6-4 แสดงการวัดค่าทางสถิติของชุดสังเคราะห์สัญญาณทั้งสองชุด โดความละเอียดของตัวอย่างที่ใช้ ขนาด 16 บิทใช้ RMS windows ขนาด 50 ms 90
ตารางที่ 6-5 แสดงจำนวนครั้งในการเรียกใช้งานฟังก์ชัน เรียงลำดับการเรียกใช้จากมากไปน้อย 97
ตารางที่ 6-6 แสดงผลการทดลองการวัดเวลาหลังจากทำการออพติไมซ์โดยใช้วิธีของ Candidate scheme for fast ACELP search 98
ตารางที่ 6-7 แสดงผลการทดลองการวัดเวลาหลังจากทำการออพติไมซ์โดยใช้วิธีของ Candidate scheme for fast ACELP search พร้อมทั้งปรับระดับของการออพติไมซ์เป็นแบบ No External Var Refs 99
ตารางที่ 6-8 แสดงผลการทดลองวัดเวลาหลังจากทำการออพติไมซ์โดยการบังคับโค้ดเซคชัน 105

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ให้อยู่ในหน่วยความจำภายในของ ดี เอส พี บอร์ด 105
ตารางที่ 6-9 แสดง ฟังก์ชัน ที่ได้ทำการแก้ไขการเรียกใช้ 106
ตารางที่ 6-10 แสดงผลการทดลองวัดเวลาหลังจากทำการออพติไมซ์โดยการบังคับโค้ดเซคชันให้อยู่ใน
หน่วยความจำภายในของ ดี เอส พี บอร์ด และใช้ ดี เอส พีไลบรารี ช่วย 106
ตารางที่ 6-10 แสดงผลการทดลองวัดเวลาหลังจากทำการออพติไมซ์โดยการบังคับโค้ดเซคชันให้อยู่ใน
internal ram ของ ดี เอส พี บอร์ด และใช้ ดี เอส พีไลบรารี ช่วย (ต่อ) 107

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 หลักการและเหตุผล

เทคโนโลยีด้านการเข้ารหัสสัญญาณเสียง (Speech Coding / Speech Compression) ใช้สำหรับการแทนสัญญาณเสียงให้อยู่ในรูปของสัญญาณดิจิตอลที่มีขนาดเล็กทำให้มีการใช้งานแบนด์วิธ (Bandwidth) ในการส่งและจัดเก็บข้อมูลได้อย่างมีประสิทธิภาพ การเข้ารหัสสัญญาณเสียงนี้เกี่ยวข้องกับการสุ่มสัญญาณและควอนไทต์ (quantize) สัญญาณ เนื่องจากการสุ่มสัญญาณถูกจำกัดด้วยทฤษฎีของ Nyguist อยู่แล้วไม่สามารถลดค่าความถี่ในการสุ่มลงได้ ดังนั้นการค้นคว้าด้านบีบอัดสัญญาณเสียง (Speech Compression) จึงมุ่งเน้นไปที่การพัฒนาการแทนสัญญาณโดยใช้ขนาดที่เล็กลงแต่ยังคงให้คุณภาพเสียงที่ดี เราแบ่งการแทนสัญญาณหรือการควอนไทต์สัญญาณออกเป็น 2 ประเภทคือการควอนไทต์โดยตรง ซึ่งเป็นการแทนสัญญาณที่ถูกสุ่มโดยตรงและการควอนไทต์แบบพารามิเตอร์ (parameter) ซึ่งเป็นการแทนแบบจำลองที่ใช้สร้างเสียงนั้นขึ้นมา

ด้วยเหตุผลข้างต้น ผู้จัดทำโครงงานจึงมีแนวคิดที่จะศึกษาวิธีการเข้ารหัส / ถอดรหัส สัญญาณเสียงตามมาตรฐานของ ITU-T G.729 (1996) ซึ่งเป็นวิธีการเข้ารหัสแบบ Conjugate-Structure Algebraic-Code-Excited-Linear-Prediction (CS-ACELP) ซึ่งอยู่ในประเภทของการควอนไทต์โดยใช้พารามิเตอร์และอิมพลีเมนต์ (implement) บน ดี เอส พี บอร์ด (DSP Board) ของเท็กซัส อินสตรูเมนต์ (Texas Instrument) รุ่น TMS320C6713

1.2 วัตถุประสงค์

1. เพื่อศึกษากระบวนการประมวลผลสัญญาณดิจิตอล
2. เพื่อศึกษาวิธีการเข้ารหัส / ถอดรหัส สัญญาณเสียงตามมาตรฐานของ ITU-T G.729
3. เพื่อศึกษาวิธีการโปรแกรมระบบฝังตัวบน ดี เอส พี เพลตฟอร์ม (DSP Platform)
4. เพื่อศึกษาวิธีการโปรแกรมตามมาตรฐานของการพัฒนาอัลกอริทึมของ TI
5. นำความรู้ที่ได้จากหัวข้อที่ 1 – 4 มาทำการดัดแปลงและออพติไมซ์ (Optimize) เพื่อโปรแกรมลงบน TMS320C6713

1.3 ผลที่คาดว่าจะได้รับ

สามารถสร้างไลบรารี (Library) ที่อ้างอิงมาตรฐานของ TI ของ โมดูล (Module) การเข้ารหัส / ถอดรหัส สัญญาณเสียงตามมาตรฐานของ ITU-T G.729 บนTMS320C6713 ได้พร้อมกันอย่างต่ำ 4 ช่องสัญญาณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.4 ขอบเขตของการพัฒนา

สำหรับโครงงานที่จัดทำนั้นจะใช้ภาษา C ในการพัฒนาอัลกอริทึมตามมาตรฐานของ เท็กซัส อินสตรูเมนต์ โดยใช้เครื่องมือในการพัฒนาคือ Code Composer Studio Version 2.0

1.4.1 สามารถเข้ารหัส / ถอดรหัส สัญญาณเสียงได้ตามมาตรฐานของ ITU-T G.729 Annex B และ C บน ดี เอส พี บอร์ด TMS320C6713

1.4.2 สามารถรองรับช่องสัญญาณได้อย่างน้อย 4 ช่องสัญญาณ

1.5 ขั้นตอนการดำเนินงาน

โครงงานนี้แบ่งการดำเนินงานออกเป็น 3 ขั้นตอน

1.5.1 เป็นการศึกษาข้อมูลที่จำเป็นในการทำโครงงานนี้ ประกอบด้วย

1. ศึกษาพื้นฐานของการประมวลผลสัญญาณดิจิตอล
2. ศึกษามาตรฐานของการเข้ารหัส / ถอดรหัส สัญญาณเสียงตามมาตรฐานของITU-T G.729
3. ศึกษามาตรฐานการเขียนโปรแกรมบน ดี เอส พี เพลตฟอร์ม ของ เท็กซัส อินสตรูเมนต์ และวิธีการงาน Code composer studio V 2.0
4. ศึกษาวิธีการเขียนโปรแกรมแบบมัลติแชนเนิล (Multi-channel)
5. ศึกษาวิธีการออพติไมซ์อัลกอริทึม

1.5.2 เป็นขั้นตอนในการพอร์ต (Port) อัลกอริทึม (Algorithm) ลงบน ดี เอส พี เพลตฟอร์ม

1. ทำการวิเคราะห์ อัลกอริทึม ในการเข้ารหัส / ถอดรหัส ว่าประกอบด้วย โมดูล อะไรบ้าง
2. ทำการ อิมพลีเมนต์ อัลกอริทึม ให้สามารถรันแบบมัลติแชนเนิลได้
3. ทำการ Implement แต่ละ โมดูล ลงบน ดี เอส พี เพลตฟอร์ม ตามมาตรฐานของ เท็กซัส อินสตรูเมนต์

1.5.3 เป็นขั้นตอนในการออพติไมซ์โปรแกรม

1. เริ่มต้นจากการวัดการทำงานของ โมดูล ย่อยในการเข้ารหัสเพื่อหาว่าโมดูลใดมีความซับซ้อนและใช้เวลาในการทำงานมากที่สุด
2. หาหัวข้อวิจัยที่เกี่ยวข้องกับ โมดูล นั้นศึกษาและนำมาประยุกต์กับขั้นตอนที่มีอยู่เดิม
3. ศึกษาการจัดการหน่วยความจำเพื่อใช้พื้นที่ในหน่วยความจำภายในได้อย่างมีประสิทธิภาพ
4. ในส่วนของฟังก์ชั่นที่เกี่ยวกับคณิตศาตร์พื้นฐานเช่น ฟังก์ชั้นตรีโกนมิติ ฟังก์ชั่นสำหรับเลขยกกำลัง สามารถทดแทนด้วยฟังก์ชั่นของ DSP
5. พัฒนาโปรแกรมให้สอดคล้องกับมาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบราลี (Standard Algorithm) และเขียนลิงก์เกอร์เพื่อสร้างไลบราลี

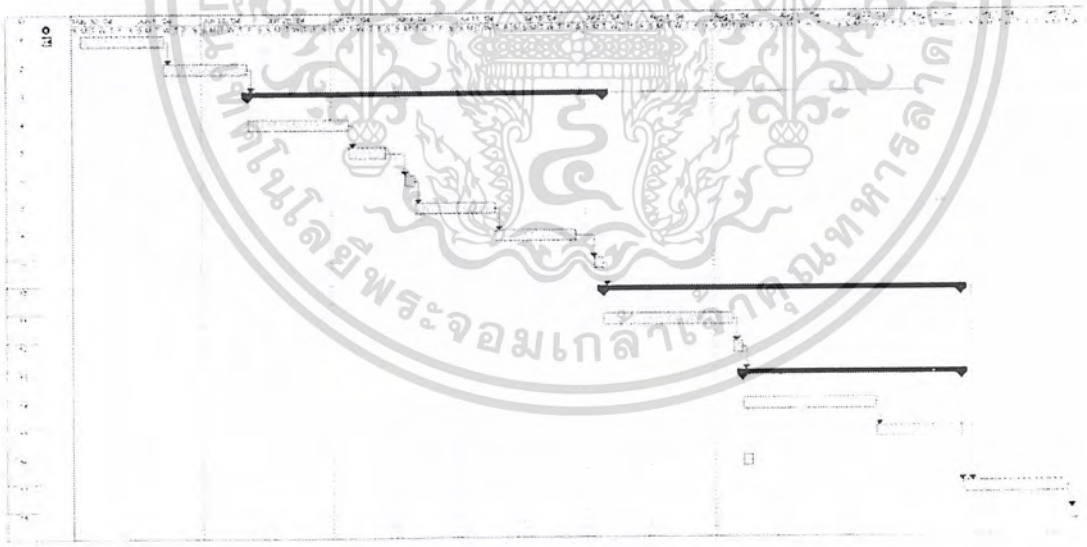
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แผนการดำเนินงานประจำภาคการศึกษาที่ 1/2547

	Task Name	WBS	Duration	Start	Finish
1	กำหนด Scope ของชิ้นงาน	1	7 days	Mon 5/31/04	Tue 6/8/04
2	ศึกษาพื้นฐานทางคณิตศาสตร์ของ DSP Algorithm ที่เกี่ยวข้อง	2	7 days	Wed 6/9/04	Thu 6/17/04
3	**ศึกษามาตรฐาน ITU-T G.729**	**3**	**27 days**	**Fri 6/18/04**	**Mon 7/26/04**
4	ศึกษาการทำงานของ LPC	3.1	7 days	Fri 6/18/04	Mon 6/28/04
5	ศึกษาการทำงานของ open-loop pitch	3.2	4 days	Tue 6/29/04	Fri 7/2/04
6	นำเสนอข้อมูลที่ได้ทำการศึกษาให้กับอ.ที่ปรึกษา	3.3	1 day	Mon 7/5/04	Mon 7/5/04
7	ศึกษาการทำงานของ closed-loop pitch	3.4	7 days	Tue 7/6/04	Wed 7/14/04
8	ศึกษาการทำงานของ fixed-code book & gain	3.5	7 days	Thu 7/15/04	Fri 7/23/04
9	นำเสนอข้อมูลที่ได้ทำการศึกษาให้กับอ.ที่ปรึกษา	3.6	1 day	Mon 7/26/04	Mon 7/26/04
10	**ศึกษาและฝึกใช้งาน board DSP TMS320c6000**	**4**	**28 days**	**Tue 7/27/04**	**Thu 9/2/04**
11	ศึกษาวิธีการใช้ tool ในการพัฒนา program ของ board	4.1	10 days	Tue 7/27/04	Mon 8/9/04
12	นำเสนอข้อมูลที่ได้ทำการศึกษาให้กับอ.ที่ปรึกษา	4.2	1 day	Tue 8/10/04	Tue 8/10/04
13	**implement standard G.729 ลงบน DSP board**	**4.3**	**17 days**	**Wed 8/11/04**	**Thu 9/2/04**
14	port โปรแกรมลงบน DSP board	4.3.1	10 days	Wed 8/11/04	Tue 8/24/04
15	ศึกษาวิธีการ optimize โดยใช้ tool	4.3.2	7 days	Wed 8/25/04	Thu 9/2/04
16	นำเสนอข้อมูลที่ได้ทำการทดลองให้กับอ.ที่ปรึกษา	4.3.3	1 day	Wed 8/11/04	Wed 8/11/04
17	สรุปผลการทดลอง ทำรายงานเพื่อนำเสนอผลงาน	5	7 days	Fri 9/3/04	Mon 9/13/04
18	ซ้อม present ให้กับอ.ที่ปรึกษา	6	1 day	Tue 9/14/04	Tue 9/14/04

Visio WBS Chart Task Selection

ตารางที่ 1-1 แสดงแผนการดำเนินงานในภาคการศึกษาที่ 1/2547



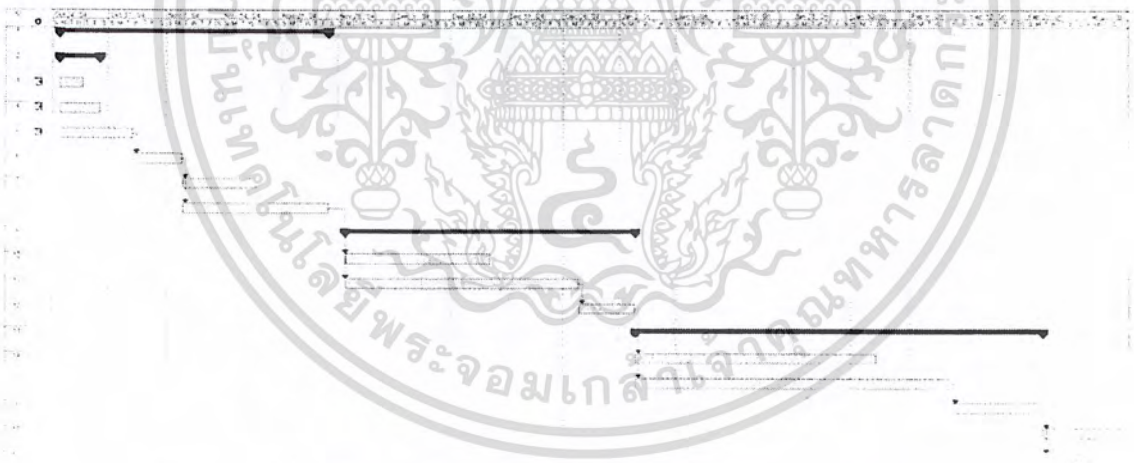
รูปที่ 1-1 แสดงการดำเนินงานในภาคการศึกษา 1/2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แผนการดำเนินงานประจำภาคการศึกษาที่ 2/2547

	Task Name	Duration	Start	Finish	Predece
1	**ปรับปรุงโครงสร้างของโปรแกรมให้รองรับการทำงานแบบ multichannel**	**25 days**	**Mon 10/25/04**	**Fri 11/26/04**	
2	**ศึกษาวิธีการเขียน Program แบบ multichannel**	**5 days**	**Mon 10/25/04**	**Fri 10/29/04**	
3	Reentrant	3 days	Mon 10/25/04	Wed 10/27/04	
4	Relocation	5 days	Mon 10/25/04	Fri 10/29/04	
5	ปรับปรุงโครงสร้างข้อมูลที่ใช้ภายในโปรแกรมส่วนของการเข้ารหัส	7 days	Mon 10/25/04	Tue 11/2/04	
6	ปรับปรุงโครงสร้างข้อมูลที่ใช้ภายในโปรแกรมส่วนของการถอดรหัส	4 days	Wed 11/3/04	Mon 11/8/04	5
7	ศึกษาการใช้งาน DSP bios	7 days	Tue 11/9/04	Wed 11/17/04	6
8	ทดสอบความถูกต้องของการทำงาน	14 days	Tue 11/9/04	Fri 11/26/04	6
9	**พัฒนา DSP Library**	**26 days**	**Mon 11/29/04**	**Mon 1/3/05**	
10	ศึกษาวิธีการสร้าง Library ให้เป็นไปตามมาตรฐานของผู้พัฒนา DSP library	14 days	Mon 11/29/04	Thu 12/16/04	8
11	implement DSP library	21 days	Mon 11/29/04	Mon 12/27/04	8
12	ทดสอบ DSP Library ที่ได้พัฒนาขึ้นมา	5 days	Tue 12/28/04	Mon 1/3/05	11
13	**ปรับปรุงประสิทธิภาพของ DSP Library**	**35 days**	**Tue 1/4/05**	**Mon 2/21/05**	
14	ศึกษาหัวข้อวิจัยที่เกี่ยวข้องกับการปรับปรุงมาตรฐาน G.729	21 days	Tue 1/4/05	Tue 2/1/05	12
15	ทดลอง Implement สิ่งที่ได้ศึกษาเพิ่มจากหัวข้อวิจัย	28 days	Tue 1/4/05	Thu 2/10/05	12
16	ตรวจสอบความถูกต้อง และวัดประสิทธิภาพของ Library ที่ปรับปรุงขึ้น	7 days	Fri 2/11/05	Mon 2/21/05	15
17	ทำรายงาน สรุปและประเมินผลการทำงานของโครงงาน	7 days	Tue 2/22/05	Wed 3/2/05	16
18	เตรียม Presentation	7 days	Tue 2/22/05	Wed 3/2/05	16

ตารางที่ 1-2 แสดงการวางแผนการดำเนินงานในภาคการศึกษาที่ 2/2547



รูปที่ 1-2 แสดงการดำเนินงานในภาคการศึกษา 2/2547

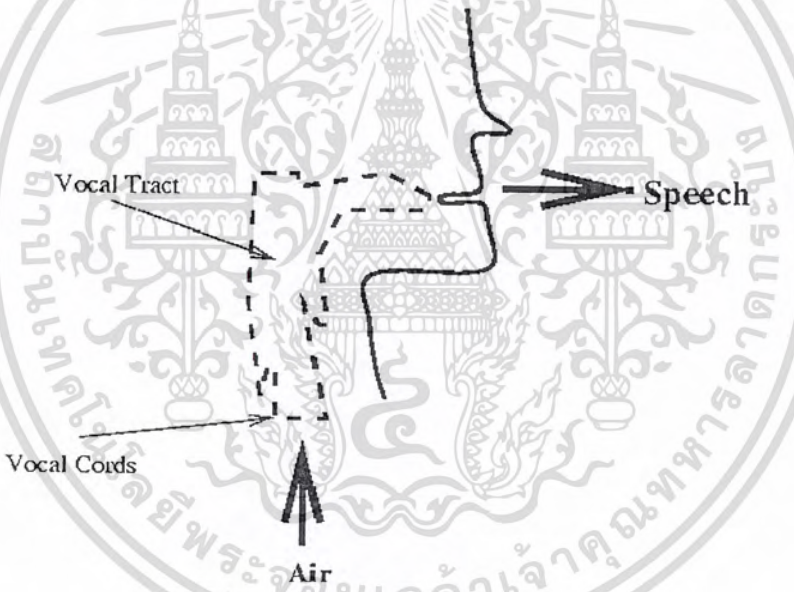
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีการสังเคราะห์เสียง

2.1 บทนำ

การเข้ารหัสสัญญาณเสียงโดยวิธีควอนไทต์โดยใช้พารามิเตอร์นั้น เราพยายามหาแบบจำลองทางคณิตศาสตร์เพื่อแทนระบบของการเกิดเสียงของมนุษย์ พารามิเตอร์ที่นำมาใช้ในการสังเคราะห์นั้น แบ่งออกเป็นสองส่วนหลัก ๆ คือ สัญญาณกระตุ้น (excitation signal) และวงจรกรอง (filter) ซึ่งเมื่อเปรียบเทียบกับการกำเนิดเสียงของมนุษย์นั้น สัญญาณกระตุ้น คือ ลมในช่องปอด และ วงจรกรอง คือ อวัยวะกำเนิดเสียง และเมื่อนำสัญญาณกระตุ้นผ่านวงจรกรอง ก็จะเกิดเป็นเสียงได้ในที่สุด

2.2 การสังเคราะห์เสียงเชิงกายภาพ (Physical Model)



รูปที่ 2-1 แสดงกลไกการเกิดเสียงของมนุษย์

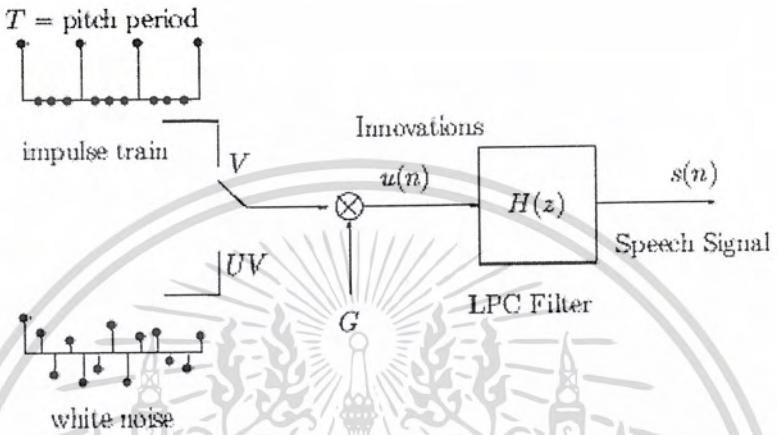
จากรูปที่ 2.1 แสดงวิธีการกำเนิดเสียงของมนุษย์ ขณะที่พูดนั้นลมจะถูกผลักออกมาจากปอดผ่านหลอดลมและออกมาทางปากกลายมาเป็นเสียง โดยลักษณะของอวัยวะออกเสียงมีผลต่อเสียงที่กำเนิด ดังนี้

1. เสียงจะมีการสั่นเป็นจังหวะ (Voiced sound) โดยคาบของการสั่นเรียกว่าพิตท์ซึ่งผู้หญิงและเด็กจะมีการสั่นของเส้นเสียงที่เร็วกว่าของผู้ชายทำให้มีเสียงที่สูงกว่าเสียงของผู้ชาย
2. เส้นเสียงจะไม่สั่นแต่จะเปิดให้ลมผ่านออกไปอย่างเดียว (Unvoiced sound)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. รูปร่างที่เปลี่ยนไปของหลอดลมทำให้เสียงเปลี่ยนไป
4. รูปร่างของหลอดลมจะเปลี่ยนไปอย่างช้า ๆ โดยทั่วไปอยู่ในระดับ 10 – 100 ms
5. ปริมาณที่ถูกผลักออกมาจากปอดนั้นเป็นสิ่งที่กำหนดความดังของเสียง

2.3 การสังเคราะห์เสียงเชิงคณิตศาสตร์ (Mathematical Model)



รูปที่ 2-2 แสดงการสังเคราะห์สัญญาณเสียง

จากรูปที่ 2.2 แสดงแบบจำลองทางคณิตศาสตร์สำหรับการสังเคราะห์เสียง โดยอธิบายได้ดังนี้

1. สัญญาณเสียงที่สังเคราะห์ขึ้นได้จากการนำสัญญาณเสียงที่เป็นคาบ (Voiced) และสัญญาณเสียงที่ไม่เป็นคาบ (Unvoiced) มาทำการมอดูเลท (modulate) และผ่านวงจรกรอง
2. เปรียบเทียบความสัมพันธ์ระหว่างรูปแบบทางกายภาพกับรูปแบบทางคณิตศาสตร์ได้ดังต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลอดลม (Vocal Tract)	⟺	*H(z)* (LPC Filter)
ลม (Air)	⟺	*u(n)* (Innovations)
การสั่นของเส้นเสียง (Vocal Cord Vibration)	⟺	*V* (Voiced)
คาบการสั่นของเส้นเสียง (Vocal Cord Vibration Period)	⟺	*T* (Pitch period)
เสียงที่ไม่มีการสั่นของเส้นเสียง (Fricatives and Plosives)	⟺	*UV* (Unvoiced)
ปริมาณของลมที่ออกมาจากปอด (Air Volume)	⟺	*G* (Gain)

ตารางที่ 2-1 ความสัมพันธ์ระหว่างรูปแบบทางกายภาพกับรูปแบบทางคณิตศาสตร์

2.3.1 การหาสัมประสิทธิ์ของวงจรกรอง LP

แนวคิดของการหาสัมประสิทธิ์ของวงจรกรอง LP คือการประมาณให้ตัวอย่างของเสียงที่เวลาใด ๆ s(n) สามารถประมาณค่าได้จากการทำความเป็นไปได้เชิงเส้น (linear combination) ของตัวอย่างสัญญาณเสียงอินพุทก่อนหน้าจำนวน p ตัวอย่าง เช่น

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + ... + a_p s(n-p) \tag{2.1}$$

ซึ่งสัมประสิทธิ์ของ $a_1, a_2, \ldots, a_p$ นั้นถือว่ามีค่าคงที่ในช่วงเฟรม ที่ทำการวิเคราะห์ เราสามารถแปลงสมการที่ (2.1) จากการประมาณให้เป็นเท่ากับได้โดยเพิ่มสัญญาณกระตุ้นเข้าไป ดังต่อไปนี้

$$s(n) = \sum_{i=1}^{p} a_i s(n-i) + Gu(n) \tag{2.2}$$

โดยที่ u(n) เป็นสัญญาณกระตุ้นที่ถูกทำให้เป็นหนึ่งหน่วย (normalize)แล้ว เมื่อ G เป็นอัตราขยายของสัญญาณกระตุ้น ซึ่งหากพิจารณาจากสมการดังกล่าวบน โดเมนของ z จะได้ความสัมพันธ์ดังนี้

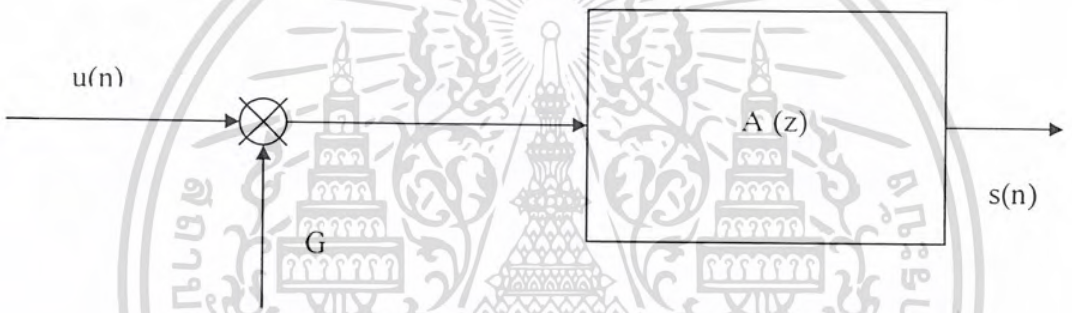
$$S(z) = \sum_{i=1}^{p} a_i z^{-i} S(z) + GU(z) \tag{2.3}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำให้เราได้ฟังก์ชันถ่ายโอน (transfer function) เป็น

$$H(z) = \frac{S(z)}{GU(z)} = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}} = \frac{1}{A(z)} \tag{2.4}$$

ความหมายของสมการนี้สามารถแสดงได้ด้วยภาพที่ 2.1 ซึ่งแหล่งกำเนิดของสัญญาณกระตุ้นปล่อยสัญญาณกระตุ้น u (n) ออกมา สัญญาณจะถูกขยายด้วยค่า G และกลายเป็นสัญญาณอินพุทที่ป้อนให้กับวงจรกรองแบบ all pole เพื่อสร้างสัญญาณเสียง s(n) ออกมา จากความรู้ในปัจจุบันเราพบว่าสัญญาณกระตุ้นของเสียงพูดนั้นเป็นแบบพัลซ์เทรน (pulsed train) หรือสัญญาณเสียงที่เป็นคาบ (voiced sound) หรือเป็นสัญญาณแบบสุ่ม (unvoiced sound) ส่วนวงจรกรองนั้นเป็นการทำงานที่ผสมผสานกันของอวัยวะต่างที่ใช้ออกเสียง (ช่องคอ, ทางเดินอากาศ, ช่องปาก และลิ้น)



รูปที่ 2-3 แสดง linear prediction mode of speech

การประมาณแบบความเป็นไปได้เชิงเส้นของสัญญาณเสียงในอดีต โดยนิยามค่าของสัญญาณที่ประมาณ ดังนี้

$$\tilde{s}(n) = \sum_{i=1}^{p} a_i s(n-i) \tag{2.5}$$

ค่าความผิดพลาดในการประมาณ e (n) แสดงได้โดย

$$e(n) = s(n) - \tilde{s}(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k) \tag{2.6}$$

โดยที่มีค่าความผิดพลาดของฟังก์ชันถ่ายโอน (error transfer function)

$$A(z) = \frac{E(z)}{S(z)} = 1 - \sum_{k=1}^{p} a_k z^{-k} \tag{2.7}$$

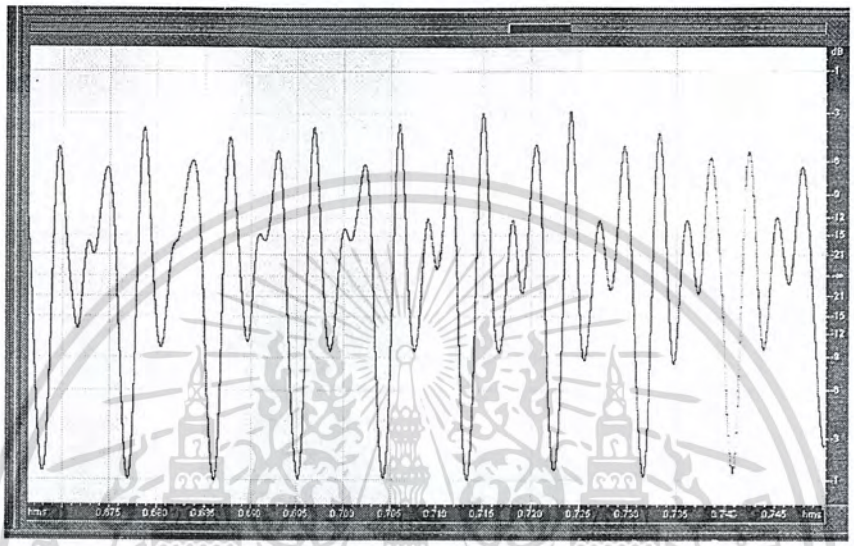
ดังนั้นเมื่อให้ s(n) เป็นสัญญาณจริงที่ถูกสร้างขึ้นมาจากระบบเชิงเส้นตามสมการที่ (2.2) จะพบว่า e(n) จะมีค่าเท่ากับ สัญญาณกระตุ้นที่ถูกขยายแล้ว Gu(n)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.2 องค์ประกอบของสัญญาณเสียง

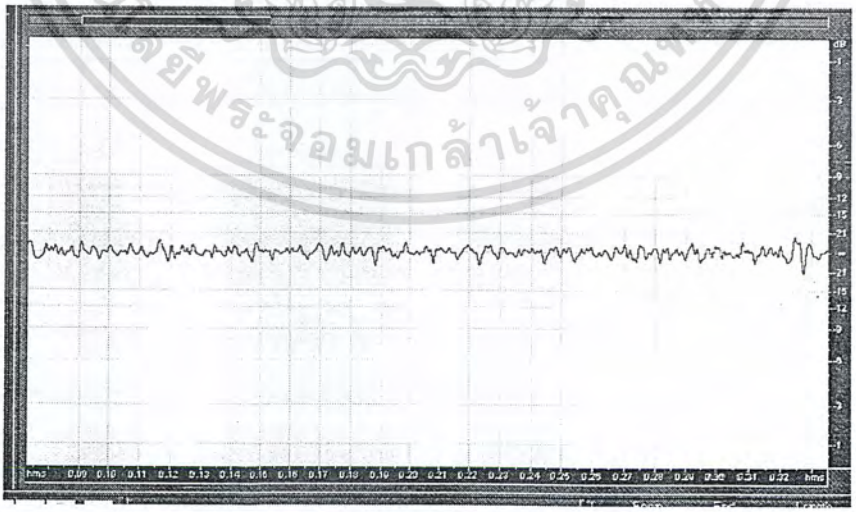
1. Voiced sound คือ สัญญาณที่มีคุณสมบัติซ้ำกันเป็นคาบ (Periodic signal)

หากเปรียบเทียบกับการออกเสียงของเราก็คือเสียงที่ออกมาแล้วรู้สึกถึงการสั่นของเส้นเสียงที่ลำคอ เช่นการออกเสียงตัวอักษร b, d, g



รูปที่ 2-4 แสดงตัวอย่างลักษณะของสัญญาณเสียงที่เป็นคาบ

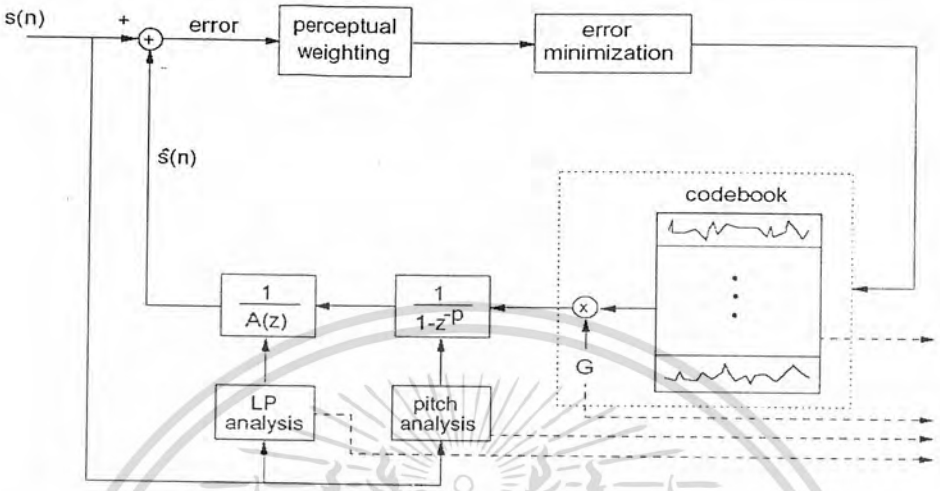
2. สัญญาณเสียงแบบสุ่ม (Unvoiced sound) ก็คือสัญญาณเสียงที่มีลักษณะของสัญญาณเป็นสัญญาณรบกวน หากเปรียบเทียบกับการออกเสียงของเราก็คือเสียงที่สร้างโดยการบังคับลมของช่องปาก เช่นการออกเสียง th, f, sh



รูปที่ 2-5 แสดงตัวอย่างลักษณะของสัญญาณ เสียงแบบสุ่ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4 Code Excited Linear Prediction Model (CELP)



รูปที่ 2-6 แสดงแบบจำลองการสังเคราะห์เสียงแบบ CELP

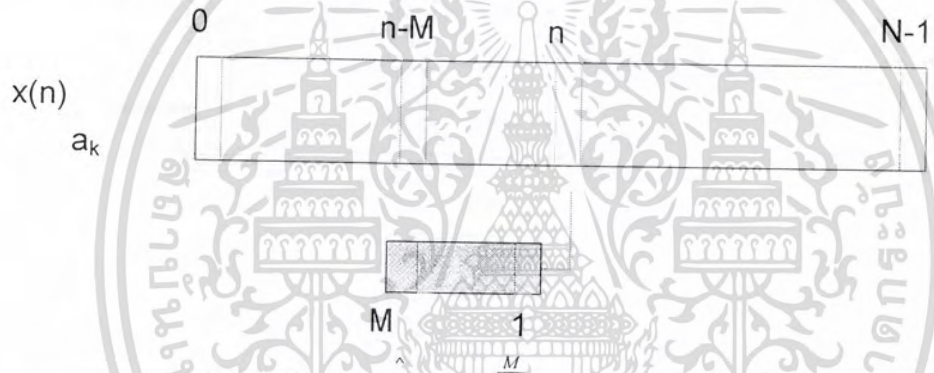
G.729 จัดอยู่ใน Speech coders ใน category CELP (Code Excited Linear Prediction) ซึ่ง Model ชนิดนี้มีพื้นฐานมาจากระบบกำเนิดเสียงของมนุษย์ โดยอวัยวะกำเนิดเสียงของคนเราจะถูกจำลองเป็นวงจรกรอง (Linear filter) ชุดหนึ่ง ซึ่งเสียงพูดนั้นจะถูกสร้างโดยอาศัยการสร้างสัญญาณที่มีคาบเท่ากับคาบการสั่นของอากาศมากระตุ้นวงจรกรองนี้ ถ้าเรามองในเชิงของ โดเมนเชิงความถี่ แล้วเสียงพูดจะถูกพิจารณาจากองค์ประกอบ 2 ส่วนคือ Smooth response (envelope) หรือสัญญาณที่มีลักษณะเป็นคาบ (Periodic signal) มาทำการมอดูเลทกับ ชุดขององค์ประกอบของความถี่ที่ไม่เป็นคาบ (discrete frequency signal) ในการแทน Coder Model ของ CELP นั้นจะแปรผันไปตามสัญญาณกระตุ้นที่ระบุ และค่าสัมประสิทธิ์ของวงจรกรอง ซึ่งทั้ง 2 ส่วนนั้นจะหามาจากช่วงของสัญญาณเสียงที่ ตัดออกมาเป็น ยูนิท (unit)หรือเรียกอีกอย่างหนึ่งว่าเฟรม ซึ่งช่วงในการตัดสัญญาณเสียงที่นำมาพิจารณาสามารถ แปรผันได้ตั้งแต่ [1ms – 100ms] ชุดของพารามิเตอร์ที่สร้างขึ้นจะถูกส่งให้กับ โมดูลถอดรหัสในแต่ละเฟรม ของเสียง โดยที่ฝั่งเข้ารหัสต้องรอสัญญาณเสียงอย่างน้อย 1เฟรม ก่อน ถึงจะสามารถเข้ารหัสเฟรมข้อมูลได้ ซึ่งก็หมายความว่ามีการดีเลย์ของระบบเกิดขึ้น ใน G.729 นั้น 1 เฟรม ที่พิจารณาใช้เวลา 10 ms หรือ 80 ตัวอย่าง (Sampling rate 8Khz) โดยจะแบ่งย่อยเป็น 2 เฟรมย่อย (5ms / เฟรมย่อย) ในส่วนของ พารามิเตอร์ของวงจรกรองจะคำนวณ 1 ครั้งต่อเฟรม ในขณะที่สัญญาณกระตุ้นจะหาแต่ละเฟรมย่อย ซึ่งในส่วนของสัญญาณกระตุ้นเป็นส่วนของสัญญาณอินพุท สำหรับสังเคราะห์เสียงประกอบด้วยสัญญาณเสียงแบบเป็นคาบ และ สัญญาณเสียงแบบสุ่ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

CELP Model ทำงานได้ดีกับเสียงที่เป็นคาบเนื่องจากเสียงที่เป็นคาบเป็นส่วนสำคัญที่ใช้พิจารณาในการหาสัมประสิทธิ์ของวงจรกรองแต่ในส่วนของสัญญาณกระตุ้น นั้นได้ทำการหาทั้ง 2 องค์ประกอบดังนั้นแบบจำลองชนิดนี้จึงสามารถใช้ได้กับทั้ง 2 กรณี

สำหรับวิธีการในการหาพารามิเตอร์ของวงจรกรองและสัญญาณกระตุ้น นั้นเรียกว่าการวิเคราะห์จากการสังเคราะห์ (analysis by synthesis) ฝั่งเข้ารหัสจะทำการ ค้นหาหาพารามิเตอร์ที่เหมาะสมโดยทำการถอดรหัสกลับในแต่ละรอบของการค้นหาโดยนำสัญญาณที่สังเคราะห์ได้มาเปรียบเทียบกับสัญญาณต้นฉบับ พารามิเตอร์ชุดที่ใกล้เคียงกับสัญญาณต้นฉบับมากที่สุดจะถูกเลือกและส่งให้กับฝั่งถอดรหัส

เนื่องจากเราพิจารณาสัญญาณเสียงในช่วงเวลาสั้น ๆ เราจึงสามารถสร้างแบบจำลองในการกำเนิดเสียงในลักษณะของระบบที่ค่าสถิติทางเวลาคงที่ (Stationary system) จากคุณสมบัติของระบบดังกล่าวเราใช้วิธีประมาณความสัมพันธ์ระหว่างค่าของสัญญาณก่อนหน้ากับค่าของสัญญาณปัจจุบันโดยใช้วิธีการประมาณพันธะเชิงเส้น (Linear Prediction) เป็นค่าสัมประสิทธิ์ของวงจรกรองตามรูปที่ 2.4



Prediction of x(n) : $\hat{x}(n) = -\sum_{k=1}^{M} a_k x(n-k)$ (2.8)

Prediction error : $e(n) = x(n) - \hat{x}(n) = x(n) + \sum_{k=1}^{M} a_k x(n-k)$ (2.9)

เมื่อ a_k คือสัมประสิทธิ์ที่ได้จากการประมาณพันธะเชิงเส้น (LP coefficient)

รูปที่ 2-7 แสดงสมการที่ใช้แสดงความสัมพันธ์ในการประมาณค่าของสัญญาณโดยใช้วิธีการของการประมาณพันธะเชิงเส้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
มาตรฐานการเข้ารหัสสัญญาณ G.729

3.1 สัมประสิทธิ์ของวงจรกรอง (Filter coefficient)

ใน G.729 นั้นวงจรกรองที่ใช้เป็นแบบ 10^{th} order all-pole และเนื่องจากวงจรกรองชุดนี้ใช้ในการสังเคราะห์เสียงตัววงจรกรองจึงถูกเรียกว่าวงจรสังเคราะห์ (synthesis filter) ในส่วนของวงจรกรองวิเคราะห์นั้นจะใช้ส่วนกลับของวงจรกรองชนิด FIR ซึ่งเมื่อผ่านเสียงเข้าไปยังวงจรกรองวิเคราะห์ ก็จะได้สัญญาณกระตุ้นสำหรับเสียงนั้นออกมา ที่จริงแล้ววงจรกรองวิเคราะห์ก็คือสัญญาณผิดผลาด (error signal) ในการทำนายเสียงจากเสียงก่อนหน้า 10 ตัวอย่าง ซึ่งสามารถมองเป็นตัวพยากรณ์เชิงเส้น (linear predictor) ได้ในการแก้สมการตรง ๆ นั้นทำได้ลำบากจึงอาศัยวิธีการหาความสัมพันธ์จากคอร์รีเลชัน (correlation)เมทริกซ์ ของสัญญาณเสียง

ในการหาความสัมพันธ์ของสัญญาณในแต่ละเฟรม นั้นจะทำการพิจารณาช่วงของสัญญาณใน 30 ms ประกอบด้วย 15 ms จากอินพุทของเฟรม ก่อนหน้า 10 ms ของเฟรม ปัจจุบันและเฟรมล่วงหน้า อีก 5 ms ซึ่งทำให้เกิด อัลกอริทึมดีเลย์ที่ 15 ms เราจะวินโดว์ (window)สัญญาณก่อนด้วยฟังก์ชันแฮมมิ่งวินโดว์ (Hamming window) และควอเตอร์โคไซน์ (quarter cosine) แทนที่จะคำนวณการหาคอร์รีเลชันโดยตรงจากสัญญาณอินพุทที่เข้ามา เพื่อช่วยปรับรูปร่างของสัญญาณให้การหาคอร์รีเลชันเน้นส่วนที่พิจารณาให้อยู่ในส่วนของเฟรมอินพุทปัจจุบันแทนที่จะไปให้น้ำหนักกับเฟรม ก่อนหน้าหรือในอนาคตมากกว่า อีกหนึ่งขั้นตอนก่อนการหาคอร์รีเลชันเนื่องจากเสียงที่มีพิตท์ที่มีความถี่สูงเป็นองค์ประกอบ (เช่นเสียงผู้หญิง) มีสเปกตรอลเอนวิโลป (spectral envelope) สูงกว่าของพิตท์ความถี่ต่ำที่ถูกมอดูเลทเข้ามาด้วย ทำให้การประมาณหาเอนวิโลปที่ความถี่ระหว่างคาบของพิตท์ (pitch periods) นั้นต่ำไป การแก้ไขในจุดนี้ทำโดยการทำคอนโวลูชันค่าความถี่ (convolution frequency)ของสเปกตรอลเอนวิโลปโดย Gaussian ฟังก์ชัน เพื่อขยายช่วงค่าสูงสุดของสเปกตรอลเอนวิโลปเพื่อเติมช่องว่างในส่วนนั้น

สัมประสิทธิ์ของวงจรกรองสังเคราะห์ LP อันดับที่ 10^{th} สามารถคำนวณหาได้โดยอาศัยคอร์รีเลชัน$r(k)$ จากสมการของ Yule-Walker โดยใช้วิธีแก้ปัญหาแบบทำซ้ำโดยอัลกอริทึมของ Lervinson-Durbin

$$\sum_{i=1}^{10} a_i r(|i-k|) = -r_k \qquad (3.1)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในขั้นตอนถัดไปก็จะทำการควอนไทด์วงจรกรอง แต่เนื่องจากเราไม่สามารถควอนไทด์มันโดยตรงได้เพราะ

1. เป็นไปได้ว่าควอนไทด์noise อาจจะให้รากคำตอบไม่สามารถหาค่ากลับได้ทำให้มีปัญหาเรื่องความไม่เสถียร์ของวงจรกรอง
2. มันยากในการควอนไทด์สัญญาณรบกวนของสัมประสิทธ์ที่ได้รับมาเนื่องจากการรับรู้สัญญาณรบกวนของคนเรามีพื้นฐานจากโดเมนเชิงความถี่

เพื่อแก้ปัญหา 2 ข้อนี้ค่าสัมประสิทธ์จะถูกเปลี่ยนไปอยู่ใน Line Spectral Frequency หรือ LSF's โดยอาศัยสมการ

$$F_1(z) = A(z) + z^{-11}A(z^{-1})$$
$$F_2(z) = A(z) - z^{-11}A(z^{-1}) \qquad (3.2)$$

LSF's เป็นรากคำตอบของโพลิโนเมียลเหล่านี้ซึ่งมีคุณสมบัติที่สำคัญดังนี้

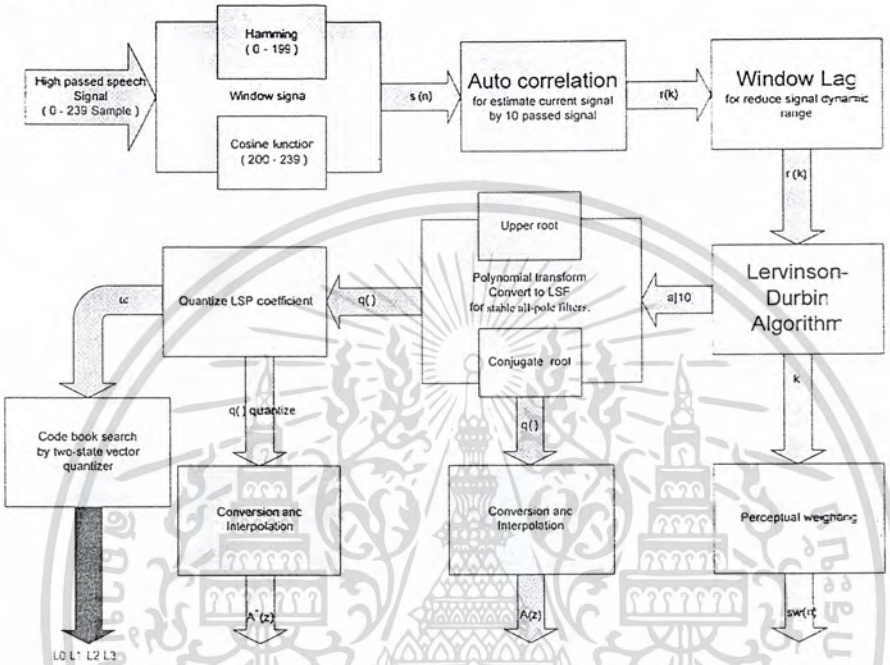
1. รากคำตอบอยู่บนวงกลมหนึ่งหน่วย
2. มีรากคำตอบเป็นคอนจูเกต (conjugate) ซึ่งกันและกัน (แสดงว่าสามารถหาส่วนกลับของวงจรกรองได้)
3. สำหรับคู่ของ โพลิโนเมียล ที่เป็นไปตาม 2 ข้อข้างต้นทำให้วงจรกรอง A(z) เป็นหนึ่งหน่วย (จากข้อ 1) และ วงจรสังเคราะห์กลับ (Inverse synthesis filter) เสถียร์
4. การเปลี่ยนแปลง LSF ใด ๆ ทำให้เกิดการเปลี่ยนแปลงวงจรกรองวิเคราะห์ (analysis filter) ในช่วงความถี่เล็ก ๆ รอบ ๆ ความถี่ LSF นั้น

จากคุณสมบัติข้อ 3 ทำให้ตัวถอดรหัสง่ายต่อการระบุว่าวงจรกรองนั้นเสถียรหรือไม่และจากคุณสมบัติข้อ 4 ช่วยให้การควอนไทด์เซชัน LSF เกี่ยวข้องกับ frequency response ของ วงจรสังเคราะห์ เพื่อลด bandwidth ในการส่งข้อมูลฝั่งเข้ารหัสและถอดรหัสจะใช้การพยากรณ์ค่าของ LSF's โดยใช้ 4th order moving average โดยจะมีชุดของ predictor ที่ใช้เลือกอยู่ 2 ชุด แล้วอาศัยบิทในการเลือกว่าจะใช้เวกเตอร์ ควอนไทด์เซชันชุดใหน การทำเวกเตอร์ ควอนไทด์เซชันแบ่งออกเป็น 2 สถานะโดยสถานะที่หนึ่ง 10-dimension codebook ประกอบด้วย 128 เอนทรีสำหรับค้นหา โดยเอนทรีที่ให้ผลลัพธ์ของการมินิไมซ์ค่ารากที่สองค่าเฉลี่ยของความผิดผลาด (mean square error) ระหว่างค่า LSF's จริงและสถานะที่สอง 10-dimensionเวกเตอร์ จะถูกลบออกจากค่าจริง (Original LSF's) ซึ่งค่าผลต่างนี้จะถูกแบ่งออกเป็นสอง 5-dimensionเวกเตอร์ โดยเอนทรีที่ให้ค่าที่ตรงกับเวกเตอร์ ชุดแรก (แทน LSF 1-5) จะถูกเลือกจาก codebook ชุดที่ 2 (โดยอาศัยการมินิไมซ์ค่ารากที่สองค่าเฉลี่ยของความผิดผลาดเช่นเดียวกับสถานะแรก) และเอนทรีที่ math กับเวกเตอร์ ชุดที่สอง (แทน LSF 6-10) ก็จะถูกเลือกออกมาจาก codebook ชุดที่ 3 ใน codebook ชุดที่ 2 และ 3 นี้ เป็นแบบ 5 – dimension ซึ่งมี 32 เอนทรีเช่นกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วิธีการหาแบบ two state structure นี้เรียกว่า Conjugate structure และ แทน "CS" ในชื่อของ coder สังเกตว่าเราจะส่งข้อมูลในการควอนไทค์เซชันไป 7 บิทเพื่อเลือก codebook ตัวแรก อีก 10 บิทสำหรับ เลือก codebook 32 เอนทรี 2 ชุด และอีก 1 บิทในการเลือกฟังก์ชัน moving average ที่ใช้

ในฝั่งถอดรหัสค่า LSF's ในเฟรมย่อย แรกจะทำการประมาณค่ากับเฟรม ที่แล้วแต่ในเฟรมย่อย ที่สอง จะใช้ค่าจริงที่ได้รับไปถอดรหัส



รูปที่ 3-1 แสดงโฟลโปรแกรมของการหาสัมประสิทธิ์ของวงจรกรอง

3.1.1 ขั้นตอนการหาสัมประสิทธิ์วงจรกรอง โดยละเอียด

3.1.1.1 เตรียมสัญญาณก่อนการประมวลผล (Pre-processing)

ประกอบด้วย 2 ฟังก์ชันที่สำคัญ คือ

1 Signal scaling ประกอบด้วยการหารด้วย 2 เพื่อป้องกันการเกิดโอเวอร์โฟล (overflow) ในกระบวนการทำ fixed point

2 วงจรกรองความถี่สูงเพื่อป้องกันสัญญาณที่มีความถี่ต่ำที่ไม่ต้องการ ตามสมการ

$$H_{h1}(z) = \frac{0.46363718 - 0.9272470z^{-1} + 0.46363718z^{-2}}{1 - 1.9059465z^{-1} + 0.9114024z^{-2}} \qquad (3.3)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งเป็นสมการใน z โดเมน เมื่อแปลง z กลับจะได้สมการใน โดเมนของเวลาดังนี้

$$y(n) = 0.46363718x(n) - 0.92724705x(n-1) + 0.46363718x(n-2) - 1.9059465y(n-1) + 0.9114024y(n-2) \quad (3.4)$$

s (n) แทน y (n) ของสัญญาณที่ผ่านขั้นตอนนี้แล้ว

3.1.1.2 วิเคราะห์หาสัมประสิทธิ์ LP และควอนไทด์ซชัน (Linear prediction analysis and quantization)

Short-term analysis และวงจรสังเคราะห์ มีพื้นฐานมาจากวงจรกรองลิเนียร์พรีดิคชันลำดับที่ 10 (10 th Order Linear Prediction (LP) filter) โดยจะถูกทำ 1 ครั้งทุก ๆ 1 เฟรม โดยใช้วิธีการออโตคอร์รีเลชัน (autocorrelation method) ซึ่งเป็นวินโดว์ที่ไม่สมมาตร (asymmetric window) 30 ms ทุก ๆ ค่าสัมประสิทธิ์ 80 ตัวอย่าง ของออโตคอร์รีเลชันของวินโดว์ของสัญญาณเสียงถูกคำนวณและเปลี่ยนกลับไปเป็นค่าสัมประสิทธิ์ LP โดยใช้อัลกอริทึมของ Levinson หลังจากนั้นค่าสัมประสิทธิ์ LP จะถูกเปลี่ยนไปเป็น LSP โดเมนสำหรับการประมาณและการทำควอนไทด์เซชันและ การประมาณที่ได้จากวงจรกรองชนิดควอนไทด์และ ไม่ควอนไทด์ จะถูกแปลงกลับไปเป็นค่าสัมประสิทธิ์วงจรกรอง LP อีกครั้ง เพื่อทำการสร้างและถ่วงน้ำหนัก สำหรับแต่ละ เฟรมย่อย

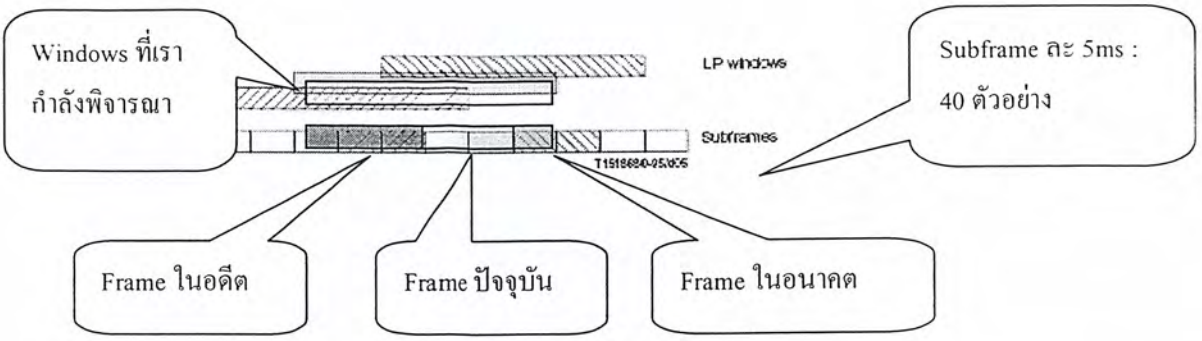
3.1.1.3 วินโดว์และการหาออโต้คอร์รีเลชัน (Windowing and autocorrelation computation)

LP analysis window ประกอบด้วย 2-ส่วน- โดยส่วนแรกเป็นฟังก์ชันแฮมมิ่งวินโดว์ และส่วนที่ 2 เป็น ¼ ของรอบของฟังก์ชันโคไซน์ (Cosine function)

$$w_{lp}(n) = \begin{cases} 0.54 - 0.46\cos\left(\dfrac{2\pi n}{399}\right) & n = 0,\ldots,199 \\ \cos\left(\dfrac{2\pi(n-200)}{159}\right) & n = 200,\ldots,239 \end{cases} \quad (3.5)$$

โดย 5 ms ที่เพิ่มขึ้นมาใน LP analysis หมายความว่า 40 ตัวอย่าง จะต้องทำมาจาก เฟรม ในอนาคต ซึ่งการแปลงนี้จะทำให้เกิดดีเลย์ขึ้น 5 ms ในขณะทำการเข้ารหัส ดังนั้น LP analysis window ต้องการ 120 ตัวอย่าง จากเฟรมในอดีต 80 ตัวอย่าง จากเฟรมในปัจจุบันที่กำลังพิจารณา และ 40 ตัวอย่างจากเฟรมในอนาคต รวมเป็น 240 ตัวอย่างและคำนวณหา $s'(n)$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-2 แสดงเฟรม ของสัญญาณอินพุทที่ใช้ในการคำนวณ

สำหรับความสำคัญที่เราใช้ในการประมวลผลสัญญาณนั้นเราให้ไม่เท่ากันในแต่ละ ตัวอย่าง จึงต้องมีการคูณด้วยค่าถ่วงน้ำหนัก ซึ่งในที่นี้คือ ค่าของ window ในตำแหน่งที่ตรงกับตัวอย่างนั้น ๆ

$$s'(n) = w_{lp}(n)s(n) \qquad n = 0,...,239 \tag{3.6}$$

ใช้ $s'(n)$ ในการหาสัมประสิทธิ์ออโตคอร์รีเลชัน $r(k)$

$$r(k) = \sum_{n=k}^{239} s'(n)s'(n-k) \qquad k = 0,...,10 \tag{3.7}$$

เพื่อป้องกันการเกิดปัญหาค่าทางคณิตศาสตร์สำหรับ $r(0)$ จะมีค่าต่ำกว่าขอบเขตที่กำหนด $r(0) = 1.0$ A ย่านของสัญญาณการขยาย 60 Hz จะถูกคูณเข้ากับ $r(k)$

$$w_{lag}(k) = \exp\left[-\frac{1}{2}\left(\frac{2\pi f_0 k}{f_s}\right)^2\right] \qquad k = 1,..,10 \tag{3.8}$$

ซึ่งแทนค่า f0 = 60 Hz และ fs ด้วย 8000 Hz ซึ่งเป็นความถี่ในการทำสุ่มตัวอย่างและ $r(0)$ จะต้องทำการคูณด้วยอัตราส่วนของ white noise correction คือ 1.0001 ซึ่งเทียบเท่ากับการบวกด้วยสัญญาณรบกวนที่ –40 dB ดังนั้น $r'(k)$ จะกลายเป็นออโตคอร์รีเลชันที่ถูกแก้ไขแล้ว

$$\begin{aligned} r'(0) &= 1.0001r(0) \\ r'(k) &= w_{lag}(k)r(k) \qquad k = 1,...,10 \end{aligned} \tag{3.9}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

3.1.1.4 อัลกอริทึมของเลอวินสัน และเดอบิน (Levinson-Durbin Algorithms)

สมการที่แสดงความสัมพันธ์ระหว่างสัมประสิทธิ์ของออโตคอร์รีเลชันที่ถูกปรับปรุง $r'(k)$ กับสัมประสิทธิ์ LP a_i $i = 1,\ldots,10$

$$\sum_{i=1}^{10} a_i r'(|i-k|) = -r'(k) \qquad k = 1,\ldots,10 \tag{3.10}$$

สำหรับวิธีการแก้สมการหานั้นใช้ความสัมพันธ์แบบรีเคอร์ซีฟดังต่อไปนี้

$$\begin{aligned}
&E^{[0]} = r'(0) \\
&for \quad i = 1 \quad to \quad 10 \\
&\qquad a_0^{[i-1]} = 1 \\
&\qquad k_i = -\left[\sum_{j=0}^{i-1} a_j^{[i-1]} r'(i-j)\right] / E^{[i-1]} \\
&\qquad a_i^{[i]} = k_i \\
&\qquad for \quad j = 1 \quad to \quad i-1 \\
&\qquad\qquad a_j^{[i]} = a_j^{[i-1]} + k_i a_{i-j}^{[i-1]} \\
&\qquad end \\
&\qquad E^{[i]} = \left(1 - k_i^2\right) E^{[i-1]} \\
&end
\end{aligned} \tag{3.11}$$

สำหรับคำตอบที่ได้จะเท่ากับ $a_j = a_j^{[10]}$, $j = 0,\ldots,10$ โดย $a_0 = 1.0$

3.1.1.5 การแปลงจากโดเมนของ LP ไปยัง LSP (LP to LSP conversion)

สัมประสิทธิ์ของ LP a_i $i = 1,\ldots,10$ จะถูกเปลี่ยนเป็นสัมประสิทธิ์ LSP สำหรับเพื่อทำควอนไทด์เซชันและ การประมาณค่า สำหรับวงจรกรอง LP ลำดับที่ 10 (10 th Order LP filter) นั้น สัมประสิทธิ์ LSP จะถูกนิยามอยู่ในรูปของผลรวมของรากของสมการและผลต่างของ โพลิโนเมียล

61521

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$F_1'(z) = A(z) + z^{-11}A(z^{-1})$$
$$and: \quad (3.12)$$
$$F_2'(z) = A(z) - z^{-11}A(z^{-1})$$

โดยโพลิโนเมียล $F'_1(z)$ เป็น symmetric และโพลิโนเมียล $F'_2(z)$ เป็น anti symmetric จะสามารถพิสูจน์ได้ว่ารากทุกรากของสมการจะอยู่บนวงกลมหนึ่งหน่วยและเป็นทางเลือก (altemate) ซึ่งกันและกัน $F'_1(z)$ มีรากที่ $z = -1$ และ $F'_2(z)$ มีราก $z = 1$ ซึ่งจะทำให้กำจัดเทอมได้และ นิยามโพลิโนเมียลใหม่อีก 2 ตัว

$$F_1(z) = F_1'(z)/(1 + z^{-1})$$
$$and: \quad (3.13)$$
$$F_2(z) = F_2'(z)/(1 - z^{-1})$$

ซึ่งแต่ละสมการจะมี 5 รากที่เป็นคอนจูเกตซึ่งกันและกันบน วงกลมหนึ่งหน่วย ซึ่งสามารถเขียนอยู่ในรูปของผลคูณได้

$$F_1(z) = \prod_{i=1,3,\ldots,9}(1 - 2q_i z^{-1} + z^{-2})$$
$$and: \quad (3.14)$$
$$F_2(z) = \prod_{i=2,4,\ldots,10}(1 - 2q_i z^{-1} + z^{-2})$$

โดย q_i ตามสมการนั้นแทนด้วย $\cos(\omega_i)$ ซึ่ง ω_i แต่ละพจน์ เรียกว่า Line Spectral Frequencies (LSF) และเรียงลำดับ $0 < \omega_1 < \omega_2 < \prec \omega_{10} < \prod$ โดย q_i แต่ละพจน์นั้นหมายถึงสัมประสิทธิ์ LSP ในโดเมนของโคไซน์เพราะว่าโพลิโนเมียล $F_1(z)$ และ $F_2(z)$ นั้นสมมาตร (symmetric) เฉพาะสัมประสิทธิ์ ห้าพจน์แรกซึ่งต้องการที่จะทำการคำนวณ โดย สัมประสิทธิ์ ดังกล่าวนั้นจะสามารถหาได้จากความสัมพันธ์แบบ รีเคอร์ซีฟ

$$f_1(i+1) = a_{i+1} + a_{10-i} - f_1(i) \qquad i = 0,..,4$$
$$f_2(i+1) = a_{i+1} - a_{10-i} + f_2(i) \qquad i = 0,..,4 \quad (3.15)$$

โดยกำหนดให้ $f_1(0) = f_2(0) = 1.0$ สัมประสิทธิ์ LSP นั้นจะหาเจอเมื่อมีการหาค่าของ $F_1(z)$ และ $F_2(z)$ แบ่งช่วงระหว่าง $0 - \prod$ ออกเป็น 60 จุดและทำการตรวจสอบเครื่องหมาย เมื่อมีการเปลี่ยนแปลงเครื่องหมายแสดงว่า เกิดรากขึ้นที่ระหว่างจุด ๆ นั้น และทำการแบ่งพื้นที่ นั้นลงไปอีก 4 ส่วนจะทำให้ได้ค่าประมาณของรากที่ดี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขั้นโพลิโนเมียลของ Chebyshev ถูกใช้ในการหาค่า $F_1(z)$ และ $F_2(z)$ ซึ่งวิธีนี้จะหารากได้เป็นโดเมนของโคไซน์

$$F(\omega) = 2e^{-j5\omega}C(x)$$
$$with: \quad C(x) = T_5(x) + f(1)T_4(x) + f(2)T_3(x) + f(3)T_2(x) + f(4)T_1(x) + f(5)/2 \tag{3.16}$$

โดย $T_m(x) = \cos(m\omega)$ และ m เป็นอันดับของโพลิโนเมียลของ Chebyshev และ f(i) เป็นสัมประสิทธิ์ของ $F_1(z)$ และ $F_2(z)$ โพลิโนเมียล C(x) จะถูกหาค่าได้ค่าที่ $x = \cos(\omega)$ โดยใช้ความสัมพันธ์แบบรีเคอร์ซีฟ

$$\begin{aligned} &for \quad k = 4 \quad down \quad to \quad 1 \\ &\qquad b_k = 2xb_{k+1} - b_{k+2} + f(5-k) \\ &end \\ &C(x) = xb_1 - b_2 + f(5)/2 \end{aligned} \tag{3.17}$$

$$init: \qquad b_5 = 1 \quad and \; b_6 = 0$$

3.1.1.6 การควอนไทด์สัมประสิทธิ์ LSP (Quantization of the LSP coefficient)

สัมประสิทธิ์ LSP q_i นั้นถูกควอนไทด์โดยใช้ LSF ω_i และ Switched 4 th order MA นั้นถูกใช้เพื่อการพยากรณ์ค่าของสัมประสิทธิ์ LSF ของเฟรม ปัจจุบัน โดยผลต่างระหว่างค่าที่คำนวณได้กับการพยากรณ์สัมประสิทธิ์จะถูกควอนไทด์โดยใช้ two stage vector quantizer โดยสถานะแรก จะเป็น VQ 7 บิท โดยใช้ L1 สำหรับ 128 เอนทรี ส่วนสถานะที่สองจะเหลือ 10 บิทแบ่งออกเป็น VQ 5 บิท2 ส่วน แทน L2 และ L3 ตามลำดับ ซึ่ง 5 บิทนั้นแทน 32 เอนทรี

ในกระบวนการถอดรหัสสัมประสิทธิ์ที่ได้รับนั้นจะอยู่ในรูปของผลรวมของทั้ง 2-codebook

$$\hat{l}_i = \begin{cases} L1_i(L1) + L2_i(L2) & i = 1,2,..,5 \\ L1_i(L1) + L3_{i-5}(L3) & i = 6,7,..,10 \end{cases} \tag{3.18}$$

โดย L1 L2 และ L3 เป็นอินเด็กของ codebook เพื่อลดการเกิด sharp resonance ในการทำควอนไทด์วงจรสังเคราะห์ LP โดยสัมประสิทธิ์ของควอนไทเซอร์เอาท์พุท (Quantizer output) จะต้องถูกจัดเรียงจนกระทั่งสัมประสิทธิ์ที่อยู่ติดกันจะมีระยะทางน้อยกว่า J โดยวิธีการจัดเรียงตามอัลกอริทึม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$
\begin{aligned}
&for \quad i = 2 \quad to \quad 10 \\
&\quad if (\hat{l}_{i-1} > \hat{l}_i - J) \\
&\quad\quad \hat{l}_{i-1} = (\hat{l}_i + \hat{l}_{i-1} - J)/2 \\
&\quad\quad \hat{l}_i = (\hat{l}_i + \hat{l}_{i-1} + J)/2 \\
&\quad end \\
&end
\end{aligned}
\tag{3.19}
$$

การจัดเรียงจะต้องทำสองครั้ง โดยครั้งแรก J = 0.0012 และ J = 0.0006 หลังจากกระบวนการเรียงแล้วสัมประสิทธิ์ของ LSF ที่ถูกควอนไทด์ (Quantized LSF coefficient) สำหรับเฟรมปัจจุบัน จะสามารถหาได้จากควอนไทด์เซอร์เวกเตอร์ ของ 4 เฟรมก่อนหน้า ดังนี้

$$
\hat{\omega}_i^{(m)} = \left(1 - \sum_{k=1}^{4} \hat{p}_{i,k}\right) \hat{l}_i^{(m)} + \sum_{k=1}^{4} \hat{p}_{i,k}\, \hat{l}_i^{(m-k)} \qquad i = 1,..,10 \tag{3.20}
$$

ซึ่ง $\hat{p}_{i,k}$ คือ switched MA predictor coefficient ซึ่งมี 2 สมการ จะใช้สมการไหนขึ้นอยู่กับบิท L0 โดยค่าเริ่มต้นของ $l_i^{(k)}$ เมื่อ k < 0 มีค่าเท่ากับ $i\prod/11$

หลังจากการทำการหาสัมประสิทธิ์ของ LSF ที่ถูกควอนไทด์แล้ว ต้องตรวจสอบความเสถียรของรากคำตอบ โดยทำขั้นตอนดังนี้

- เรียงลำดับสัมประสิทธิ์ของ LSF ที่ถูกควอนไทด์จากน้อยไปมาก
- ถ้าสัมประสิทธิ์ของ LSF ที่ถูกควอนไทด์มีค่าน้อยกว่า 0.005 กำหนดให้ เท่ากับ 0.005
- ถ้า $\omega_{i+1} - \omega_i - 0.0391$ แล้ว กำหนดให้ $\omega_{i+1} = \omega_i + 0.0391$
- ถ้า $\omega_{10} > 3.135$ แล้ว $\omega_{10} = 3.135$

สำหรับกระบวนการเข้ารหัส LSF นั้น จะมีสมการที่ใช้ในการพยากรณ์ 2 สมการ จะให้ค่าประมาณที่ดีที่สุดของสัมประสิทธิ์ LSF ซึ่งการประมาณที่ดีที่สุดนั้นจะต้องทำให้ weighted mean-squared error (MSE)

$$
E_{lsf} = \sum_{i=1}^{10} w_i (\omega_i - \hat{\omega}_i)^2 \tag{3.21}
$$

สำหรับ weight (w) นั้นหาได้จากฟังก์ชัน ของสัมประสิทธิ์ LSF ที่ไม่ควอนไทด์ ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$w_i = \begin{cases} 1.0 & if \quad \omega_2 - 0.04\pi - 1 > 0 \\ 10(\omega_2 - 0.04\pi - 1)^2 + 1 & otherwise \end{cases}$$

$$w_i \quad 2 \le i \le 9 = \begin{cases} 1.0 & if \quad \omega_{i+1} - \omega_{i-1} - 1 > 0 \\ 10(\omega_{i+1} - \omega_{i-1} - 1)^2 + 1 & otherwise \end{cases} \qquad (3.22)$$

$$w_{10} = \begin{cases} 1.0 & if \quad -\omega_9 + 0.92\pi - 1 > 0 \\ 10(-\omega_9 + 0.92\pi - 1)^2 + 1 & otherwise \end{cases}$$

สำหรับ w_5 และ w_6 นั้นคูณด้วย 1.2

ควอนไทด์เซอร์เวกเตอร์ ของเฟรมปัจจุบัน ได้มาจาก สมการดังต่อไปนี้

$$l_i = \left[\omega_i^{(m)} - \sum_{k=1}^{4} \hat{p}_{i,k} \hat{l}_i^{(m-k)} \right] / \left(1 - \sum_{k=1}^{4} \hat{p}_{i,k} \right) \qquad i = 1,....,10 \qquad (3.23)$$

โดย L1 จะถูกหาและ เอนทรี ของ L1 ที่ทำให้ค่าของรากที่สองค่าเฉลี่ยของความผิดผลาดน้อยที่สุดจะถูกเลือก และ L2 จะถูกหาและเลือกเช่นเดียวกับ L1 แต่จะเป็นส่วนของ 5 บิทล่างของสถานะที่สอง L2 ที่หามาได้จะต้อง และจัดเรียงเพื่อประกันระยะทางสั้นสุดคือ 0.0012 และใช้ L1 และ L2 ที่ถูกเลือกเพื่อหา L3 อีกครั้ง ต้องมีการเรียงเพื่อประกันระยะทางสั้นสุดคือ 0.0012 และเวกเตอร์ L3 จะต้องให้ผลลัพธ์ความผิดพลาดที่น้อยที่สุดควอนไทด์เซอร์เวกเตอร์ จะต้องถูกนำมาเรียงใหม่โดยรับประกันระยะทางสั้นสุด 0.0006 ซึ่ง process นี้เสร็จสิ้นจะต้องทำให้เสร็จสำหรับ MA predictor ทั้งสองสมการ จะทำการหาค่า L0 และ MA predictor จะเลือก L0 ที่ให้ค่าของรากที่สองค่าเฉลี่ยของความผิดผลาดต่ำสุดจะถูกเลือก

3.1.1.7 การประมาณค่าของสัมประสิทธ์ LSP (interpolation of LSP coefficient)

ควอนไทด์(และสัมประสิทธิ์ LP ที่ไม่ควอนไทด์) จะถูกใช้สำหรับเฟรมย่อยที่ 2 สำหรับเฟรมย่อยแรกควอนไทด์ (และสัมประสิทธิ์ LP ที่ไม่ควอนไทด์) จะได้มาจากทำการประมาณค่าเชิงเส้น (Linear interpolation) ซึ่งจะเกี่ยวข้องกับพารามิเตอร์ของเฟรมย่อยที่อยู่ติดกัน กำหนดให้ $q^{(current)}$ เป็นสัมประสิทธิ์ LSP ซึ่งคำนวณได้จากเฟรม 10 ms ปัจจุบัน และ $q^{(previous)}$ เป็น LSP coefficient ซึ่งคำนวณได้จากเฟรม 10 ms ก่อนหน้า สำหรับสัมประสิทธิ์ของ LSP ที่ไม่ควอนไทด์ ของแต่ละ 2 เฟรมย่อย ดังนี้

$$\begin{aligned} Subframe\,1: \quad & \hat{q}_i = 0.5 * \hat{q}_i^{(previous)} + 0.5 * \hat{q}_i^{(current)} && i = 1,2,...,10 \\ Subframe\,2: \quad & \hat{q}_i = \hat{q}_i^{(current)} && i = 1,2,...,10 \end{aligned} \qquad (3.24)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.1.8 การแปลงจากโดเมน ของ LSP ไปยัง LP (LSP to LP conversion)

แต่ละครั้งที่สัมประสิทธิ์ LSP ถูกควอนไทด์ และหาค่า จะต้องถูกเปลี่ยนกลับมาเป็นสัมประสิทธิ์ LP ซึ่งขั้นตอนแสดงความสัมพันธ์แบบรีเคอซีฟตาม อัลกอริทึม

$$
\begin{aligned}
&for \;\; i = 1 \;\; to \;\; 5 \\
&\quad f_1(i) = -2q_{2i-1} f_1(i-1) + 2f_1(i-2) \\
&\quad for \;\; j = i-1 \;\; down \;\; to \;\; 1 \\
&\quad\quad f_1^{[i]}(j) = f_1^{[i-1]}(j) - 2q_{2i-1} f_1^{[i-1]}(j-1) + f_1^{[i-1]}(i-2) \\
&\quad end \\
&end
\end{aligned}
\tag{3.25}
$$

กำหนดให้ $f_1(0) = 1$ และ $f_1(-1) = 0$ และ $f_2(i)$ ก็สามารถคำนวณได้เช่นเดียวกันแต่ต้องแทนค่า q_{2i+1} ด้วย q_{2i}

แต่ละครั้งที่เจอค่าสัมประสิทธิ์ $f_1(i)$ และ $f_2(i)$ $F_1(z)$ และ $F_2(z)$ จะต้องถูกคูณด้วย $(1 + z^{-1})$ และ $(1 - z^{-1})$ เพื่อที่จะได้ $F'_1(z)$ และ $F'_2(z)$ จะได้

$$
\begin{aligned}
f'_1(i) &= f_1(i) + f_1(i-1) \qquad i = 1,..,5 \\
f'_2(i) &= f_2(i) + f_2(i-1) \qquad i = 1,..,5
\end{aligned}
\tag{3.26}
$$

ท้ายที่สุดสัมประสิทธิ์ LP จะถูกคำนวณจาก $f'_1(i)$ และ $f'_2(i)$

$$
a_i = \begin{cases} 0.5 f'_1(i) + 0.5 f'_2(i) & i = 1,...,5 \\ 0.5 f'_1(11-i) + 0.5 f'_2(11-i) & i = 6,...,10 \end{cases}
\tag{3.27}
$$

ซึ่งเป็นวิธีโดยตรง ซึ่งหามาได้จากความสัมพันธ์ $A(z) = (F'_1(z) + F'_2(z)) / 2$ เพราะว่า $F'_1(z)$ และ $F'_2(z)$ นั้นเป็น symmetric และ anti symmetric

3.2 สัญญาณกระตุ้น (Excitation)

ในขั้นตอน ต่อไปจะกล่าวถึงวิธีการที่ใช้หาสัญญาณกระตุ้นซึ่งจะหาสำหรับแต่ละเฟรมย่อย สัญญาณกระตุ้นถูกแทนด้วยผลรวมขององค์ประกอบสองส่วน อย่างที่แรกก็คือ Version ที่ถูกดีเลย์ของสัญญาณกระตุ้นที่ใช้ ในส่วนที่สองคืออิมพัลซ์ (impulse) ในตำแหน่งต่าง ๆ ของสัญญาณ 4 ตำแหน่ง ส่วนประกอบส่วนแรกนั้นเรียกว่า adaptive codebook เป็นแบบจำลองของสัญญาณเสียงที่มีคุณสมบัติเป็นคาบ ซึ่งจริง ๆ แล้วก็คือพิทช์ดีเลย์ ในสัญญาณเสียงนั้น

ในขั้นตอนแรกของการหาพิทช์ดีเลย์นั้น จะหาพิทช์ดีเลย์แบบหยาบ ๆ ก่อนโดยจะนำสัญญาณเสียงมาทำออโตคอร์รีเลชันซึ่งแบ่งช่วงของการทำเป็น 3 ช่วงแล้วเลือกอินเด็ก(delay) ของการทำออโตคอร์รีเลชันที่ให้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

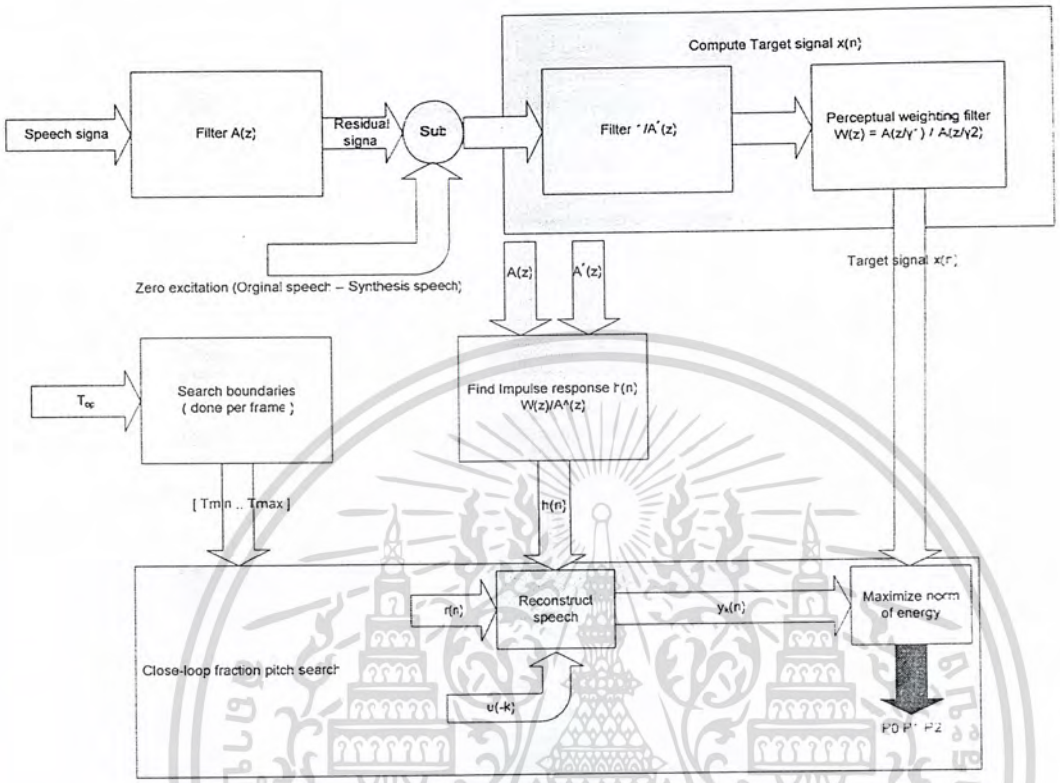
ค่าสูงสุดในแต่ละช่วงมาเพื่อเลือกอีกทีโดยจะให้น้ำหนักกับ Harmonic ที่อยู่ในช่วงต่ำกว่ามากกว่า Harmonic ช่วงสูง เรียกการค้นหาแบบหยาบ ๆ ในขั้นตอนนี้ว่า Open-loop pitch analysis แล้วนำค่าพิตท์แบบหยาบ ๆ ที่ได้มาหาแบบละเอียดรอบ ๆบริเวณนั้น ในขั้นตอนต่อไปโดย การนำสัญญาณกระตุ้นของเฟรม ก่อนหน้ามาผ่านวงจรสังเคราะห์ LP แล้วนำผลลัพธ์ที่ได้มาทำการคอร์รีเลชันกับสัญญาณเสียงจริง หารด้วยขนาดของเอาท์พุทของวงจรสังเคราะห์ (เพื่อไม่ต้องพิจารณาอัตราขยายสัญญาณในขั้นตอนการ search)ดีเลย์ที่ทำให้พลังงานสูงที่สุด ค่าอัตราขยายสัญญาณจะหาได้จากผลลัพธ์ที่ได้เลือกมา เราจะใช้ส่วนที่เหลือจากการนำดีเลย์ที่เหมาะสมถูกเลือกขึ้นมาผ่าน วงจรสังเคราะห์ LP ลบออกจาก สัญญาณอินพุทแล้วนำส่วนที่เหลือนี้ไปใช้หาสัญญาณกระตุ้นส่วนที่เหลือ (unvoiced component) ต่อไป

ในขั้นตอนนี้เรียกว่า fixed codebook contribution โดยสัญญาณกระตุ้นที่เหลือนี้จะถูกแทนด้วยอิมพัลซ์ที่มีแอมพลิจูด $+/_-1$ อยู่ 4 ตัว codebook เวกเตอร์ ที่ถูกเลือกโดยอิมพัลซ์ เหล่านั้นจะถูกเพิ่มองค์ประกอบของ Harmonic เพื่อเพิ่มคุณภาพของสัญญาณเสียงที่จะสังเคราะห์ขึ้นมาโดยวงจรกรองแบบฮาร์โมนิก (Harmonic filter) ในการค้นหานั้นจะได้ตำแหน่งของพัลซ์ และเครื่องหมาย โดยจะเลือกผลลัพท์ของเมทริกซ์ที่ค้นหาที่ให้ค่ามากที่สุดของการกระตุ้นวงจรกรองด้วยสัญญาณกระตุ้นไปยังวงจรสังเคราะห์ คูณด้วย target signal (speech input ส่วนที่เหลือจากการหา Open-loop) หารด้วยขนาดพลังงาน ของเอาท์พุท จากวงจรสังเคราะห์ (เพื่อไม่ต้องพิจารณาอัตราขยายสัญญาณในขั้นตอนการ search) หลังจากนั้นจึงหาอัตราขยายสัญญาณจากสัญญาณกระตุ้นที่ได้เลือกไว้

ในขั้นนี้แต่ละเฟรมย่อย ได้หา พารามิเตอร์พิทช์ดีเลย์, อัตราขยายสัญญาณ adaptive codebook. สัญญาณกระตุ้นของ fixed codebook และอัตราขยายสัญญาณ ซึ่งพารามิเตอร์เหล่านี้จะถูกควอนไทต์และส่งไปให้กับฝั่งถอดรหัสในเฟรมย่อยแรกพิทช์ดีเลย์จะถูกส่งออกไปปรกติ (8 บิท) แต่ในเฟรมย่อยที่ 2 นั้นจะใช้ค่า relative จากเฟรมย่อยแรกทำให้ใช้เพียง (5 บิทในการส่ง) ส่วนของ fixed codebook นั้นก็ทำการส่งออกไปตรง ๆ โดยใช้ 4 บิทสำหรับเครื่องหมายและ 13 บิทสำหรับการระบุตำแหน่งของพัลซ์ แต่ในส่วนของอัตราขยายสัญญาณนั้น (อัตราขยายสัญญาณ adaptive codebook และ fixed codebook) ใช้การทำนายจากเฟรม ก่อนหน้าโดยอาศัยองค์ประกอบอัตราขยายสัญญาณ (gain factor) ซึ่งเป็นความสัมพันธ์ของอัตราขยายสัญญาณทั้งสองในรูปแบบของผลคูณเพื่อทำนายความผิดพลาดโดยองค์ประกอบอัตราขยายสัญญาณ และอัตราขยายสัญญาณ fixed codebook ใช้การรวมความสัมพันธ์แล้ว ควอนไทต์แบบ two stage vectorควอนไทต์เซชัน นสถานะแรกประกอบด้วย 3 บิท two dimension codebook หนึ่งชุด สถานะถัดไปประกอบดว้ย 4 บิท two dimension หนึ่งชุด ซึ่งผลรวมของ codebook ทั้งคู่จะแทนองค์ประกอบอัตราขยายสัญญาณ และอัตราขยายสัญญาณ fixed codebook ที่ใช้ในการส่งออกไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.1 การหาองค์ประกอบสัญญาณเสียงที่เป็นคาบ (Voice sound)



รูปที่ 3-3 แสดงวิธีการหาสัญญาณกระตุ้นที่มีลักษณะของสัญญาณเป็นคาบ (Voice sound)

3.2.1.1 การปรับลักษณะของสัญญาณ (Perceptual weighting)

Perceptual weighting นั้นมีพื้นฐานมาจากสัมประสิทธิ์ของ LP ที่ไม่ควอนไทด์

$$W(z) = \frac{A(z/\gamma 1)}{A(z/\gamma 2)} = \frac{1+\sum_{i=1}^{10}\gamma_1^i a_i z^{-i}}{1+\sum_{i=1}^{10}\gamma_2^i a_i z^{-i}} \qquad (3.28)$$

โดยค่าของ γ_1 และ γ_2 นำไปสามารถนำไปหาผลของการตอบสนองเชิงความถี่ของวงจรกรอง W(z) สำหรับการปรับที่ค่าของ γ_1 และ γ_2 เหมาะสมจะทำให้การทำถ่วงน้ำหนักเกิดประสิทธิภาพมาก ซึ่งจะต้องทำให้ γ_1 และ γ_2 นั้นเป็นฟังก์ชัน spectral แบบเดียวกับสัญญาณอินพุทซึ่งการทำนี้จะทำเพียงครั้งเดียวใน 10 ms และมีการประมาณในเฟรมย่อยแรกของแต่ละเฟรมจะทำให้การปรับสัญญาณให้เรียบขึ้น สำหรับฟังก์ชัน Spectral นั้นจะสามารถหาได้จากวงจรกรองแบบการทำนายเชิงเส้นอันดับที่สอง (2^{nd} order

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

linear prediction) ซึ่งได้มาจากผลพลอยได้ของ Lervinson – Durbin recursion โดยสัมประสิทธิ์การสะท้อน (reflection coefficient) k ถูกเปลี่ยนไปเป็นสัมประสิทธิ์ของอัตราส่วน Log Area (LAR) o

$$o_t = \log\frac{(1.0+k_t)}{(1.0-k_t)} \qquad i = 1,2 \tag{3.29}$$

ค่าของสัมประสิทธิ์ LAR ในเฟรมปัจจุบัน ใช้ในเฟรมย่อยที่ 2 สัมประสิทธิ์ LAR สำหรับเฟรมย่อย แรกนั้น ค่าที่ใช้แทนนั้นหาได้จากการประมาณค่าจากเฟรมก่อนหน้ากับเฟรมปัจจุบัน

$$\begin{aligned} Subframe1: \quad & o_t^{(1)} = 0.5o_t^{(previous)} + 0.5o_t^{(current)} \qquad i = 1,2 \\ Subframe2: \quad & o_t^{(2)} = o_t^{(current)} \qquad i = 1,2 \end{aligned} \tag{3.30}$$

โดยคุณสมบัติของ Spectral Envelope ของสัญญาณ แบ่งออกเป็น flat (flat = 1) หรือ tilt (flat = 0) สำหรับแต่ละเฟรมย่อย คุณสมบัติเหล่านี้ขึ้นอยู่กับสัมประสิทธิ์ LAR ดังสมการ

$$flat^{(m)} = \begin{cases} 0 & if \quad o_1^{(m)} < -1.74 \quad and \quad o_2^{(m)} > 0.65 \quad and \quad flat^{(m-1)} = 1 \\ 1 & if \quad (o_1^{(m)} > -1.52 \quad or \quad o_2^{(m)} < 0.43) \quad and \quad flat^{(m-1)} = 1 \\ flat^{(m-1)} & otherwise \end{cases} \tag{3.31}$$

ดังที่กล่าวมาข้างต้นว่า γ_1 และ γ_2 มีค่าขึ้นอยู่กับฟังก์ชัน Spectrum ดังนั้น ค่าของ γ_1 และ γ_2 กำหนด ดังนี้

- flat (flat = 1) weight factor γ_1 จะเท่ากับ 0.94 และ γ_2 จะเท่ากับ 0.6
- tilted (flat = 0) weight factor γ_1 จะเท่ากับ 0.98 และ γ_2 จะปรับเปลี่ยนโดยขึ้นอยู่กับความแข็งของ resonance ใน วงจรสังเคราะห์ LP แต่อยู่ในช่วง 0.4 – 0.7 ถ้า แข็งแรงมากมาก γ_2 ก็จะเข้าใกล้ 0.7 การปรับนี้จะมากหรือน้อยโดยขึ้นอยู่กับ ระยะทางที่น้อยที่สุดระหว่างสัมประสิทธิ์ LSP ของเฟรมนั้น

$$d_{\min} = \min[\omega_{i+1} - \omega_i] \qquad i = 1,...,9 \tag{3.32}$$

โดยค่าของ γ_2 จะมีความสัมพันธ์กับ d ดังนี้

$$\gamma_2 = -6.0d_{\min} + 1.0 \qquad bounded \quad by \quad 0.4 \le \gamma_2 \le 0.7 \tag{3.33}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Weighted speech signal (sw (n)) หาได้ตามสมการ ดังต่อไปนี้

$$sw(n) = s(n) + \sum_{i=1}^{10} a_i \gamma_1^i s(n-i) - \sum_{i=1}^{10} a_i \gamma_2^i s(n-i) \qquad n = 0,...,39 \tag{3.34}$$

3.2.1.2 การหาค่าพิตท์อย่างหยาบ (Open loop pitch analysis)

เพื่อที่จะลดความซับซ้อนในการหาสำหรับดีเลย์ของ Adaptive codebook (Pitch) ดังนั้นจุดในการหาจะถูกกำหนดให้อยู่รอบ ๆ พิตท์คร่าว ๆ จากการทำ Open loop pitch analysis ซึ่งการทำขั้นตอนนี้จะทำเพียงครั้งเดียวต่อเฟรม ซึ่งจะต้องใช้หลักการของความเหมือนกัน (correlation) ของสัญญาณเป็นคาบ ๆ

$$R(k) = \sum_{i=0}^{79} sw(n)sw(n-k) \tag{3.35}$$

แบ่งช่วงออกเป็น 3 ช่วง ดังนี้

$i = 1$: 80,...,143

$i = 2$: 40,...,79

$i = 3$: 20,...,39

โดย R′ (k) คือ R(k) ที่ถูกทำให้เป็นหนึ่งหน่วยตามสมการ

$$R'(t_i) = \frac{R(t_i)}{\sqrt{\sum_n sw^2(n-t_i)}} \qquad i = 1,...,3 \tag{3.36}$$

โดยเลือกค่าที่ทำให้ R′ (k) มีค่ามากที่สุด ช่วงละ 1 ตัว ดังนั้น จะมีตัวที่ถูกเลือกขึ้นมา 3 ตัว สำหรับการเลือกนั้น อัลกอริทึม จะมีแนวโน้มจะเลือกค่าที่มีการดีเลย์น้อย โดยวิธีการเลือกมีดังต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$
\begin{aligned}
&T_{op} = t_1 \\
&R'(T_{op}) = R'(t_1) \\
&if \quad R'(t_2) \geq 0.85R'(T_{op}) \\
&\qquad R'(T_{op}) = R'(t_2) \\
&\qquad T_{op} = t_2 \\
&end \\
&if \quad R'(t_3) \geq 0.85R'(T_{op}) \\
&\qquad R'(T_{op}) = R'(t_3) \\
&\qquad T_{op} = t_3 \\
&end
\end{aligned}
\tag{3.37}
$$

3.2.1.3 การคำนวณหาผลการตอบสนองเชิงอิมพัลส์ (Computation of the impulse response)

การตอบสนองอิมพัลซ์ h (n) ของการถ่วงน้ำหนักวงจรสังเคราะห์ถูกใช้ในการหา Adaptive codebook และ Fix codebook h (n) ถูกคำนวณไว้สำหรับแต่ละเฟรมย่อย โดยการกรองสัญญาณซึ่งประกอบไปด้วยสัมประสิทธิ์ของวงจรกรอง A (z/γ_1) และ ทำการวงจรกรอง ต่อโดยใช้ 1/ $\hat{A}(z)$ (z) และ 1/A (z/γ_2)

3.2.1.4 การคำนวณหาสัญญาณกระตุ้น (Computation of the target signal)

Target signal x (n) ใช้สำหรับหา Adaptive codebook สามารถคำนวณได้จากนำ LP residual signal (r (n)) ลบออกด้วยความผิดพลาดของสัญญาณเสียงจริงกับสัญญาณเสียงที่สังเคราะห์ขึ้น และกรอง LP residual signal r(n) ผ่านการความเป็นไปได้ ระหว่างวงจรสังเคราะห์ 1/ $\hat{A}(z)$ (z) และวงจรกรองถ่วงน้ำหนัก A(z/γ_1) / A(z/γ_2) จะได้สัญญาณกระตุ้นที่ผ่านมา (y(n)) x(n) และ y(n) ถูกเลื่อนออกไปด้วยดีเลย์ k แล้วหาความเหมือนกันเป็นคาบ เพื่อหา พิตท์ที่ละเอียดขึ้น

3.2.1.5 การค้นหาอเด็บทีพโคดบุ๊ค (Adaptive codebook search)

พารามิเตอร์ของ Adaptive codebook ประกอบด้วย พิตท์และอัตราขยายสัญญาณโดย adaptive codebook นั้นถูกสร้างขึ้นโดยใช้สัญญาณกระตุ้นของวงจรกรองของพิตท์ (pitch filter) โดยการหา Adaptive codebook จะกระทำทุก 5ms หรือทุก ๆ เฟรมย่อย ในเฟรมย่อยแรกนั้นพิตท์ T1 ถ้าประมาณแล้วมีค่าต่ำกว่า 85 จะต้องถูกหาให้ละเอียดขึ้น โดยการประมาณค่าเศษส่วนซึ่งมีความละเอียด 1/3 ([19 1/3, 84 2/3]) และกรณีที่มากกว่า 85 จะประมาณเป็นจำนวนเต็มเท่านั้น ([85,143]) สำหรับเฟรมย่อย ที่ 2 ดีเลย์ T2 จะถูกใช้ร่วมกับการประมาณเศษส่วนด้วยความละเอียด 1/3 และอยู่ในช่วงของ ([int(T1) – 5 2/3, int(T1) + 4 2/3])

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับพิตท์ของแต่ละเฟรมย่อยแรกใช้ closed loop analysis เพื่อลดรากที่สองค่าเฉลี่ยของความผิดผลาดที่ถ่วงน้ำหนักพบว่าจะมีค่าอยู่รอบ Open loop pitch ซึ่งจะไม่เกิน 6 ตัวอย่าง ดังนั้นจะช่วงลดการหาพิตท์ลงไปได้โดยกำหนดช่วงของการหาได้โดยอยู่ระหว่าง tmin และ tmax

$$
\begin{aligned}
&t_{\min} = T_{op} - 3 \\
&if \;\; t_{\min} \;<\; 20 \;\; then \;\; t_{\min} = 20 \\
&t_{\max} = t_{\min} + 6 \\
&if \;\; t_{\max} \;>\; 143 \;\; then \\
&\qquad t_{\max} = 143 \\
&\qquad t_{\min} = t_{\max} - 6 \\
&end
\end{aligned}
\tag{3.38}
$$

สำหรับเฟรมย่อยที่ 2 ค่าของพิตท์จะอยู่รอบ ๆ ของพิตท์ที่ถูกเลือกโดย T1 ขอบเขตของการค้นหาจะอยู่ในช่วงของ tmin – 2/3 และ tmax + 2/3 โดย tmin และ tmax

$$
\begin{aligned}
&t_{\min} = \text{int}(T_1) - 5 \\
&if \;\; t_{\min} \;<\; 20 \;\; then \;\; t_{\min} = 20 \\
&t_{\max} = t_{\min} + 9 \\
&if \;\; t_{\max} \;>\; 143 \;\; then \\
&\qquad t_{\max} = 143 \\
&\qquad t_{\min} = t_{\max} - 9 \\
&end
\end{aligned}
\tag{3.39}
$$

สำหรับการหา Close loop pitch จะต้องลดรากที่สองค่าเฉลี่ยของความผิดผลาดที่ถ่วงน้ำหนักระหว่าง x (n) กับ y (n) ซึ่ง ซึ่งต้องพยายามทำให้สมการที่ 3.40 มีค่ามากที่สุด

$$
R(k) = \sum_{i=0}^{39} x(n) y_k(n-k) \Big/ \sqrt{\sum_{i=0}^{39} y_k(n) y_k(n)} \tag{3.40}
$$

โดย y (n) ถูกเลื่อนไปเรื่อย ๆ โดย k อยู่ในช่วงของ tmin + 1 ถึง tmax ซึ่งค่าที่ได้จะเป็นพิตท์ที่ละเอียดขึ้น

โดย u (n) คือบัฟเฟอร์ของสัญญาณกระตุ้น และ y_{k-1} (-1) = 0 ซึ่งในขั้นตอนในการหานั้น แต่ละตัวอย่างของ u (n), n = 0 39 นั้นยังไม่รู้ และยังต้องการให้พิตท์ดีเลย์น้อยกว่า 40 และเพื่อทำให้การค้นหาง่ายขึ้น LP residual จะถูกกำหนดค่าให้กับ u (n) จึงทำให้เกิดความสัมพันธ์ ดังต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$y_k(n) = y_{k-1}(n-1) + u(-k)h(n) \qquad n = 39,...,0 \qquad (3.41)$$

สำหรับการหาค่า T2 และ T1 ถ้าพิตท์ที่หาได้มีค่าน้อยกว่า 85 จะต้องทำการหาต่อด้วยความละเอียด 1/3 รอบ ๆ พิตท์ที่หามาได้จะถูกทดสอบ หาค่ามากที่สุด โดยการประมาณค่าโดยใช้วงจรกรองชนิด FIR b12 มีพื้นฐานจากฟังก์ชัน ฟังก์ชันแฮมมิ่งวินโดว์ซิงก์ (Hamming windowed sinc) ด้วย ปัด sinc –11 หรือ –11 และ pad ด้วย zeros ที่ + 12 หรือ –12 (b_{12} (12) = 0) โดยวงจรกรอง มีความถี่ cut off ที่ –3 dB ที่ 3600 Hz และประมาณค่า R (k) สำหรับ fraction –2/3, -1/3, 0, 1/3, 2/3

$$R(k)_t = \sum_{i=0}^{3} R(k-i)b_{12}(t+3i) + \sum_{i=0}^{3} R(k+1+i)b_{12}(3-t+3i) \qquad t = 0,1,2 \qquad (3.42)$$

โดยที่ T = 0, 1, 2 เกี่ยวข้องกับ fraction 0, 1/3, 2/3 ตามลำดับ

3.2.1.6 การสร้างอเด็บทีฟโคดบุ๊คเวกเตอร์ (Generation of the adaptive-codebook vector)

เมื่อหาพิทช์ดีเลย์ได้แล้ว เวกเตอร์ v (n) จะถูกคำนวณ โดยแท้จริงแล้ว v (n) คือ u (n) ที่มีค่าพิตท์เป็นดีเลย์ k fraction t เท่านั้น

$$v(n) = \sum_{i=0}^{9} u(n-k+i)b_{30}(t+3i) + \sum_{i=0}^{9} u(n-k+1+i)b_{30}(3-t+3i) \qquad n = 0,...,39 \qquad t = 0,1,2 \qquad (3.43)$$

โดยการประมาณค่าใช้วงจรกรอง b 30 มีพื้นฐานมาจากฟังก์ชัน ฟังก์ชันแฮมมิ่งวินโดว์ซิงก์ (Hamming windowed sinc) ปัดเศษที่ +29 หรือ –29 และเติมด้วยศูนย์ที่ + 30 หรือ –30 (b_{30} (30) = 0) โดยมีความถี่ cut off = -3 dB ที่ 3600 Hz

3.2.1.7 การคำนวณหาดีเลย์ของอแด็บทีฟโคดบุ๊ค (Codeword computation for adaptive-codebook delays)

พิทช์ T1 นั้นเข้ารหัส 8 บิทในเฟรมย่อยแรก และพิทช์ของเฟรมย่อยที่ 2 ซึ่งสัมพัทธ์กับเฟรมย่อยแรกจะเข้ารหัสด้วย 5 บิทดีเลย์ที่เป็นเศษส่วน T ถูกแสดงโดยตัวที่เป็นจำนวนเต็มของ T int (T) และส่วนที่เป็นเศษส่วน frac/3, frac = -1,0,1 พิทท์อินเด็ก P1 จะถูกเข้ารหัส ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$P1 = \begin{cases} 3(\text{int}(T_1) - 19) + frac - 1 & if \quad T_1 = [19,...,85], frac = [-1,0,1] \\ (\text{int}(T_1) - 85) + 197 & if \quad T_1 = [86,...,143], frac = 0 \end{cases} \tag{3.44}$$

สำหรับ T2 ก็เช่นเดียวกับ T1 แต่สัมพัทธ์กับ T1

$$P2 = 3(\text{int}(T_2) - t_{\min}) + frac + 2 \tag{3.45}$$

เพื่อความทนของตัวเข้ารหัส ในการได้รับการสุ่มความผิดพลาดต่าง ๆ จึงมีการเพิ่ม พาริตี้บิท P0 เข้าไปโดยคำนวณจาก P1 พาริตี้บิทจะถูกสร้างโดย XOR โอเปอร์เรชันโดยใช้ 6 บิท สูงของ P1 ที่ตัวถอดรหัสพาริตี้บิทจะถูกคำนวณกลับ และถ้าไม่ตรงกับที่ส่งมา ตัวถอดรหัสจะทำการกลบเกลื่อนความผิดพลาด (error concealment)

3.2.1.8 การคำนวณหาอัตราขยายของอแด็บทีพโคดบุ๊ค (Computation of the adaptive-codebook gain)

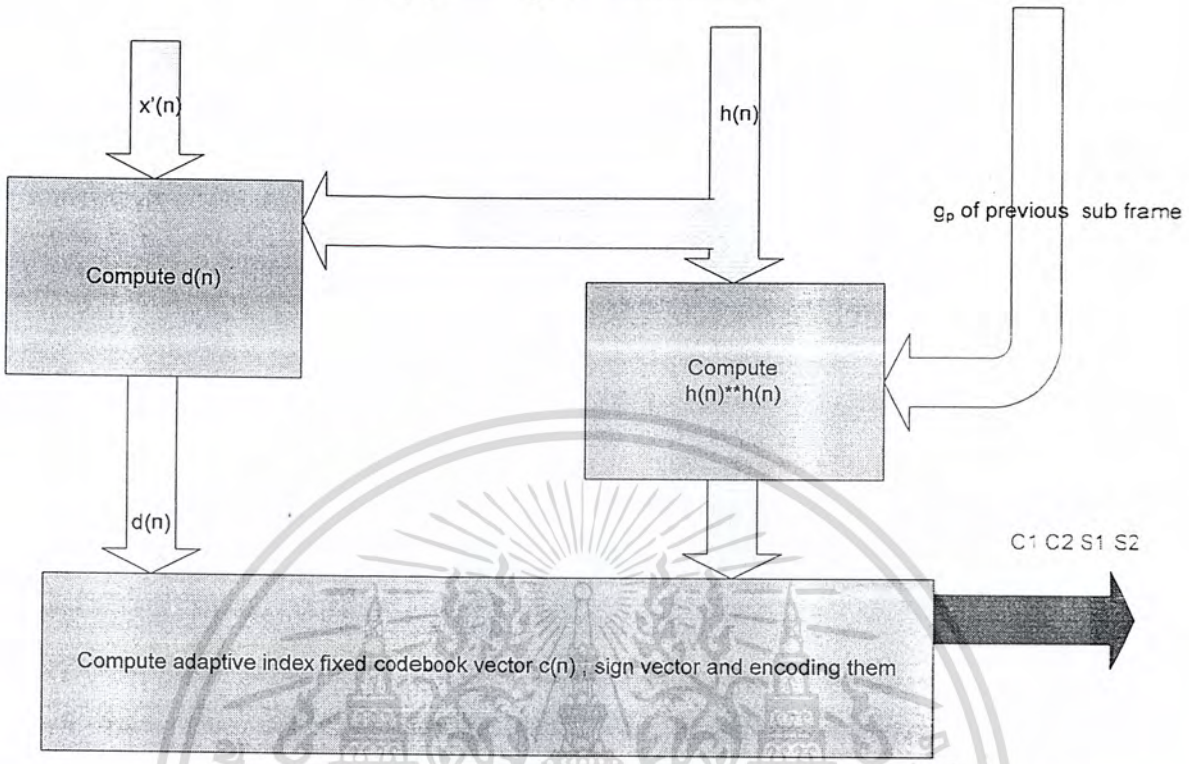
เมื่อดีเลย์ของ Adaptive codebook ถูกคำนวณแล้ว จะต้องมีการหาอัตราขยายสัญญาณ adaptive-codebook

$$g_p = \sum_{n=0}^{39} x(n)y(n) / \sum_{n=0}^{39} y(n)y(n)$$
$$y(n) = \sum_{i=0}^{n} v(i)h(n-i) \qquad n = 0,...,39 \tag{3.46}$$

ขณะที่ x (n) เป็น target signal y (n) เป็น adaptive codebook เวกเตอร์ ที่ถูกกรอง ซึ่งจะได้มาจากเป็นการคอนโวลูชันระหว่าง v (n) กับ h (n)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.2 การหาองค์ประกอบสัญญาณเสียงสุ่ม (Unvoiced sound)



รูปที่ 3-4 แสดงการหาองค์ประกอบสัญญาณเสียงสุ่ม (Unvoiced sound)

3.2.2.1 ฟิกซ์โคดบุ๊ค (Fix codebook)

Fix codebook มีพื้นฐานมาจาก algebraic codebook โดยใช้ ISPP ซึ่ง codebook แต่ละ codebook เวกเตอร์ จะประกอบด้วยพัลซ์ ที่ไม่ใช่ 0 4 พัลซ์ ซึ่งแต่ละพัลซ์ จะมีทั้งแอมพลีจูดที่เป็น +1 หรือ –1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Pulse	Sign	Positions
i0	***s0:*** ± ***1***	***m*0: 0, 5, 10, 15, 20, 25, 30, 35**
i1	***s1:*** ± ***1***	***m*1: 1, 6, 11, 16, 21, 26, 31, 36**
i2	***s2:*** ± ***1***	***m*2: 2, 7, 12, 17, 22, 27, 32, 37**
i3	***s3:*** ± ***1***	***m*3: 3, 8, 13, 18, 23, 28, 33, 38 4, 9, 14, 19, 24, 29, 34, 39**

ตารางที่ 3-1 แสดงโครงสร้างของ fixed codebook

Codebook เวกเตอร์ c (n) ถูกสร้างขึ้นโดยเป็นเวกเตอร์ 0 ทั้งหมด 40 และมี 4 พัลซ์หนึ่งหน่วย(unit pulse) ที่ตำแหน่งที่เจอ และคูณด้วยเครื่องหมายที่เกี่ยวข้องในตาราง

$$c(n) = s_0\delta(n-p_0) + s_1\delta(n-p_1) + s_2\delta(n-p_2) + s_3\delta(n-p_3) \qquad n = 0,...,39$$

เมื่อ $\delta(n)$ แทนฟังก์ชันอิมพัลซ์หนึ่งหน่วย (unit impulse function) (3.47)

s แทนเครื่องหมายแสดงค่า +/- ของแอมพลิจูด

Codebook ที่ถูกเลือกมาจะต้องผ่าน adaptive pre-filter P (z) ซึ่งจะเพิ่ม harmonic คอมโพเนินท์และพัฒนาคุณภาพของสัญญาณเสียง

$$P(z) = 1/(1-\beta z^{-T}) \tag{3.48}$$

เมื่อ T คือเวลาที่เป็นจำนวนเต็มของพิตท์ในเฟรมย่อยนั้น ๆ และ β คือ อัตราขยายสัญญาณพิตท์ค่าของ β หาได้จาก นำอัตราขยายสัญญาณของ adaptive codebook ที่ถูกควอนไทด์ของเฟรมย่อยก่อนหน้า

$$\beta = \hat{g}_p^{(m-1)} \qquad bounded\ \ by\ \ 0.2 \le \beta \le 0.8 \tag{3.49}$$

สำหรับพิตท์ที่น้อยกว่า 40 codebook c (n) จะถูกแก้ไข ดังต่อไปนี้

$$c(n) = \begin{cases} c(n) & n = 0,...,T-1 \\ c(n) + \beta c(n-T) & n = T,...,39 \end{cases} \tag{3.50}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เช่นเดียวกันกับ h (n) ที่ต้องถูกแก้ไข เช่นเดียวกัน

$$h(n) = \begin{cases} h(n) & n = 0,...,T-1 \\ h(n) + \beta h(n-T) & n = T,...,39 \end{cases} \quad (3.51)$$

3.2.2.2 การค้นหาฟิกซ์โคดบุ๊ค (Fixed codebook search procedure)

Fixed codebook ถูกค้นหาโดย ลด MSE ระหว่าง ความเข้มของสัญญาณเสียงที่เข้ามาตาม x′ (n) กับสัญญาณที่เราจะสร้างขึ้นมาคือ c (n)

$$x'(n) = x(n) - g_p y(n) \qquad n = 0,...,39 \quad (3.52)$$

เมื่อ y (n) เป็น adaptive codebook เวกเตอร์ ที่ถูก และ g_p เป็นอัตราขยายสัญญาณ adaptive codebook

Matrix H ถูกนิยามเป็น lower triangular Toepliz convolution เมทริกซ์ กับ diagonal h (0) และ lowers diagonal h (1) h (39) เมทริกซ์ $\phi = H'H$ ประกอบด้วยคอร์รีเลชันของ h (n) และอีลีเมนต์ของเมทริกซ์ที่สมมาตร

$$\phi(i,j) = \sum_{n=j}^{39} h(n-i)h(n-j) \qquad i = 0,...,39 \qquad j = i,..39 \quad (3.53)$$

สัญญาณคอร์รีเลชัน (Correlation signal d (n)) ได้มาจาก x′(n) และการตอบสนองอิมพัลซ์ (impulse response)h (n)

$$d(n) = \sum_{i=n}^{39} x'(n).h(i-n) \qquad ,n = 0,...,39 \quad (3.54)$$

ถ้า c_k เป็นพจน์ที่ k ของ Fix – codebook เวกเตอร์ แล้ว codebook จะถูกหาโดยต้องทำให้ correlation ระหว่าง d (n) และ c (n) สมการที่ (3.55) มีค่ามากที่สุด เมื่อ t คือสัญลักษณ์ของการทรานส์โพส

$$\frac{C_k^2}{E_k} = \frac{\left(\sum_{n=0}^{39} d(n)c_k(n)\right)^2}{c_k' \phi c_k} \quad (3.55)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Algebraic ของ codebook ทำให้การค้นหารวดเร็วเพราะว่า codebook เวกเตอร์ c_k ประกอบด้วยเฉพาะพัลซ์ ที่ไม่ใช่ 0 จำนวน 4 พัลซ์ จึงทำให้คอร์รีเลชันในที่เป็นตัวเศษของสมการลดรูปลงเหลือเพียง

$$C = \sum_{i=0}^{3} s_i d(m_i) \tag{3.56}$$

โดย m แทนตำแหน่งของพัลซ์ และ s แทนแอมพลีจูดของพัลซ์ และ พลังงาน ที่เป็นตัวหารนั้น จะกลายเป็น

$$E = \sum_{i=0}^{3} \phi(m_i, m_i) + 2\sum_{i=0}^{2} \sum_{j=i+1}^{3} s_i s_j \phi(m_i, m_j) \tag{3.57}$$

เพื่อทำให้การค้นหาง่ายลงแอมพลีจูดของพัลซ์ จะถูกคาดเดาก่อนโดยการทำควอนไทด์สัญญาณ d (n) ซึ่งทำได้โดยกำหนดแอมพลีจูดของพัลซ์ ในตำแหน่งที่แน่นอนที่เท่ากับเครื่องหมายของ d(n) ที่ตำแหน่งนั้น ๆ ก่อนทำการหา codebook ขั้นตอนแรกจะต้องสร้างกลับ (decompose d(n)) ออกเป็น 2 ส่วน โดยแบ่งเป็นขนาดของ |d(n)| และเครื่องหมาย ขั้นตอนที่ 2 ทำการแก้เมทริกซ์ ϕ โดยการรวมเครื่องหมายด้วย

$$\phi'(i,j) = sign[d(i)]sign[d(j)]\phi(i,j) \qquad i = 0,..39 \qquad j = i+1,...,39 \tag{3.58}$$

และทะแยงมุมหลัก (Main diagonal) ใน ϕ จะถูกสเกลโดยเอาอัตราส่วน ของ 2 ออก

$$\phi'(i,j) = 0.5\phi'(i,i) \qquad i = 0,...,39 \tag{3.59}$$

3.2.2.3 การเข้ารหัสฟิกซ์โคดบุ๊ค (Codeword computation of the fixed codebook)

ตำแหน่งของพัลซ์ i_0 i_1 i_2 จะเข้ารหัส โดยใช้ 3บิท ในขณะที่ i_3 นั้นจะเข้ารหัส โดยใช้ 4บิท แต่ละพัลซ์แอมพลีจูดนั้นจะใช้ 1บิท ในการเข้ารหัส ทำให้ทั้งหมดมี 17บิท สำหรับ 4 พัลซ์ โดย s = 1 ถ้าเครื่องหมายเป็น + และ s = 0 ถ้าเครื่องหมายเป็น – sign codeword

$$S = s_0 + 2s_1 + 4s_2 + 8s_3 \tag{3.60}$$

และ fixed codebook ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$C = (m_0 / 5) + 8(m_1 / 5) + 64(m_2 / 5) + 512(2(m_3 / 5) + jx) \quad (3.61)$$

โดยที่ jx = 0 ถ้า m3 = 3,8, ,38 และ jx = 1 ถ้า m3 = 4,9, ,39

3.2.2.4 การทำควอนไทด์เซชันของอัตราขยาย (Quantization of the gain)

อัตราขยายสัญญาณ Adaptive codebook (pitch gain) และ fix codebook เป็น เวกเตอร์ที่ถูกควอนไทด์โดยใช้ 7-bit ในการหา codebook ของ อัตราขยายสัญญาณจะต้องทำให้ MSE ระหว่างสัญญาณเสียงเดิมกับสัญญาณเสียงที่สร้างขึ้นมาใหม่ให้น้อยที่สุด สมการ

$$E = x'x + g_p^2 y'y + g_c^2 z'z - 2g_p x'y - 2g_c x'z + 2g_p g_c y'z \quad (3.62)$$

โดยที่ x เป็นเวกเตอร์ target y เป็นวงจรกรอง adaptive codebook เวกเตอร์ และ z เป็น fixed codebook เวกเตอร์ ที่ทำการคอนโวล์ (convolved) กับ h (n)

$$z(n) = \sum_{i=0}^{n} c(i)h(n-i) \qquad n = 0,....39 \quad (3.63)$$

3.2.2.5 การพยากรณ์อัตราขยาย (Gain prediction)

อัตราขยายสัญญาณ Fixed codebook g_c สามารถหาได้จาก

$$g_c = \gamma . g_c' \quad (3.64)$$

โดย g'_c คืออัตราขยายสัญญาณที่ได้จากการพยากรณ์โดยใช้พลังงานของ fixed codebook ของเฟรม ก่อนหน้า และ γ คืออัตราส่วนของความถูกต้อง

พลังงานเฉลี่ยของ Fixed codebook contribution นั้น กำหนดโดยสมการ

$$E = 10.\log\left(\frac{1}{40}\sum_{n=0}^{39} c(n)^2\right) \quad (3.65)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลังจากทำ Scale ของ c (n) ด้วยอัตราขยายสัญญาณ fixed codebook g_c พลังงานของสเกล fixed codebook คือ 20 log g_c + E สมมติให้ E_m เป็น Mean removed energy (dB) ของ Scaled fixed codebook contribution ที่เฟรมย่อยที่ m จะได้ดังสมการ ดังนี้

$$E^{(m)} = 20\log g_c + E - \overline{E} \tag{3.66}$$

โดยที่ E_{av} = 30 dB ซึ่งเป็นค่าเฉลี่ยพลังงานของสัญญาณกระตุ้น Fixed codebook โดย g_c สามารถเขียนอยู่ในรูปฟังก์ชัน ของพลังงานได้ ดังนี้

$$g_c = 10^{(E_m + \bar{E} - E)/20} \tag{3.67}$$

อัตราขยายสัญญาณที่ใช้สำหรับการพยากรณ์ g_c (predict gain) ถูกคาดการณ์ได้จาก log ของพลังงานของ Fixed codebook contribution ก่อนหน้า โดยใช้ 4th Order MA Prediction

$$\tilde{E}^{(m)} = \sum_{i=1}^{4} b_i \hat{U}^{(m-i)} \tag{3.68}$$

โดย [b1 b2 b3 b4] = [0.68 0.58 0.34 0.19] คือสัมประสิทธิ์ MA และ $\tilde{E}^{(m)}$ คือควอนไทด์ของความผิดพลาดจากการพยากร (prediction error) ที่เฟรมย่อยที่ m

$$U^{(m)} = E^{(m)} - \hat{E}^{(m)} \tag{3.69}$$

3.2.2.6 การค้นหาอัตราขยายที่ควอนไทด์ (Codebook search for gain quantization)

อัตราขยายสัญญาณ Adaptive codebook g_p และอัตราส่วนของ γ เป็นเวกเตอร์ ที่ถูกควอนไทด์โดยใช้ two stage conjugate structured codebook โดยสถานะแรกประกอบด้วย 3บิท 2 เวกเตอร์ codebook GA และสถานะที่สองประกอบด้วย 4 บิท 2 เวกเตอร์ codebook GB โดยอีลีเมนต์แรกของแต่ละ codebook นั้นแทน quantized อัตราขยายสัญญาณ adaptive codebook *gp* และอีลีเมนต์ที่สองแทน quantized fixed codebook gain correction factor $\hat{\gamma}$ กำหนดให้vอินเด็กของ codebook GA และ GB สำหรับ GA และ GB อัตราขยายสัญญาณ adaptive codebook ที่ถูกควอนไทด์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\hat{g}_p = GA_1(GA) + GB_1(GB)$$
$$and\ \ the\ \ quantized\ \ fixed-codebook\ \ gain\ \ by: \qquad (3.70)$$
$$\hat{g}_c = g_c^{'}\hat{\gamma} = (GA_2(GA) + GB_2(GB))$$

3.2.2.7 การเข้ารหัสอัตราขยายที่ถูกควอนไทด์แล้ว (Codeword computation for gain quantizer)

Codeword GA และ GB สำหรับควอนไทด์เซอร์ของอัตราขยายสัญญาณ จะได้รับโดยอินเด็ก ที่เกี่ยวข้อง เพื่อที่จะช่วยลดผลกระทบความผิดผลาดแบบบิทเดียว (single bit error) ในอินเด็กของ codebook ที่ทำการ map ไป

3.2.2.8 การปรับปรุงข้อมูลในหน่วยความจำ (Memory Update)

การอัพเดตสถานะเพื่อทำการสังเคราะห์และถ่วงน้ำหนักที่ต้องการคำนวณเพื่อหา target signal ในเฟรมย่อยถัดไป หลังจากที่อัตราขยายสัญญาณทั้ง 2 ถูกควอนไทด์แล้ว u(n) ในเฟรมย่อยปัจจุบันจะหาได้จาก

$$u(n) = \hat{g}_p v(n) + \hat{g}_c c(n) \qquad n = 0,...,39 \qquad (3.71)$$

สถานะของวงจรกรองจะต้องถูก update โดยการกรอง r(n) – u(n) หรือความแตกต่างระหว่าง residual กับสัญญาณกระตุ้นผ่านวงจรกรอง $1/\hat{A}(z)$ และ A (z/γ_1)*A (z/γ_2) สำหรับ 40 ตัวอย่างเฟรมย่อย และทำการเก็บสถานะของวงจรกรองไว้ โดย สร้างสัญญาณขึ้นมาใหม่ s(hat)(n) ซึ่งจะถูกคำนวณโดยการกรองสัญญาณกระตุ้นผ่าน $1/\hat{A}(z)$ เอาท์พุท ของวงจรกรองขึ้นอยู่กับ r(n) – u(n) ซึ่งจะเท่ากับ e(n) = s(n) – $\hat{s}$ (n) ดังนั้นสถานะของวงจรสังเคราะห์ $1/\hat{A}(z)$ จะถูกให้โดย e(n) โดยที่ n = 30, ,39 อัพเดตสถานะของวงจรกรอง A (z/γ_1)*A (z/γ_2) โดยการกรองสัญญาณความผิดพลาดผ่านวงจรกรองเพื่อนำไปหา ew(n)

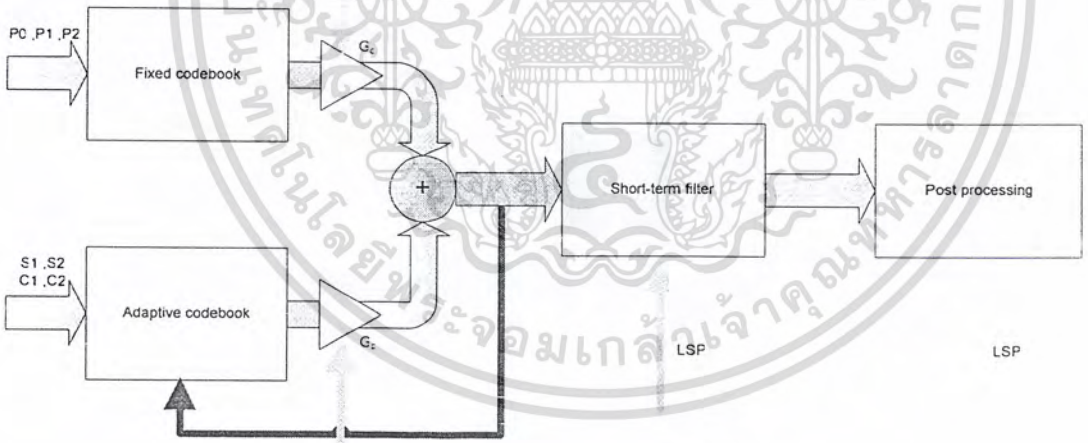
$$ew(n) = x(n) - \hat{g}_p y(n) - \hat{g}_c z(n) \qquad (3.72)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Parameter	codeword	Subframe 1 (bits)	Subframe 2 (bits)	Total bits per frame
Line spectrum pairs	*L0*, *L1*, *L2*, *L3*			18
Adaptive-codebook delay	*P1*, *P2*	8	5	13
Pitch-delay parity	*P0*	1		1
Fixed-codebook index	*C1*, *C2*	13	13	26
Fixed-codebook sign	*S1*, *S2*	4	4	8
Codebook gains (stage 1)	*GA1*, *GA2*	3	3	6
Codebook gains (stage 2)	*GB1*, *GB2*	4	4	8
Total				80

ตารางที่ 3-2 แสดงบิทฟิลด์ที่ส่งให้ฝั่งถอดรหัส

3.3 การถอดรหัส



รูปที่ 3-5 แสดงโฟลโปรแกรมของการถอดรหัส

การสังเคราห์ะสัญญาณเสียงกลับนั้นวิธีการได้อยู่ในฝั่งเข้ารหัส อยู่แล้วเมื่อฝั่งถอดรหัส ได้รับพารามิเตอร์แล้วสังเคราะห์สัญญาณเสียงออกมา สัญญาณเสียงที่ได้จะถูกปรับปรุงคุณภาพโดยอาศัย Postfilter

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.1 การปรับปรุงคุณภาพของสัญญาณ (Post Processing)

เมื่อตัวถอดรหัส ได้รับพารามิเตอร์แล้วสร้างสัญญาณเสียงขึ้นมา สัญญาณเสียงที่สร้างขึ้นจะถูกนำมาปรับปรุงคุณภาพโดยใช้ Post Processing ซึ่งมีองค์ประกอบด้วยกัน 4 ส่วนคือ

1. Long term postfilter แทนโดยสัญลักษณ์ Hp(z)
2. Short term postfilter แทนโดยสัญลักษณ์ Hf(z)
3. Tilt compensation filter แทนโดยสัญลักษณ์ Ht(z)
4. Gain compensation factor แทนโดยสัญลักษณ์ g

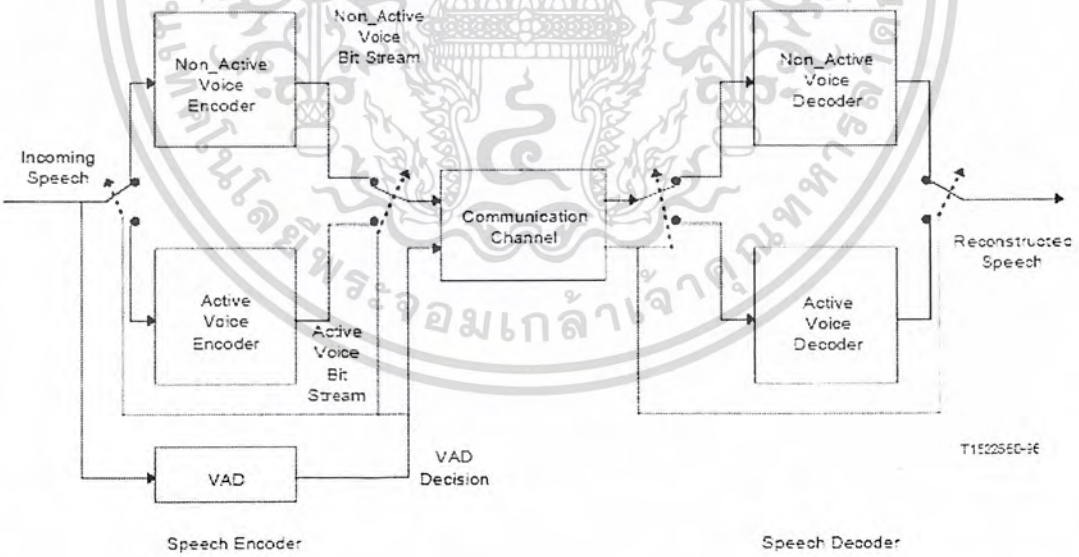
Long term postfilter สร้างจากถอดรหัสหาอัตราขยายของสัญญาณและพารามิเตอร์ของพิทช์ดีเลย์เป็นวงจรกรองฟังก์ชัน 1st order all-zero ที่ช่วยเน้นช่วงความถี่รอบ ๆ คาบของพิตท์

Short term postfilter ออกแบบมาให้ช่วยเน้น formants ซึ่งเป็นช่วงความถี่ของพลังงานในวงจรสังเคราะห์

Tilt คือความชัน (slope) ของพลังงานใน โดเมนเชิงความถี่ ในสัญญาณเสียงโดยวงจรกรองแบบ Tilt compensation ออกแบบมาเพื่อช่วยลดความผิดเพี้ยนที่เกิดจากการทำ Short term postfilter

Gain compensation factor เป็นผลหารของพลังงานของการถอดรหัสแบบเดิมหารด้วยพลังงานหลังผ่านส่วนของ post filter แล้ว เป็นตัวที่ใช้เก็บความเพี้ยนของสัญญาณเสียง

3.4 การตรวจสอบว่ามีการสนทนาหรือไม่ (Voiced Activity Detection (VAD))



รูปที่ 3-6 แสดง Speech communication system with VAD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

G.729 Annex B เป็นส่วนเพิ่มเติมจาก G.729 สำหรับการเข้ารหัสสัญญาณเสียงเฟรม ที่ไม่ใช่เสียงพูด โดยการแทนรหัสสัญญาณในเฟรม นั้นเป็นสัญญาณสัญญาณรบกวน โดยปกติสัญญาณเสียงจะประกอบด้วย ช่วงที่มีเสียงพูด (talkspurts) และตามด้วยช่วงที่ไม่มีเสียงพูด (silence period) ส่วนที่เพิ่มเติมเข้าไปสำหรับ Annex B นี้คือ การแยกได้ว่าช่วงใดที่มีเสียงพูดและช่วงใดที่ไม่มีเสียงพูด ด้วยการเข้ารหัส G.729 เดิมนั้นเฟรมที่ไม่มีเสียงพูด G.729 นั้นจะมีการส่งสัญญาณที่เข้ารหัสแล้ว 2 แบบ คือไม่ส่งค่าพารามิเตอร์อะไรออกไปเลย หรือ ส่งค่าพลังงานของสัญญาณเสียงนั้น เพื่อที่ตัวถอดรหัสจะสามารถแทนสัญญาณที่เข้ารหัสมาด้วย white noise แต่อย่างไรก็ตาม ในสภาพแวดล้อมที่มีเสียงดังและประกอบด้วยสัญญาณรบกวนที่มีค่าสถิติไม่คงที่ (non stationary background noise) นั้น การส่งสัญญาณทั้ง 2 ทั้ง 2 แบบนั้น ไม่มีประสิทธิภาพเพียงพอ จึงต้องมี Annex B เพิ่มเติมขึ้นมาสำหรับการตัดสินใจว่าสัญญาณเฟรม ที่สนใจนั้นเป็นช่วงที่มีเสียงพูดหรือไม่นั้น ใช้วิธีการของ Voice Activity Detection (VAD) ในแต่ละเฟรม การตัดสินใจว่าเฟรม ใด เป็นเสียงพูดหรือไม่นั้น ใช้ค่าเฉลี่ยของพารามิเตอร์4 ตัวประกอบการตัดสินใจ โดยพารามิเตอร์ทั้ง 4 นั้นมีดังต่อไปนี้

1. LSF ของช่วงที่ไม่ใช่เสียงพูด
2. ค่าพลังงาน Full band ของสัญญาณเสียง ซึ่งหาได้จาก Log ของ สัมประสิทธิ์ตัวแรกของการทำออโตคอร์รีเลชันของช่วงที่ไม่ใช่เสียงพูด
3. ค่าพลังงาน Low band ของสัญญาณเสียง ซึ่งหาได้จากการนำค่าสัมประสิทธิ์ของการทำออโตคอร์รีเลชันกรองผ่านวงจรกรองชองช่วงที่ไม่ใช่เสียงพูด
4. ค่า zero crossing rate ของสัญญาณเสียง ของช่วงที่ไม่ใช่เสียงพูด

ในแต่ละเฟรม พารามิเตอร์ที่กล่าวนั้นจะถูกคำนวณและเปรียบเทียบกับค่าเฉลี่ย ผลต่างของค่าจริงและค่าเฉลี่ยนั้นจะใช้ในการตัดสินใจว่าเฟรม ใดเป็นเสียงพูดหรือไม่ใช่เสียงพูด มากกว่านั้นค่าที่คำนวณได้จะนำไป update ค่าเฉลี่ย จะได้ค่าเฉลี่ยใหม่สำหรับใช้ในการคำนวณในเฟรมถัดไป

หลังจากนั้นสัญญาณเสียง (เสียงพูด และ ไม่ใช่เสียงพูด) จะถูกกรอง โดยใช้พารามิเตอร์ที่ได้จาก 2 เฟรม ก่อนหน้า ถ้าตัดสินใจได้ว่าเฟรม ปัจจุบันนั้นเป็นเฟรม ที่ไม่ใช่เสียงพูด ขั้นตอนถัดไปคือตัดสินใจว่า จะส่งเฟรมข้อมูลแบบ Silence Insertion Description (SID) หรือไม่ส่งข้อมูลใดเลย (a null frame) สำหรับเฟรม SID นั้นประกอบด้วยพารามิเตอร์ที่ให้ตัวถอดรหัสสามารถสร้าง comfort noise ขึ้นมาได้ โดยประกอบด้วย พลังงานของสัญญาณกระตุ้น (5 บิท) และความผิดพลาดของสัมประสิทธิ์ LSF (10 บิท) เฟรม SID จะส่งออกไปก็ต่อเมื่อพบว่าพารามิเตอร์ของ background noise มีการเปลี่ยนแปลงไปหลังจากการส่งเฟรม SID ครั้งที่แล้ว โดยพิจารณาโดยตัวเข้ารหัส จากสัมประสิทธิ์ของวงจรกรองและพลังงานที่เปลี่ยนแปลงเกินค่า thresholds

การแยกความแตกต่างระหว่างเฟรม ทั้ง 3 ประเภท speech (G.729 เดิม), SID , และ null โดยดูจาก header ของเฟรม นั้น ๆ คือขนาดของเฟรม 80, 15, 0 ตามลำดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
การปรับปรุงความเร็วในการเข้ารหัส (OPTIMIZATION)

4.1 แนวทางในการพัฒนา

ในการปรับปรุงความเร็วของ อัลกอริทึมนั้นเราสามารถแบ่งแนวทางที่เป็นไปได้อยู่ 2 รูปแบบคือ การแก้ที่ตัวมาตรฐานของชุดเข้ารหัส แต่วิธีการนี้อาจจะมีข้อจำกัดในเรื่องของระดับของคุณภาพเสียงที่ได้ด้วยจึงต้องพิจารณาควบคู่กันไป ส่วนแนวทางการพัฒนาอีกทางเป็นการปรับปรุงโดยอาศัยความสามารถของเครื่องมือ (Tool) ที่ใช้พัฒนาคือพยายามดึงความสามารถของเครื่องมือของ ดี เอส พี บอร์ด ที่ใช้ออกมาให้มากที่สุด

4.2 การปรับปรุงความเร็วโดยการแก้ไขที่มาตรฐานการเข้ารหัส

ในการพัฒนาในส่วนนี้จำเป็นต้องทราบก่อนว่าชุดหรือกลุ่มของการคำนวณส่วนใดมีการใช้งานมากเป็นพิเศษ หรือใช้เวลาในการคำนวณนานที่สุดซึ่งจากการทดลองทำให้ทราบว่าเวลาส่วนใหญ่ของการเข้ารหัสอยู่ที่โมดูลของการคำนวณหา fixed codebook

Module	จำนวนครั้งในการเรียกเฉลี่ย(ครั้ง)	เวลาเฉลี่ยต่อเฟรม (us)
d4i40_17()	82	1285.430
cor_h_cp()	82	164.570
ACELP_codebook()	82	16.51
corr_xy2()	63	19.53
cor_h_x()	62	81.290
update_bwd()	54	14.320
tst_bwd_dominant()	53	5.430
ener_DB()	53	30.890
vad()	32	99.770
MakeDec()	31	6.440

ตารางที่ 4-1 แสดงเวลาที่ใช้ในแต่ละโมดูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มีผู้ที่เสนอทฤษฎีต่าง ๆ ในการที่จะลดความซับซ้อนในการคำนวณการค้นหาตำแหน่งของอินเด็ก ใน codebook ไว้หลายวิธี หนึ่งในนั้นก็คือ Candidate Scheme for fast ACELP search ซึ่งเป็นวิธีที่อาศัย pilot ฟังก์ชัน ในการลดจำนวณบริเวณที่ใช้ในการค้นหา (search region) ลงก่อนทำการค้นหาซึ่งก็ต้องแลกกับคุณภาพของเสียงที่จะลดลงด้วยดังนั้นจะต้องทำการทดลองหาจำนวนพื้นที่ที่จะลดลงแลกกับคุณภาพของเสียงที่ยอมรับได้

4.2.1Candidate Scheme for fast ACELP search

การหา Algebraic code excited linear prediction (ACELP) ถูกใช้ในหลาย ๆ มาตราฐานการเข้ารหัสสัญญาณเสียง เนื่องจากมีความซับซ้อนในการคำนวณต่ำกว่าวิธีอื่น ๆ และได้สัญญาณที่มีคุณภาพสูงด้วยวิธีการออพติไมซ์แบบ analysis-by-synthesis สำหรับการลดความซับซ้อนในการคำนวณของ ACELP นั้นอาศัยหลักการออกแบบฟังก์ชัน pilot เพื่อพยากรณ์ตำแหน่งของพัลซ์ที่เป็นไปได้ก่อน ทำให้ลดการคำนวณหาฟังก์ชันของคอร์รีเลชันของพัลซ์ที่ไม่จำเป็น ทำให้ตัวเข้ารหัสแบบ ACELP ซึ่งใช้สำหรับมาตราฐานของ ITU G.723.1 และ ITU G.729 ลดความซับซ้อนในการคำนวณลง 50-80 % โดยที่สัญญาณเสียงมีคุณภาพไม่ต่างจากเดิมมากนัก

1 . วิธีการ Candidate position

เพื่อที่จะสร้างสัญญาณ เสียงแบบสุ่ม ตามมาตราฐานนั้น เราจำเป็นที่จะต้องคำนวณค่า ϕ และค่า d สำหรับแต่ละ d (m) นั้น เราใช้การคูณทั้งหมด (L-m) ครั้ง และการบวกทั้งหมด (L-m-1) ครั้ง ดังนั้นถ้าเราจะหา d ทั้งหมดนั้น เราจะต้องใช้การคูณทั้งหมด (L+1)*L/2 ครั้ง และการบวกทั้งหมด (L-1)*L/2 ครั้ง

$$d(m) = \sum_{i=n}^{L-1} r[i]h[i-m], \qquad m = 0,1,...,L-1 \tag{4.1}$$

ในการหา ϕ(I,j) เราใช้การคูณทั้งหมด (L-max(I,j)) และการบวกทั้งหมด (L-max(I,j)-1) ครั้ง และสำหรับการคำนวณค่า ϕ เริ่มต้นจาก ϕ(L-1,L-1) = h(0)h(0) เราสามารถเขียนสมการแบบรีเคอร์ซีฟดังนี้

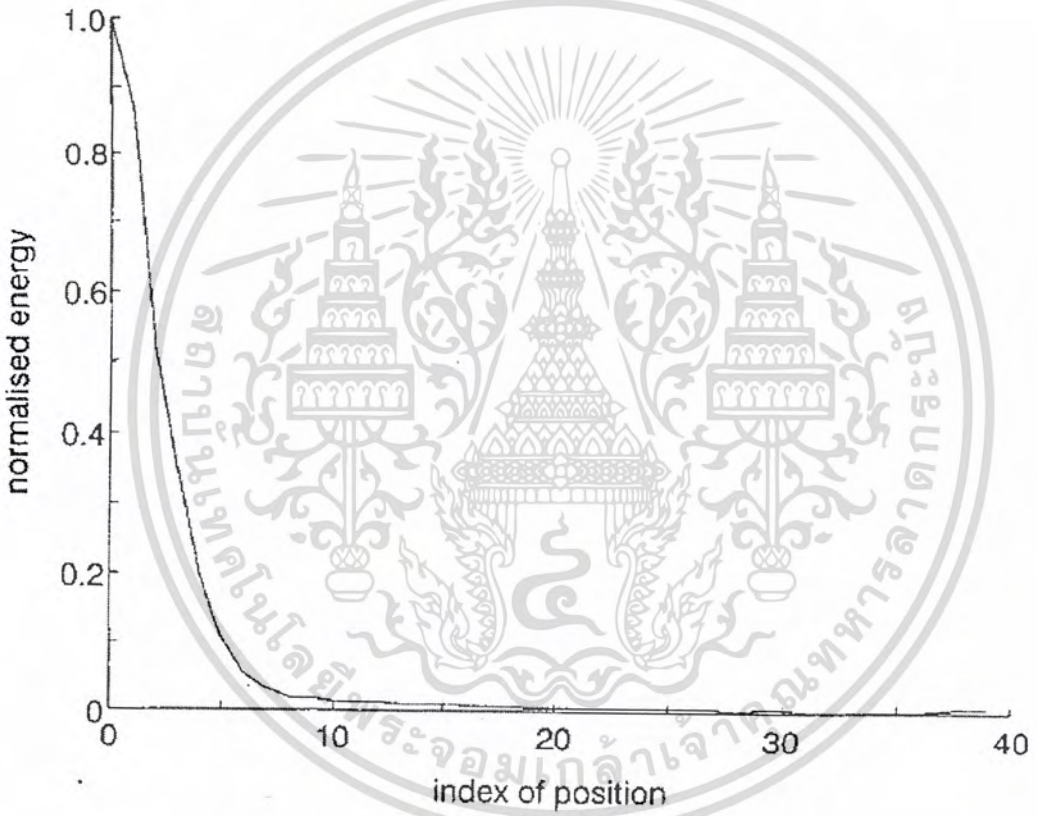
$$\phi(i,j) = \sum_{n=j}^{L-1} \mathrm{h[n - i]h[n - j]}, \qquad i = 0,1,...,L-1, \quad and \quad j = i, i+1,...,L-1 \tag{4.2}$$

ดังนั้น เราจะต้องใช้การคูณทั้งหมด (L+1)*L/2 ครั้ง และการบวกทั้งหมด (L-1)*L/2 ครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถ้าเราสามารถประมาณว่าตำแหน่งของพัลซ์ มีโอกาสเป็นสมาชิกของเสียงแบบสุ่มเท่า ๆ กันทั้งหมด แล้วเราสามารถที่จะลดขั้นตอนการค้นหาตำแหน่งของพัลซ์ ได้ และท้ายที่สุดการคำนวณหาค่า ϕ และค่า d ก็สามารถลดลงได้ด้วยเช่นกัน

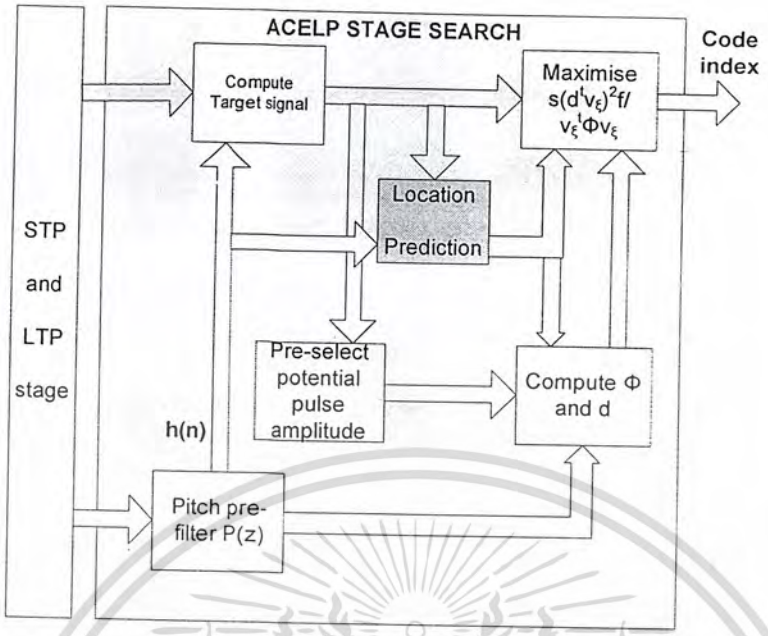
เราเสนอวิธีการของ candidate position method ว่ามีประสิทธิภาพในการพยากรณ์ตำแหน่งที่มีโอกาสเป็นพัลซ์ในส่วนของสัญญาณเสียงที่ไม่เป็นคาบ (unvoiced sound) ได้ เพราะว่าการวิเคราะห์โดยใช้ LPC นั้น เป็นการแทนแบบจำลองของทางเดินเสียง (vocal tract models) เพื่อที่จะเป็นมินิมัมเฟสฟิลเตอร์ (minimum phase filter) และวงจรสังเคราะห์ h[n] จะต้องมีความเสถียรภาพและมีมินิมัมเฟสแลก (minimum phase-lag) และมินิมัมพลังงานดีเลย์พรอบเพอร์ตี (delay properties) ตามรูปที่ 4-1 แสดงเส้นโค้งของ $E[h[n]]^2$



รูปที่ 4-1 แสดงเส้นโค้งของ $E[h[n]]^2$

หลังจากใช้ค่าเฉลี่ยของ 10000 vocal tract models พบว่า มากกว่า 95 % ของพลังงานทั้งหมดตกอยู่ใน 5 พัลซ์แรก ๆ ดังนั้นลักษณะของสัญญาณที่จะสังเคราะห์ได้นั้นจะต้องใกล้เคียงกับตำแหน่งของสัญญาณกระตุ้น และด้วยการสังเกตนี้เราจึงได้ออกแบบวิธีการเพื่อออพติไมซ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4-2 แสดง ACELP STAGE SEARCH

จากรูปที่ 4-2 เป็นการออกแบบโครงสร้างของโปรแกรมเพื่อปรับปรุงมาตราฐานการเข้ารหัสที่ใช้การหาสัญญาณกระตุ้นด้วย ACELP แบบเดิม ประยุกต์เข้ากับวิธีการใหม่ คือ Candidate Position Method

ถ้าเราสามารถคาดเดาได้ว่าพื้นที่ใดที่มีโอกาสเกินพัลซ์ ที่เราต้องการแทนในสัญญาณกระตุ้น เราจะสามารถลดการคำนวณหา d และ ϕ ลงได้อย่างมาก เพื่อที่จะคาดการถึงตำแหน่งที่เป็นไปได้ของพัลซ์ นั้น เราจะแบ่งเฟรมย่อยของเสียงที่จะทำการวิเคราะห์จำนวน L

$$\hat{r}(k) = \sum_{n \in R_k} \Lambda(r(n)), \qquad k = 0,1,\ldots, K-1 \tag{4.3}$$

ตัวอย่างออกเป็น K ส่วน ดังนั้นสัญญาณ target r(n) จะเป็นสมการดังนี้

โดยที่ Rk นั้นแทนพื้นที่ลำดับที่ k และ Λ แทน ฟังก์ชัน condensing โดยค่า K จะน้อยกว่า L และเพื่อที่จะดึงเอาส่วนที่มีความถี่ต่ำออกมาฟังก์ชัน condensing จะต้องทำการกรองข้อมูลโดยการลดความถี่ของการสุ่มตัวอย่าง (down sampling) เพื่อที่จะคำนวณหาพื้นที่ที่โดดเด่นของ r(n) ซึ่งถ้าเราใส่ฟังก์ชัน condensing ลงไปใน vocal tract model เราจะได้สมการ ดังนี้

$$\hat{h}(k) = \sum_{n \in R_k} \Lambda(h(n)), \qquad k = 0,1,\ldots, K-1 \tag{4.4}$$

ถ้าความละเอียดถูกลดลงโดยค่าที่มากกว่า 5 เราจะพบว่า $h(k) \approx \delta(k)$ เพราะว่าพลังงานที่มากที่สุดจะปรากฏอยู่ใน 5 พัลซ์ แรก สำหรับวิธีนี้ พบว่าจำนวนพื้นที่ ที่เหมาะสม K จะประมาณได้เท่ากับ L/5 ในทางเดียวกัน ถ้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราใส่ฟังก์ชัน condensing ลงไปกับสัญญาณกระตุ้น จะได้รับพื้นที่ที่มีความเด่นชัดมากที่สุด ซึ่งในกรณีนี้สัญญาณกระตุ้นซึ่งอยู่ในรูปของตำแหน่งของพัลซ์ นั้น จะถูกแบ่งออกเป็น K พื้นที่

	R_0	R_1	R_2	R_3	R_4	R_5	R_6	R_7
m_0	0	5	10	15	20	25	30	35
m_1	1	6	11	16	21	26	31	36
m_2	2	7	12	17	22	27	32	37
m_3	3	8	13	18	23	28	33	38
	4	9	14	19	24	29	34	39

ตารางที่ 4-2 แสดงตำแหน่งพัลซ์ และพื้นที่ ของการค้นหาอินเด็ก

จากตารางแสดงให้เห็นถึงพื้นที่ที่ถูกแบ่ง (R, i = 0 7) ในมาตราฐานของ G.729 เนื่องจาก ตัวเข้ารหัส G.729 นั้น L=40 และเลือก K=8

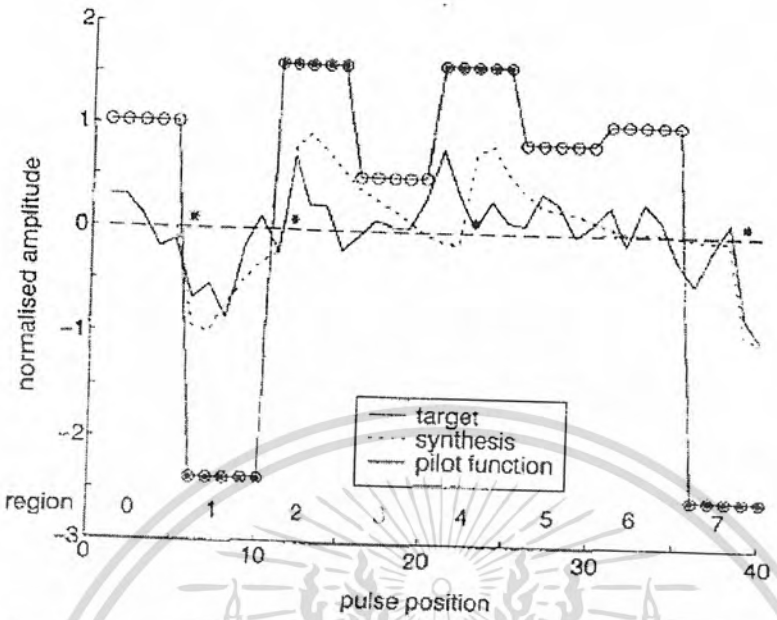
ในภาพใหญ่ ถ้าเรามีสัญญาณ target ที่มีความละเอียดต่ำ (r(n)) และการตอบสนองของวงจรกรอง ($h(n) \approx \delta(k)$) เราสามารถสังเกต r(k) เพื่อที่จะพยากรณ์ตำแหน่งของพัลซ์ที่จะเกิดขึ้น และขั้นตอนถัดไปเราจะหาฟังก์ชัน condensing ที่เหมาะสม

สำหรับวิธีของ Candidate position scheme นั้น เราต้องพัฒนาฟังก์ชัน condensing ที่เหมาะสม เพื่อที่จะพยากรณ์ตำแหน่งของพัลซ์ได้อย่างมีประสิทธิภาพมากที่สุด ถ้าพื้นที่ของสัญญาณที่ถูกเลือกแล้ว เราจะต้นทำการนิยามของฟังก์ชัน pilot E(k) เพื่อที่จะแสดงลักษณะของพลังงานที่อยู่ในพื้นที่แต่ละพื้นที่ ดังสมการต่อไปนี้

$$E(k) = p_k \sum_{n \in R_k} |r(n)|, \qquad k = 0,1,2,\ldots, K-1 \tag{4.5}$$

โดยที่ $p_k = sign(\sum_{n \in R_k} r(n))$ เพื่อที่จะแสดงขั้วของ E(k) ในพื้นที่ที่ k จากรูปแสดงกราฟของสัญญาณ target , สัญญาณที่สังเคราะห์ได้ , ฟังก์ชัน pilot , และตำแหน่งของพัลซ์ที่ได้จากการค้นโดยใช้วิธีการเดิม แสดงโดยรูปที่ 4-3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4-3 แสดงฟังก์ชัน Pilot, target และสัญญาณ synthesis ใน ตัวเข้ารหัสแบบ ACELP

ในเฟรมย่อยโดยทั่วไป แล้วพื้นที่ใดที่ให้ค่าพลังงาน E(k) มีแอมพลิจูดมากที่สุด มักจะเกิดพัลซ์มากที่สุด ดังนั้นถ้าเราเลือกที่จะค้นหาพื้นที่ M พื้นที่เราจะเลือกพื้นที่ (region) ที่ให้ค่า E(k) สูงสุด M อันดับแรกมาพิจารณาหาพัลซ์เฉพาะภายในพื้นที่ที่เลือกขึ้นมาเท่านั้น โดยตำแหน่งที่จะถูกค้นหาลดลงเหลือ ML/K เท่านั้น แทนที่จะต้องหาทุก ๆ ความเป็นไปได้ที่เกิดขึ้น คือ 2^{13} จะลดลงเหลือเพียง $2M^4$ เนื่องจากตามมาตราฐานของ G.729 จะมีการเปลี่ยนแปลงสัญญาณกระตุ้น v(n) และผลตอบสนองของวงจรกรอง h(n) ในกรณีที่ ค่าพิตท์ของสัญญาณเสียงน้อยกว่าความยาวของเฟรมย่อยดังนั้นฟังก์ชันจะถูกเปลี่ยน ในกรณีที่ค่าพิตท์น้อยกว่าความยาวของเฟรมย่อยดังสมการดังนี้

$$\hat{r}(k) = \widetilde{E}(k) = \begin{cases} E(k) + \beta E(k+\widetilde{T}) & k = 0,\ldots,(K-1-\widetilde{T}) \\ E(k) & k = (K-\widetilde{T}),\ldots,(\widetilde{T}-1) \\ (1-\beta p_k\, p_{k-\widetilde{T}})E(k) & k = \widetilde{T},\ldots,(K-1) \end{cases} \qquad (4.6)$$

โดยที่ T = T/5

ท้ายที่สุด เมื่อเราเลือกพื้นที่ได้แล้ว การค้นหาตำแหน่งของพัลซ์นั้นเราจะค้นหาเฉพาะพื้นที่ที่เราคาดว่าจะเกิดพัลซ์ ขึ้นเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับการคำนวณนั้น สามารถลดการคำนวณ หาค่า d , ϕ และการค้นหาได้อย่างมาก ยกตัวอย่างเช่น ถ้า M = 3 เราจะคำนวณ เฉพาะ ตำแหน่งที่เกี่ยวข้องกับ 3 พื้นที่ที่เราเลือกเท่านั้น ในตาราง เราแสดงความซับซ้อนในการคำนวณของวิธีเดิมเมื่อเทียบกับวิธี Candidate position method พบว่า การคำนวณลดลงประมาณ 27 – 48 % ยกตัวอย่างเช่น เมื่อเลือก M=5 พบว่า การค้นหาจะลดลงจาก 1440 เหลือเพียง 220 ครั้งสำหรับ G.729 และ 320 เหลือเพียง 140 สำหรับ G.729A ที่ใช้การค้นหาแบบ dept-first-search

No. of selected regions	Exhaustive search		Focused search(maximum)		Depth-first tree search	
	Original	Proposed	Original	Proposed	Original	Proposed
M	8192	$2M^5$	<1440	$<2M\lfloor 90/(8/M)^3 \rfloor$	320	$4(2M+M^2)$
4	8192	512	<1440	<96	320	96
5	8192	1250	<1440	<220	320	140

ตารางที่ 4-3 แสดงจำนวนของความเป็นไปได้ในวิธีการค้นหาแบบ ACELP สำหรับ G.729 coders

สำหรับค่า M ที่เลือกขึ้นมา จะมีผลต่อคุณภาพของเสียงที่สังเคราะห์ขึ้นเช่นเดียวกัน ถ้าเลือก M มาก จะทำให้คุณภาพเสียงไม่เพี้ยนมากนัก แต่ถ้าเลือก M น้อย ๆ พบว่าคุณภาพเสียงก็จะแย่ลง ดังนี้

Fast ACELP	M=8	M=7	M=6	M=5	M=4	M=3	M=2
MOS degradation	0	~0*	~0	~0	~0	-0.02	-0.1

* ~0: หมายถึงผู้ฟังไม่สามารถบอกระดับความแตกต่างของเสียงถอดรหัสต้นฉบับและเสียงที่ถูกแก้ไข

ตารางที่ 4.4 MOS degradation for the proposed candidate

จากการทดลองพบว่าระดับที่เหมาะสมในด้านของความซับซ้อนในการคำนวณและคุณภาพของเสียง ควรจะเลือกค่า M ที่เท่ากับ 5 แต่ถึงแม้จะลดลงไปมากกว่านี้ คุณภาพของเสียงก็ไม่แตกต่างจากเดิมมากนัก ดังนั้น สำหรับโครงงานนี้ เราจะเลือกการออพติไมซ์โดยที่ M=1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3 การปรับปรุงความเร็วโดยการแก้ไขที่เครื่องมือ (การจัดการหน่วยความจำ & DSP BIOS Tool)

งานทางด้านการประมวลผลสัญญาณดิจิตอล (Digital signal processing) นั้นเป็นงานที่ต้องใช้การคำนวณทางคณิตศาสตร์สูงดังนั้นจึงมีผู้ผลิตอุปกรณ์ที่ใช้งานทางด้านนี้โดยเฉพาะอยู่หลายเจ้าด้วยกัน เท็กซัส อินสตรูเมนต์ ก็ถือว่าเป็นบริษัทที่มีชื่อเสียงและถือส่วนแบ่งในการตลาดในการผลิตอุปกรณ์ทางด้านนี้มากกว่า 50 % ซึ่งทางผู้จัดทำได้ใช้บอร์ดรุ่น TMS320C6713 เป็นบอร์ดที่สามารถคำนวณเลขทศนิยม (floating point unit) ได้และมี เอ แอล ยู (ALU) ถึง 4 ชุด สรุปคุณสมบัติที่สำคัญ ๆ ดังนี้

	TMS320C6713-225
Cycle time (ns)	4.4
Data/Program memory (bits)	4KByte L1D Data Cache; 4KByte L1P Program Cache; 64KByte L2 Cache; 192 Kbytes L2 SRAM
SDRAM	8 MByte
DMA	16 (EDMA)
External memory interface	GDP: 32 bits
Host port /Exp. Bus / PCI	(1)16-bits HPI
McBSP	2
McASP	2
I2C	2
Timers	(2)32-bits
GPIO	(1)16-bits
Core Supply (Volts)	1.26
IO Supply (Volts)	3.3

ตารางที่ 4-5 แสดงคุณสมบัติของบอร์ดที่ใช้

จากความสามารถของบอร์ดที่ใช้ในการทดลองดังกล่าวข้างต้นจึงมีความจำเป็นที่จะต้องเข้าใจและดึงเอาความสามารถของบอร์ดที่ใช้อยู่ออกมาให้ได้มากที่สุดซึ่งแบ่งออกได้เป็น 2 กรณีดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. การออพติไมซ์โดยอาศัยออฟชันของเครื่องมือที่ใช้พัฒนา (build option)
2. การออพติไมซ์โดยใช้ความสามารถของ Real-time os ในเรื่องของการจัดการหน่วยความจำ
3. การใช้ไลบรารี ของ โอเปอร์เรชัน พื้นฐานด้านการคำนวณมาช่วย

4.3.1 การออพติไมซ์โดยอาศัยความสามารถของเครื่องมือที่ใช้พัฒนา

เนื่องจากเครื่องมือที่ใช้ในการพัฒนาโปรแกรม (Code composer studio V2.0) มีความสามารถในการเลือกระดับในการ build code ของโปรแกรมได้หลายระดับรูปแบบด้วยกันซึ่งสามารถดูผลได้จากบทที่เกี่ยวกับการทดลองทำการออพติไมซ์โมดูล G.729 โดยอาศัยเครื่องมือ Code Composer Studio V2.0

4.3.2 การออพติไมซ์โดยใช้ความสามารถของ Real-time OS ในเรื่องของการจัดการหน่วยความจำ

เมื่อเทียบเวลาในการเข้าถึงข้อมูลระหว่างหน่วยความจำภายในกับหน่วยความจำภายนอก หน่วยประมวลผลแล้วจะพบว่าการเข้าถึงข้อมูลของหน่วยความจำภายใน(Internal memory) เร็วกว่าหน่วยความจำภายนอกถึง 25 wait state (ดี เอส พี บอร์ด รุ่น TMS320C6713) ดังนั้นจึงควรที่จะนำ อัลกอริทึมไปเก็บในหน่วยความจำภายใน แต่ เนื่องจากทรัพยากรของหน่วยความจำภายในหน่วยประมวลผลนั้นมีจำกัดดังนั้นหากขนาดของโปรแกรมที่ทำการพัฒนามีขนาดใหญ่แล้วก็เป็นไปไม่ได้ที่จะนำโค้ดเซคชันและเซคชันข้อมูลทั้งหมดเข้าไปใช้ในหน่วยความจำภายในได้จึงจำเป็นที่จะต้องให้ความสำคัญในการจัดการกับทรัพยากรทางด้านนี้เป็นอย่างมากซึ่งวิธีที่ได้นำมาใช้ในการจัดการนั้นมีดังนี้

4.3.1.1 Overlay

การทำ Overlay เป็นการแบ่งเซคชันของโปรแกรมออกเป็นส่วนย่อย ๆ โดยจะโหลดเซคชันส่วนที่ใช้งานเข้าไปไว้ในหน่วยความจำหลักโดยแบ่งออกเป็นสองกรณีคือ ส่วนของโค้ดเซคชันและเซคชันข้อมูลในส่วนของโค้ดเซคชันนั้นมีปัจจัยในการพิจารณาว่าจำเป็นที่ต้องทำ overlay หรือไม่คือ

1. ดี เอส พี บอร์ด ที่ใช้นั้นมีแคช (Cache) หรือไม่ หากมีแคชส่วนนั้นมีขนาดเท่าไหร่และมีชนิดอย่างไร (Program-Cache , Data-Cache ,P/D-Cache)
2. กรณีที่มีโปรแกรมแคช หรือ P/D-Cache แล้วต้องพิจารณาต่อว่าขนาดของแคช นั้นเพียงพอที่จะเก็บส่วนของโมดูลของวัตถุที่ใหญ่ที่สุดของเราได้หรือไม่
3. หากมีขนาดของแคช เพียงพอก็ไม่จำเป็นที่จะต้องทำ Overiay โค้ดเซคชัน

เนื่องจากแคช ของ ดี เอส พี บอร์ด ที่ใช้เป็นแบบ P/D แคชซึ่งมี L1Pแคชโปรแกรม ขนาด 4KByte ซึ่งเมื่อพิจารณาดูแล้วอัลกอริทึมที่ได้พัฒนาขึ้นมานั้นฟังก์ชัน ที่มีขนาดใหญ่ที่สุดมีขนาดไม่เกิน 4Kbyte จึงไม่จำเป็นต้องทำ Overlay ในส่วนของโค้ดเซคชัน

การทำ Overlay ในส่วนของเซคชันข้อมูลนั้นเราใช้วิธีของมาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบรารี โดยการรวมกลุ่มของข้อมูลที่ใช้ในช่วงเดียวกันให้อยู่ด้วยกัน โดยอนุญาติให้ผู้ใช้งานอัลกอริทึมสามารถที่จะคัดลอกเซคชันข้อมูล (data-section) ส่วนที่ต้องการใช้เข้ามาไว้ในหน่วยความจำภายใน ได้ ส่วน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โมดูล ที่ไม่ถูกใช้งานก็สามารถคัดลอกออกไปเก็บไว้ที่พื้นที่ภายนอกได้ซึ่งการอิมพลีเมนต์ นั้นจะอยู่ในส่วนของ ALG_MOVE ในภาคผนวกเรื่อง มาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบราลี

4.3.1.2 Pragma directive

Pragma จริงแล้วก็คือการบอกอะไรบางอย่างกับ compiler เพื่อเป็นประโยชน์ในการสร้าง build code และ การลิงก์โมดูล ต่าง ๆ เช่น ใช้ในการกำหนดเซคชันของฟังก์ชัน ต่าง ๆ ว่าจะจัดกลุ่มให้อยู่ในเซคชันส่วนใดทำให้เราสามารถ แบ่งเซคชันที่ใช้ในการ inint กับ Exit ออกมาเก็บ ใน Write-once บัฟเฟอร์ได้ ซึ่งเมื่อโมดูล ส่วนนี้ถูกใช้งานไปครั้งหนึ่งแล้วก็จะสามารถนำหน่วยความจำกลับมาใช้ใหม่ ข้อดีอีกอย่างหนึ่งก็คือกรณีที่ในไฟล์หนึ่ง ๆ นั้นมีหลายฟังก์ชัน หากเราไม่ได้กำหนด pragma ให้แต่ละฟังก์ชัน แล้ว การโหลด code ขึ้นมานั้นจะถูกโหลดขึ้นมาทั้ง page หรือทั้งโมดูลของวัตถุ ทำให้เสียเวลาและพื้นที่ไปกับฟังก์ชัน ที่ไม่ได้ถูกเรียกใช้งานแต่หากเรากำหนด pragma กำหนดเซคชันให้กับฟังก์ชัน symbol ก็จะช่วยให้การโหลดนั้น Load เฉพาะฟังก์ชัน ที่ใช้งานจริง ๆ ไม่ได้โหลดมาทั้งไฟล์

4.3.3 การออพติไมซ์โดยการแทนฟังก์ชันบางฟังก์ชั่นใน math.h ด้วยไลบราลี fastrts67x.h ของ ดี เอส พี

พบว่าการใช้ไลบราลี ของดี เอส พี แทนไลบราลี ของ math.h ทำให้จำนวน Instruction ลดลง รวมทั้งขนาดของโค้ดของโปรแกรม ที่เล็กลง ดูผลการทดลองได้ในบทการทดลอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
การออกแบบและสร้างไลบรารี

เนื่องจากในการพัฒนาโปรแกรมหรืองานหนึ่ง ๆ นั้นผู้พัฒนาไม่จำเป็นต้องเริ่มพัฒนาตั้งแต่ต้นใหม่ตลอดการนำฟังก์ชัน ที่เคยเขียนมา reuse ใหม่ช่วยให้การพัฒนาเป็นไป ได้อย่างรวดเร็ว concept การทำงานในลักษณะนี้ทำให้มีผู้พัฒนาไลบรารี ออกมาขายให้กับผู้ที่ไม่ต้องการพัฒนาในรายละเอียดบางอย่าง งานด้าน Digital Signal Processing ก็เช่นเดียวกันได้มีผู้พัฒนา อัลกอริทึมต่าง ๆ ออกมาขายอย่างมากมายแต่ เมื่อมองในมุมมองของความเป็นไปได้ในการประกอบรวมอัลกอริทึมจากผู้ผลิตต่างเจ้าก็เป็นไปได้ยากหรือแทบเป็นไปไม่ได้เลยที่จะไม่ต้องแก้ไขโค้ดบางส่วนเพื่อประกอบรวมอัลกอริทึมเหล่านั้น จึงมีความสำคัญที่จะต้องมีมาตรฐานในการเรียกฟังก์ชัน สำหรับ อัลกอริทึมของ DSP นั้นจำเป็นที่จะต้องมีคุณสมบัติสำคัญอีกประการหนึ่งคือมัลติแชนเนิลซึ่งการทำงานจะเป็นหลาย ๆ Thead พร้อม ๆ กัน โดยจะต้องมีคุณสมบัติเป็น reentrant ซึ่งจำเป็นต้องมีการเก็บสถานะ และค่าของตัวแปร รวมทั้งเป็น relocatable อีกด้วย

5.1 อัลกอริทึมแบบมัลติแชนเนิล

อัลกอริทึมแบบมัลติแชนเนิลมักจะถูกใช้เพื่อตอบสนองความต้องการดัง ต่อไปนี้

- การใช้ อัลกอริทึมของ ดี เอส พี (DSP algorithm) ตัวเดียวกันกับหลายช่องสัญญาณ
- ชนิดของ อัลกอริทึมและ ช่องสัญญาณ ที่ใช้สามารถ เปลี่ยนแปลงได้ระหว่าง run-time

โดยที่ อัลกอริทึมลักษณะนี้จะต้องมีคุณสมบัติ reentrant และ relocatable

5.1.1 โปรแกรมที่มีคุณสมบัติ Reentrant

คือคุณสมบัติที่โปรแกรมนั้นอาจจะถูกเรียกเข้าไปทำงานซ้ำ ๆ หรืออาจจะถูกเรียกเข้าไปทำงานก่อนที่การเรียกก่อนหน้านั้นจะเสร็จสมบูรณ์ โดยที่การทำงานนั้นไม่ขึ้นต่อกัน หรืออีกนัยหนึ่งก็คือ ในส่วนของโค้ดเซคชันและเซคชันข้อมูลนั้นสามารถอ่านได้อย่างเดียว

โดยทั่วไปแล้วชนิดของตัวแปรมีอยู่ 2 แบบคือ

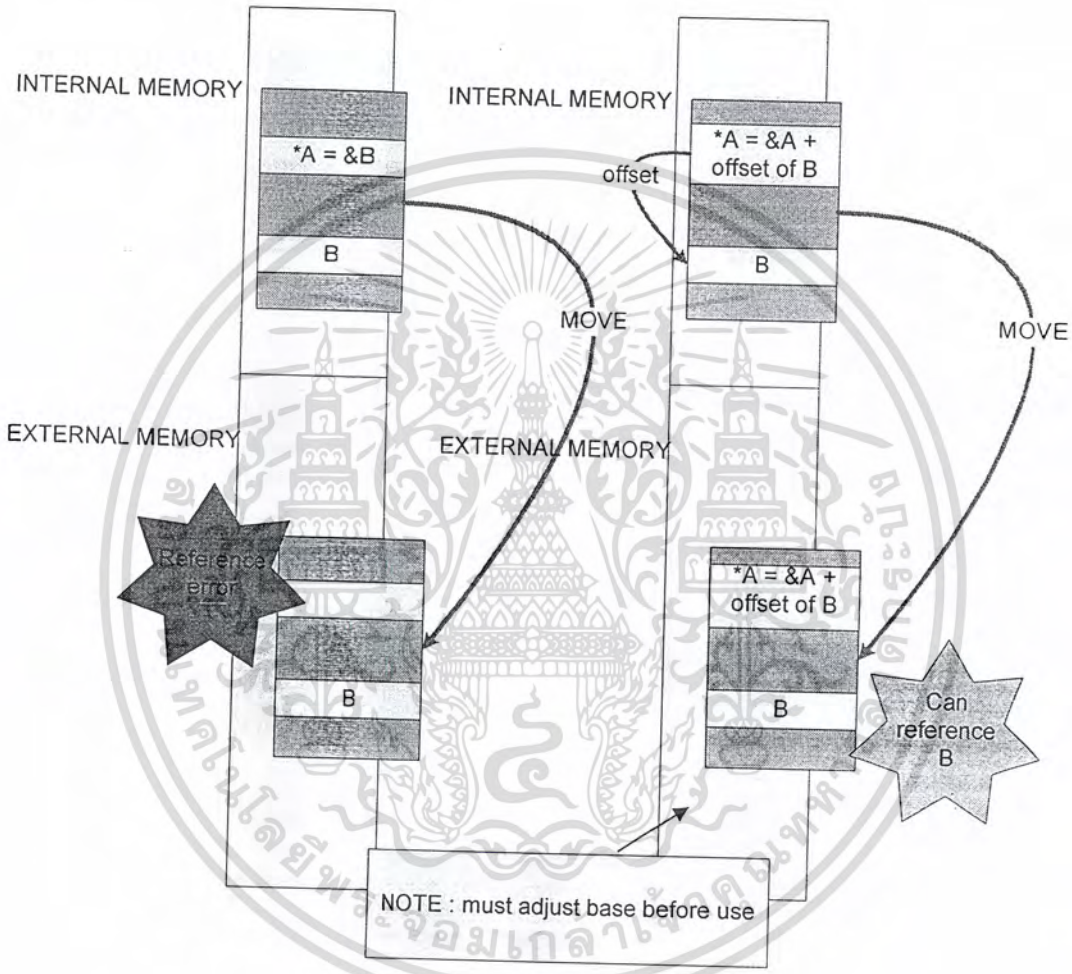
- ช่วงชีวิตของตัวแปรโลคอล (Local lifetime) ตัวแปรชนิดนี้จะถูกจองใหม่ลงใน block ที่ได้ ประกาศไว้ในแต่ละครั้งที่ execution control ผ่านและถูกคืนเมื่อรันคำสั่ง return()
- ช่วงชีวิตของตัวแปรโกลบอล (Global lifetime) ตัวแปรชนิดนี้จะถูกจองเมื่อเริ่มต้นโปรแกรม และทำการคืนเมื่อจบการทำงานของโปรแกรม

ตัวแปรชนิดโกลบอลจะต้องเป็น constant เพื่อที่จะทำให้โปรแกรม เป็น reentrant ในแต่ละ Channel จำเป็นที่จะต้องมีตัวแปรที่ใช้ในการเก็บ States โดยที่การส่งค่าตัวแปรนั้นอาศัยการส่งโดยใช้ตำแหน่งของตัวแปร (pass by reference หรือ โดย pointer)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.1.2 Relocatable

เนื่องจากปัญหาทางด้านข้อจำกัดของหน่วยความจำภายในมีขนาดไม่เพียงพอที่จะนำอัลกอริทึมทุกตัวเข้าไปทำงานพร้อมกันได้จึงจำเป็นที่ข้อมูลทุกส่วนของอัลกอริทึมต้องเป็น relocatable ซึ่งจะช่วยให้เราสามารถเคลื่อนย้าย อัลกอริทึมไปมาระหว่างหน่วยความจำภายนอกและภายใน (off-chip และ on-chip) ระหว่างการทำ context switching ได้



รูปที่ 5-1 แสดงการทำ Data relocatable

จากรูปที่ 5-1 รูปทางขวามือได้ทำการแก้ไขโปรแกรมให้มีคุณสมบัติของ Data Relocatable โดยแก้ไขตัวแปร pointer ในโครงสร้างของเราให้เก็บ offset ที่ชี้ไปยังหน่วยความจำส่วนที่อ้างถึงแทนที่จะเก็บ Absolute address เหมือนลักษณะการเขียนของรูปทางซ้ายมือซึ่งทำให้การอ้างตำแหน่งของหน่วยความจำของ pointer จะผิดหากมีการย้ายตำแหน่ง block ของหน่วยความจำ แต่รูปทางขวามือจำเป็นต้องปรับการชี้ตำแหน่งก่อนเรียกใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งในการที่เราจะทำให้เกิดคุณสมบัติที่กล่าว เราจำเป็นที่จะออกแบบโครงสร้างที่ใช้เก็บข้อมูลที่เหมาะสม โดยการออกแบบโครงสร้างนั้น เราแบ่งตามการทำงานในแต่ละ Process ที่ตัวแปรตัวนั้นเกี่ยวข้องอยู่ ซึ่งแสดงได้ดังต่อไปนี้

Memtab	Size(Byte)
coderMemBlk	10,376
preProcMemBlk	32
exc_err_mblk	32
vad_mblk	520
cod_cng_mblk	768
bwfwfunc_mblk	8
pwf_mblk	24
acelp_cp_d4i40_17_mblk	4
q_gaincp_mblk	32
phdisp_mblk	72
Total	11,868

ตารางที่ 5-1 แสดงขนาดของ Memtabแต่ละตารางหน่วยความจำของตัวเข้ารหัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Memtab	Size(Byte)
decld8cp_mblk	4,000
post_pro_mblk	32
exc_err_mblk	32
phdisp_mblk	72
pstcp_mblk	2,288
bwfwfunc_mblk	8
degaincp_mblk	32
dec_sid_mblk	96
decodecp_mblk	2,696
Total	9,256

ตารางที่ 5.2 แสดงขนาดของ Memtabแต่ละตารางหน่วยความจำของตัวถอดรหัส

Table Name	Size(Byte)
G729_DTX_TABLE	808
G729_LD8K_TABLE	3,962
G729_LD8CP_TABLE	2,260

ตารางที่ 5.3 แสดงขนาดของตารางค่าคงที่

5.1.3 การแก้ไขเชิงปฏิบัติในการทำมัลติแชนเนิล (Practical Approaches)

5.1.3.1 การแก้ไขที่ ฟังก์ชันที่มีตัวแปรสแตติกหรือโกลบอลเป็นสมาชิก (Functions with Multiple Static/Global Variables)

ตัวแปรที่เป็น โกลบอลและสแตติกต้องถูกจัดกลุ่มเข้าด้วยกันเป็นโครงสร้างดังตัวอย่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
// File Pre_Proc.c
static FLOAT x0, x1;        /* high-pass fir memory        */
static FLOAT y1, y2;        /* high-pass iir memory        */
void init_pre_process( void )
{
      x0 = x1 = (F)0.0;
      y2 = y1 = (F)0.0;
      return;
}
```

แก้ไขใหม่ดังนี้

```
typedef struct{
FLOAT x0;
FLOAT x1;
FLOAT y1;
FLOAT y2;
}preProcMemBlk,*preProcMemBlk_handle ;

void init_pre_process( preProcMemBlk_handle hmem)
{
          hmem->x0 = hmem->x1 = (F)0.0;
          hmem->y2 = hmem->y1 = (F)0.0;
        return;
}
```

ในกรณีที่โมดูล ใหญ่มีการเรียกใช้โมดูลย่อยซึ่งแต่ละโมดูล ก็มีสแตติกหรือตัวแปรชนิดโกลบอล ในกรณีนี้มีวิธีการจัดการอยู่ 2วิธีคือ

1. ประกาศตัวแปรสแตติกและโกลบอลของทุกโมดูล เป็นโครงสร้างใหญ่โครงสร้างเดียวเลยแล้วอาศัยการส่งpointer ให้แก่ทุก ๆ โมดูลย่อย (sub module)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ทำการจัดกลุ่มตัวแปรสแตติกของแต่ละ โมดูลย่อยให้แยกเป็นโครงสร้างของใครของมัน โดยให้ ตัวแปรโกลบอลและวัตถุของโมดูลย่อย อยู่ในโครงสร้างของโมดูลของโนดพาร์เรน แล้วส่งให้แก่โมดูลย่อยที่ต้องการ

```
// File codercp.c
typedef struct{
coderMemBlk_handle coder_handle;
preProcMemBlk_handle preProc_handle;
exc_err_mblk_handle exc_err_handle;
vad_mblk_handle vad_handle;
cod_cng_mblk_handle cod_cng_handle;
bwfwfunc_mblk_handle bwfwfunc_handle;
pwf_mblk_handle pwf_handle;
acelp_cp_d4i40_17_mblk_handle acelp_cp_d4i40_17_handle;
q_gaincp_mblk_handle q_gaincp_handle;
phdisp_mblk_handle phdisp_handle;

}ENCODER_G729_MEM_BLK,*ENCODER_G729_MEM_BLK_HANDLE;

void ENCODER_G729_MEM_BLK_init(ENCODER_G729_MEM_BLK_HANDLE *hmem );
void ENCODER_G729_MEM_BLK_free(ENCODER_G729_MEM_BLK_HANDLE hmem);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void ENCODER_G729_MEM_BLK_free(ENCODER_G729_MEM_BLK_HANDLE hmem){
        coderMemBlk_free(hmem->coder_handle);
        preProcMemBlk_free(hmem->preProc_handle);
        exc_err_mblk_free(hmem->exc_err_handle);
        vad_mblk_free(hmem->vad_handle);
        cod_cng_mblk_free(hmem->cod_cng_handle);
        bwfwfunc_mblk_free(hmem->bwfwfunc_handle);
        acelp_cp_d4i40_17_mblk_free(hmem->acelp_cp_d4i40_17_handle);
        pwf_mblk_free(hmem->pwf_handle);
        q_gaincp_mblk_free(hmem->q_gaincp_handle);
        phdisp_mblk_free(hmem->phdisp_handle);

        free(hmem);
}
```

5.1.3.2 ตารางค่าคงที่ (Constant Tables)

แม้ว่าตารางค่าคงที่ จะไม่ถูกแก้ไขในการประมวลผล แต่ว่ามันก็ต้อง support relocatable ด้วยเพื่อที่จะได้ย้ายเข้าไปอยู่ หน่วยความจำภายในก่อนการประมวลผลช่วยให้ประมวลผลได้รวดเร็วขึ้นและสามารถย้ายออกมาได้เมื่อพื้นที่ไม่เพียงพอ โดยทั่วไปแล้วตารางจะถูกประกาศไว้ที่หน่วยความจำข้อมูลภายนอก (external data memory) แล้วค่อยย้ายเข้ามาในหน่วยความจำภายในก่อนการประมวลผล ซึ่งต้องใช้เวลามากสำหรับ อัลกอริทึมที่มีตารางค่าคงที่ หลายตาราง ในการระบุว่า subroutine ไหนใช้ตารางไหน เราใช้วิธีการเช่นเดียวกับการจัดกลุ่มตารางที่เป็นค่าคงที่ (context data constant table) ก็สามารถทำให้ relocatable ได้ในช่วงเวลาสั้น ๆ

วิธีการ

1. ทำการจัดกลุ่มค่าคงที่เข้ามาใน โครงสร้างเพียงโครงสร้างเดียว
2. ประกาศข้อมูล pointer กับชนิดของโครงสร้างซึ่ง Pointer ตัวนี้จะเป็นตัวแบ่รชนิดโกลบอล
3. เพิ่มการนิยามข้อมูลลงไปในไฟล์ที่ก่อนแก้ไขที่มีค่าคงที่อยู่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
// Table.h
typedef struct{
/* VAD constants */
FLOAT lbf_corr[NP+1];
/* SID gain quantization */
FLOAT fact[NB_GAIN+1];
FLOAT tab_Sidgain[32];
/* SID LSF quantization */
int PtrTab_1[32];
int PtrTab_2[2][16];
FLOAT noise_fg[MODE][MA_NP][M];
FLOAT noise_fg_sum[MODE][M];
FLOAT noise_fg_sum_inv[MODE][M];

FLOAT Mp[MODE];
}G729_DTX_TABLE,*G729_DTX_TABLEPTR;

G729_DTX_TABLEPTR G729_DTX_TablePtr ;
void initTab_dtx();
/* VAD constants */
#define lbf_corr G729_DTX_TablePtr->lbf_corr
/* SID gain quantization */
#define fact G729_DTX_TablePtr->fact
#define tab_Sidgain G729_DTX_TablePtr->tab_Sidgain
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* SID LSF quantization */
#define PtrTab_1 G729_DTX_TablePtr->PtrTab_1
#define PtrTab_2 G729_DTX_TablePtr->PtrTab_2
#define noise_fg G729_DTX_TablePtr->noise_fg
#define noise_fg_sum G729_DTX_TablePtr->noise_fg_sum
#define noise_fg_sum_inv G729_DTX_TablePtr->noise_fg_sum_inv
#define Mp G729_DTX_TablePtr->Mp
```

5.1.3.3 Data Relocation

กรณีที่มีตัวแปรที่เป็น static/global/constant pointers ต้องทำการปรับวิธีการอ้างถึงค่าของตัวแปร โดยการเก็บoff set แทน โดยอาศัย base จากตำแหน่งของโครงสร้างที่ถูกจัดกลุ่ม

ตัวอย่าง

```
/*-----------------------------------------------*
 *       Staticหน่วยความจำallocation.            *
 *-----------------------------------------------*/

/* Speech vector */
static FLOAT old_speech[L_TOTAL];
static FLOAT *speech, *p_window;
FLOAT  *new_speech;              /* Global variable */
/* Weighted speech vector */
static FLOAT old_wsp[L_FRAME+PIT_MAX];
static FLOAT *wsp;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 /* Filter's memory */
 static FLOAT mem_err[M_BWD+L_SUBFR], *error;
 /* Weighted speech vector */
 static FLOAT old_wsp[L_FRAME+PIT_MAX];
 static FLOAT *wsp;


 /* Excitation vector */
 static FLOAT old_exc[L_FRAME+PIT_MAX+L_INTERPOL];
 static FLOAT *exc;
 /* Zero vector */
 static FLOAT ai_zero[L_SUBFR+M_BWDP1];
 static FLOAT *zero;

void init_coder_ld8c(
   int dtx_enable   /* input : DTX enable flag */
){
  new_speech = old_speech + L_TOTAL - L_FRAME;      /* New speech     */
  speech     = new_speech - L_NEXT;                 /* Present frame  */
  p_window   = old_speech + L_TOTAL - L_WINDOW;     /* For LPC window */

  /* Initialize static pointers */
  wsp    = old_wsp + PIT_MAX;
  exc    = old_exc + PIT_MAX + L_INTERPOL;
  zero   = ai_zero + M_BWDP1;
  error  = mem_err + M_BWD;
return ;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แก้ไขใหม่ได้ดังนี้

```
void coderMemBlk_init(coderMemBlk_handle *hmem){


FLOAT lsp_oldTmp[M]=
{ (F)0.9595, (F)0.8413, (F)0.6549, (F)0.4154, (F)0.1423,
(F)-0.1423, (F)-0.4154, (F)-0.6549, (F)-0.8413, (F)-0.9595};
int lag_bufTmp[5]={20,20, 20, 20,20};
FLOAT pgain_bufTmp[5]={(F)0.7,(F)0.7, (F)0.7, (F)0.7,(F)0.7};


  (*hmem) = (coderMemBlk *)malloc(sizeof(coderMemBlk));
        copy(lsp_oldTmp,(*hmem)->lsp_old,M);
        copyArrI(lag_bufTmp,(*hmem)->lag_buf , 5);
        copy(pgain_bufTmp,(*hmem)->pgain_buf , 5);


        /* New speech    */
  (*hmem)->new_speech = (FLOAT *)(((char *)&((*hmem)->old_speech)) + ((L_TOTAL -
L_FRAME)*sizeof(FLOAT)) -(int)(*hmem));
    /* Present frame  */
  (*hmem)->speech    = (FLOAT *)(((char *)&((*hmem)->new_speech ))-
(L_NEXT*sizeof(FLOAT)) -(int)(*hmem));
    /* For LPC window */
  (*hmem)->p_window   = (FLOAT *)(((char *)&((*hmem)->old_speech ))+ ((L_TOTAL -
L_WINDOW)*sizeof(FLOAT))-(int)(*hmem));
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
   /* Initialize static pointers */
   (*hmem)->wsp   = (FLOAT *)(((char *)&((*hmem)->old_wsp ))+
(PIT_MAX*sizeof(FLOAT))-(int)(*hmem));
   (*hmem)->exc   = (FLOAT *)(((char *)&((*hmem)->old_exc ))+ ((PIT_MAX +
L_INTERPOL)*sizeof(FLOAT))-(int)(*hmem));
   (*hmem)->zero  = (FLOAT *)(((char *)&((*hmem)->ai_zero ))+
(M_BWDP1*sizeof(FLOAT))-(int)(*hmem));
   (*hmem)->error = (FLOAT *)(((char *)&((*hmem)->mem_err ))+
(M_BWD*sizeof(FLOAT))-(int)(*hmem));
         return ;
}
```

5.2 Abstract algorithm interface

การสร้างไลบราลี ตามมาตรฐานของ เท็กซัส อินสตรูเมนต์ นั้น เราจะนำโค้ดที่เราเขียนด้วยภาษา C นั้น มาครอบด้วย interface ของผู้ผลิต เพื่อสามารถนำไลบราลี นั้น ไปใช้งานกับแอพพลิเคชันอื่น ๆ ได้ โดยไม่ต้องเปลี่ยนแปลงโค้ดมาก โดย interface ดังที่กล่าวมานั้น มีประโยชน์สำหรับการจัดการหน่วยความจำ ของไลบราลี นั้น ๆ อีกด้วย

โดยส่วนที่สำคัญคือ Abstract Algorithm Interfaces นั้น เป็น interface ที่มีลักษณะเป็น polymorphism โดย แต่ละไลบราลี จะมี Abstract Algorithm Interfaces หรือ IALG เดียวกัน แต่มีการอิมพลีเมนต์ ต่างกัน โดยจะมีการชี้ pointer ของ IALG ของแต่ละไลบราลี ไปยังฟังก์ชัน ที่เฉพาะของแต่ละไลบราลี

IALG นั้นจะต้องถูกอิมพลีเมนต์ สำหรับทุก ๆ ไลบราลี ที่เราต้องการใช้งาน เพื่อใช้สำหรับการจัดการหน่วยความจำและเพิ่มประสิทธิภาพของการใช้หน่วยความจำ ภายใน chip

ทุก ๆ ไลบราลี นั้นจะต้องมีการกำหนดค่าเริ่มต้นให้กับโครงสร้างของ IALG_Fxns ซึ่งเป็น pointer ที่ใช้ไปยังแต่ละฟังก์ชัน ที่เฉพาะเจาะจงสำหรับแต่ละไลบราลี ซึ่งหมายความว่า pointer ของ IALG_Fxns จะต้องชี้ไปยังฟังก์ชัน ที่จะต้อง Implement

สำหรับฟังก์ชัน ของ IALG_Fxns นั้นแบ่งออกเป็น 3 ประเภท ดังนี้

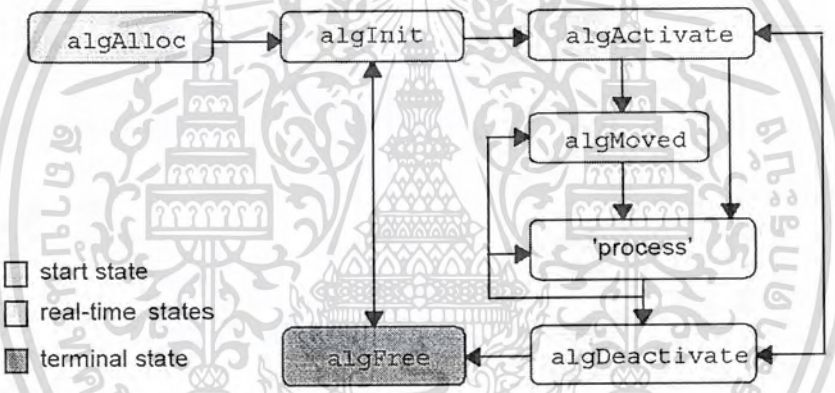
1) การสร้างวัตถุของไลบราลี นั้น การกำหนดค่าเริ่มต้นและการทำลาย
2) การทำงานของไลบราลี
3) การควบคุมการทำงานของไลบราลี และ relocation

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การสร้างวัตถุนั้นมีความซับซ้อน เพราะว่ามีวัตถุมากมายที่ทำงานร่วมกันในแอพพลิเคชันดังนั้นการตัดสินใจเกี่ยวกับ memory overlays และการ preempt จะขึ้นอยู่กับผู้ประกอบแอพพลิเคชันนั้น ๆ จึงมีความจำเป็นที่จะต้องมี interface ที่ให้ผู้ประกอบแอพพลิเคชันได้ติดต่อกับตัวไลบราลี นั้น ๆ ได้แก่ algAlloc () , algInit () และ algFree ()

เมื่อวัตถุถูกสร้างขึ้น มันสามารถประมวลผลข้อมูลได้ เมื่อผู้ประกอบแอพพลิเคชันเป็นผู้เรียก algActivate () เพื่อเริ่มต้นการทำงานของวัตถุนั้น และเมื่อวัตถุนั้นทำงานสิ้นสุดลง ผู้ประกอบแอพพลิเคชันจะเรียก algDeactivate () เพื่อเก็บสถานะ และข้อมูลทั้งหมดในหน่วยความจำชั่วคราว (scratch memory) ไปยังหน่วยความจำถาวร (persistant memory) เพื่อให้ไลบราลี อื่นสามารถใช้งานหน่วยความจำชั่วคราวส่วนนั้นได้

อีกสองฟังก์ชัน คือ algControl () และ algMoved () สำหรับ algControl นั้นใช้สำหรับควบคุมและตรวจดูสถานะของวัตถุนั้น และ algMoved () ใช้สำหรับผู้ประกอบแอพพลิเคชันที่จะย้ายวัตถุนั้นไปยังหน่วยความจำในตำแหน่งอื่น ๆ



รูปที่ 5-2 แสดงลำดับการทำงานของวัตถุ

5.2.1 algActivate

algActivate	กำหนดค่าเริ่มต้นของหน่วยความจำแบบ scratch ก่อนที่จะทำการประมวลผล
Synopsis	Void algActivate (IALG_Handle handle);
Arguments	IALG_Handle handle; /* algorithm instance handle */
Return Value	Void

algActivate ใช้สำหรับการกำหนดค่าเริ่มต้นของหน่วยความจำชั่วคราวโดยใช้ค่าของหน่วยความจำถาวรซึ่งเป็นส่วนหนึ่งของวัตถุนั้น โดยอาร์กิวเมนต์ที่ส่งเข้ามานั้นจะเป็นตัวกำหนดว่าเป็นหน่วยความจำของวัตถุใด ก่อนที่จะเรียกวัตถุนั้นให้ทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2.2 algInit

algInit อธิบายความต้องการหน่วยความจำในส่วนต่าง ๆ ของวัตถุของไลบรารี นั้น

Synopsis
```
numRecs = algAlloc (const IALG_Params *params , IALG_Fxns **parentFxns ,
    IALG_MemRec memTab []);
```

Arguments
```
IALG_Params *params;          /* algorithm specific attribute */
IALG_Fxns **parentFxns;       /* output parent algorithm functions */
IALG_MemRec memTab[] ;        /* output array of mem records */
```

Return Value
```
Int numRecs;                  /* number of initialized record in memTab[] */
```

algAlloc () มีการส่งค่าตารางของหน่วยความจำที่อธิบายขนาด, alignment, ชนิดของข้อมูลและประเภทของหน่วยความจำที่ใช้ของแต่ละไลบรารี เมื่อมีการทำงานเสร็จสิ้น จะมีการส่งค่าที่เป็นบวก และไม่ใช่ศูนย์ออกมา คือจำนวนของ record ที่ใช้ในการ initialize ซึ่งฟังก์ชัน นี้ไม่สามารถกำหนดค่าเริ่มต้นของหน่วยความจำที่มีจำนวน record มากกว่าจำนวนที่ถูกส่งออกมาจาก algNumAlloc () ถ้า algNumAlloc ไม่ได้ถูกอิมพลีเมนต์ จะสามารถกำหนดค่าเริ่มต้นได้มากกว่าจำนวนที่มากที่สุดที่กำหนดไว้ คือ IALG_DEFMEMRECS

ตัวอย่าง

```
Int G729ENC_KMITL_alloc(const IALG_Params *G729ENCParams, IALG_Fxns **fxns,
IALG_MemRec memTab[])
{
  const IG729ENC_Params *params = (Void *)G729ENCParams;
  /* TODO: implement algAlloc */
    if (params == NULL) {
    params = &IG729ENC_PARAMS;  /* set default parameters */
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* Request memory for G729ENC object */
    memTab[0].size = sizeof(G729ENC_KMITL_Obj);
    memTab[0].alignment = (0 * 8) / CHAR_BIT;
    memTab[0].space = IALG_EXTERNAL;
    memTab[0].attrs = IALG_PERSIST;

/*************** WORK BUFFER *****************/
        /* G729 Implementation coderMemBlk */
    memTab[1].size = sizeof(coderMemBlk);
    memTab[1].alignment = (0 * 8) / CHAR_BIT;
    memTab[1].space = IALG_DARAM0;
    memTab[1].attrs = IALG_SCRATCH;

   /*************** HISTORY BUFFER *****************/
   /* G729 Implementation coderMemBlk */
    memTab[11].size = sizeof(coderMemBlk);
    memTab[11].alignment = (0 * 8) / CHAR_BIT;
    memTab[11].space = IALG_EXTERNAL;
    memTab[11].attrs = IALG_PERSIST;

  return (MTAB_NRECS);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2.3 algControl

Synopsis retval = algControl(IALG_Handle handle,
IALG_Cmd cmd, IALG_Status *status);

Arguments IALG_Handle handle; /* algorithm instance handle */
IALG_Cmd cmd; /* algorithm specific command */
IALG_Status *status; /* algorithm specific status */

Return Value Int retval;

algControl ทำหน้าที่ส่งคำสั่งเฉพาะรวมทั้งสถานะของอินพุท/เอาท์พุท ให้กับอาร์กิวเมนต์ของอัลกอริทึมตัวแรกของ algControl() คือพอยน์เตอร์ของวัตถุของอัลกอริทึมซึ่งสามารถเรียกได้หลังจาก การเรียก algInit () แล้ว ถูกเรียกก่อน algActivate() ต้องไม่ถูกเรียกหลังการเรียก algFree() อาร์กิวเมนต์ตัวที่สอง และสามเป็นพารามิเตอร์ของ อัลกอริทึมและในส่วนของ command values ต้องมีค่าน้อยกว่า IALG_SYSCMD เสมอ ค่าที่มากกว่าจะถูกสงวนไว้สำหรับ version ที่ใหม่กว่าของ อินเตอร์เฟสของ IALG หากฟังก์ชัน ทำงานเสร็จสิ้นโดยสมบูรณ์ก็จะ return IALG_EOK ไม่เช่นนั้นก็จะ return IALG_EFAIL หรือค่าเฉพาะของความผิดผลาดนั้น ๆ ในสภาพแวดล้อมแบบ preemptive algControl อาจจะ preempt ฟังก์ชั่นอื่น ๆ ของโมดูล ได้เช่น ทำการ preempt processing methods

ในการอิมพลีเมนต์ตัว algControl() นั้นเป็นออพชันให้เราเลือกว่าจะอิมพลีเมนต์ หรือไม่ก็ได้หากไม่ก็กำหนดค่าในตารางของฟังก์ชัน ให้ชี้ไปที่ NULL

ตัวอย่าง

```
Int G729ENC_KMITL_control(IALG_Handle handle, IALG_Cmd cmd, IALG_Status * status)
{
   IG729ENC_Status *sPtr = (IG729ENC_Status *)status;
   G729ENC_KMITL_Obj *G729ENC = (Void *)handle;
   switch ((IG729ENC_Cmd)cmd) {
     case IG729ENC_GETSTATUS:
        sPtr->vadEnable = G729ENC->vadEnable;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    sPtr->annexC = G729ENC->annexC;
        sPtr->annexD = G729ENC->annexD;
        sPtr->annexE = G729ENC->annexE;
        return(IALG_EOK);

    case IG729ENC_SETSTATUS:
        G729ENC->vadEnable = sPtr->vadEnable;
        G729ENC->annexC = sPtr->annexC;
        G729ENC->annexD = sPtr->annexD;
        G729ENC->annexE = sPtr->annexE;
        return(IALG_EOK);
    default: break;
  }
        return(IALG_EFAIL);
}
```

5.2.4 algDeactivate

Synopsis Void algDeactivate(IALG_Handle handle);

Arguments IALG_Handle handle; /* algorithm instance handle */

Return Value Void

algDeactivate() บันทึกข้อมูลสถานะลงในหน่วยความจำถาวรโดยอาศัยหน่วยความจำถาวรซึ่งเป็นพื้นที่ส่วนหนึ่งในวัตถุของอาร์กิวเมนต์ของอัลกอริทึม ก็คือ pointer ของ ตัววัตถุของ อัลกอริทึมซึ่งใช้ในการระบุบัฟเฟอร์ที่จะใช้เก็บข้อมูลก่อนรอบถัดไปของ algActivate() และ ส่วนการคำนวนในการอิมพลีเมนต์ ตัว algDectivate() นั้นถือเป็นออพชันซึ่งจะทำก็ต่อเมื่อต้องการถ้าโมดูล ต้องการกำหนดค่าเริ่มต้นของโค้ดที่สามารถทำงานไปแล้วครั้งหนึ่งก่อนเพื่อประมวลผลเฟรมข้อมูล หลาย ๆ ลำดับต่อเนื่องกันไป แต่หากไม่ก็กำหนดค่าในตารางของฟังก์ชัน ให้ชี้ไปที่ NULL

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2.5 algFree

Synopsis numRecs = algFree(IALG_Handle handle, IALG_MemRec memTab[]);

Arguments IALG_Handle handle: /* algorithm instance handle */

IALG_MemRec memTab[]; /* output array of mem records */

Return Value Int numRecs; /* number of initialized records in memTab[] */

algFree() คืนพื้นที่ในส่วนของ การบรรยายขนาด base address , alignment ,ชนิด และพื้นที่ของทุก ๆ buffers ที่ใช้ก่อนทำการจอง algorithm's instance รวมทั้ง algorithm's instance เองด้วย ฟังก์ชั่นนี้จะ returns ค่าบวกเสมอ

ตัวอย่าง

```
typedef struct EncoderObj {
IALG_Obj ialgObj /* IALG object MUST be first field */
Int *workBuf;
Int workBufLen;
...;
} EncoderObj;
Int algFree(IALG_Handle handle, IALG_MemRec memTab[])
{
EncoderObj *inst = (EncoderObj *)handle;
algAlloc(NULL, memTab); /* get default values first */
memTab[0] .size = sizeof (EncoderObj);
memTab[0] .base = inst;
memTab[1].size = inst->workBufLen * sizeof(Int);
memTab[1].base = (Void *)inst->workBuf;
return(2);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Int G729ENC_KMITL_free(IALG_Handle handle, IALG_MemRec memTab[])
{
   Int n;
   G729ENC_KMITL_Obj *G729ENC = (Void *)handle;
   n = G729ENC_KMITL_alloc(NULL, NULL, memTab);
   memTab[0].base = handle;
   return (n);
}
```

5.2.6 algInit

Synopsis status = algInit(IALG_Handle handle, IALG_MemRec memTab[],
IALG_Handle parent, IALG_Params *params,);

Arguments
```
IALG_Handle handle;        /* algorithm's instance handle */
IALG_memRec memTab[];      /* array of allocated buffers */
IALG_Handle parent;        /* handle algorithm's parent instance */
IALG_Params *params;       /* ptr to algorithm's instance args */
```

Return Value `Int status;` /* status indicating success or failure */

algInit() ทำการกำหนดค่าเริ่มต้นให้กับ อัลกอริทึมในการสร้างวัตถุขณะรันไทม์ ก่อนการประมวลผลข้อมูล อาร์กิวเมนต์ตัวแรกของ algInit() คือ handle ของวัตถุของ อัลกอริทึมเพื่อใช้กำหนดค่าเริ่มต้นให้กับโครงสร้างของ IALG_Obj ซึ่งถูกระบุโดย memTab[0].base อาร์กิวเมนต์ตัวที่ 2 ก็คือตารางของ memory records ที่บรรยาย base address, size, alignment, type และขนาดของพื้นที่ที่ใช้ทั้งหมด จำนวนของ initialized records ถูกระบุโดยจำนวนที่ถูก return จาก algAlloc() ก่อนหน้านี้ อาร์กิวเมนต์ตัวที่ 3 คือ handle ของวัตถุของ อัลกอริทึมตัวอื่นซึ่งมักจะเป็น NULL ซะเป็นส่วนใหญ่ นั่นหมายถึงไม่มีวัตถุพาเรนท์ ตัวแปรนี้ช่วยให้ clients สามารถใช้วัตถุของ อัลกอริทึมร่วมกันได้ ตัวอย่างเช่นวัตถุพาเรนท์อาจจะมีตารางค่าคงที่ที่ใช้ร่วมกันให้กับโมดูลเข้ารหัสหลาย ๆ ชนิดเป็นต้น อาร์กิวเมนต์ตัวสุดท้ายเป็น Pointer ไปยังพารามิเตอร์ของอัลกอริทึมที่จำเป็นในการสร้าง และ กำหนดค่าเริ่มต้นของวัตถุซึ่งเป็นตัวเดียวกันกับที่ส่งให้กับ algAlloc() ซึ่งสามารถเป็น NULL ได้ในกรณีที่ยึดเอาดีฟอลพารามิเตอร์ของอัลกอริทึมส่วน client ต้องไม่กำหนดว่าจะไปอยู่ในหน่วยความจำที่จำเพาะ แต่อย่างไรก็ตามกรณีของ อัลกอริทึมที่ใช้ DMA อาจจะขอให้จองพื้นที่ในส่วนของหน่วยความจำภายในได้ Client อาจจะจองพื้นที่ส่วนที่เป็น IALG_WRITEONCE บัพเพอร์ได้และใช้พื้นที่ส่วนนี้ใช้ร่วมกัน หลาย ๆ วัตถุ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่าง

```
typedef struct EncoderObj {
IALG_Obj ialgObj                /* IALG object MUST be first field */
int workBuf;                    /* pointer to on-chip scratch memory */
Int workBufLen;                 /* workBuf length (in words) */
....;
} EncoderObj;
Int algInit(IALG_Handle handle,
IALG_MemRec memTab[], IALG_Handle p, IALG_Params *algParams)
{
EncoderObj *inst = (EncoderObj *)handle;
EncoderParams *params = (EncoderParams *)algParams;
if (params == NULL) {
params = &ENCODERATTRS; /* use default parameters */
}
inst->workBuf = memTab[1].base;
inst->workBufLen = params->frameDuration * 8;
....
return (IALG_EOK);
}
```

5.2.7 algMoved

Synopsis

```
Void algMoved (IALG_Handle handle,
 IALG_MemRec memTab [ ], IALG_Handle
parent, const IALG_Params *params);
```

Arguments

```
IALG_Handle handle;          /* algorithm's instance handle */
IALG_Handle parent;          /* handle algorithm's parent instance */
IALG_Params *params;         /*ptr to algorithm's instance args */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

IALG_memRec memTab []; /* array of allocated buffers */

Return Value Void

algMoved() จัดการในส่วนของการกำค่าเริ่มต้นใหม่ในกรณีที่วัตถุถูกเคลื่อนย้ายไปโดยผู้ประกอบอัลกอริทึม ข้อมูลที่อยู่ภายในหน่วยความจำภายในทุกตัวจะต้องถูกปรับตำแหน่ง ตำแหน่งใหม่ อาร์กิวเมนต์ของ algMoved() ถูกระบุโดยอาร์กิวเมนต์ที่ถูกส่งมายัง algInit() ความจริงแล้วหลาย ๆ กรณีที่ตัวอัลกอริทึมอาจจะใช้ฟังก์ชัน algInit() ในการอิมพลีเมนต์ algMoved() อย่างไรก็ตามสิ่งที่ต้องตระหนักถึงก็คือ algMoved() นั้นถูกเรียกในขณะที่ อัลกอริทึมทำงาน (real-time)แต่ algInit() ไม่ การ initialization บางอย่างที่ algInit() อาจจะไม่จำเป็นใน algMoved() จึงเป็นหน้าที่ของผู้ประกอบ อัลกอริทึมที่ต้องคัดลอกสถานะของอัลกอริทึมก่อนในการที่จะย้ายตำแหน่งของ ตัว อัลกอริทึม และเปลี่ยนแปลงตำแหน่งของข้อมูลที่อยู่ในหน่วยความจำภายใน แม้ว่า parameters ของ อัลกอริทึมจะถูกส่งผ่านเข้าไปยัง algMoved() แต่ก็มีข้อยกเว้นในส่วนของ ค่า pointer ค่าเหล่านั้นจะต้องถูกส่งเข้า algInit() ข้อมูลใด ๆ ในโครงสร้างของพารามิเตอร์ที่ถูกอ้างถึงโดยการใช้ pointer ต้องถูกส่งเข้าไปใน algInit() เนื่องจากการอ้างตำแหล่งอาจจะมีการเปลี่ยนแปลงได้ ซึ่งการอิมพลีเมนต์ในส่วนของ algMoved() นั้นเป็น Optional แต่ก็ถือว่ามีความสำคัญเป็นอย่างยิ่งที่ต้องอิมพลีเมนต์หากโมดูล ใดที่ไม่ได้อิมพลีเมนต์ก็จะไม่สามารถย้ายตัว อัลกอริทึมได้

5.2.8 algNumAlloc()

Synopsis num = algNumAlloc(Void);

Arguments Void

Return Value Int num; /* number of allocation requests for algAlloc() */

1. algNumAlloc() returns ค่า Block ของหน่วยความจำที่มากที่สุดที่สามารถจองได้
2.ฟังก์ชัน algAlloc() จำเป็นที่จะต้องเรียก algNumAlloc() เพื่อดูว่าผู้ประกอบ อัลกอริทึมต้องการจำนวนหน่วยความจำ blockเท่าใด กรณีที่เกิดการ ผิดผลาดขึ้นเนื่องจาก พื้นที่ไม่เพียงพอ
3. algNumAlloc() อาจจะถูกเรียกเมื่อใดก็ได้และต้องไม่เปลี่ยนแปลงค่า
4. algNumAlloc() เป็น optional ในการเลือกที่จะอิมพลีเมนต์หรือไม่หาก ไม่ก็จะ return ค่ามากสุดในการจองมาให้ซึ่ง define โดยตัวแปร IALG_DEFMEMRECS.

ตัวอย่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*The example below shows how an algorithm can use another algorithm.*/
#define NUMBUF 3 /* max number of my memory requests */
extern IALG_Fxns *subAlg; /* sub-algorithm used by this alg */
Int algNumAlloc(Void)
{
return (NUMBUF + subAlg->algNumAlloc());
}
Int algAlloc(const IALG_Params *p, struct IALG_Fxns **pFxns,
IALG_MemRec memTab)
{
Int n;
/* initialize my memory requests */
...
/* initialize sub-algorithm's requests */
n = subAlg->algAlloc(..., memTab);
return (n + NUMBUF);
}
```

5.3 ลิงก์เกอร์คอมมานด์ (Linker Command)

ในการเขียนลิงก์เกอร์คอมมานด์จำเป็นที่ต้องรู้จักองค์ประกอบ 2 ส่วนก็คือลิงก์เกอร์(Linker) และอาร์ไครฟ์เวอร์ (Archiver) ดังนั้นในหัวข้อนี้จะอธิบายหลักการเบื้องต้นของลิงก์เกอร์ และอาร์ไครฟ์เวอร์ ดังต่อไปนี้

5.3.1 อาร์ไครฟ์เวอร์ (Archiver)

ในการสร้างไลบรารี เราสามารถสร้างได้จากไฟล์หลาย ๆ ชนิดทั้ง Assembler และลิงก์เกอร์จำเป็นต้องมี archive libraries เป็นอินพุท โดย Assembler ใช้ไลบรารี ที่ประกอบไปด้วยไฟล์ต้นฉบับและในขณะที่ลิงก์เกอร์ใช้ไลบรารี ที่ประกอบไปด้วยไฟล์ออบเจค

โดยหนึ่งในแอพพลิเคชันที่มีความสำคัญมากที่สุดก็คืออาร์ไครฟ์เวอร์ซึ่งใช้สำหรับสร้างไลบรารี ของโมดูลของวัตถุ ยกตัวอย่างเช่น เมื่อเราเขียนฟังก์ชัน ที่เกี่ยวข้องกับคณิตศาสตร์หลาย ๆ ฟังก์ชัน เราต้องการเก็บรวบรวมฟังก์ชัน เหล่านี้ไว้ด้วยกันเราจะใช้อาร์ไครฟ์เวอร์ในการรวบรวม ซึ่งเราสามารถกำหนดวัตถุของไล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บราลี เฉพาะฟังก์ชัน คณิตศาสตร์เหล่านั้นเป็นอินพุท ของอาร์ไครฟ์เวอร์โดยลิงก์เกอร์จะทำหน้าที่ค้นหาไลบราลี และไฟล์ที่เกี่ยวข้องกับไลบราลี ทางคณิตศาสตร์ที่เราจะสร้าง เราสามารถใช้อาร์ไครฟ์เวอร์ในการสร้างมาโครไลบราลีได้อีกด้วยโดยเราจะสร้างจากหลาย ๆ ไฟล์ต้นฉบับซึ่งแต่ละไฟล์ต้นฉบับนั้นประกอบด้วย 1 macro เราใช้อาร์ไครฟ์เวอร์ในการเก็บ macro เหล่านั้นเข้าไว้ด้วยกันซึ่งเราใช้ .mlib directive ในการกำหนดว่า macro ไหนควรจะถูกใช้ในอาร์ไครฟ์เวอร์

5.3.2ลิงก์เกอร์ (Linker)

ลิงก์เกอร์ของThe TMS320C6000 อนุญาติให้เราสามารถคอนฟิกหน่วยความจำของระบบไว้ที่เอาท์พุทเซคชันเพื่อเพิ่มประสิทธิภาพของ Memory map ขณะที่ลิงก์เกอร์เชื่อมไฟล์ออบเจคต่าง ๆ เข้าด้วยกันจะมีลำดับขั้นตอนดังนี้

1. จองเซคชันของหน่วยความจำในบอร์ด(target system's configured memory)
2. Relocate Symbol และเซคชันที่จะ Assign ให้กับ Symbol ต่าง ๆ ไว้ที่ตำแหน่งปลายทาง
3. หาตำแหน่งของ Symbols ที่อ้างไปยังไฟล์อื่น

ลิงก์เกอร์คอมมานด์ทำหน้าที่ควบคุมการคอนฟิกหน่วยความจำ, เอาท์พุทเซคชัน และตำแหน่งที่อ้างถึง เราสามารถคอนฟิกหน่วยความจำของระบบได้ โดยการกำหนดและสร้างแบบจำลองหน่วยความจำซึ่งมี 2 directive ที่เกี่ยวข้องคือหน่วยความจำและเซคชันซึ่งมีขั้นตอนในการสร้างแบบจำลองหน่วยความจำดังต่อไปนี้

1. ระบุและจองเซคชัน ในหน่วยความจำส่วนต่าง ๆ
2. ลิงก์แต่ละเซคชันเข้ากับไฟล์ออบเจค
3. Define หรือ redefine global symbols ในขณะลิงก์

5.3.3 ตัวอย่างการลิงก์ไลบราลี

ในการสร้างไลบราลี นั้นเมื่อเราได้ Compile และ build โปรเจคแบบ Release mode เราจะได้ไฟล์ออบเจคมาหลังจากนั้นเราต้องเขียนไฟล์สคริปเพื่อลิงก์ไฟล์ออบเจคต่าง ๆ ที่ใช้เป็นไลบราลี ตัวอย่างของไฟล์สคริปมีดังนี้

```
@echo NOTE: Please execute the c:\ti\DosRun.bat file located in
lnk6x g729enc.cmd
erase g729enc_kmitl.l67
erase g729enc.l67
ar6x -r g729enc_kmitl.l67 g729enc.o67
ar6x -r g729enc.l67 g729enc.obj
```

รูปที่ 5-3 แสดงลิงก์เกอร์คอมมานด์ไฟล์makelib.bat

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

lnk6x เป็นลิงก์เกอร์ที่รับพารามิเตอร์ 2 ตัว ตัวแรกบอกถึง Optional ส่วน ตัวที่ 2 คือชื่อไฟล์ที่จะทำการ ลิงก์มีมากกว่า 1 ก็ได้

aAr6x เป็นคำสั่งที่ใช้สำหรับอาร์ไครฟ์เวอร์ซึ่งมีออพชันที่บอกอาร์ไครฟ์เวอร์ว่าจะจัดการกับสมาชิกของไลบรารี ที่ปรากฏอยู่และชื่อไฟล์ได้อย่างไรซึ่ง

Option -r ใช้สำหรับแทนชื่อของสมาชิกในไลบรารี ดังนั้นถ้าเราไม่กำหนดชื่อไฟล์ที่เฉพาะลงไป achiver จะแทนชื่อสมาชิกของไลบรารี ด้วยไฟล์ที่มีชื่อเช่นเดียวกันใน directory ปัจจุบัน แต่ถ้าเกิดชื่อไม่มีอยู่ในไลบรารี Achiver จะเพิ่มเติมชื่อลงไปแทนที่จะแทนที่มัน

Option	Description
–a	Produces an absolute, executable module. This is the default; if neither –a nor –r is specified, the linker acts as if –a were specified.
–ar	Produces a relocatable, executable object module.
–b	Disables merge of symbolic debugging information.
–c	Autoinitializes variables at run time.
–cr	Initializes variables at load time.
–e *global_symbol*	Defines a global symbol that specifies the primary entry point for the output module.
–f *fill_value*	Sets the default fill value for holes within output sections; *fill_value* is a 32-bit constant.
–g *symbol*	Makes *symbol* global (overrides –h).
–h	Makes all global symbols static.
–heap *size*	Sets heap size (for the dynamic memory allocation in C) to *size* words and defines a global symbol that specifies the heap size. Default = 1K words.
–i *pathname*†	Alters the library-search algorithm to look in a directory named with *pathname* before looking in the default location. This option must appear before the –l option.
–j	Disables conditional linking.
–l *filename*†	Names an archive library or linker command *filename* as linker input.
–m *filename*†	Produces a map or listing of the input and output sections, including holes, and places the listing in *filename*.
–n	Ignores all fill specifications in MEMORY directives.
–o *filename*†	Names the executable output module. The default filename is a.out.
–q	Suppresses the banner and all progress information (linker runs in quiet mode).
–r	Produces a nonexecutable, relocatable output module.
–s	Strips symbol table information and line number entries from the output module.
–stack *size*	Sets C system stack size to *size* words and defines a global symbol that specifies the stack size. Default = 1K words.
–u *symbol*	Places an unresolved external *symbol* into the output module's symbol table.
–w	Displays a message when an undefined output section is created.
–x	Forces rereading of libraries, which resolves back references.

† The *pathname* or *filename* must follow operating system conventions.

ตารางที่ 5-4 แสดงออพชันของคำสั่ง lnk6x

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
-r
-m C:\ti\myprojects\Xdais\Release\g729enc.map
-o C:\ti\myprojects\Xdais\Release\g729enc.out
-h
-g _G729ENC_KMITL_IG729ENC
-g _G729ENC_KMITL_IALG
-g _G729ENC_KMITL_alloc
-g _G729ENC_KMITL_free
.....
/* TODO: List all OBJ files which make up the algorithm here */
C:\ti\myprojects\Xdais\Release\G729ENC_KMITL_ialg.obj
C:\ti\myprojects\Xdais\Release\G729ENC_KMITL_ialgvt.obj
C:\ti\myprojects\Xdais\Release\iG729ENC.obj
C:\ti\myprojects\Xdais\Release\acelp_cp.obj
...
SECTIONS
{
   .text:algAlloc {}
   .text:algInit {}
   .text:algFree {}
/* The G729ENC_KMITL_activate & G729ENC_KMITL_deactivate routines are
 only used if scratch memory is being used.
*/ .text:algActivate {}
   .text:algDeactivate {}
   .text:algControl {}
   .text:init {}
   .text:exit {}
}
```

รูปที่ 5-4 แสดงตัวอย่างของลิงก์เกอร์คอมมานด์ไฟล์ g729enc.cmd

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6

การทดลอง

6.1 การทดลองตรวจสอบความถูกต้องของ ชุดเข้า/ถอดรหัส G.729 ที่โปรแกรมลงบน ดี เอส พี บอร์ด

<u>วัตถุประสงค์</u>

1. เพื่อทดสอบชุดเข้ารหัส/ ถอดรหัสเมื่อเราที่โปรแกรมลงบน ดี เอส พี บอร์ด แล้วยังทำงานได้อย่างถูกต้อง

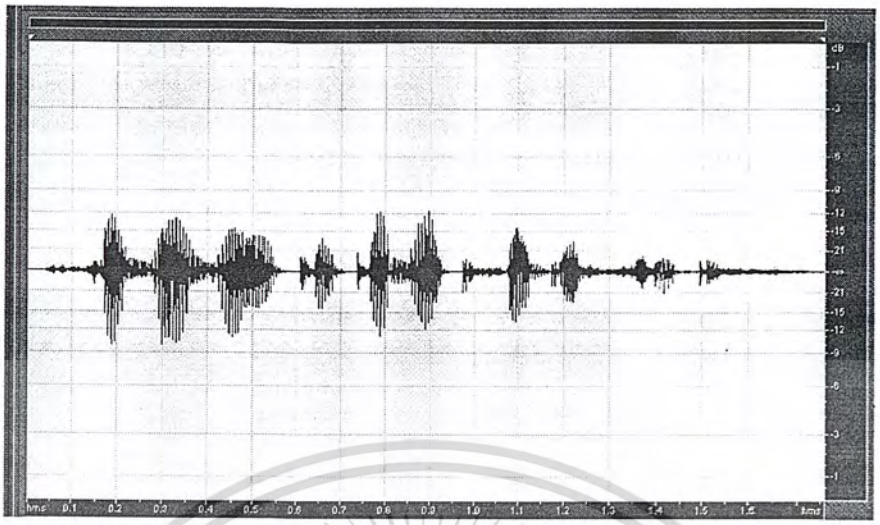
<u>วิธีการทดลอง</u>

1. นำไฟล์เสียงอินพุท ผ่านโมดูล เข้ารหัสบน PC จะได้ไฟล์ของเสียงที่เข้ารหัสแล้วบน PC แล้วทำอย่างเดียวกันบน ดี เอส พี บอร์ด แล้วนำไฟล์ทั้ง 2 มาเปรียบเทียบกัน โดยใช้คำสั่งไฟล์เปรียบเทียบแบบบิทต่อบิท (fc/b) บนดอส (dos)
2. นำไฟล์ทั้ง 2 ผ่านโมดูล ในการถอดรหัสบน PC และ ดี เอส พี บอร์ด ตามลำดับจะได้ file ของสัญญาณเสียงที่ถูกถอดรหัสออกมาแล้วนำไฟล์ทั้ง 2 มาเปรียบเทียบกัน โดยใช้วิธีเดียวกับข้อที่ 1
3. นำไฟล์เสียงอินพุท เทียบกับไฟล์เสียงที่ผ่านการเข้ารหัสและถอดรหัส G.729 มาเปรียบเทียบกันโดยใช้โปรแกรม Adobe Audition 1.5

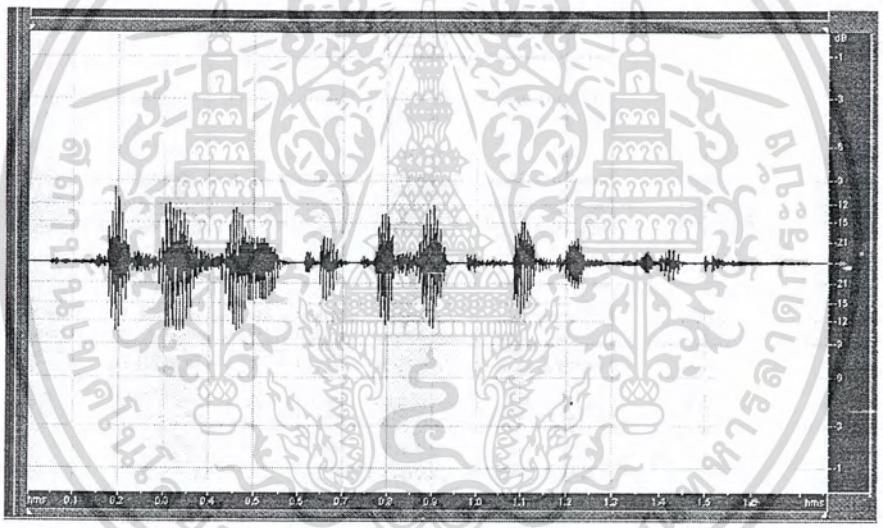
<u>ผลการทดลอง</u>

1. พบว่าไฟล์ที่ได้จากการเข้ารหัส และถอดรหัส บน PC และ ดี เอส พี บอร์ด ไม่แตกต่างกัน
2 เมื่อนำไฟล์เสียงอินพุท เทียบกับไฟล์เสียงที่ผ่านการเข้ารหัสและถอดรหัส G.729 พบว่ามีลักษณะของสัญญาณเป็นดังรูปต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-1 แสดงสัญญาณเสียงต้นฉบับ



รูปที่ 6-2 แสดงสัญญาณเสียงสังเคราะห์โดยโมดูล G.729

3. เมื่อฟังเสียงทั้งสองเทียบกันไม่สามารถแยกความแตกต่างได้

<u>สรุปผลการทดลอง</u>

1. ชุดเข้ารหัสและถอดรหัส G.729 ที่โปรแกรมลงบน ดี เอส พี บอร์ด ทำงานได้อย่างถูกต้อง
2. สัญญาณเสียงที่ผ่านชุดเข้ารหัสและถอดรหัส G.729 จะมีลักษณะของสัญญาณที่แตกต่างไปจากสัญญาณเสียงต้นฉบับเล็กน้อย เพราะโมดูล G.729 เป็นการเข้ารหัสสัญญาณแบบ loosy ซึ่งเป็นการเข้ารหัสสัญญาณที่ยอมให้ผลลัพธ์หลังจากการถอดรหัสกลับมาแล้วไม่จำเป็นต้องเหมือนกับสัญญาณต้นฉบับทุกประการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.2 การทดลองปรับปรุงความเร็วของชุดเข้ารหัสและถอดรหัส G.729 โดยอาศัย Build ออพชันของ Code Composer Studio V2.0

วัตถุประสงค์

1. เพื่อศึกษาระดับการออพติไมซ์ของเครื่องมือ code composer ว่าแต่ละระดับจะมีผลต่อประสิทธิภาพการทำงานอย่างไร
2. ตรวจสอบว่าเมื่อเลือกระดับการออพติไมซ์แบบต่าง ๆ แล้วผลลัพธ์การทำงานยังคงถูกต้องอยู่หรือไม่

วิธีการทดลอง

1. เลือกระดับของการ ออพติไมซ์สำหรับการ compile และ build สำหรับ code composer
2. ทำการทดลองที่ 1 ซ้ำโดยการเปรียบเทียบไฟล์ในลักษณะเดิมเพื่อดูว่าไฟล์ที่ได้หลังจากการออพติไมซ์แล้วนั้นยังคงเหมือนเดิมอยู่หรือไม่
3. บันทึกขนาดของไฟล์ที่ใช้ในการรัน ในการออพติไมซ์แบบต่าง ๆ
4. บันทึกเวลาการทำงานของโมดูล G.729 ในการออพติไมซ์แบบต่าง ๆ

ผลการทดลอง

แสดงผลการทดลองการระดับของการออพติไมซ์ (Optimize level) สำหรับโมดูล การเข้ารหัส

Optimize Level	File size (KB)	Max (ms)	Avg (ms)
size most critical (None)	467	13.44	12.72
size most critical (Register)	421	8.23	7.77
size most critical (Local)	396	7.56	7.10
size most critical (Function)	401	7.56	7.10
size most critical (File)	399	6.42	6.09
speed critical (None)	470	13.20	12.50
speed critical (Register)	424	7.93	7.46
speed critical (Local)	399	7.26	6.81
speed critical (Function)	421	4.24	3.97
speed critical (File)	420	4.17	3.89

ตารางที่ 6-1 แสดงผลการทดลองเมื่อเลือกระดับของการออพติไมซ์แบบต่าง ๆ

สำหรับโมดูล การเข้ารหัส G.729

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Optimize Level	File size (KB)	Max (ms)	Avg (ms)
speed most critical (None)	470	13.23	12.50
speed most critical (Register)	424	7.94	7.46
speed most critical (Local)	399	7.26	6.82
speed most critical (Function)	426	4.25	3.97
speed most critical (File)	425	4.19	3.92

ตารางที่ 6-1 แสดงผลการทดลองเมื่อเลือกระดับของการออพติไมซ์แบบต่าง ๆ

สำหรับโมดูล การเข้ารหัส G.729 (ต่อ)

แสดงผลการทดลองการระดับของการออพติไมซ์สำหรับโมดูล การถอดรหัส

Optimize Level	File size (KB)	Max (ms)	Avg (ms)
size most critical (None)	307	3.12	2.85
size most critical (Register)	290	1.94	1.77
size most critical (Local)	280	1.68	1.54
size most critical (Function)	279	1.42	1.28
size most critical (File)	279	1.42	1.28
speed critical (None)	308	3.07	2.80
speed critical (Register)	291	1.89	1.71
speed critical (Local)	281	1.64	1.49
speed critical (Function)	289	0.84	0.77
speed critical (File)	289	0.74	0.68
speed most critical (None)	308	3.07	2.80
speed most critical (Register)	291	1.88	1.71
speed most critical (Local)	281	1.65	1.50
speed most critical (Function)	291	0.86	0.79
speed most critical (File)	291	0.76	0.71

ตารางที่ 6-2 แสดงผลการทดลองเมื่อเลือกระดับของการออพติไมซ์แบบต่าง ๆ

สำหรับโมดูล การถอดรหัส G.729

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สรุปผลการทดลอง

1. เมื่อเปรียบเทียบด้านขนาดของไฟล์พบว่า

   - การเข้ารหัสระดับของการออพติไมซ์แบบ size most critical (Local) ให้ผลลัพธ์ที่มีขนาดไฟล์ที่เล็กที่สุด

   - การถอดรหัสระดับของการออพติไมซ์แบบ size most critical (Function / File) ให้ผลลัพธ์ที่มีขนาดไฟล์ที่เล็กที่สุด

2. เมื่อเปรียบเทียบด้านความเร็ว พบว่า

   - การเข้ารหัสระดับของการออพติไมซ์แบบ speed critical (File) ให้ผลลัพธ์ที่ใช้เวลาน้อยที่สุด

   - การถอดรหัสระดับของการออพติไมซ์แบบ speed critical (File) ให้ผลลัพธ์ที่ใช้เวลาน้อยที่สุด

3. เมื่อเปรียบเทียบได้คุณสมบัติ ขนาดของไฟล์และ ความเร็วในการทำงาน พบว่า

   - การเข้ารหัสระดับของการออพติไมซ์แบบ speed critical (File) ให้ผลลัพธ์ที่ดีที่สุด

   - การถอดรหัสระดับของการออพติไมซ์แบบ speed critical (File) ให้ผลลัพธ์ที่ดีที่สุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.3 การทดลองเพื่อทดสอบความถูกต้องของโปรแกรม หลังแก้ไขให้โปรแกรมมีคุณสมบัติ Reentrant

วัตถุประสงค์

1. เพื่อตรวจสอบว่าชุดเข้ารหัสและถอดรหัสที่ได้แก้ไขให้เป็นโปรแกรมที่เป็น Reentrantแล้ว สามารถใช้งานได้อย่างถูกต้อง

เป็นพื้นฐานในการพัฒนาโปรแกรมแบบ Multi-channel

วิธีการทดลอง

1. ทำการแก้ไขโปรแกรมการเข้ารหัสและถอดรหัสให้มีคุณสมบัติของการ Reentrant โดยจัดกลุ่มกลุ่มของตัวแปรที่เป็นโกลบอลและสแตติกให้อยู่ในโครงสร้างตามที่ได้ออกแบบไว้
2. นำไฟล์เสียง 2 ชุดมาทำการเข้ารหัสและถอดรหัสโดยชุดเข้ารหัสและถอดรหัสตามมาตรฐานเก็บไฟล์ผลลัพธ์ที่ได้ไว้อ้างอิง
3. นำไฟล์เสียงชุดเดิมมาทำการเข้ารหัสและถอดรหัสโดยให้ทำการเข้ารหัสและถอดรหัส สลับกันที่ละเฟรม
4. นำไฟล์ผลลัพธ์ที่ได้ไปตรวจสอบกับไฟล์ผลลัพธ์ที่ได้ในข้อที่ 2 มาเปรียบเทียบกัน โดยใช้คำสั่งไฟล์ เปรียบเทียบแบบบิทต่อบิท (fc/b) บน dos
5. ทำการตรวจสอบเช่นเดียวกันบน ดี เอส พี บอร์ด

ผลการทดลอง

จากการทดลองบนคอมพิวเตอร์ และ ดี เอส พี บอร์ด ไฟล์ทั้งสองชุดเหมือนกันทุกประการ

สรุปผลการทดลอง

เนื่องจากผลที่ได้จากการเปรียบเทียบไฟล์ทั้ง 2 ชุดได้ผลเหมือนกันทุกประการทำให้สรุปได้ว่าโปรแกรมของเรานั้นเป็น Reentrant แล้วเพราะหากโปรแกรมยังมีการเก็บสถานะโดยอาศัยตัวแปรชนิดโกลบอลอยู่ผลลัพธ์ที่ได้จากการเข้ารหัสและถอดรหัสของไฟล์ 2 ไฟล์นั้นต้องมีผลลัพธ์ที่แตกต่างไปจากชุดต้นฉบับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.4 การทดลองเพื่อทดสอบความถูกต้องของโปรแกรมหลังแก้ไขให้โปรแกรมเป็น Relocatable

วัตถุประสงค์

1. เพื่อตรวจสอบว่าชุดเข้ารหัสและถอดรหัสที่ได้แก้ไขให้เป็น Relocatable โปรแกรม แล้วสามารถใช้งานได้อย่างถูกต้อง
2. ปีนพื้นฐานในการพัฒนาโปรแกรมแบบ Multi-channel

วิธีการทดลอง

1. ทำการแก้ไขโปรแกรมการเข้ารหัสและถอดรหัสให้มีคุณสมบัติของการ Relocatable โดยแก้ไขตัวแปร pointer ในโครงสร้างของเราให้เก็บ offset ที่ชี้ไปยังหน่วยความจำส่วนที่อ้างถึงแทนที่จะเก็บ Absolute address โดยจะปรับการชี้ตำแหน่งเมื่อเรียกใช้
2. หลังจากที่ได้แก้ไขโปรแกรมเรียบร้อยแล้ว เริ่มการตรวจสอบโดยนำไฟล์เสียง 2 ชุดมาทำการเข้ารหัสและถอดรหัสโดยชุดเข้ารหัสและถอดรหัสตามมาตรฐานเก็บไฟล์ผลลัพธ์ที่ได้ไว้อ้างอิง
3. นำไฟล์เสียงชุดเดิมมาทำการเข้ารหัสและถอดรหัสโดยให้ทำการเข้ารหัสและถอดรหัส สลับกันที่ละเฟรม โดยช่วงหนึ่งระหว่างการเข้ารหัสหรือถอดรหัสใช้การคัดลอกหน่วยความจำของวัตถุการเข้ารหัส ไปยังหน่วยความจำตำแหน่งอื่น
4. นำไฟล์ผลลัพธ์ที่ได้ไปตรวจสอบกับไฟล์ผลลัพธ์ที่ได้ในข้อที่ 2 มาเปรียบเทียบกัน โดยใช้คำสั่งไฟล์ เปรียบเทียบแบบบิทต่อบิท (fc/b) บน dos
5. ทำการตรวจสอบเช่นเดียวกันบน ดี เอส พี บอร์ด

ผลการทดลอง

จากการทดลองบนคอมพิวเตอร์ และ ดี เอส พี บอร์ด ไฟล์ทั้งสองชุดเหมือนกันทุกประการ

สรุปผลการทดลอง

เนื่องจากผลที่ได้จากการเปรียบเทียบไฟล์ทั้ง 2 ชุดได้ผลเหมือนกันทุกประการทำให้สรุปได้ว่าโปรแกรมของเรานั้น Relocatable และ Reentrant เพราะหากโปรแกรมยังมีการเก็บสถานะโดยอาศัยตัวแปรชนิดโกลบอลอยู่รวมทั้งมีการย้ายตำแหน่งของบลอคของหน่วยความจำ (memory block) ผลลัพธ์ที่ได้จากการเข้ารหัสและถอดรหัสของไฟล์ 2 ไฟล์นั้นต้องมีผลลัพธ์ที่แตกต่างไปจากชุดต้นฉบับ

จากคุณสมบัติเรื่องของการทำ Relocatable และ Reentrant ทำให้โปรแกรมของเราสามารถสร้างเป็นอัลกอริทึม แบบมัลติแชนเนิลได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.5 การทดลองเพื่อทดสอบความถูกต้องของโปรแกรมหลังแก้ไขให้โปรแกรมเป็นมาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบราลี

วัตถุประสงค์

1. เพื่อทดสอบความถูกต้องของโปรแกรมที่ได้ปรับเปลี่ยนให้เป็น มาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบราลี โดยใช้อินเตอร์เฟสของ IALG
2. เพื่อเตรียมสร้างไลบราลี

วิธีการทดลอง

1. ทำการช่วยสร้างอินเตอร์เฟสเบื้องต้นโดยใช้คอมโพเนินท์ wizard for eXpressDSP ในการช่วยสร้างโดยสามารถดูรายละเอียดในการสร้างได้ ที่คู่มือ Programmer
2. ทำการอิมพลีเมนต์ฟังก์ชัน ของอินเตอร์เฟสของ IALG ที่เรียกใช้และกำหนดตาราง หน่วยความจำของ
   เซคชันข้อมูลของเรา ตัดโค้ดส่วนที่ทำการจองพื้นที่เองในส่วนของโปรแกรมออก ให้อาศัย ALG_create() ในการจองพื้นที่จากตารางหน่วยความจำที่เรากำหนดไว้แทน
3. ทดลองเปรียบเทียบไฟล์ต้นฉบับที่ได้จากการเข้ารหัสและ ถอดรหัส กับไฟล์ที่ได้พัฒนาแป็นมาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบราลี
4. ทำการดูค่าตำแหน่งหน่วยความจำที่เราต้องการกับค่าจริงที่จองได้ว่าตรงตามที่ต้องการหรือไม่

ผลการทดลอง

1. จากการทดลองปรากฏว่าไฟล์ทั้งสองชุดเหมือนกันทุกประการ
2. address ของหน่วยความจำที่ได้กับตำแหน่งที่ได้กำหนดในตารางหน่วยความจำนั้นไม่ตรงกันใน

สรุปผลการทดลอง

จากการทดลองสรุปได้ว่าการทำงานของ อัลกอริทึมทำงานได้อย่างปกติแต่พื้นที่ของตารางหน่วยความจำที่ได้นั้นไม่ตรงกับที่ระบุไว้ เนื่องจากฟังก์ชัน ในการจองพื้นที่ของ ALG ไม่ได้มีการกำหนดเซคชันในการ สร้างตารางของหน่วยความจำดังนั้นตำแหน่งของพื้นที่ที่ได้จึงเป็นพื้นที่ที่ได้เรากำหนดไว้ว่าใช้สำหรับ การจองพื้นที่หน่วยความจำทั่วไป (โดยทั่วไปจะกำหนดไว้ที่หน่วยความจำภายนอก)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.6 การทดลองเพื่อทดสอบความถูกต้องของโปรแกรมหลังแก้ไขให้โปรแกรมเป็นมาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบรารี และเปลี่ยนอินเตอร์เฟสของฟังก์ชัน ของ Abstract algorithm interface

วัตถุประสงค์

1. เพื่อทดสอบความถูกต้องของโปรแกรมที่ได้ปรับเปลี่ยนให้เป็น มาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบรารี โดยใช้อินเตอร์เฟสของ ALGRF
2. เพื่อพิจารณาตำแหน่งของหน่วยความจำที่ได้จริงกับที่ได้กำหนดไว้
3. เพื่อเตรียมสร้างไลบรารี

วิธีการทดลอง

1. ทำการเปลี่ยนฟังก์ชัน abstract Interface จาก เดิมที่ใช้ ALG มาเป็น ALGRF
2. ทำการกำหนดชื่อเซคชันให้กับ IRAM เป็น INTERNALHEAP และ SRAM เป็น EXTERNALHEAP พร้อมทั้งกำหนดขนาดของสูงสุดในการสร้างพื้นที่หน่วยความจำฮีฟ (heap memory) ของ แต่ละเซกชันให้เพียงพอกับขนาดของตารางหน่วยความจำทั้งหมด
3. ทำการ extern int ตัวแปร INTERNALHEAP และ EXTERNALHEAP
4. ในส่วนแรกของ main โปรแกรมเรียก ALGRF_setup(INTERNALHEAP,EXTERNALHEAP) เพื่อกำหนดค่าของเซคชันของหน่วยความจำในการจองหน่วยความจำฮีฟ
5. ทดลองเปรียบเทียบไฟล์ต้นฉบับที่ได้จากการเข้ารหัสและ ถอดรหัส กับไฟล์ที่ได้พัฒนาเป็น มาตราฐานการเขียนอัลกอริทึมของผู้ผลิตไลบรารี
6. ทำการ Watch ค่าตำแหน่งหน่วยความจำที่เราต้องการกับค่าจริงที่จองได้ว่าตรงตามที่ต้องการหรือไม่

ผลการทดลอง

1. จากการทดลองปรากฏว่าไฟล์ทั้งสองชุดเหมือนกันทุกประการ
2. ตำแหน่งของหน่วยความจำที่ได้กับตำแหน่งที่ได้กำหนดในตารางหน่วยความจำนั้นตรงกัน

สรุปผลการทดลอง

จากการทดลองสรุปได้ว่าการทำงานของ อัลกอริทึมทำงานได้อย่างปกติและพื้นที่ของตารางหน่วยความจำที่ได้นั้นตรงกับที่ระบุไว้ ดังนั้นการใช้ abstract interface function ALGRF จึงมีการจัดการหน่วยความจำที่ดีกว่า abstract interface function ALG

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.7 การทดลองเปรียบเทียบลักษณะสัญญาณคุณภาพเสียงที่ได้ระหว่างชุดสังเคราะห์เสียง G.729 กับ G.723.1

<u>วัตถุประสงค์</u>

1. เพื่อเปรียบเทียบคุณภาพของเสียงที่ได้ระหว่าง 2 มาตรฐาน

<u>วิธีการทดลอง</u>

1. นำไฟล์เสียงต้นฉบับชุดเดียวกันมาทำการเข้ารหัสและถอดรหัส แล้วนำไฟล์ที่ถอดรหัสได้มากลับเป็นอินพุท ใหม่ทำไป 10 รอบ
2. นำมาหาค่ารากที่สองค่าเฉลี่ยของความผิดผลาด (RMSe) จากสมการ

$$\sqrt{\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{n-1}} \qquad (6\text{-}1)$$

เมื่อ x_i คือ อินพุทตัวอย่างที่ i

y_i คือ เอาท์พุท ตัวอย่างที่ i

n คือจำนวนตัวอย่างทั้งหมด

3. นำมาหาค่า Signal to noise ratio (Snr) จากสมการ

$$SNR = 10.0 * \log_{10}(energy / err) \qquad (6\text{-}2)$$

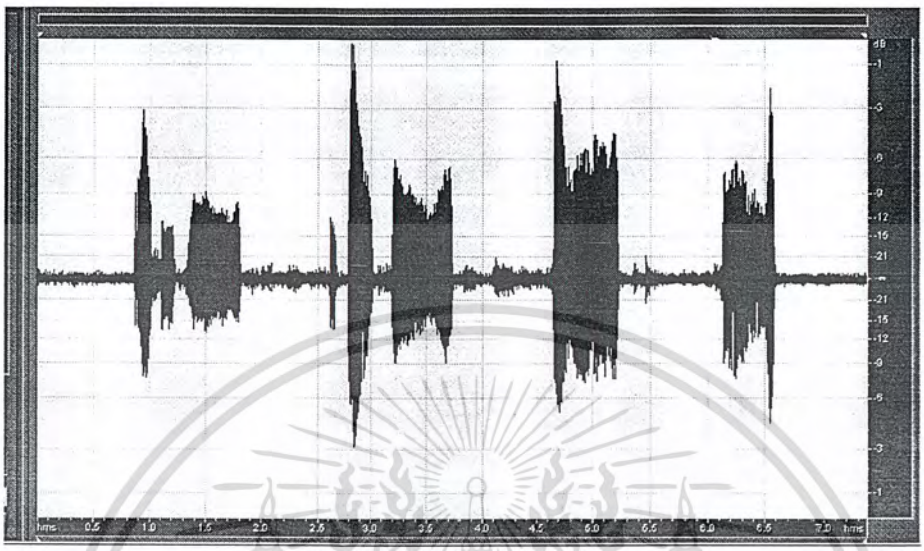
โดยที่ค่าของพลังงานหาได้จากผลรวมของไฟล์ต้นฉบับกับสัญญาณเสียงสังเคราะห์ ทั้งสองไฟล์คูณกันแบบตัวอย่างต่อตัวอย่าง และพลังงานของความผิดพลาดหาจากผลต่างกำลังสองของไฟล์ต้นฉบับกับสัญญาณเสียงสังเคราะห์

4. พิจารณาจากรูปกราฟที่ได้ ทดลองฟังเสียงในแต่ละรอบของเอาท์พุท เทียบกัน
5. ใช้ Adobe Audition 1.5 ช่วยในการหาค่าทางสถิติ ของ RMS (ค่าพลังงานของสัญญาณ) โดยกำหนดให้ window size เท่ากับ 50 ms
6. ทดลองเปลี่ยนโดเมนของ เอาท์พุททั้งสองชุดเปรียบเทียบผลโดยเปรียบเทียบในเทอมขององค์ประกอบของความถี่และเทอมของ Spectral โดเมนโดยอาศัย Adobe Audition 1.5 ช่วยในการแปลงโมเมนของสัญญาณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<u>ผลการทดลองแสดงลักษณะสัญญาณเสียงในโดเมนของเวลา</u>

แสดงลักษณะของสัญญาณที่สังเคราะห์ได้ของทั้งสอง อัลกอริทึมในรอบการสังเคราะห์ที่ 1, 5 และ 10

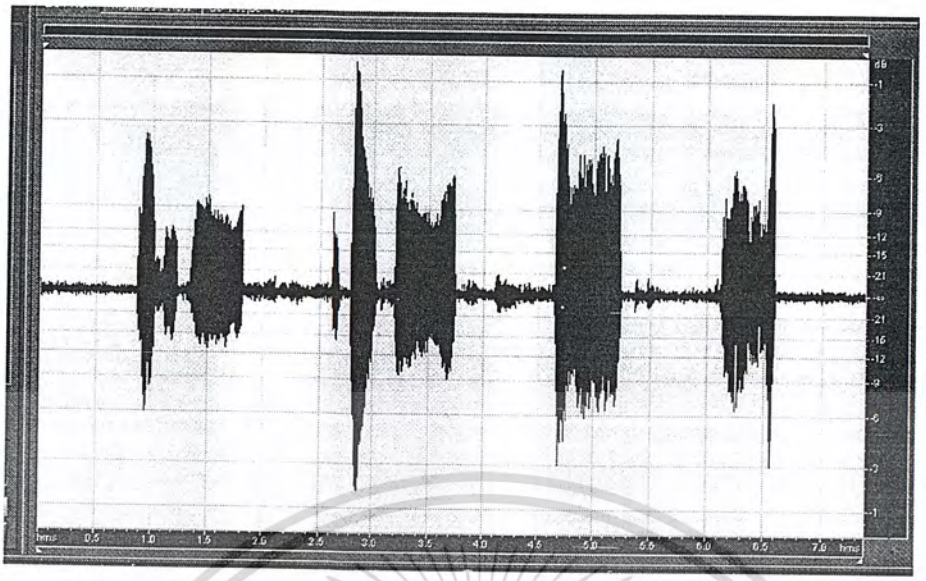


รูปที่ 6-3 แสดงลักษณะของสัญญาณเสียงต้นฉบับ ในโดเมนของเวลา

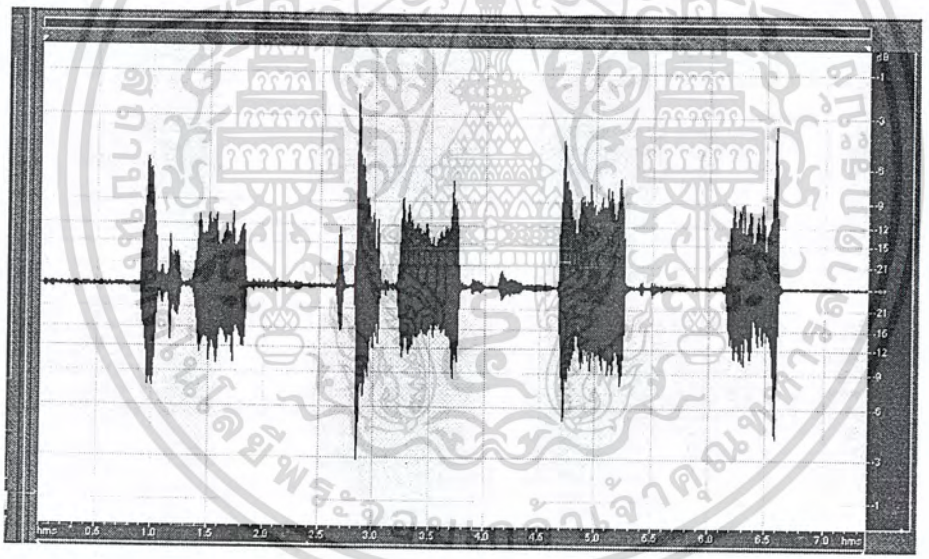
รูปที่6-4 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของเวลา

ในการสังเคราะห์เสียงรอบที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

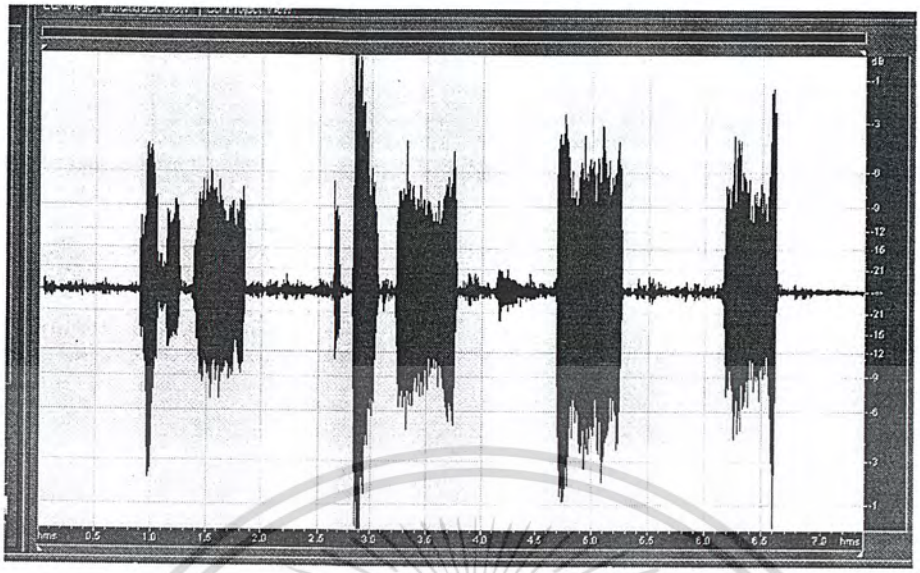


รูปที่ 6-5 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.723.1 ในโดเมนของเวลในการสังเคราะห์เสียงรอบที่ 1



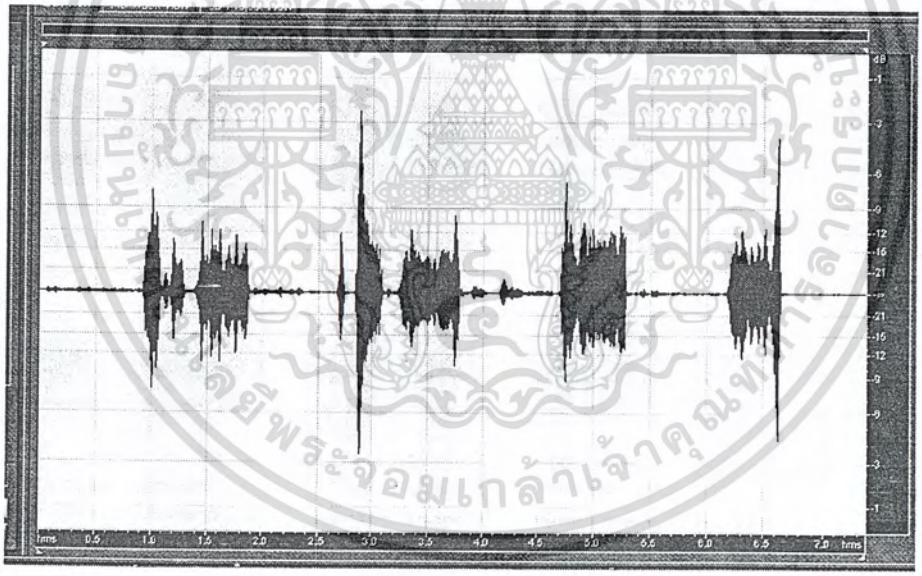
รูปที่6-6 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของเวลาในการสังเคราะห์เสียงรอบที่ 5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-7 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.723.1 ในโดเมนของเวล

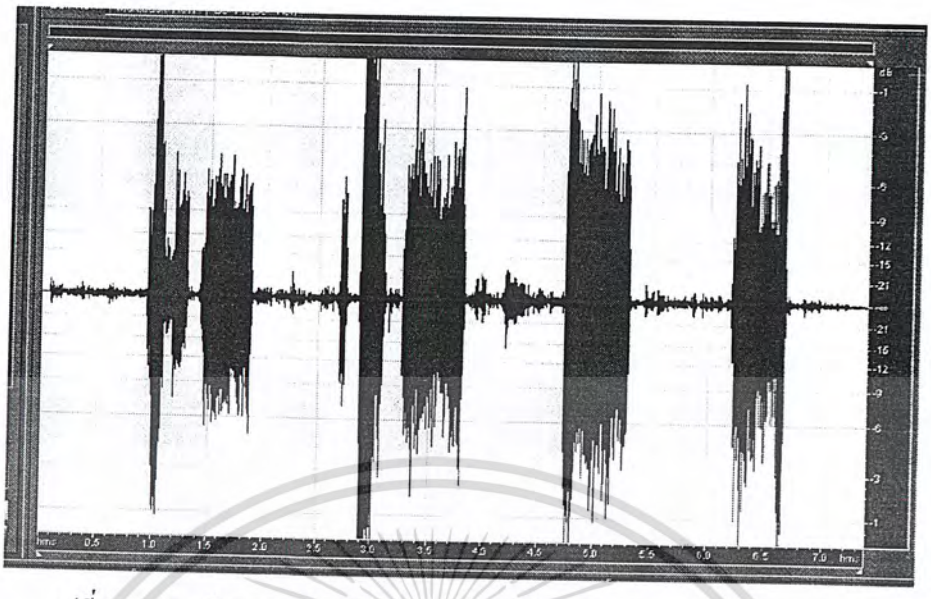
ในการสังเคราะห์เสียงรอบที่ 5



รูปที่ 6-8 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของเวล

ในการสังเคราะห์เสียงรอบที่ 10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-9 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.723.1 ในโดเมนของเวล

ในการสังเคราะห์เสียงรอบที่10

ผลการทดลองแสดงการเปรียบเทียบค่า RMSe และ Snr ของชุดสังเคราะห์เสียงทั้ง 2 วิธี

ผลลัพธ์รอบที่	G.729(CELP 8Kbps)		G.723.1(MP-MLQ 6.3Kbps)	
	Snr	RMSe	Snr	RMSe
1	-2.66	3487.23	-3.12	3677.11
2	-2.34	3361.79	-3.52	3849.16
3	-2.34	3362.36	-3.73	3943.28
4	-1.93	3205.27	-4.13	4131.60
5	-1.85	3178.38	-4.52	4322.97
6	-1.73	3133.97	-4.65	4383.60
7	-1.70	3121.87	-4.92	4522.80
8	-1.57	3075.41	-5.30	4725.49
9	-1.45	3034.98	-5.70	4951.46
10	-1.23	2958.69	-6.10	5183.54

ตารางที่ 6-3 แสดงการเปรียบเทียบค่า RMSe และ Snr ของทั้งสอง Algorithm

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

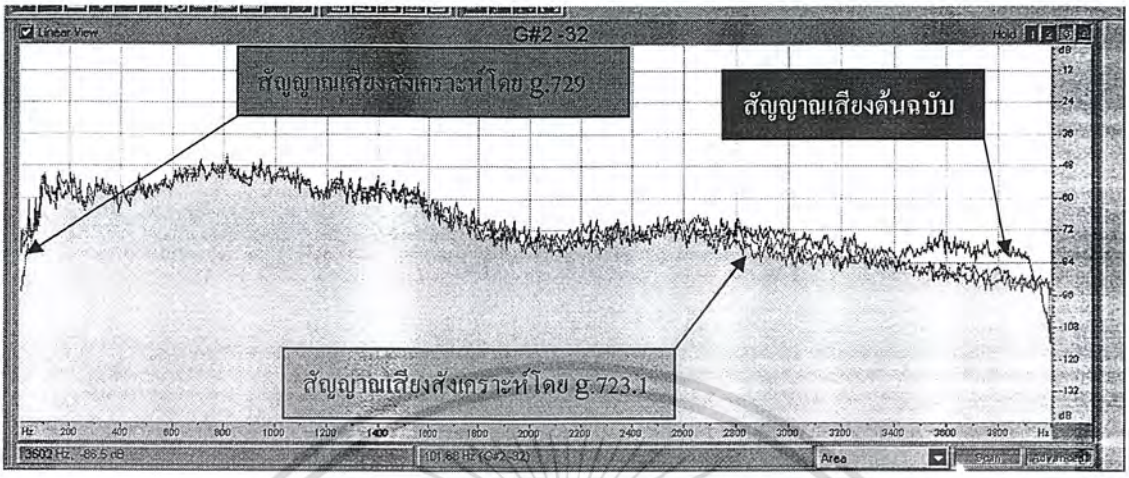
ผลการทดลองแสดงการเปรียบเทียบค่าทางสถิติ RMS

STATISTIC TYPE	INPUT	การวัดรอบที่1		การวัดรอบที่5		การวัดรอบที่10	
		G729	G723.1	G729	G723.1	G729	G723.1
Min Sample Value	-23050	-25808	-26997	-23576	-32768	-22219	-32768
Max Sample Value	32130	30113	31764	26304	32767	24877	32767
Peak Amplitude(db)	-0.14	-0.73	-0.27	-1.91	0	-2.39	0
Possibly Clipped	0	0	0	0	8	0	67
DC Offset	-0.34	0	0.025	0	0.021	0	-0.001
Minimum RMS Power (db)	-42.68	-46.64	-45.53	-60.81	-51.96	-96.43	-81.56
Maximum RMS Power(db)	-10.59	-10.92	-10.14	-11.78	-7.75	-12.36	-5.04
Average RMS Power(db)	-25.48	-26.25	-25.36	-28.53	-23.66	-31.04	-21.19
Total RMS Power(db)	-22.14	-22.7	-21.8	-24.51	-19.85	-26.73	-17.27

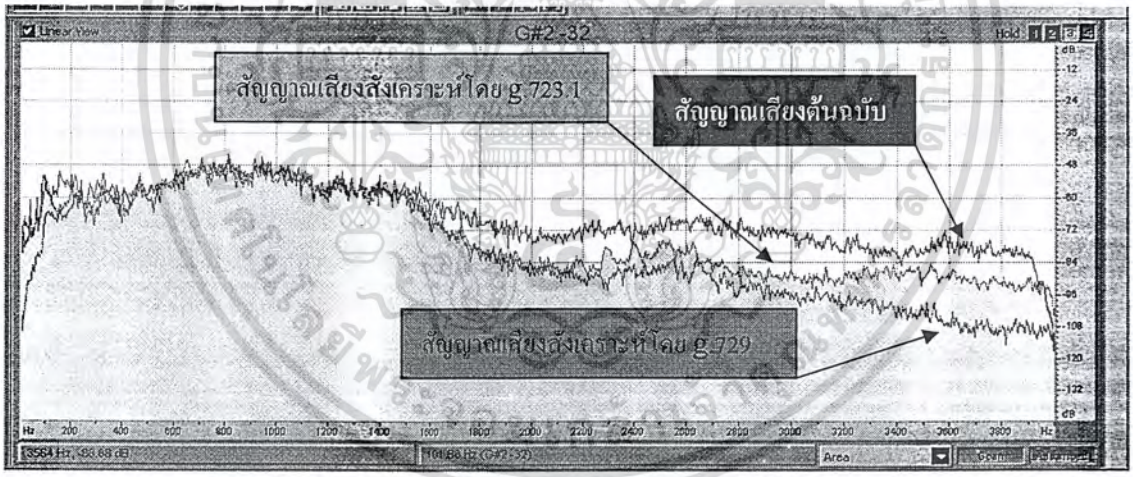
ตารางที่ 6-4 แสดงการวัดค่าทางสถิติของชุดสังเคราะห์สัญญาณทั้งสองชุด โดความละเอียดของตัวอย่างที่ใช้ขนาด 16 บิทใช้ RMS windows ขนาด 50 ms

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลการทดลองเปรียบเทียบลักษณะของสัญญาณเมื่อแปลงสัญญาณจากโดเมนทางเวลาเป็นโดเมนทางความถี่

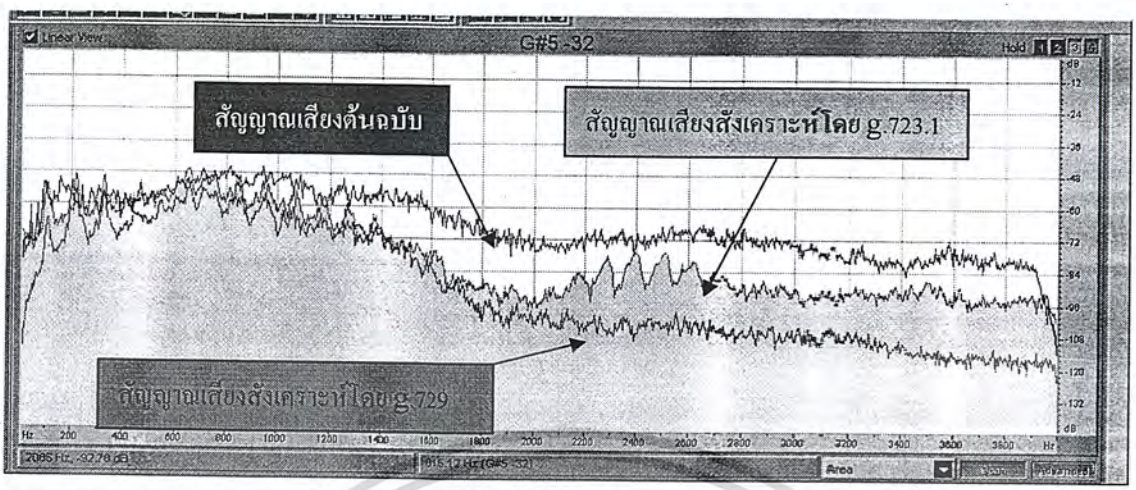


รูปที่ 6-10 แสดงการเปรียบเทียบสัญญาณต้นฉบับ กับสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 และ G.723.1 ในการสังเคราะห์เสียงรอบที่ 1

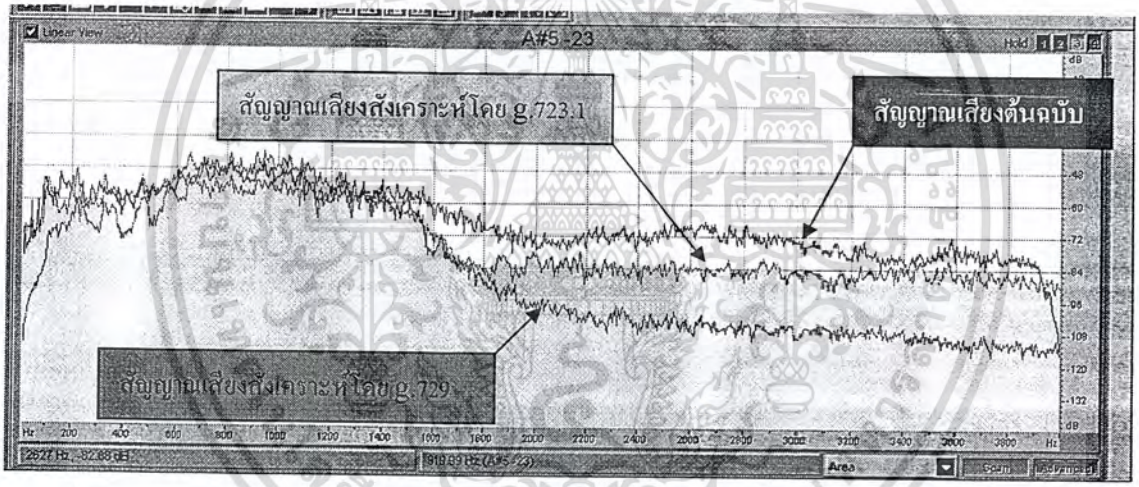


รูปที่ 6-11 แสดงการเปรียบเทียบสัญญาณต้นฉบับ กับสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 และ G.723.1 ในการสังเคราะห์เสียงรอบที่ 5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-12 แสดงการเปรียบเทียบสัญญาณต้นฉบับ กับสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 และ G.723.1 ในการสังเคราะห์เสียงรอบที่ 7

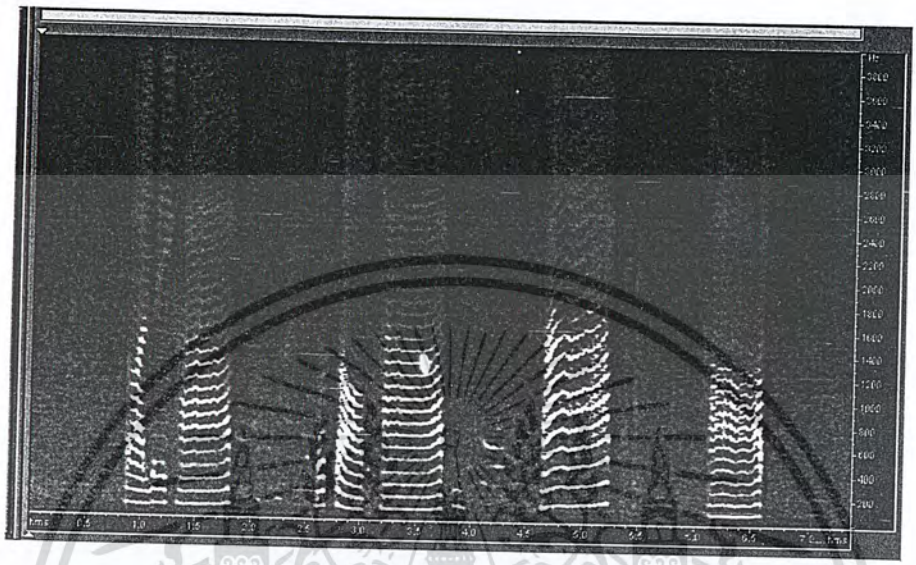


รูปที่ 6-13 แสดงการเปรียบเทียบสัญญาณต้นฉบับ กับสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 และ G.723.1 ในการสังเคราะห์เสียงรอบที่ 10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<u>ผลการทดลองเปรียบเทียบลักษณะของสัญญาณเมื่อแปลงสัญญาณจากโดเมนทางเวลาเป็นโดเมน Spectral</u>

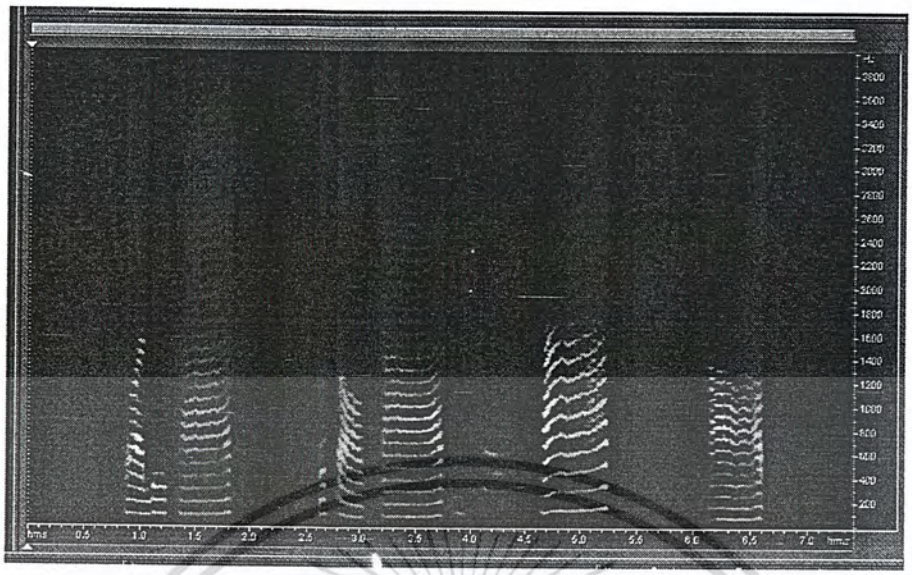
การทดลองแปลงโดเมนจากโดเมนทางเวลาเป็นโดเมนของ Spectral แสดงให้เห็นถึงองค์ประกอบเชิงความถี่และค่าพลังงาน



รูปที่ 6-14 แสดงสัญญาณต้นฉบับในโดเมนของ Spectral

รูปที่ 6-15 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของ Spectral ในการสังเคราะห์เสียงรอบที่ 1

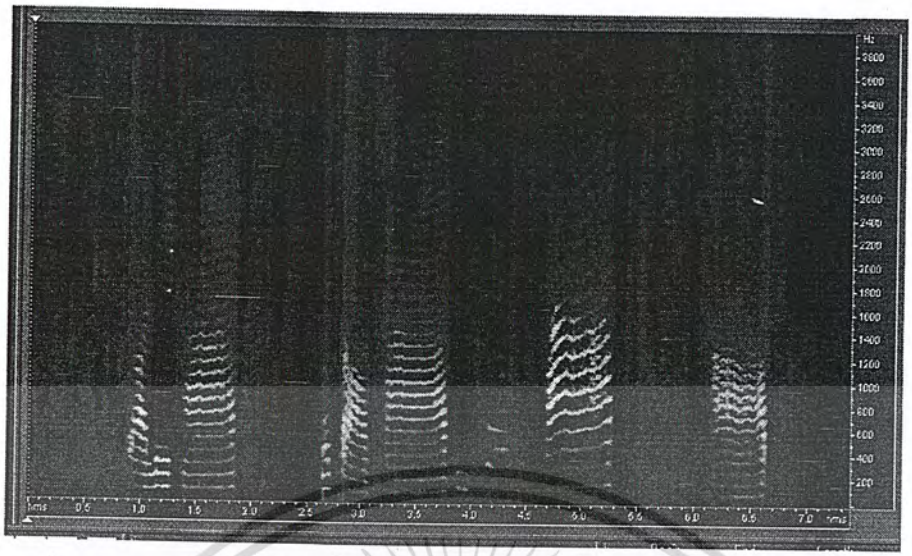
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-16 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส ณ.723.1 ในโดเมนของ Spectral ในการสังเคราะห์เสียงรอบที่ 1

รูปที่ 6-17 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของ Spectral ในการสังเคราะห์เสียงรอบที่ 5

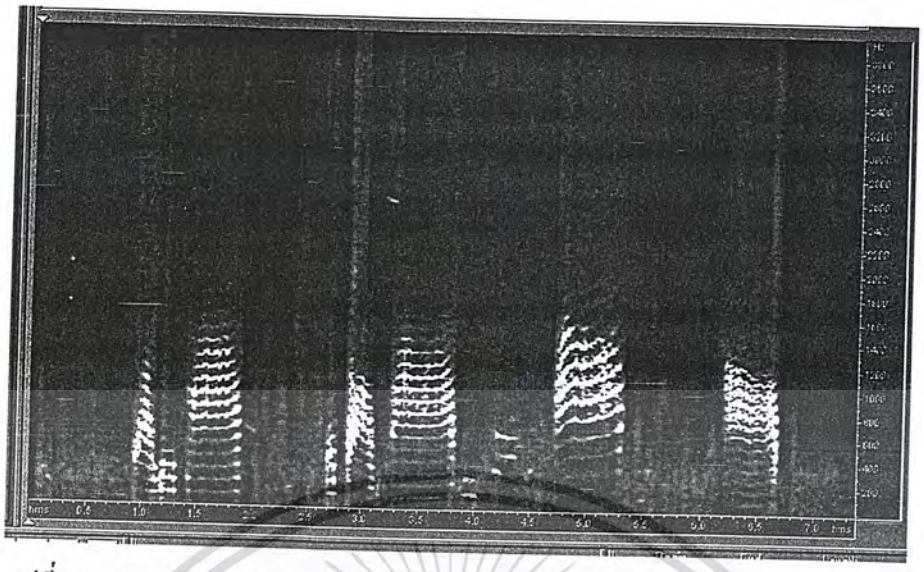
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-18 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส ณ.723.1 ในโดเมนของ Spectral ในการสังเคราะห์เสียงรอบที่ 5

รูปที่ 6-19 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.729 ในโดเมนของ Spectral ในการสังเคราะห์เสียงรอบที่ 10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-20 แสดงสัญญาณเสียงสังเคราะห์จากชุดเข้ารหัส G.723.1 ในโดเมนของ Spectral ในการสังเคราะห์เสียงรอบที่ 10

<u>สรุปผลการทดลอง</u>

1. ในการวัดค่า RMSe , Snr พบว่าในการสังเคราะห์เสียงด้วย G.729 ให้ผลลัพธ์ที่ดีกว่า G.723.1 คือมีค่า RMSe ที่น้อยและ Snr ที่สูงกว่าและมีข้อสังเกตของ G.729 ยิ่งเมื่อการสังเคราะห์เสียงกลับยิ่งมากรอบขึ้นกลับให้ผลของ ความผิดพลาดที่ลดลง
2. ในการวัดหาค่า RMS ของทั้งสองสรุปได้ว่ายิ่งมีการเมื่อการสังเคราะห์เสียงกลับยิ่งมากรอบขึ้น แนวโน้มของพลังงานของ G.729 จะลดลง ส่วน G.723.1 กลับเพิ่มขึ้นจาก สัญญาณเสียงต้นฉบับ
3. เมื่อพิจารณาใน Spectral โดเมน จะสังเกตุได้ว่า ที่ความถี่ที่สูง ๆ (ความถี่ที่มากกว่า 3000 เฮิร์ต) การรักษาระดับพลังงานของ ชุดเข้ารหัสแบบ g.723.1 จะรักษาความคลายคลึงกับต้นฉบับได้มากกว่าแบบ g.729 เมื่อนำมาจารณาด้านการคงรูปแบบความคล้ายคลึงของพลังงานเมื่อมีการสังเคราะห์เสียงกลับมากรอบขึ้น G.723.1 จะทำได้ดีกว่าแต่ในการใช้งานในแอพพลิเคชันจริง ๆ ไม่ได้มีสภาพแวดล้อมอย่างนี้
4. จากการเปรียบเทียบโดยใช้ Snr เป็นการเปรียบเทียบแบบหยาบ ๆ ซึ่งเป็นการเปรียบเทียบแต่ละตัวอย่างของเสียงเท่านั้นไม่ได้เปรียบเทียบในลักษณะรูปร่างของสัญญาณและ G.729 มี บิทเรท (bitrate)ที่สูงกว่า G.723.1 รวมทั้งอัลกอริทึมดีเลย์ที่ต่ำกว่า ทำให้การเปรียบเทียบไม่สามารถทำได้ว่ามาตรฐานใดดีกว่ากันทั้งนี้ต้องขึ้นอยู่กับการนำไปใช้งานให้เหมาะสมตาม Application
5. การพิจารณาเรื่องคุณภาพของสัญญาณเสียงที่สังเคราะห์ขึ้นนั้นวิธีการวัดที่ดีสุดคือใช้การฟังของมนุษย์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.8 การทดลองทำการวัดเวลาสำหรับโมดูล ต่าง ๆ สำหรับการเข้ารหัส

วัตถุประสงค์

เพื่อวัดว่าโมดูล ใดมีการทำงานมากที่สุดสำหรับการเข้ารหัสเพื่อเป็นแนวทางในการเลือกโมดูลสำหรับการออพติไมซ์ต่อไป

วิธีการทดลอง

1. ทำการ enable clock ของระบบก่อน จาก Menu profiler->enable clock
2. เลือกเครื่องมือที่ใช้ในการวัด performance จาก Menu profiler->start new session

ผลการทดลอง

Module	จำนวนครั้งในการเรียกเฉลี่ย (ครั้ง)	เวลาเฉลี่ยต่อเฟรม (us)
d4i40_17()	82	1285.430
cor_h_cp()	82	164.570
ACELP_codebook()	82	16.51
corr_xy2()	63	19.53
cor_h_x()	62	81.290
update_bwd()	54	14.320
tst_bwd_dominant()	53	5.430
ener_DB()	53	30.890
vad()	32	99.770
MakeDec()	31	6.440

ตารางที่ 6-5 แสดงจำนวนครั้งในการเรียกใช้งานฟังก์ชัน เรียงลำดับการเรียกใช้จากมากไปน้อย

สรุปผลการทดลอง

1.พบว่าการคำนวณส่วนใหญ่จะอยู่ในฟังก์ชั่น d4i40_17() ซึ่งใช้ในการหาตำแหน่งของ พัลซ์ ที่เป็นไปได้จำนวน 4 พัลซ์ในสัญญาณสุ่ม (unvoiced sound)

2.เราควรที่จะออพติไมซ์โมดูล ในส่วนของการค้นหา fix codebook

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.9 การทดลองทำการวัดเวลาหลังจากทำการออพติไมซ์ตามหัวข้อวิจัยของ F.-K CHEN, J.-F YANG และ Y. –L YAN เรื่อง Candidate scheme for fast ACELP search

วัตถุประสงค์

เพื่อวัดเวลาหลังจากการออพติไมซ์เปรียบเทียบกับก่อนการออพติไมซ์เวลาที่วัดได้จากการเข้ารหัสและถอดรหัสรวมกัน

วิธีการทดลอง

1. ทำการสร้างวัตถุของ STS ของ DSP Bios เพื่อใช้ในการช่วยวัดเวลา
3. กำหนด environment ให้ทั้งโค้ดและเซคชันข้อมูลทำงานอยู่ที่หน่วยความจำภายนอก
4. แทรกโค้ดสำหรับวัดเวลาไว้ที่หัวและท้ายของ โมดูล ที่ต้องการวัด
5. เรียก Statistical view ออกมา
6. รันโปรแกรมตามปกติ

ผลการทดลอง

จำนวนพื้นที่ในการออพติไมซ์สำหรับการ search	ALL MAX(ms)	ALL AVG(ms)	ACELP MAX(ms)	ACELP AVG(ms)
8(แบบเดิม)	6.47	4.19	1.30	0.85
7	5.38	4.17	1.13	0.67
6	5.01	3.93	0.91	0.53
5	4.75	3.79	0.70	0.41
4	4.42	3.63	0.49	0.32
3	4.26	3.55	0.36	0.26
2	4.15	3.49	0.30	0.23
1	4.12	3.45	0.28	0.21

ตารางที่ 6-6 แสดงผลการทดลองการวัดเวลาหลังจากทำการออพติไมซ์โดยใช้วิธีของ Candidate scheme for fast ACELP search

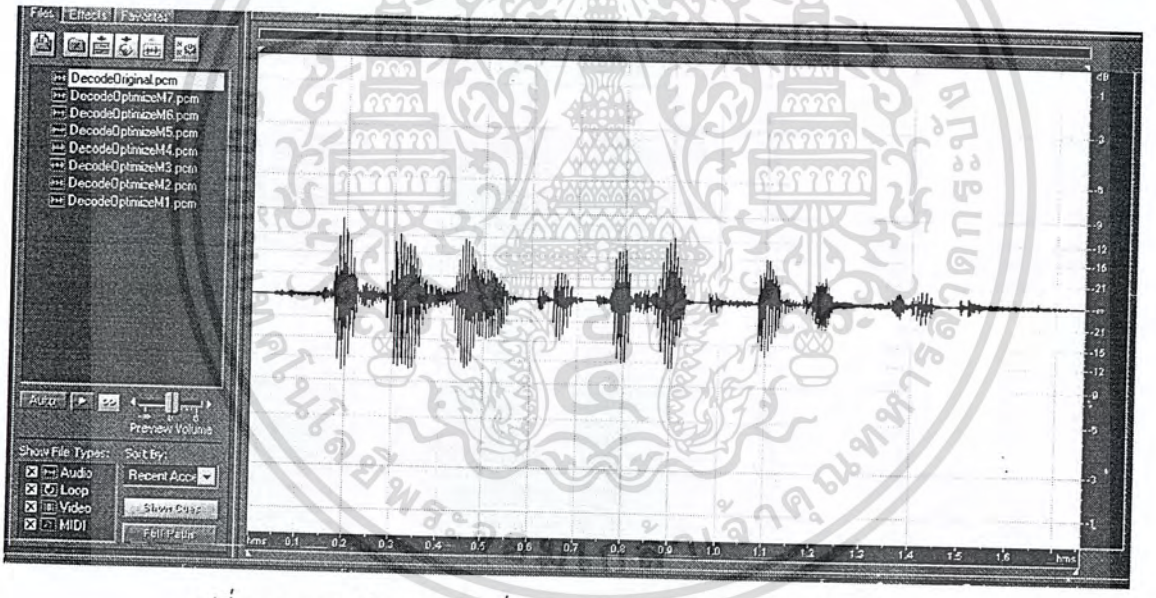
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อทดลองทำการปรับโปรแกรมระดับการออพติไมซ์เป็น No External Var Refs
ได้ผลลัพธ์ดังนี้

จำนวนพื้นที่	ALL MAX(ms)	ALL AVG(ms)	ACELP MAX(ms)	ACELP AVG(ms)
1	3.99	3.36	0.29	0.22

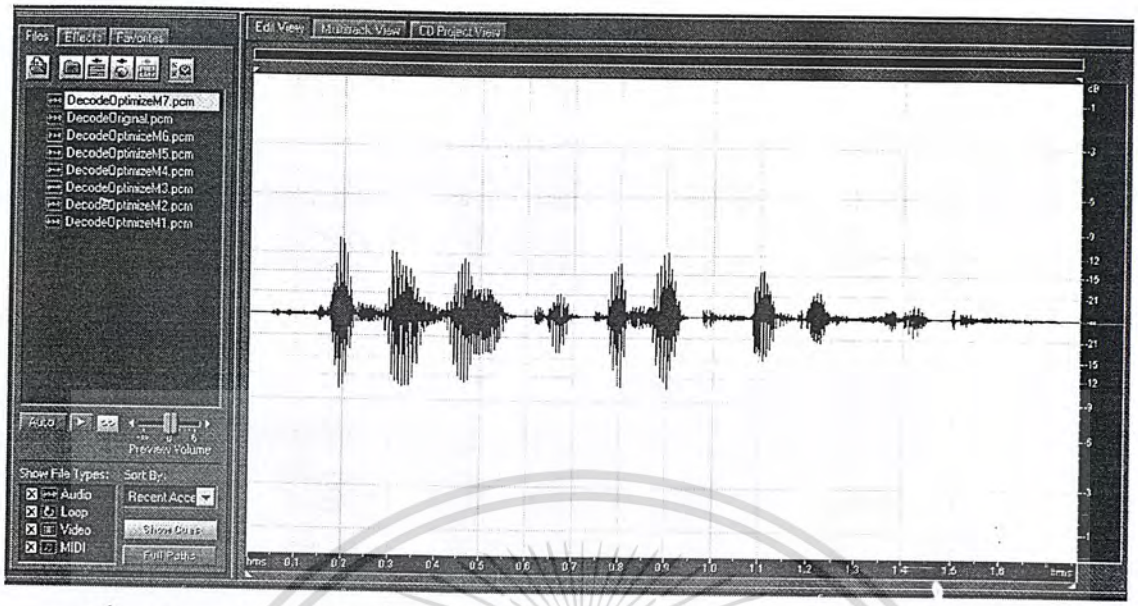
ตารางที่ 6-7 แสดงผลการทดลองการวัดเวลาหลังจากทำการออพติไมซ์โดยใช้วิธีของ Candidate scheme for fast ACELP search พร้อมทั้งปรับระดับของการออพติไมซ์เป็นแบบ No External Var Refs

กราฟเปรียบเทียบรูปคลื่นสัญญาณเสียงที่ได้ทำการเลือกจำนวนพื้นที่ที่จะทำการค้นหาสำหรับการออพติไมซ์ตั้งแต่ 8-1

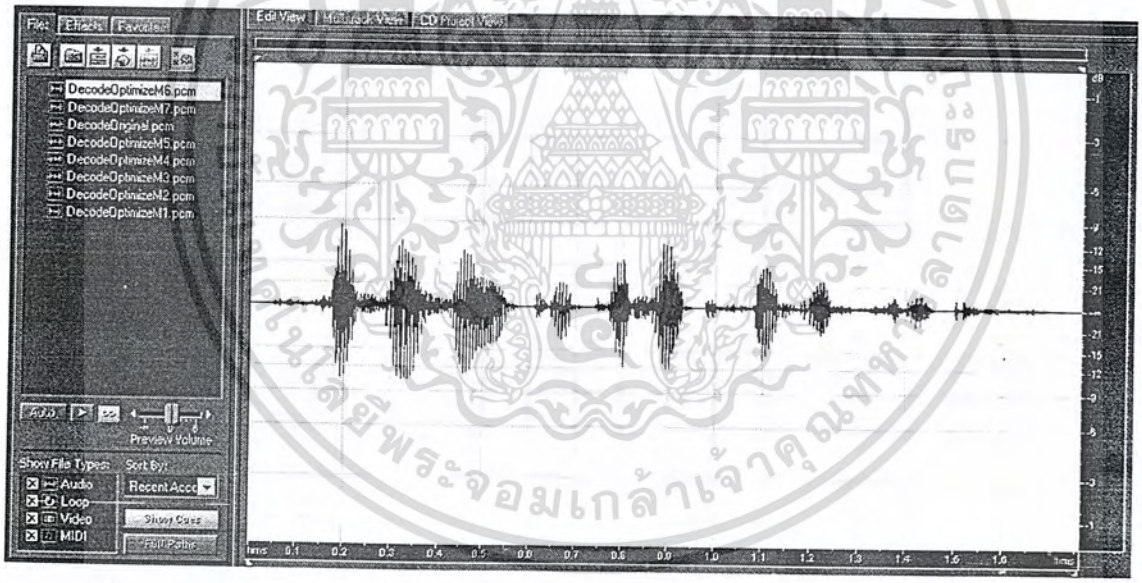


รูปที่ 6-21 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสตามมาตรฐาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

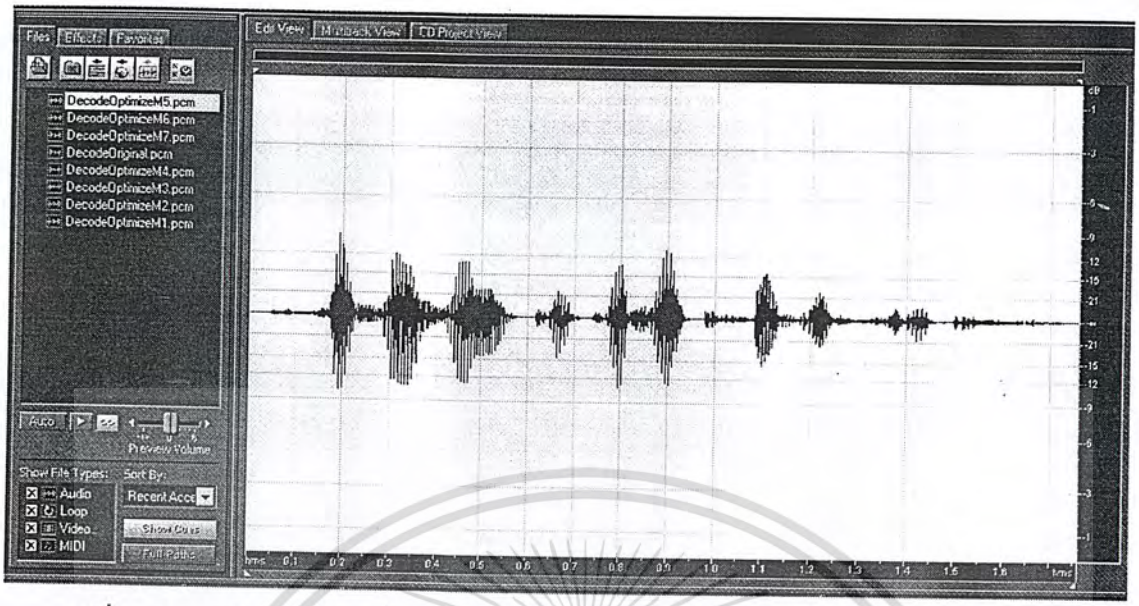


รูปที่ 6-22 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 7

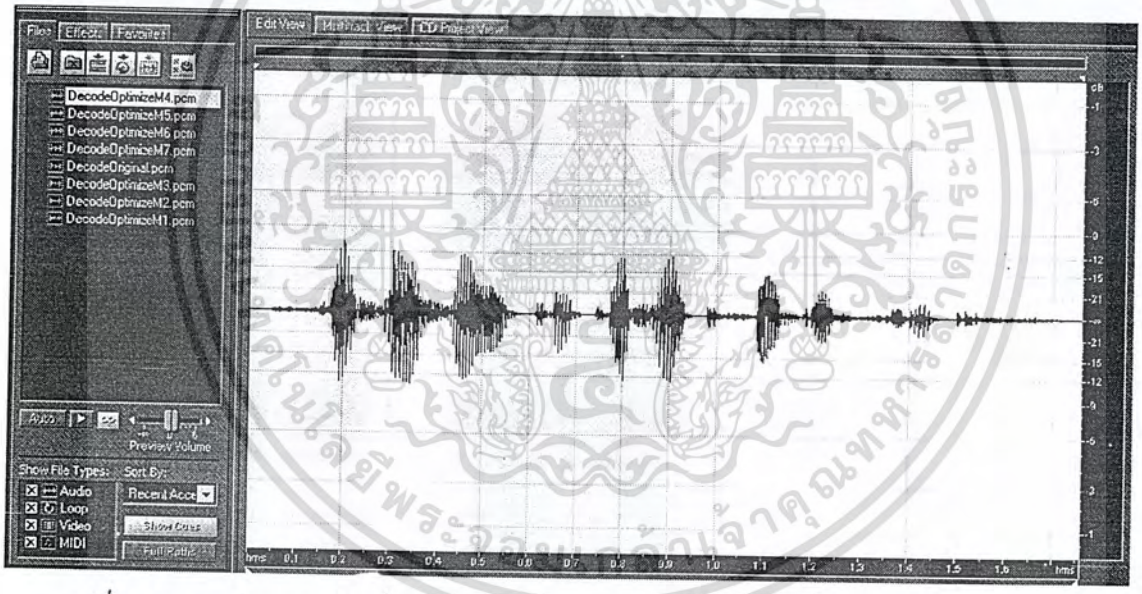


รูปที่ 6-23 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

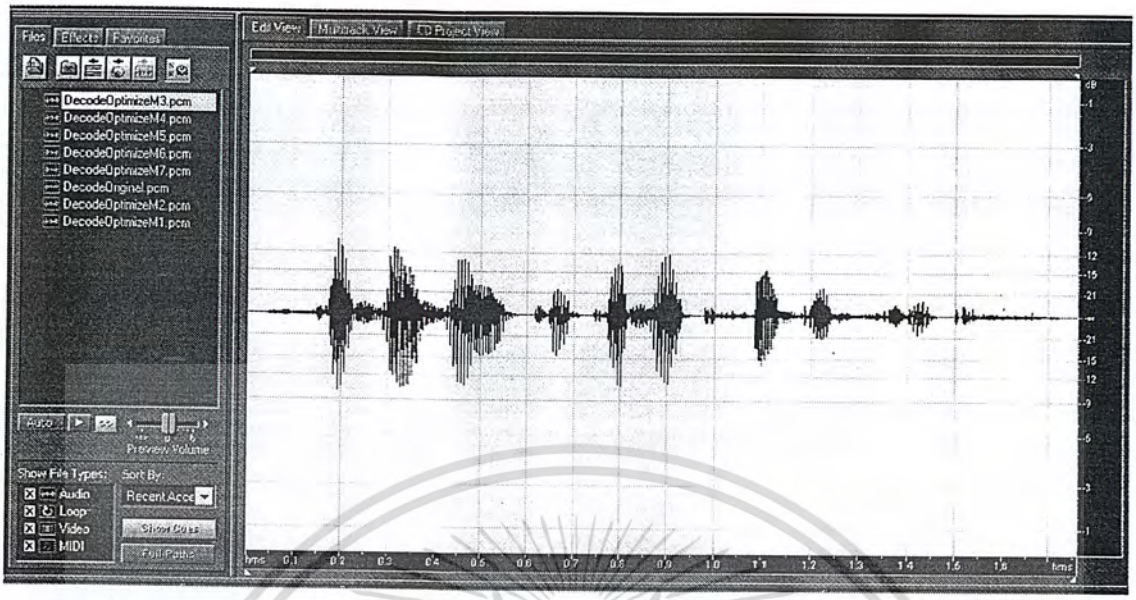


รูปที่ 6-24 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 5

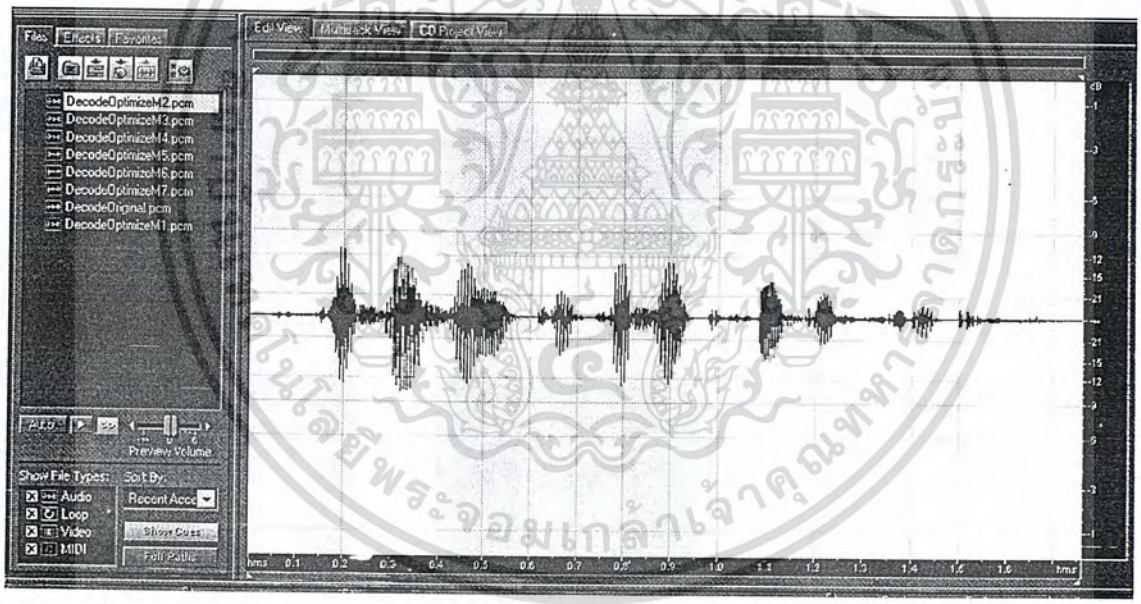


รูปที่ 6-25 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 4

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

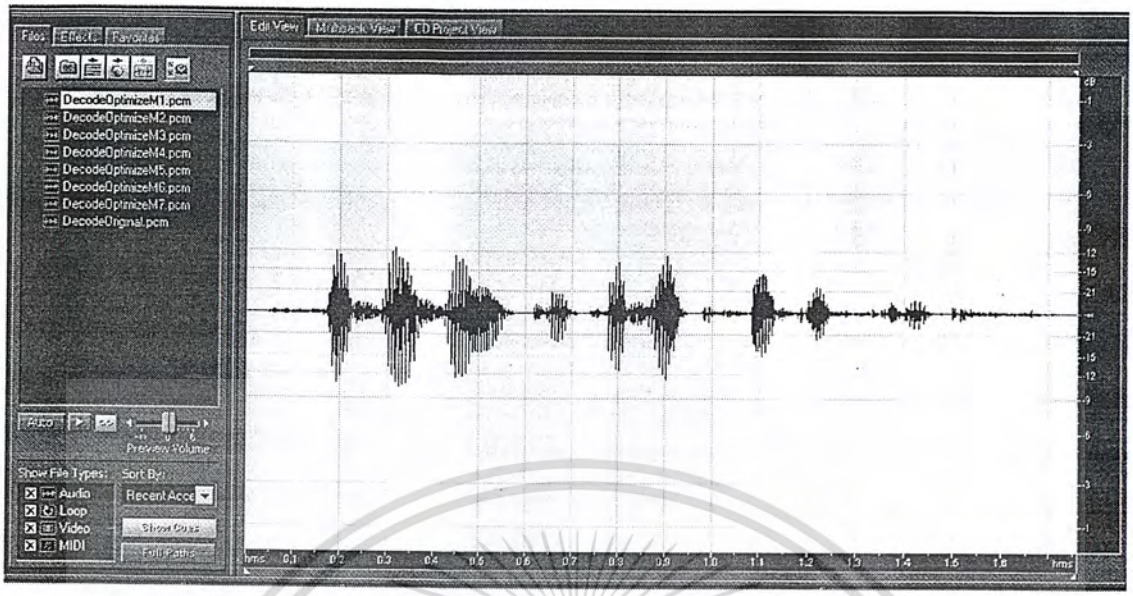


รูปที่ 6-26 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 3



รูปที่ 6-27 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-28 แสดงสัญญาณเสียงที่ถูกถอดรหัสโดยชุดถอดรหัสที่ใช้จำนวนพื้นที่สำหรับการค้นหา = 1

<u>สรุปผลการทดลอง</u>

1. พบว่าจำนวนพื้นที่สำหรับการค้นหา ลดลงทำให้ความเร็วในการเข้ารหัสสัญญาณเสียงเร็วขึ้น
2. ถึงแม้ว่าจำนวนพื้นที่สำหรับการค้นหา จะลดลงแต่ลักษณะของสัญญาณเสียงแทบจะไม่แตกต่างจากของเดิม และเมื่อเปรียบเทียบจากการฟังพบว่าหูคนไม่สามารถแยกความแตกต่างนี้ได้
3. เราตัดสินใจเลือกการออพติไมซ์ด้วยวิธีนี้โดยเลือกพื้นที่สำหรับการค้นหา = 1 เพราะการเข้ารหัสสามารถทำได้เร็วที่สุดโดยที่คุณภาพของเสียงที่ได้ยังยอมรับได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

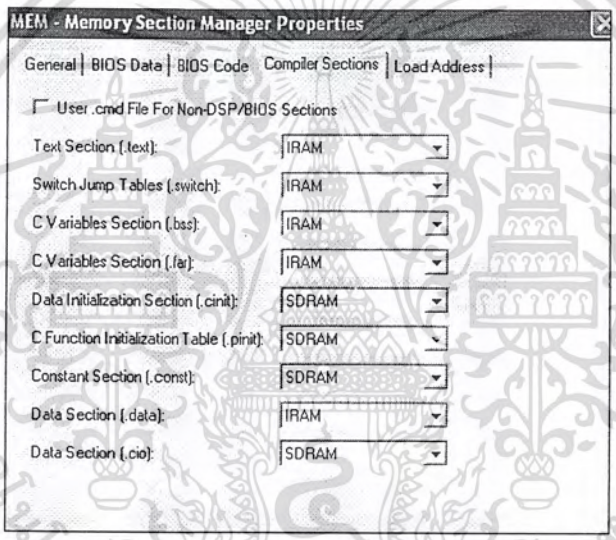
6.10 การทดลองทำการวัดเวลาหลังจากทำการออพติไมซ์โดยการบังคับโค้ดเซคชันให้อยู่ในหน่วยความจำภายในของ ดี เอส พี บอร์ด

<u>วัตถุประสงค์</u>

เพื่อวัดเวลาหลังจากการออพติไมซ์เปรียบเทียบกับก่อนการออพติไมซ์เวลาที่วัดได้จากการเข้ารหัสและถอดรหัสรวมกัน

วิธีการทดลอง

1. ลบฟังก์ชัน ที่เกี่ยวข้องกับ stdio.h และ stdlib.h เพื่อลดขนาดขนาดของโค้ด
2. แก้คอนฟิกกูเรชันไฟล์ .cdb โดยให้ section.text ไปอยู่ใน IRAM (internal ram)
3. กำหนดออพติไมซ์ระดับ Fiel ,โปรแกรมระดับการออพติไมซ์เป็น No External Var Refs



รูปที่ 6-29 แสดงการกำหนด Memory Map ให้กับ Segment ต่าง ๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลการทดลอง

จำนวนพื้นที่ ในการออพติไมซ์สำหรับการ search	ALL MAX(ms)	ALL AVG(ms)
8(แบบเดิม)	4.73	3.50
7	4.39	3.34
6	4.10	3.11
5	3.81	2.96
4	3.30	2.73
3	3.28	2.73
2	3.21	2.68
1	3.18	2.65

ตารางที่ 6-8 แสดงผลการทดลองวัดเวลาหลังจากทำการออพติไมซ์โดยการบังคับโค้ดเซคชัน ให้อยู่ในหน่วยความจำภายในของ ดี เอส พี บอร์ด

สรุปผลการทดลอง

การนำโค้ดเซคชัน เข้าไปทำงานใน internal ram สามารถทำงานได้เร็วกว่าเมื่อเปรียบเทียบกับการทำงานเมื่ออยู่ที่ Ram ภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.11 การทดลองทำการวัดเวลาหลังจากทำการออพติไมซ์โดยการแทนฟังก์ชันบางฟังก์ชั่นใน math.h ด้วยไลบรารี fastrts67x.h ของ ดี เอส พี

วัตถุประสงค์

เพื่อวัดเวลาหลังจากการออพติไมซ์เปรียบเทียบกับก่อนการออพติไมซ์เวลาที่วัดได้จากการเข้ารหัสและถอดรหัสรวมกัน

วิธีการทดลอง

1. ทำการเพิ่มไลบรารี fastrts67x เข้ามาในโปรเจค
2. เปลี่ยนฟังก์ชั่นที่ใน Math.h ที่สามารถใช้ fastrts67x.h แทนได้ ดังนี้

Function	Instruction
Log10	extern double log10dp (double x) ;
inv_sqrt	extern double rsqrtdp (double x) ;
pow	extern double powdp(double x);
sqrt	extern double sqrtdp (double x) ;
acos	extern double acosdp (double x) ;

ตารางที่ 6-9 แสดง ฟังก์ชัน ที่ได้ทำการแก้ไขการเรียกใช้

ผลการทดลอง

จำนวนพื้นที่ ในการออพติไมซ์สำหรับการ search	ALL MAX(ms)	ALL AVG(ms)	ACELP MAX(ms)	ACELP AVG(ms)
8(แบบเดิม)	4.39	3.32	1.19	0.79
7	4.04	3.06	0.99	0.61
6	3.81	2.82	0.77	0.47
5	3.48	2.65	0.66	0.36

ตารางที่ 6-10 แสดงผลการทดลองวัดเวลาหลังจากทำการออพติไมซ์โดยการบังคับโค้ดเซคชันให้อยู่ใน หน่วยความจำภายในของ ดี เอส พี บอร์ด และใช้ ดี เอส พีไลบรารี ช่วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จำนวนพื้นที่ ในการออพติไมซ์สำหรับการ search	ALL MAX(ms)	ALL AVG(ms)	ACELP MAX(ms)	ACELP AVG(ms)
4	3.18	2.53	0.39	0.27
3	2.96	2.44	0.26	0.21
2	2.87	2.40	0.21	0.18
1	2.85	2.36	0.19	0.17

ตารางที่ 6-10 แสดงผลการทดลองวัดเวลาหลังจากทำการออพติไมซ์โดยการบังคับโค้ดเซคชันให้อยู่ใน internal ram ของ ดี เอส พี บอร์ด และใช้ ดี เอส พีไลบราลี ช่วย (ต่อ)

สรุปผลการทดลอง

พบว่าการใช้ไลบราลี ของ DSP แทนไลบราลี ของ math.h ทำให้จำนวน Instruction ลดลง รวมทั้งขนาดของโค้ดของโปรแกรม ที่เล็กลง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 7
บทวิจารณ์และสรุป

7.1 สรุปผลที่ได้จากการทำโครงงาน

จากการศึกษาเราได้เรียนรู้ว่าการเข้ารหัสสัญญาณเสียงมีหลายประเภทอันได้แก่เวฟฟอร์มควอนไทดเซชัน (Wave form quantization) และพาราเมตริกควอนไทด์เซชัน (Parametric quantization) ซึ่งแต่ละวิธีนั้นก็เหมาะกับการประยุกต์ใช้งานที่แตกต่างกันเช่น งานที่ต้องการสัญญาณเสียงที่มีคุณภาพสูงและไม่สนใจเรื่องของการส่งผ่านเครือข่ายก็ควรใช้เวฟฟอร์มควอนไทดเซชันในขณะที่การส่งสัญญาณเสียงผ่านเครือข่ายนั้นจะเหมาะกับการเข้ารหัสเสียงแบบพาราเมตริกควอนไทด์เซชันเนื่องจากข้อมูลที่ใช้ในการส่งมีขนาดเล็กกว่า เวฟฟอร์มควอนไทดเซชันแต่ก็มีข้อเสียที่ต้องอาศัยทฤษฎีการคำนวณมากทำให้เราต้องใช้เวลาในการศึกษาการทำงานของการเข้ารหัสและถอดรหัส ในขณะเดียวกันนั้น แอพพลิเคชันที่ทำงานเกี่ยวกับสัญญาณเสียงนั้นส่วนใหญ่จะเป็นแบบเรียลไทม์แอพพลิเคชันซึ่งเราต้องคำนึงถึงเดทไลน์ (Deadline) และความซับซ้อนในการประมวลผลสัญญาณซึ่งถ้าการประมวลผลนั้นเกินระยะเวลาเดทไลน์ออกไปจะมีผลต่อคุณภาพเสียงที่ได้รับ ดังนั้นจึงมีความจำเป็นที่จะต้องลดความซับซ้อนในการประมวลผลลง โดยในโครงงานนี้ได้ศึกษาวิธีการจากวารสาร IEEE และนำทฤษฎีมาประยุกต์ใช้บน DSP ไมโครโพรเซสเซอร์พบว่าสามารถลดความซับซ้อนลงได้อย่างมากแต่ก็ต้องแลกกับคุณภาพของสัญญาณเสียงที่ลดลงทั้งนี้ต้องขึ้นอยู่กับผู้นำเอาอัลกอริทึมไปใช้งานว่าจะยอมรับได้ถึงระดับไหน

ศึกษาวิธีการเขียนโปรแกรมบนระบบฝังตัว (embedded system) แบบหนึ่งโดยใช้ DSP เป็น plat form ในการพัฒนา การพัฒนาบนระบบแบบฝังตัวนั้นมีสิ่งที่ต้องพิจารณาอยู่หลายเรื่อง เรื่องที่สำคัญที่สุดคือหน่วยความจำที่เราสามารถใช้งานได้โดยระบบฝังตัวนั้นมีหน่วยความจำน้อยมากเมื่อเทียบกับเครื่องคอมพิวเตอร์ส่วนบุคคลที่มีหน่วยความจำให้ใช้อย่างไม่จำกัด ดังนั้นการจัดการหน่วยความจำให้ใช้งานได้อย่างมีประสิทธิภาพจึงมีผลโดยตรงกับการทำงานของแอพพลิเคชันนั้น ๆ เช่น สมมติว่าโปรแกรมถูกเก็บไว้ในหน่วย ความจำภายนอกซึ่งไมโครโพรเซสเซอร์ต้องใช้เวลาในการเข้าถึงโปรแกรม = 25 clock cycles เมื่อเทียบกับหน่วยความจำภายในตัวไมโครโพรเซสเซอร์เองใช้เวลาในการเข้าถึงโปรแกรม ประมาณ 0 clock cycles ดังนั้นถ้าเราสามารถจัดการหน่อยความจำได้ดีจะทำให้ประสิทธิภาพโดยรวมของระบบพัฒนาขึ้นอย่างมาก ซึ่งเราสามารถจัดการได้โดยแบ่งโค้ดเซคชัน ออกเป็นส่วน ๆ และกำหนดให้อยู่ในหน่วยความจำส่วนต่าง ๆ ตามความต้องการใช้งานของแอพพลิชัน

หลังจากที่เราได้พัฒนาโปรแกรมเรียบร้อยแล้วพบว่าการนำโปรแกรมเข้าไปใช้ร่วมกับ แอพพลิเคชันอื่นนั้นทำได้ยากเนื่องจากโปรแกรมที่ได้พัฒนามายังไม่มีฟังก์ชัน มาตรฐานในการติดต่อระหว่าง แอพพลิเค

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชันดังนั้นเราจึงได้ศึกษาการโปรแกรมตามมาตรฐานของผู้ผลิต (Texas instrument Standard) และได้ปรับปรุงโปรแกรมให้เป็นตามมาตรฐานของผู้ผลิต

7.2 แนวทางในการพัฒนาต่อ

หลังจากการทดสอบพบว่าการทำงานนั้นยังทำงานได้ไม่ตรงกับเป้าหมายที่ได้วางไว้คือ การเข้ารหัสและถอดรหัสได้พร้อมกัน 4 ช่องสัญญาณแต่โปรแกรมทำได้เพียง 3 ช่องสัญญาณเท่านั้นการพัฒนาต่อไปจึงอาจทำการออพติไมซ์ได้ในระดับของภาษา Assembly

สำหรับ แอพพลิเคชันที่เหมาะกับการนำโครงงานนี้ไปใช้ได้แก่ VOIP โดยการนำไลบรารี ที่เกี่ยวข้อง เช่น DTMF, Echo Canceller นำมา integrate เข้าด้วยกันเป็นระบบ VOIP ได้ในที่สุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

[1] ITU-T Recommendation G.729 Annex A , B and C+ , 1996

[2] Venkatraman Atti and Andreas Spanias,A Simulation tool for introducing algebraic CELP (ACELP) coding concepts in a DSP course ,

[3] Antti Kiviluoto, Speech Coding Standards

[4] Climent Nadeu, Speech analysis

[5] Jonathan D. Rosenberg, Lucent Technologies, Bell Laboratories & Columbia University, G.729 Error Recovery for Internet Telephony

[6] F.-K Chen, J.-F Yang and Y.-L Yan, Candidate scheme for fast ACELP search, IEEE 2002

[7] TMS320 DSP Algorithm Standard API Reference, Literature Number: SPRU360C

[8] TMS320 DSP Algorithm Standard Rule & Guidelines, Literature Number: SPRU352E

[9] TMS320 DSP Algorithm Standard Algorithm Standard Developer's Guide , Literature Number: SPRU424C

[10] A Consumer's Guide to using eXpressDSP-Compliant Algorithms for DSP Software Products, Literature Number: SPRA810

[11] A Technical Overview of eXpressDSP-Complianct Algorithms for DSP Software Producers, Literature Number: SPRA579C

[12] Using the TMS320 DSP Algorithm Standard in Static System, Literature Number: SPRA557B

[13] Using the TMS320 DSP Algorithm Standard in Dynamic System, Literature Number: SPRA580B

[14] Techniques for implementing Shared Relocatable Buffers Using the TMS320 DSP Algorithm Standard, Literature Number: SPRA790

[15] TMS320C6000 Assembly Language Tools User's Guide, Literature Number: SPRU186I ,April 2001

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้