

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การสร้างชุดเข้ารหัส Convolution และชุดถอดรหัส Viterbi
IMPLEMENTATION OF CONVOLUTION ENCODER AND VITERBI DECODER



2/พ.
83/4 ก
2547

เลขหมู่.....
เลขทะเบียน.....**61735**
วัน,เดือน,ปี.....**2 1 ก.ค. 2549**

b. 115883 12
i.

ปริญญานิพนธ์เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

hm

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดลอกหรือเผยแพร่เนื้อหาของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสาร
วิศวกรรมโทรคมนาคม

การสร้างชุดเข้ารหัส Convolution และชุดถอดรหัส Viterbi
IMPLEMENTATION OF CONVOLUTION ENCODER AND VITERBI DECODER



ปริญญานิพนธ์เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิศวกรรมโทรคมนาคม
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2547

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การสร้างชุดเข้ารหัส Convolution และชุดถอดรหัส Viterbi

IMPLEMENTATION OF CONVOLUTIONAL ENCODER AND VITERBI DECODER

ผู้จัดทำ

นายรังสรรค์ แก้วพานิชย์ 45015027

นายวรวิทย์ ตีรณคมน์ 45015031

นายประเสริฐ ผิวงาม 45015061

P. Suph. อาจารย์ที่ปรึกษา
(ดร.พรชัย ททรัพย์นิต)

อ. สรวัดน์ ชิวปรีชา อาจารย์ที่ปรึกษา
(อ. สรวัดน์ ชิวปรีชา)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเข้ารหัสขั้ว Convolution และขั้วถอดรหัส Viterbi
IMPLEMENTATION OF CONVOLUTION ENCODER AND VITERBI DECODER

โดย นายรังสรรค์ แก้วพานิชย์ 45015027

นายวรวิทย์ ตรีธณกมล 45015031

นายประเสริฐ ผิวงาม 45015061

อาจารย์ที่ปรึกษา ดร.พรชัย ทรัพย์นิธิ

อ. สรวิทย์ ชิวปรีชา

บทคัดย่อ

การเข้ารหัสขั้วคอนโวลูชันและการถอดรหัส Viterbi เป็นเทคนิคหนึ่งที่ใช้แก้ไขความผิดพลาดจากการรับส่งข้อมูลในระบบสื่อสารเชิงดิจิทัล เพื่อเพิ่มประสิทธิภาพของการสื่อสารดังกล่าวในช่องสัญญาณเสียง (Voice Service) ของระบบโทรศัพท์เคลื่อนที่ ได้นำการเข้ารหัสแบบคอนโวลูชันและการถอดรหัส Viterbi นี้มาใช้งาน ซึ่งโครงการนี้ได้นำเสนอการเข้ารหัสขั้วและถอดรหัสด้วยอุปกรณ์ FPGA โดยอธิบายการทำงานของวงจรที่ออกแบบด้วยภาษา Verilog HDL แล้วสังเคราะห์ลงบนอุปกรณ์ FPGA เพื่อทดสอบการทำงานของอุปกรณ์ที่สร้างขึ้น

ABSTRACT

Convolutional codes with the combination of Viterbi decoding is a powerful error-correction technique for information transmission in digital communication. These techniques for voice transmission. The purpose of this project is to study the dynamics of this code and implement the standardized convolution code. This work is done by using Verilog HDL and implemented on platform for FPGA

สารบัญ

หน้า

บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
สารบัญ	III
สารบัญตาราง	IV
สารบัญรูป	VIII
รายการคำย่อ	IX
รายการสัญลักษณ์	XI
บทที่ 1 บทนำ	1
บทที่ 2 หลักการและการใช้งานรหัสแก้ไขความผิดพลาดล่วงหน้าและรหัสคอนไวลูชัน	3
2.1 พื้นฐานรหัสแก้ไขความผิดพลาด	3
2.1.1 รูปแบบระบบสื่อสารดิจิทัล	3
2.1.2 วิธีการแก้ไขความผิดพลาดของข้อมูลในระบบสื่อสารดิจิทัล	4
2.1.3 ชนิดของการเกิดความผิดพลาดในระบบสื่อสาร	5
2.1.4 รีตันแดนซีและอัตราการเข้ารหัส	5
2.1.5 อัตราการขยายการเข้ารหัส (Coding Gain)	5
2.2 สัญญาณรบกวน	6
2.2.1 สัญญาณรบกวนแบบ Additive	6
2.2.1.1 สัญญาณรบกวนเนื่องจากอุณหภูมิ (Thermal noise)	6
2.2.1.2 สัญญาณรบกวนจากบรรยากาศ (Atmospheric noise)	7
2.2.1.3 สัญญาณรบกวนจากอวกาศ (Galactic noise)	7
2.2.1.4 สัญญาณรบกวนจากมนุษย์ (Man-made noise)	7
2.2.2 สัญญาณรบกวนแบบ Multiplicative	7
2.3 การส่งข้อมูลผ่านช่องสัญญาณ	8
2.3.1 ช่องสัญญาณแบบ AWGN (The Addition White Gaussian Noise Channel)	9
2.3.2 ความจุของช่องสัญญาณแบบ Continuous AWGN	10
2.4 ทฤษฎีของการเข้ารหัสช่องสัญญาณ	11
2.5 รหัสแบบบล็อก (Block Code)	12
2.5.1 การเข้ารหัสแบบลิเนียร์บล็อก(Linear Block Code)	12
2.5.2 การถอดรหัสแบบลิเนียร์บล็อก	12
2.5.3 ความสามารถในการตรวจจับและแก้ไขความผิดพลาด	13

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6 รหัสคอนโวลูชัน (Convolution Code)	14
2.6.1 การเข้ารหัสคอนโวลูชัน	14
2.6.2 การวิเคราะห์การทำงานของการทำงานการเข้ารหัสคอนโวลูชัน	15
2.6.3 แผนภาพแสดงการทำงานของการทำงานการเข้ารหัสคอนโวลูชัน	15
2.6.3.1 แผนภาพสเตต (State Diagram)	15
2.6.3.2 แผนภาพทรี (Tree Diagram)	16
2.6.3.3 แผนภาพ Trellis (Trellis Diagram)	17
2.6.4 การถอดรหัสคอนโวลูชัน	19
2.7 การถอดรหัสด้วยวิธี Viterbi	19
2.7.1 รูปแบบการทำงานของวงจรถอดรหัสด้วยวิธี Viterbi	19
2.7.2 การถอดรหัสด้วย Viterbi Algorithm	20
2.7.3 การถอดรหัสคอนโวลูชันที่ R เท่ากับ 1/2 และ K เท่ากับ 3 ด้วยวิธี Viterbi	21
บทที่ 3 เอฟพีจีเอพื้นฐานและการออกแบบ	24
3.1 ความรู้พื้นฐานวงจรรวมแบบดิจิทัล	24
3.1.1 วงจรรวมแบบลอจิกมาตรฐาน (Standard Logic)	24
3.1.2 วงจรรวมแบบพีแอลดี (Programmable Logic Device: PLD)	24
3.2 ความรู้พื้นฐานอุปกรณ์เอฟพีจีเอ (Field Programmable Gate Array: FPGA)	25
3.2.1 โครงสร้างของเอฟพีจีเอของบริษัท Xilinx	26
3.2.2 โครงสร้างภายในเอฟพีจีเอ Xilinx	26
3.2.3 รายละเอียดของอุปกรณ์ภายในเอฟพีจีเอ Xilinx	27
3.2.3.1 บล็อกอินพุตเอาต์พุต (Input/Output Block: IOB)	27
3.2.3.2 บล็อก Configurable Logic Block (CLB)	27
3.2.3.3 บล็อกหน่วยความจำ 18 กิโลบิต	30
3.2.4.4 บล็อกวงจรคูณขนาด 18x 18 บิต	30
3.2.4.5 บล็อก Digital Clock Manager (DCM)	31
3.2.4.6 บล็อกเชื่อมต่อและหาเส้นทาง (Routing Resources)	32
3.3 การออกแบบและติดตั้งวงจรดิจิทัลบนเอฟพีจีเอ	33
3.4 ภาษาเวริล็อก (Verilog HDL)	34
3.4.1 ประวัติความเป็นมา	34
3.4.2 โครงสร้างของภาษาเวริล็อก	35
บทที่ 4 โครงสร้างชุดถอดรหัสและการออกแบบ	36
4.1 โครงสร้างและการทำงานของชุดถอดรหัส Viterbi	36
4.1.1 โครงสร้างภาค Branch Metric Unit (BMU)	36
4.1.2 โครงสร้างภาค Add-Compare-Select Unit (ACSU)	38

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.3 โครงสร้างภาค Survivor Memory Unit (SMU)	40
4.1.4 การเปรียบเทียบ โครงสร้างของชุดถอดรหัส Viterbi แบบต่างๆ	42
4.2 การออกแบบสร้างสำหรับปริภูมิพหุนาม	43
บทที่ 5 ผลการออกแบบและทดสอบการทำงาน	44
5.1 มอดูลชุดเข้ารหัส	44
5.1.1 ผลการจำลองการทำงานมอดูลชุดเข้ารหัส	44
5.2 มอดูลชุดถอดรหัส	45
5.2.1 ผลการจำลองการทำงานมอดูล BMU (Branch metric Unit)	45
5.2.2 ผลการจำลองการทำงานมอดูล ACSU (Add-Compare-Select Unit)	47
5.2.3 ผลการจำลองการทำงานมอดูล SMU (Survivor Memory Unit)	48
5.2.4 ผลการจำลองชุดถอดรหัส	49
5.3 ผลการทดลองที่ได้จากการสังเคราะห์โปรแกรมการเข้ารหัสและถอดรหัสลงบน อุปกรณ์ FPGA	51
5.4 การทดสอบประสิทธิภาพของ ชุดเข้ารหัส และ ชุดถอดรหัส ในช่องสัญญาณ AWGN	53
บทที่ 6 สรุปผลและข้อเสนอแนะ	55
ภาคผนวก	
กิตติกรรมประกาศ	
หนังสืออ้างอิง	

สารบัญรูป

	หน้า
รูปที่ 2.1 บล็อกไดอะแกรมองค์ประกอบพื้นฐานของระบบสื่อสารดิจิทัล	4
รูปที่ 2.2 วงจรเข้ารหัสคอนวูลูชันที่ค่า R เท่ากับ 1/2 และค่า K เท่ากับ 3	14
รูปที่ 2.3 แผนภาพสเตทที่ R เท่ากับ 1/2 และ K เท่ากับ 3	16
รูปที่ 2.4 แผนภาพทรีที่ R เท่ากับ 1/2 และ K เท่ากับ 3	17
รูปที่ 2.5 แผนภาพ Trellis ที่ R เท่ากับ 1/2 และ K เท่ากับ 3	18
รูปที่ 2.6 ตัวอย่างการใช้งานแผนภาพ Trellis	18
รูปที่ 2.7 ค่า Branch Metric ในแผนภาพ Trellis	21
รูปที่ 2.8 Path Metric ในเวลา t_1 ถึง t_3	22
รูปที่ 2.9 เส้นทางของข้อมูลที่ได้อจากการถอดรหัสด้วยวิธี Viterbi	22
รูปที่ 3.1 ไดอะแกรมเทคโนโลยีวงจรรวมแบบดิจิทัล	25
รูปที่ 3.2 โครงสร้างทั่วไปของเอฟพีจีเอ	27
รูปที่ 3.3 โครงสร้างของบล็อกอินพุตเอาต์พุต	28
รูปที่ 3.4 โครงสร้างของ Slice	29
รูปที่ 3.5 การเชื่อมต่อบล็อกหน่วยความจำ 18 กิโลบิตและบล็อกวงจรคูณขนาด 18×18 บิต	30
รูปที่ 3.6 โครงสร้างของบล็อกเชื่อมต่อและหาเส้นทาง	31
รูปที่ 3.7 ขั้นตอนการออกแบบและติดตั้งวงจรดิจิทัลบน FPGA	36
รูปที่ 4.1 บล็อกไดอะแกรมของชุดถอดรหัส Viterbi	37
รูปที่ 4.2 ไดอะแกรมภาค BMU(ก) วิธีการเปิดตาราง (ข) วิธีการบวกที่ K เท่ากับ 1/2	38
รูปที่ 4.3 ไดอะแกรม Add-Compare-Select (ACS)	38
รูปที่ 4.4 โครงสร้าง ACS แบบ Butterfly	39
รูปที่ 4.5 วงจร SMU แบบ Register Exchange ที่ค่า K เท่ากับ 3	41
รูปที่ 4.6 ไดอะแกรม SMU แบบ Trace back ที่ค่า K เท่ากับ 3	42
รูปที่ 5.1 แสดงการจำลองการทำงานของโปรแกรมเข้ารหัสเมื่ออินพุต (11110000)	44
รูปที่ 5.2 แสดงการจำลองการทำงานของโปรแกรมเข้ารหัสเมื่ออินพุต (1111111100001111)	44
รูปที่ 5.3 ผังแสดงวงจรระดับ RTL ที่ได้อจากการสังเคราะห์ชุด เข้ารหัส	45
รูปที่ 5.4 แสดงการจำลองการทำงานของชุด BMU	45
รูปที่ 5.5 ผังแสดงวงจรระดับ RTL ที่ได้อจากการสังเคราะห์ชุด BMU	46
รูปที่ 5.6 แสดงการจำลองการทำงานของชุด ACSU	47
รูปที่ 5.7 ผังแสดงวงจรระดับ RTL ที่ได้อจากการสังเคราะห์ชุด ACSU	48
รูปที่ 5.8 แสดงการจำลองการทำงานของชุด SMU	48
รูปที่ 5.9 ผังแสดงวงจรระดับ RTL ที่ได้อจากการสังเคราะห์ชุด SMU	49

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5.10 แสดงการจำลองการทำงานของโปรแกรมถอดรหัสเมื่อผ่านการเข้ารหัสด้วย อินพุต (11110000)	49
รูปที่ 5.11 แสดงการจำลองการทำงานของโปรแกรมถอดรหัสเมื่อผ่านการเข้ารหัสด้วยอิน พุต(1111111100001111)	50
รูปที่ 5.12 ผังแสดงวงจรระดับ RTL ที่ได้จากการสังเคราะห์ชุดถอดรหัส	50
รูปที่ 5.13 แสดงเอาต์พุตที่ได้จากการป้อนข้อมูล 00011111	51
รูปที่ 5.14 แสดงเอาต์พุตที่ได้จากการป้อนข้อมูล 11110000	52
รูปที่ 5.15 แสดงเอาต์พุตที่ได้จากการป้อนข้อมูล (11111100)	52
รูปที่ 5.16 แสดงบล็อกไดอะแกรมการทดสอบประสิทธิภาพของ ชุดเข้ารหัส และ ชุดถอดรหัส ในช่องสัญญาณ AWGN	53
รูปที่ 5.17 กราฟแสดงค่าผลของอัตราความผิดพลาดบิต (BER) กับค่ากำลังของสัญญาณต่อสัญญาณ รบกวน (E_b/N_0) ที่ค่า $K=3, R=1/2$ เทียบกับกรณีเข้ารหัสและไม่เข้ารหัส ในช่องสัญญาณ AWGN โดยใช้โปรแกรม Matlab	54



สารบัญตาราง

	หน้า
ตารางที่ 3.1 ขนาดหน่วยความจำที่สามารถเลือกใช้ได้ในบล็อกลหน่วยความจำ	29
ตารางที่ 4.1 ผลเปรียบเทียบข้อดีข้อเสียของโครงสร้างของชุดถอดรหัส Viterbi แบบต่างๆ	43



รายการคำย่อ

3 rd Generation Partnership Project	3GPP
Application Specific Integrated Circuit	ASIC
Add Compare Select	ACS
Add Compare Select Unit	ACSU
Additive White Gaussain Noise	AWGN
Binary Phase Shift Keying	BPSK
Bit Error Rate	BER
Branch Metric Unit	BMU
Code Devision Multiple Access	CDMA
Complex Programmable Logic Device	CPLD
Configurable Logic Block	CLB
Cyclic Redundancy Check	CRC
Digital Signal Processing	DSP
Forward Error Correction	FEC
Frequency Maximum	Fmax
Field Programmable Gate Array	FPGA
Hardware Description Language	HDL
Intellectaul Property	IP
Look Up Table	LUT
Mega bit per second	Mbps
Motion Picture Experts Group	MPEG
Non-Recurring Engineering	NRE
Open Verilog International	OVI
Programmable Array Logic	PAL
Programmable Logic Array	PLA
Programmable Logic Device	PLD
Path Metric	PM
Path Metric Memory	PMM
Programmable Read Only Memory	PROM
Erasable Programmable Read Only Memory	EPROM
Random Access Memory	RAM
Register Exchange	RE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Read Only Memory	ROM
Register Transfer Level	RTL
Signal to Parallel	SP
Survivor Memory Unit	SMU
Signal-to-Noise Ration	SNR
Static Random Access Memory	SRAM
Third Generation Mobile System	3G
Trace Back	TB
Very High Speed Integrated Circuit	VHSIC
Very Large Scale Integrated Circuit	VLSI
VHISC Hardware Description Language	VHDL
Wideband CDMA	W-CDMA
Wireless Local Access Network	WLAN



รายการสัญลักษณ์

Branch Metric	BM
Coding Rate	R
Constraint Length	K
Decision Bit	DB
Generator Polynomial	G และ g
Minimum Hamming Distance	d_{min}
Path Metric	PM
Redudant	r
State Metric	SM
Syndrome	S



บทที่ 1

บทนำ

ในปัจจุบันความต้องการสำหรับระบบการสื่อสารและจัดเก็บข้อมูลดิจิทัลที่มีประสิทธิภาพและเชื่อถือได้นั้นได้เพิ่มขึ้นอย่างรวดเร็ว ความต้องการเหล่านี้ได้ถูกกระตุ้นจากการเกิดขึ้นของโครงข่ายแลกเปลี่ยนข้อมูลข่าวสารที่มีขนาดใหญ่และความเร็วในการส่งข้อมูลที่สูงขึ้นโดยสิ่งสำคัญที่นอกออกแบบระบบจะต้องพิจารณาถึงคือการควบคุมความผิดพลาดในการรับส่งข้อมูลเพื่อที่เราจะสามารถนำหรือสร้างข้อมูลกลับคืนมาได้เป็นอย่างดี

ในปี ค.ศ. 1948 นาย Claude Shannon ได้นำเสนอทฤษฎีซึ่งเป็นรากฐานสำคัญของทฤษฎีข่าวสาร (Information Theory) โดยทางทฤษฎีของ Shannon นั้นได้แสดงถึงคุณลักษณะของช่องสัญญาณ (Channel) โดยค่าของระดับสัญญาณรบกวน แบนวิดท์ และระดับกำลังงานของสัญญาณ ซึ่งถูกใช้ในการกำหนดค่าตัวแปร C หรือความจุของช่องสัญญาณ (Channel Capacity) โดยเป็นตัวกำหนดขอบเขตสูงสุดของอัตราการส่งสัญญาณข้อมูลผ่านช่องสัญญาณซึ่งยังคงรักษาระดับความน่าเชื่อถือที่ทางด้านรับสามารถรับได้ ดังนั้นตราบดีที่เราจะส่งข้อมูลด้วยอัตราที่ต่ำกว่า C แล้ว ความน่าจะเป็นของความผิดพลาดในการรับส่งข้อมูลนั้นสามารถทำให้มีค่าน้อยลงตามต้องการได้ โดยการใช้สัญญาณเข้ารหัสที่มีความยาวของสัญญาณที่เหมาะสม หลังจากการนำเสนอของ Shannon ก็ทำให้เกิดการวิจัยอย่างกว้างขวางในการคิดสร้างรหัสที่ใช้ในการควบคุมการผิดพลาด ซึ่งในปัจจุบันเราสามารถแบ่งรหัสออกเป็น 2 ประเภทคือ ลิเนียร์บล็อกโค๊ด (Linear block codes) และการเข้ารหัสประสาน (Convolutional codes)

สำหรับการออกแบบระบบนั้น ในการที่จะลดความน่าจะเป็นของความผิดพลาดในการส่งข้อมูลผ่านช่องสัญญาณนั้นสามารถกระทำได้ 2 วิธีคือ การเพิ่มค่ากำลังงานที่ใช้ในการส่งสัญญาณหรือการเข้ารหัสควบคุมการผิดพลาด แต่สำหรับวิธีการเพิ่มค่ากำลังงานนั้นจำเป็นต้องปรับเปลี่ยนอุปกรณ์ขนาดและน้ำหนักของอุปกรณ์จะเพิ่มขึ้นนั้นหมายถึงจะทำให้ต้องเสียค่าใช้จ่ายที่สูงขึ้นด้วย ส่วนการใช้การเข้ารหัสควบคุมความผิดพลาดนั้นจะทำให้ค่าแบนวิดท์ของระบบเพิ่มขึ้น ดังนั้นนักออกแบบระบบจำเป็นต้องคำนึงถึงทรัพยากรที่มีอยู่ เพื่อให้เกิดประสิทธิภาพสูงสุดในการใช้งาน

วัตถุประสงค์ของปริญญานิพนธ์

1. เพื่อศึกษาเกี่ยวกับหลักการการแก้ปัญหาความผิดพลาดของข้อมูลในการส่งข้อมูลระบบดิจิทัลโดยใช้วิธีการการเข้ารหัสและถอดรหัส
2. เพื่อศึกษาเกี่ยวกับการเขียนโปรแกรมภาษา Verilog HDL ที่ใช้ในการออกแบบทางลอจิก
3. เพื่อศึกษาเกี่ยวกับการนำอุปกรณ์เอ็พไฟจีเอมาใช้งานในการทดสอบการเข้ารหัสและถอดรหัส
4. เพื่อนำเอาทฤษฎีการเข้ารหัสและถอดรหัสมาเปรียบเทียบกับผลการทดลองบนอุปกรณ์เอ็พไฟจีเอว่าจะได้ผลการทดลองถูกต้องตรงกับทฤษฎีก่อนนำไปใช้งาน

ขอบเขตของปริญญานิพนธ์

โครงการนี้เป็นการออกแบบชุดการเข้ารหัสโดยใช้การ Convolution และการถอดรหัสโดยใช้วิธีการ Viterbi ซึ่งจะใช้ภาษา Verilog HDL เป็นตัวอธิบายการทำงาน และจำลองการทำงานลงบนตัวของอุปกรณ์เอฟพีจีเอ ซึ่งจะช่วยลดความผิดพลาดของข้อมูลและเพิ่มประสิทธิภาพให้กับการสื่อสารข้อมูลในระบบดิจิทัล

เนื้อหาของปริญญานิพนธ์

บทที่ 1 จะกล่าวถึงความเป็นมาแนวคิดในการแก้ปัญหาคำผิดพลาดของข้อมูลในการสื่อสารระบบดิจิทัล วัตถุประสงค์และขอบเขตของปริญญานิพนธ์

บทที่ 2 จะกล่าวถึงการเข้ารหัสและถอดรหัสข้อมูลของการสื่อสารระบบดิจิทัลซึ่งจะบอกถึงวิธีการเข้ารหัส Convolution และถอดรหัสแบบ Viterbi และสาเหตุของความผิดพลาดของข้อมูลในการสื่อสารระบบดิจิทัล

บทที่ 3 จะกล่าวถึงโครงสร้างของอุปกรณ์เอฟพีจีเอ และวิธีการนำอุปกรณ์เอฟพีจีเอมาใช้งาน

บทที่ 4 จะกล่าวแนวความคิดในการออกแบบตัวเข้ารหัส และตัวถอดรหัสและส่วนต่าง ๆ ของชุดทดลอง

บทที่ 5 การทดลอง และผลการทดลอง

บทที่ 6 จะกล่าวถึง บทวิจารณ์และสรุปผลของการทำโครงการนี้
หนังสืออ้างอิง

บทที่ 2

หลักการและการใช้งานรหัสแก้ไขความผิดพลาดและรหัสคอนวอลูชัน

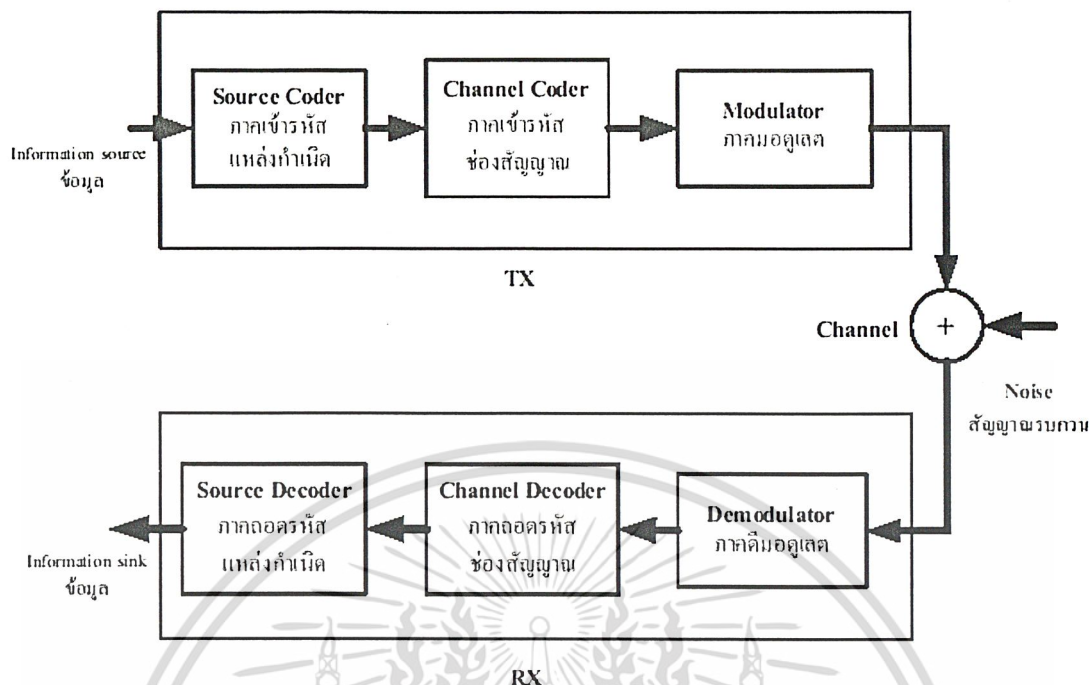
รหัสแก้ไขความผิดพลาด (Error Control Coding) เป็นเทคนิคที่ใช้ในการแก้ไขความผิดพลาดจากการรับส่งข้อมูลในระบบสื่อสารดิจิทัล ซึ่งความผิดพลาดที่เกิดขึ้นนั้นเกิดจากสาเหตุหลายประการ และสาเหตุเหล่านี้ไม่สามารถคาดคะเนเวลาเกิดได้ โดยเฉพาะสาเหตุที่สำคัญคือ การรบกวนจากสัญญาณรบกวน (Noise) ดังนั้นจึงจำเป็นต้องใช้รหัสดังกล่าวเพื่อป้องกันหรือแก้ไขความผิดพลาดของสาเหตุเหล่านี้ โดยในบทนี้กล่าวถึงพื้นฐานของรหัสแก้ไขความผิดพลาด รหัสแบบบล็อก (Block Code) รหัสคอนวอลูชัน (Convolution Code) การถอดรหัสคอนวอลูชันด้วยวิธี Viterbi

2.1 พื้นฐานรหัสแก้ไขความผิดพลาด

รหัสแก้ไขความผิดพลาดนั้นเป็นส่วนที่สำคัญในระบบสื่อสารดิจิทัล ซึ่งการใช้งานรหัสแก้ไขความผิดพลาดทำให้ค่าอัตราความผิดพลาด (Error Rate) ในการส่งข้อมูลลดลงและทำให้ประสิทธิภาพของระบบสื่อสารเพิ่มขึ้น ซึ่งส่วนนี้กล่าวถึงหลักการพื้นฐานต่างๆ ของรหัสแก้ไขความผิดพลาดนี้

2.1.1 รูปแบบระบบสื่อสารดิจิทัล

พื้นฐานระบบสื่อสารดิจิทัลทั่วไปนั้นประกอบด้วยภาคส่ง (Transmitter : Tx) ภาครับ (Receiver : Rx) และช่องสัญญาณการสื่อสาร (Channel) ในการทำงานของภาคส่งนั้นข้อมูลข่าวสาร (Information Source) ต้องถูกเข้ารหัสข้อมูลโดยภาคเข้ารหัสแหล่งกำเนิด (Source Coder) เพื่อให้ขนาดของข้อมูลข่าวสารที่ถูกส่งลดลง ตัวอย่างการเข้ารหัสนี้ได้แก่ การบีบอัดข้อมูลภาพแบบ JPEG หรือ MPEG เป็นต้น จากนั้นข้อมูลนี้จะถูกเข้ารหัสเพื่อป้องกันความผิดพลาดอีกครั้งที่ภาคเข้ารหัสช่องสัญญาณ (Channel Coder) เพื่อป้องกันหรือลดความผิดพลาดต่างๆ ที่เกิดสัญญาณรบกวนหรืออื่นๆ ในการส่งข้อมูลผ่านช่องสัญญาณการสื่อสาร ซึ่งข้อมูลที่ถูกเข้ารหัสแก้ไขความผิดพลาดนี้ประกอบด้วย ข้อมูลข่าวสารและส่วนของข้อมูลที่เพิ่มขึ้นมา ซึ่งเรียกว่า รีดันแดนซี (Redundancy) สุดท้ายข้อมูลนี้จะถูกมอดูเลต (Modulate) เช่น การมอดูเลตเชิงเฟส (Phase Shift Keying: FSK) ที่ภาคมอดูเลต (Modulator) เพื่อให้เหมาะสมกับช่องสัญญาณต่อไป



รูปที่ 2.1 บล็อกไดอะแกรมองค์ประกอบพื้นฐานของระบบสื่อสารดิจิทัล

ส่วนในภาครับนั้น การทำงานของส่วนต่างๆ จะสอดคล้องกับภาคส่งสัญญาณข้อมูลที่ได้รับได้ถูกดีมอดูเลตกลับคืนมาที่ภาคดีมอดูเลต (Demodulator) ซึ่งข้อมูลที่ได้รับนี้อาจมีความผิดพลาดรวมเข้ามาด้วย ข้อมูลนี้จะถูกถอดรหัสและถูกแก้ไขความผิดพลาดที่เกิดขึ้นให้ถูกต้องที่ภาคถอดรหัสช่องสัญญาณ (Channel Decoder) จากนั้นข้อมูลที่ได้จากภาคถอดรหัสช่องสัญญาณนี้จะถูกแปลงให้เป็นข้อมูลข่าวสารเดิม โดยภาคถอดรหัสแหล่งกำเนิด (Source Decoder) โดยบล็อกไดอะแกรมองค์ประกอบพื้นฐานของระบบสื่อสารดิจิทัลทั่วไปนั้นแสดงดังรูปที่ 2.1

2.1.2 วิธีการแก้ไขความผิดพลาดของข้อมูลในระบบสื่อสารดิจิทัล

วิธีที่ใช้ในการป้องกันหรือแก้ไขความผิดพลาดของข้อมูลในระบบสื่อสารดิจิทัลนั้นแบ่งได้เป็นสองลักษณะใหญ่ๆ ด้วยกันคือ วิธีการส่งซ้ำอัตโนมัติ (Automatic Retransmission Query) หรือเรียกว่า “ARQ” และวิธีการแก้ไขความผิดพลาดล่วงหน้า (Forward Error Correction) หรือเรียกว่า “FEC” วิธีการทำงานของ ARQ นั้น เมื่อภาครับตรวจสอบได้ว่าข้อมูลที่ได้รับเกิดความผิดพลาดก็จะทำการร้องขอให้มีการส่งข้อมูลใหม่อีกครั้ง การตรวจสอบความผิดพลาดที่เกิดขึ้นนั้นใช้ส่วนของรีดักแตรนซ์ในการทำงาน วิธี ARQ นั้นใช้ในการสื่อสารแบบสองทาง (Duplex) เท่านั้นส่วนวิธีการทำงานของ FEC นั้น เมื่อภาครับตรวจสอบได้ว่าข้อมูลที่ได้รับเกิดความผิดพลาดภาครับก็สามารถทำการแก้ไขข้อมูลที่ผิดพลาดให้ถูกต้องได้ โดยใช้ส่วนของข้อมูลข่าวสารและรีดักแตรนซ์ในแก้ไขข้อมูลที่ผิดพลาดดังกล่าว วิธี FEC สามารถใช้ในการสื่อสารแบบทางเดียว (Simplex) หรือสองทางก็ได้

การทำงานของวิธี ARQ นั้นง่ายและใช้รีดันแดนซ์น้อยกว่า FEC แต่มีความเร็วที่ใช้ในการรับส่งข้อมูลนั้นไม่แน่นอน เนื่องจากต้องมีการส่งข้อมูลใหม่อีกครั้งเมื่อตรวจสอบได้ว่ามีความผิดพลาดเกิดขึ้น ส่วนวิธี FEC นั้นมีความซับซ้อนกว่า ARQ แต่วิธี FEC นี้สามารถรับส่งข้อมูลได้อย่างรวดเร็ว สำหรับระบบสื่อสารแบบดิจิทัลที่ต้องการประสิทธิภาพสูงนั้นการป้องกันการผิดพลาดจะใช้สองวิธีมารวมกัน ซึ่งเรียกวิธีนี้ว่า การแก้ไขความผิดพลาดแบบไฮบริดจ์ (Hybrid Error Control) วิธีการป้องกันและแก้ไขใช้วิธี FEC ในขั้นตอนแรกและส่วนเกินความสามารถของวิธี FEC ก็ใช้วิธี ARQ

2.1.3 ชนิดของการเกิดความผิดพลาดในระบบสื่อสาร

การเกิดความผิดพลาดกับการรับส่งข้อมูลในระบบสื่อสารแบบดิจิทัลนั้นสามารถแบ่งได้ออกเป็นสองประเภทคือ การเกิดความผิดพลาดแบบสุ่ม (Random) และการเกิดความผิดพลาดแบบ Burst การเกิดความผิดพลาดแบบสุ่มนั้นเป็นการเกิดความผิดพลาดที่สามารถเกิดขึ้นได้อย่างอิสระกับข้อมูลที่รับส่งในแต่ละบิต ตัวอย่างการเกิดความผิดพลาดเช่น ความผิดพลาดที่เกิดจากการสอดแทรก (Interference) จากสัญญาณอื่นๆ เป็นต้น ส่วนการเกิดความผิดพลาดแบบ Burst นั้นเป็นการเกิดความผิดพลาดที่เกิดกับข้อมูลที่รับส่งหลายๆ บิตติดกัน โดยจำนวนบิตที่ผิดพลาดจากบิตแรกจนถึงบิตสุดท้ายเรียกว่า Burst Length การเกิดความผิดพลาดแบบสุ่มสามารถใช้รหัสแก้ไขความผิดพลาดในแก้ไขได้ ส่วนความผิดพลาดแบบ Burst นั้นการแก้ไขยากกว่าแบบสุ่ม การแก้ไขต้องใช้รหัสแก้ไขความผิดพลาดร่วมกับส่วน Interleaver

2.1.4 รีดันแดนซ์และอัตราการเข้ารหัส

วิธีการเข้ารหัสข้อมูลเพื่อแก้ไขความผิดพลาดนั้น เป็นการนำเอาข้อมูลข่าวสารมาแบ่งออกเป็นส่วนๆ ซึ่งขนาดเท่ากับ k บิต โดยข้อมูลที่ได้จากการเข้ารหัสแล้วจำนวน n บิต เรียกว่า คำรหัส หรือ Codeword ซึ่งประกอบด้วยข้อมูลข่าวสารเดิมและส่วนของข้อมูลที่เพิ่มขึ้นซึ่งเรียกว่า รีดันแดนซ์ (Redundancy) หรือ r ซึ่งรีดันแดนซ์นี้ใช้ในการตรวจสอบหรือแก้ไขข้อมูลผิดพลาด สามารถคำนวณได้จาก

$$r = n - k \quad (2.1)$$

ส่วนจำนวนข้อมูลข่าวสารและจำนวนข้อมูลที่ได้จากการเข้ารหัสนั้นเรียกว่า อัตราการเข้ารหัส หรือ Coding Rate (R) ซึ่งนิยมเขียนเป็น (n, k) หรือ k/n

2.1.5 อัตราการขยายการเข้ารหัส (Coding Gain)

การเข้ารหัสข้อมูลเพื่อแก้ไขความผิดพลาดนั้นทำให้ค่าอัตราการเกิดความผิดพลาดในการรับส่งข้อมูลลดลง ซึ่งค่าอัตรา ขยายการเข้ารหัส (Coding Gain) เป็นค่าที่แสดงถึงความสามารถในการป้องกันความผิดพลาดของข้อมูล โดยค่าอัตราขยายการเข้ารหัสเป็นค่าความแตกต่างระหว่างกำลังงานที่ใช้ในการส่งข้อมูลต่อสัญญาณรบกวนหรือ E_b/N_0 ซึ่งมีหน่วยเป็น dB ในกรณีที่มีการเข้ารหัสข้อมูลกับกรณีที่ไม่มี การเข้ารหัสข้อมูล (Uncoded) ที่ค่าของอัตราการเกิดความผิดพลาด (Error Rate) เท่ากัน ซึ่งถ้าค่า

อัตราขยายการเข้ารหัสของการเข้ารหัสใดมีค่าสูงก็แสดงว่ามีการผิดพลาดในการส่งข้อมูลน้อยเมื่อเทียบกับกรณีที่ไม่มีกรเข้ารหัสข้อมูล

2.2 สัญญาณรบกวน

ในธรรมชาตินั้นมีปริมาณทางกายภาพหลายอย่างที่มีการเปลี่ยนแปลงอยู่ตลอดเวลา โดยหา กฎเกณฑ์ที่แน่นอนไม่ได้ ในเรื่องของสัญญาณไฟฟ้าก็เช่นกัน สัญญาณบางชนิดเกิดการเปลี่ยนแปลงอยู่ตลอดเวลาตามธรรมชาติและอาจจะมารบกวนสัญญาณข่าวสารที่ต้องการให้เกิดความผิดพลาดถ้าหากเป็น สัญญาณที่เราไม่พึงปรารถนาแล้วเราจึงมั่นใจว่าเป็นสัญญาณรบกวนทั้งสิ้น โดยการรบกวนของสัญญาณจะถูกกล่าวว่าเป็นแบบ Additive หรือ Multiplication ก็ขึ้นอยู่กับลักษณะที่มันเข้าไปกระทำกับสัญญาณ ข่าวสารซึ่งการรบกวนแบบ Additive นั้นจะมีลักษณะแค่เพียงไปทับ (Superimposed) กับสัญญาณข่าวสาร ส่วนสัญญาณรบกวนแบบ Multiplication นั้น สามารถมองได้ว่าเป็นการมอดูเลตทางแอมพลิจูดของ สัญญาณข่าวสารโดยสัญญาณรบกวน

2.2.1 สัญญาณรบกวนแบบ Additive

เราสามารถแบ่งชนิดของสัญญาณรบกวนแบบ Additive ที่เครื่องรับได้ดังนี้

2.2.1.1 สัญญาณรบกวนเนื่องจากอุณหภูมิ (Thermal Noise)

โดยปกติเมื่ออะตอมของสารได้รับพลังงานความร้อน อิเล็กตรอนจะถูกทำให้เคลื่อนไหวย่าง ไร้กฎเกณฑ์ มากหรือน้อยขึ้นอยู่กับพลังงานที่ได้รับ การเคลื่อนไหวของอิเล็กตรอนทำให้เกิดกระแสหรือ ศักไฟฟ้าขึ้นในสารนั้น ปรากฏการณ์เช่นนี้ ทำให้เกิดสัญญาณรบกวนขึ้นมีชื่อเรียกว่า “ สัญญาณรบกวนเนื่องจากอุณหภูมิ ” ซึ่งเมื่ออาศัยหลักกลศาสตร์ควอนตัม (Quantum Mechanic) จะรู้ว่าค่าความหนาแน่นกำลังเชิงสเปกตรัมหรือ PSD ของสัญญาณรบกวนเนื่องจากอุณหภูมิ $S_{n_T}(\omega)$ นี้มีค่าขึ้นกับความถี่ ω และอุณหภูมิ T แต่จะไม่ขึ้นกับค่าความต้านทาน R ของสารนั้นเลย กล่าวคือ

$$S_{n_T}(\omega) = h|\omega| \left[1 + 1/\Pi \{ e^{h|\omega|/2\pi KT} - 1 \} \right] \quad (2.2)$$

โดย h คือ ค่าคงที่ของพลังค์ (Plank's constant) = 6.2×10^{-34} จูล - วินาที

ที่อุณหภูมิห้องปกติ (≈ 300 K) และคิดที่ความถี่ $f < 1000$ GHz ค่า $h|\omega|/2\pi KT$ จะมีค่า 0.024 จึง

ทำให้ประมาณได้ว่า $e^{h|\omega|/2\pi KT} \approx \frac{1 + h|\omega|}{2\pi KT}$ ดังนั้นสมการที่ (2.2) จะประมาณว่ามีค่าดังนี้คือ

$$\begin{aligned} S_{n_T}(\omega) &\approx h|\omega| \{ 1 + 2\pi KT / h|\omega| \} \\ &\approx 2KT \quad \text{W/Hz} \end{aligned}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเห็นได้ว่า PSD ของสัญญาณรบกวนเนื่องจากอุณหภูมิมีค่าประมาณคงที่ ในย่านความถี่ต่ำกว่า 1000 GHz ที่อุณหภูมิปกติ จึงถือได้ว่าสัญญาณรบกวนเนื่องจากอุณหภูมินั้นเป็นตัวอย่างของสัญญาณรบกวนขาว (White noise) ที่มีย่านความถี่จำกัด

2.2.1.2 สัญญาณรบกวนจากบรรยากาศ (Atmospheric noise)

การปลดปล่อยประจุที่เกิดจากฟ้าผ่าหรือฟ้าแลบนั่นคือ แหล่งกำเนิดสำคัญของสัญญาณรบกวนชนิดนี้และระดับของมันจะเปลี่ยนแปลงตามความถี่ของสัญญาณ ลักษณะภูมิประเทศ ฤดู เวลา และสภาพอากาศ จากการสังเกตเราพบว่าค่าของสัญญาณรบกวนชนิดนี้จะมีระดับลดลง ณ บริเวณตำแหน่งของละติจูดที่เพิ่มขึ้น

2.2.1.3 สัญญาณรบกวนจากอวกาศ (Galactic noise)

สัญญาณรบกวนทุกชนิดที่มีแหล่งกำเนิดจากนอกโลก เราถือว่าเป็นสัญญาณรบกวนอวกาศทั้งสิ้นโดยแหล่งกำเนิดที่สำคัญของสัญญาณรบกวนชนิดนี้มาจากดวงอาทิตย์ สัญญาณรบกวนชนิดนี้มีอิทธิพลต่อสัญญาณที่มีความถี่ในย่าน 15MHz — 100GHz โดยบรรยากาศไอโอโนสเฟียร์ (Ionosphere) จะแสดงหน้าที่เป็นตัวกรองความถี่สูงผ่าน และบรรยากาศ (Atmosphere) จะแสดงหน้าที่เป็นตัวกรองความถี่ต่ำผ่าน

2.2.1.4 สัญญาณรบกวนจากมนุษย์ (Man — made noise)

สัญญาณรบกวนที่เกิดจากการกระทำของมนุษย์นั้นมีสาเหตุมาจาก มอเตอร์ไฟฟ้า สายไฟ เครื่องจักร และอื่นๆ โดยจะมีในบริเวณตัวเมืองมากกว่าชนบท ซึ่งบางครั้งค่ากำลังงานของสัญญาณรบกวนในสองบริเวณนี้อาจแตกต่างกันถึง 16 dB ระดับของสัญญาณรบกวนชนิดต่างๆ

โดยค่ากำลังงานของสัญญาณที่เครื่องรับ F_a แสดงในหน่วยของเดซิเบลเหนือสัญญาณรบกวนเนื่องจากอุณหภูมิ

2.2.2 สัญญาณรบกวนแบบ Multiplicative

คลื่นวิทยุเมื่อส่งออกไปในอากาศแล้ว สัญญาณที่ส่งออกไปจะถูกมอดูเลตทางแอมพลิจูดจากสัญญาณรบกวนแบบสุ่ม ซึ่งเกิดจากการสะท้อนของคลื่นหลายทิศทางที่มารวมกันทำให้เกิดปรากฏการณ์ที่เรียกว่า “ การจางหายของสัญญาณ ” หรือเฟดดิ้ง (Fading)

ซึ่งเราสามารถแบ่งชนิดของเฟดดิ้งออกได้เป็น 2 แบบคือ Long — term fading และ Short — term fading โดย Long — term fading นั้นจะมีลักษณะของฟังก์ชันความหนาแน่นของความน่าจะเป็น (PDF) เป็นแบบปกติ (Normal distribution) ส่วน Short — term fading นั้นสามารถแยกประเภทออกเป็นอีก 2 กรณีคือ

- 1) กรณีที่พิจารณาสัญญาณรวมที่เครื่องรับที่เกิดจากการสะท้อนของคลื่นสะท้อน (Indirect path) เท่านั้น ซึ่งจะมีค่า PDF ในลักษณะของการแจกแจงแบบเรย์ลี (Rayleigh distribution)
- 2) กรณีที่พิจารณาสัญญาณรวมที่เครื่องรับที่เกิดจากทั้งคลื่นตรงและคลื่นสะท้อน ซึ่งจะมีค่า PDF ในลักษณะของการแจกแจงแบบไรเซียน (Rician distribution) ซึ่งเขียนได้ดังนี้

$$p(r) = (r/\sigma^2) \exp\left\{-\frac{(r^2 + a^2)}{2\sigma^2}\right\} I_0(ar/\sigma^2) \quad \text{เมื่อ } r \geq 0 \quad (2.3)$$

$$= 0 \quad \text{สำหรับกรณีอื่น}$$

โดย a คือ ค่าแอมพลิจูดของคลื่นตรง

I_0 คือ ค่าการแปลงฟังก์ชันเบสเซล (Bessel Function) ลำดับศูนย์

2.3 การส่งข้อมูลผ่านช่องสัญญาณ

จากที่เคยกล่าวไว้แล้วข้างต้นว่า เอนโทรปี H ของข้อมูลสามารถใช้วัดความไม่คงที่แน่นอนเบื้องต้น (Prior Uncertainty) เกี่ยวกับข้อมูลที่กำเนิดโดยแหล่งกำเนิดข้อมูลได้ ในการสื่อสารตามความจริงนั้นไม่ได้เป็นการสื่อสารแบบปราศจากความผิดพลาด (Error — Free) ดังนั้นจึงควรมีการพิจารณาถึงการใช้ฟังก์ชันที่เหมาะสมในการวัดค่าความไม่แน่นอนภายหลัง (Posteriori Uncertainty) ของข้อมูลหลังจากการรับข้อมูลผ่านช่องสื่อสารที่มีสัญญาณรบกวน (Noisy Channel)

โดยเราพิจารณาช่องสื่อสารดิจิทัลแบบไร้ความจำ (Discrete Memoryless Channel) ซึ่งกำหนดคุณลักษณะโดยกลุ่มของอินพุต $x_1, x_2, \dots, x_m$ มีค่า Priori Probability $\{P(x_i)\}$ และกลุ่มของเอาต์พุต $y_1, y_2, \dots, y_Q$ และกลุ่มของค่า Transition Probability $\{P(y_j|x_i)\}$ หรือความน่าจะเป็นของสัญญาณทางด้านรับ y_j เมื่อกำหนดสัญลักษณ์ x_i ที่ถูกส่งโดยรูปแบบของช่องสัญญาณนี้

จากการใช้กฎเกณฑ์ของเบสส์สำหรับตัวแปรสุ่มเราสามารถเขียนค่า Posteriori Probability ของ x_i เมื่อกำหนดค่า y_j ที่ด้านรับ ได้ดังนี้

$$P(y = x_i | y_j) = \frac{P(x_i, y_j)}{P(y_j)}$$

$$= \frac{P(y_j | x_i) P(x_i)}{P(y_j)} \quad (2.4)$$

จากที่ทราบว่าค่า Priori Uncertainty ของ x_i สามารถวัดได้จาก $-\log P(x_i)$ ดังนั้นเรากำหนดค่า Posteriori Uncertainty ของ x_i โดยค่าทางด้านรับเท่ากับ y_j เป็น $-\log P(x_i | y_j)$ โดยความแตกต่างของทั้งสองค่านี้ถูกเรียกว่า “Mutaul Information”

$$I(X; Y) = \log[P(y_j | x_i) / P(y_j)] \quad (2.5)$$

ตอนนี้เรากำหนดค่าอัตราทั้งหมดของการส่งข้อมูลผ่านช่องสัญญาณโดยการเฉลี่ยค่า $I(x_i; y_j)$

ซึ่งเรียกว่า “Average Mutaul Information” กำหนดโดย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$I(X;Y) = \sum_{i=1}^M \sum_{j=1}^Q P(x_i, y_j) \log \{P(y_j|x_i)/P(y_j)\} \quad (2.6)$$

ซึ่งมีหน่วยเป็น Bits/Symbol เมื่อให้ค่าฐานของลอการิทึมเป็น 2 เพราะค่าความน่าจะเป็นของสัญลักษณ์เอาต์พุต $P(y^*)$ คือ

$$I(X;Y) = \sum_{i=1}^M P(x_i)P(y_j|x_i) \quad (2.7)$$

จะเห็นว่า $I(x; y)$ นั้นสามารถกำหนดโดยตรงจากค่าความน่าจะเป็นของการเกิด $P(x_i)$ และกลุ่มของค่า Transition Probability $P(y_j|x_i)$ ซึ่งค่า Average Mutual Information นี้คืออัตราเฉลี่ยของการส่งข้อมูลผ่านช่องสื่อสาร โดยกำหนดการกระจายของ Source Symbols และ Channel Transition Probabilities ถ้าเราให้ค่า Transition Probabilities เป็นการกำหนดช่องสัญญาณอัตราการส่งข้อมูลผ่านช่องสัญญาณจะขึ้นอยู่กับค่าความน่าจะเป็นของการเกิด Input Symbols ความจุ (Capacity) ของช่องสัญญาณถูกกำหนดที่ค่าสูงสุดของ $I(x; y)$ นั่นคือ

$$C = \max_{p(x_i)} I(X;Y) \quad (2.8)$$

2.3.1 ช่องสัญญาณแบบ AWGN (The Addition White Gaussian Noise Channel)

ชนิดของช่องสัญญาณ DMC ที่มีความสำคัญในทางปฏิบัติ คือ ช่องสัญญาณที่ให้ค่าสัญญาณเอาต์พุตที่เกิดจากสัญญาณอินพุตรวมกับสัญญาณรบกวนเกาส์แบบแถบกว้าง (Broadband Gaussian Noise) ซึ่งโดยทั่วไปเราจะเรียกว่า “White Gaussian Noise” โดยเป็นสัญญาณที่มีค่าเฉลี่ยเป็นศูนย์และมีการแจกแจงของค่ากำลังเท่ากันตลอดย่านความถี่ $-\infty \leq f \leq \infty$ ด้วยระดับ $N_0/2$ W/Hz หรือแบนแกนความถี่ข้างเดียวเท่ากับ N_0 W/Hz

AWGN นั้นสามารถเขียนอธิบายในรูปแบบอย่างง่ายในเทอมของสัญญาณอินพุต x และสัญญาณเอาต์พุต y ดังนี้

$$y = x + n_G \quad (2.9)$$

โดย n_G คือค่าตัวแปรสุ่มแบบเกาส์ที่มีค่าเฉลี่ยเป็นศูนย์และค่าความแปรปรวน σ^2

แบบจำลองโครงสร้างช่องสื่อสารแบบ AWGN นี้เป็นแบบจำลองที่ถูกต้องที่ใช้ในระบบสื่อสารต่างๆเช่น ดาวเทียมและการสื่อสารในอวกาศโดยมีค่าอิทธิพลเด่นในการจำกัดประสิทธิภาพของการสื่อสารก็คือ Thermal Noise หรือ Galactic Noise

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.2 ความจุของช่องสัญญาณแบบ Continuous AWGN

จากหัวข้อที่รู้จักกันดี Sampling Theory ซึ่งมีอัตราไนควิสต์ $2W$ โดย W คือแบนวิทซ์ของช่องสัญญาณ ถ้ากำหนดให้คลื่นที่ส่งมีรูปแบบคลื่นเป็น $x(t)$ และ Additive White Gaussian Noise เป็น $n(t)$ ดังนั้นสัญญาณที่เครื่องรับ คือ $y(t) = x(t) + n(t)$ และสามารถกำหนดค่ากำลังงานเฉลี่ยของสัญญาณที่รับได้หรือตัวอย่างสัญญาณรบกวนเป็น $\overline{x^2} = s$ และ $\overline{n^2} = N$ ตามลำดับ

หากเราสมมุติว่าค่ากำลังงานเฉลี่ยเหล่านี้ถูกทำให้คงที่ เราจะทำการคิดแปลง โดยแทนค่า $P(x_i)$ และ $P(y_j)$ ด้วย PDF $p(x)$ และ $p(y)$ ตามลำดับ และแทน $P(y_j|x_i)$ ด้วย $p(x|y)$ โดยค่า $p(x|y)$ นั้นจะขึ้นอยู่กับ y และ x โดยค่าผลต่าง $y - x$ และสำหรับกรณีของ Gaussian Noise เราจะได้ว่า

$$p(y|x) = \frac{1}{(\sqrt{2\pi N}) \exp[-(y-x)^2/2N]} \quad (2.10)$$

โดยค่า N คือค่ากำลังเฉลี่ยหรือความแปรปรวนของตัวอย่างสัญญาณรบกวน

สัญญาณรบกวนแบบบวก (Addition) เป็นคุณสมบัติของช่องสัญญาณต่อเนื่อง (Continuous Channel) ซึ่งเกี่ยวข้องกับสอดคล้องโดยตรงกับคุณสมบัติของอินพุตที่เป็นยูนิฟอร์ม ซึ่งถูกกำหนดให้

$$C = \max_{p(x_i)} [H(Y)] - H(Y|X) \quad (2.11)$$

โดยค่าเอนโทรปีเอนต์ฟุต $H(y)$ คิดจากลำดับของตัวอย่างสัญญาณที่รับได้ $y(t)$ และค่าของเอนโทรปีอย่างมีเงื่อนไข $H(Y|X)$ ขึ้นอยู่กับการกระจายทางสถิติของตัวอย่างสัญญาณรบกวนเท่านั้น นั่นคือ ตัวอย่าง (Sample) ของ $n(t)$ ดังนั้นเราสามารถเขียนสมการที่ (2.11) ใหม่ได้ดังนี้

$$C = \max_{p(x_i)} [H(Y)] - H(n) \quad (2.12)$$

ดังนั้นตอนนี้เราสามารถกำหนดความจุของ Continuous — Waveform AWGN Channel ได้เท่ากับค่าสูงสุดของเอนโทรปีของสัญญาณที่รับได้ลบกับค่าเอนโทรปีของสัญญาณรบกวนค่าสูงสุดสามารถทำได้โดยเลือกรูปแบบคลื่นที่ส่ง $x(t)$ ให้เหมาะสม

เริ่มแรกเราพิจารณา $H(n)$ จากตัวอย่างสมการที่ (2.13) ซึ่งค่าเอนโทรปีของตัวอย่างจากการกระจายแบบเกาส์ คือ $(1/2) \log(2\pi e \sigma^2)$ โดย σ^2 คือค่าความแปรปรวน (variance) ของการกระจาย ดังนั้นเรากำหนดให้เอนโทรปีของตัวอย่างของสัญญาณรบกวนแบบเกาส์คือ $(1/2) \log(2\pi e N)$ ซึ่ง N

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คือ ค่าความแปรปรวน หรือค่ากำลังงานเฉลี่ยของตัวอย่างสัญญาณรบกวนเอนโทรปีของลำดับของ Noise Samples ที่ $2WT$ ในระหว่างช่วงเวลาเท่ากับ T วินาที คือ

$$H(n) = WT \log(2\pi eN) \quad (2.13)$$

ตอนนี้เราพิจารณา $H(y)$ จากสมการที่ (2.14) เรากำหนดให้ตัวแปรสุ่มมีค่าความแปรปรวนที่คงที่และค่าเอนโทรปีของตัวแปรสุ่มจะมีค่ามากที่สุดเมื่อให้ PDF เป็นฟังก์ชันแบบเกาส์ ดังนั้น ถ้าเรากำหนดให้ค่าสัญญาณ $y(t)$ มีค่ากำลังงานเฉลี่ย $S + N$ จะได้ว่าเอนโทรปีของสัญญาณที่รับได้ $H(y)$ จะมีค่ามากที่สุดเมื่อให้ $y(t)$ มีรูปแบบทางสถิติของ Gaussian Noise ซึ่งแต่ละ Sample มีค่าความแปรปรวน $S + N$ จะได้ลำดับของ $2WT$ Samples ดังนี้

$$\max_{p(x_i)} H(Y) = WT \log[2\pi e(S + N)] \quad (2.14)$$

ดังนั้นค่าสูงสุดของอัตราการส่งข้อมูล คือ

$$I_{\max} = WT \log\left\{\frac{(S + N)}{N}\right\} \quad (2.15)$$

ซึ่งถูกวัดในหน่วย Bits Transmitted Per T Seconds นั่นคือ

$$C = W \log(S + N/N) = W \log(1 + S/N) \quad \text{บิต/วินาที} \quad (2.16)$$

นี่คือสูตรความจุของ Shannon สำหรับ Band — Limited Continuous AWGN Channel

2.4 ทฤษฎีของการเข้ารหัสช่องสัญญาณ

ทฤษฎีของการเข้ารหัสช่องสัญญาณนั้นกล่าวว่าทุกช่องสื่อสาร C และที่ค่าอัตราการส่งข้อมูลใดๆ $R < C$ แล้วจะมีรหัสความยาวบล็อก (Block Length) n และอัตรา R ซึ่งมีความน่าจะเป็นของการถอดรหัสผิดพลาด $P(E)$ ถูกกำหนดโดย

$$P(E) \leq 2^{-nE_b(R)} \quad (2.17)$$

โดยค่าของ $E_b(R)$ คือค่าบวกของฟังก์ชัน R สำหรับ $R < C$ และเป็นการพิจารณาโดยคุณลักษณะของช่องสัญญาณเท่านั้น สำหรับการนำสมการที่ 2.17 ไปใช้นั้น คือ ที่อัตราการส่งข้อมูลใดๆ น้อยกว่า C แล้ว ค่าความน่าจะเป็นของความผิดพลาดสามารถทำให้น้อยลงตามต้องการได้โดยการเพิ่ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความยาวรหัส Block Length n ในขณะที่ยังคงรักษาอัตราเข้ารหัส (Code Rate) ให้คงที่ โดยเงื่อนไขดังกล่าวสามารถเขียนได้เช่นเดียวกันสำหรับการเข้ารหัสการประสาน (Convolutional Codes) ซึ่ง n ถูกแทนด้วย k (Code Constraint Length)

2.5 รหัสแบบบล็อก (Block Code)

รหัสแบบบล็อก (Block Code) เป็นหนึ่งในรหัสที่ใช้ป้องกันความผิดพลาดล่วงหน้า (FEC) ซึ่งในการเข้ารหัสแบบบล็อกนั้น ข้อมูลข่าวสารจะถูกแบ่งออกเป็นชุดหรือบล็อกย่อยๆ ก่อนการเข้ารหัสซึ่งรหัสแบบบล็อกสามารถตรวจสอบและบอกตำแหน่งข้อมูลที่ผิดพลาดได้ ซึ่งขึ้นอยู่กับเงื่อนไขของการเข้ารหัส โดยรหัสแบบบล็อกนี้มีอยู่หลายรูปแบบ เช่น รหัสแบบลิเนียร์บล็อก (Linear Block Code) รหัสแบบแฮมมิงรหัสแบบ Cyclic และรหัสแบบ BCH เป็นต้น

2.5.1 การเข้ารหัสแบบลิเนียร์บล็อก (Linear Block Code)

การเข้ารหัสแบบลิเนียร์บล็อกนั้นเป็นพื้นฐานที่สำคัญของรหัสแบบบล็อก รหัสแบบลิเนียร์บล็อกเป็นการตรวจเช็คพาริตีชนิดหนึ่ง โดยการเข้ารหัสแบบลิเนียร์บล็อกนั้นเป็นการนำข้อมูลข่าวสารที่ต้องการเข้ารหัสมาแบ่งเป็นชุดๆ หรือบล็อกย่อยๆ ซึ่งแต่ละชุดนั้นมีขนาดเท่ากับ k บิต แล้วนำมาหาค่ารหัสหรือ Codeword ขนาด n บิต ซึ่งโดยทั่วไปแทนด้วยสัญลักษณ์ (n,k) การเข้ารหัสจะใช้ Generator Matrix ในการคำนวณดังนี้

$$U = mG \quad (2.18)$$

โดย U คือ ข้อมูลที่ได้หลังจากการเข้ารหัส m คือ ชุดของข้อมูลจำนวน k บิต ซึ่งนำมาเข้ารหัส และ G คือ Generator Matrix ของการเข้ารหัส

2.5.2 การถอดรหัสแบบลิเนียร์บล็อก

การตรวจจับความผิดพลาดของข้อมูลที่ถูกส่งไปของรหัสแบบบล็อกนั้น วิธีการคือการนำข้อมูลที่รับได้มาทำการคูณกับเมตริกที่ใช้สำหรับตรวจจับความผิดพลาดของข้อมูล (H) ซึ่งผลลัพธ์ที่ได้จากการคูณนั้นเรียกว่าค่า Syndrome (S) ซึ่งเป็นค่าแสดงถึงลักษณะความผิดพลาดของข้อมูลที่เกิดขึ้นกับข้อมูลชุดนั้นๆ ค่า Syndrome ทำให้สามารถนำไปใช้แก้ไขความผิดพลาดของข้อมูลให้ถูกต้องได้ โดยลักษณะการถอดรหัสแบบลิเนียร์บล็อกมีการทำงานดังสมการ

$$S = (U + e)H^T \quad (2.19)$$

โดย e คือค่าความผิดพลาดที่เกิดขึ้น และ H จะมีความสัมพันธ์กับ G ซึ่งทำให้ UH^T มีค่าเป็นศูนย์ ดังนั้นค่า Syndrome ที่ได้นั้นมีค่าเท่ากับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$S = eH^T \quad (2.20)$$

ค่า Syndrome ที่ได้จากการคำนวณนั้นใช้ในการตรวจสอบว่ามีการผิดพลาดเกิดขึ้นที่บิตใดทำให้สามารถทำการแก้ไขข้อมูลที่เกิดความผิดพลาดให้ถูกต้องได้แต่รหัสแบบลิเนียร์บล็อกนั้นมิข้อจำกัดในการตรวจสอบหาตำแหน่งผิดพลาดถ้าความผิดพลาดเกิดขึ้นนั้นเกินขอบเขตของรหัส

2.5.3 ความสามารถในการตรวจจับและแก้ไขความผิดพลาด

ความสามารถในการตรวจจับและแก้ไขความผิดพลาด เป็นตัวบอกระสิทธิภาพของรหัสการวิเคราะห์คุณสมบัติต่างๆ ของการเข้ารหัสต้องมีการพิจารณาถึงองค์ประกอบหลายส่วน ได้แก่ น้ำหนักแฮมมิง (Hamming Weight) และระยะห่างแฮมมิง (Hamming Distance) สำหรับค่าน้ำหนักแฮมมิงหรือ $w(U)$ นั้นหมายถึงจำนวนบิตของข้อมูลที่ไม่เท่ากับ “0” ซึ่งในกรณีของข้อมูลไบนารีหมายถึงจำนวนข้อมูลที่เป็น “1” สำหรับค่าระยะห่างแฮมมิงระหว่างข้อมูล U และ V หรือ $d(U, V)$ นั้นหมายถึงจำนวนบิตของข้อมูลทั้งสองที่มีค่าแตกต่างกันซึ่งทั้งสองค่านี้มีความสัมพันธ์กันดังสมการ

$$D(U, V) = w(U+V) \quad (2.21)$$

ข้อมูลที่ผ่านมาการเข้ารหัสนั้นค่าระยะห่างแฮมมิงที่อาจเท่ากันหรือแตกต่างกันก็ได้ แต่ในการวิเคราะห์การทำงานจะพิจารณาเพียงข้อมูลที่มีค่าระยะห่างแฮมมิงที่มีค่าน้อยที่สุด ซึ่งเรียกว่า ค่าระยะห่างแฮมมิงต่ำสุด (Minimum Hamming Distance) หรือ $d_{\min}$ โดย $d_{\min}$ จะถูกนำไปใช้ในการคำนวณความสามารถในการแก้ไขความผิดพลาดของข้อมูล สำหรับการเข้ารหัสแบบบล็อกนั้นความสัมพันธ์ระหว่างค่า $d_{\min}$ และจำนวนบิตที่มากที่สุดของข้อมูลที่เกิดความผิดพลาดที่ยังสามารถแก้ไขให้ถูกต้องได้ (t) แสดงดังสมการ

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor \quad (2.22)$$

ส่วนค่าของจำนวนบิตที่มากที่สุดของข้อมูลที่ยังสามารถตรวจจับได้ว่าเกิดความผิดพลาดขึ้น (Error Detecting Capability) นั้นแสดงดังสมการ

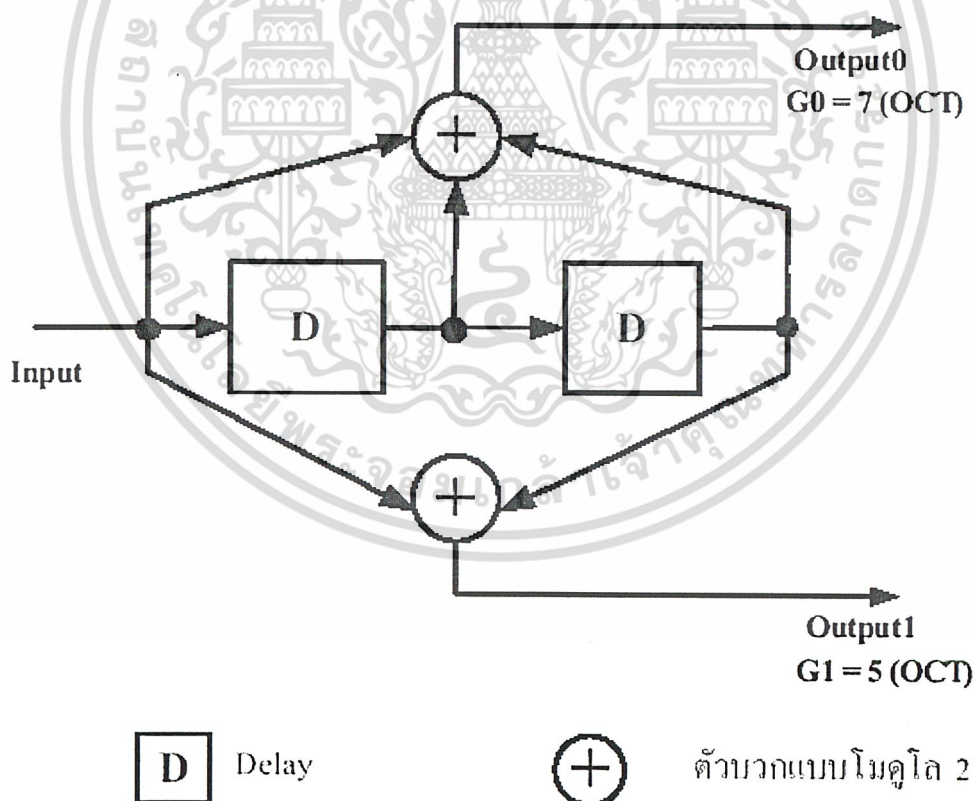
$$e = d_{\min} - 1 \quad (2.23)$$

2.6 รหัสคอนโวลูชัน (Convolution Code)

รหัสโวลูชัน (Convolution Code) ถูกนำมาเผยแพร่ในปี 1955 โดย Elias การเข้ารหัสคอนโวลูชันเป็นการหาความสัมพันธ์ของกลุ่มข้อมูลอินพุตที่เรียงลำดับอย่างต่อเนื่องโดยวงจรเข้ารหัสคอนโวลูชันประกอบด้วยตัวดีเลย์ (D) หรือชิฟรีจิสเตอร์ (Shift Register) และตัวบวกแบบมอดูโล 2 (Modulo-Adder) หรือ Exclusive OR Gate

2.6.1 การเข้ารหัสคอนโวลูชัน

การทำงานของวงจรเข้ารหัสคอนโวลูชันนั้น ข้อมูลอินพุต (k) จะถูกส่งเข้าอินพุตของวงจรเข้ารหัสคอนโวลูชันแบบต่อเนื่อง โดยข้อมูลเอาต์พุตของการเข้ารหัส (n) เกิดจากการนำข้อมูลอินพุต (k) และข้อมูลในชิฟรีจิสเตอร์มาบวกแบบมอดูโล 2 ซึ่งขึ้นอยู่กับค่า Constraint Length (K) ซึ่งเป็นค่าที่แสดงจำนวนของข้อมูลอินพุต (k) และข้อมูลสูงสุดในชิฟรีจิสเตอร์ที่นำมาบวกแบบมอดูโล 2 ในการหาค่าเอาต์พุต และค่า Generator Polynomial (g) ซึ่งเป็นค่าที่ใช้แสดงตำแหน่งข้อมูลอินพุตและข้อมูลในชิฟรีจิสเตอร์ที่นำมาบวกแบบมอดูโล 2 ค่า Generator Polynomial นี้เขียนแบบในรูปเลขฐานแปด เช่น ตำแหน่งข้อมูลที่นำมาบวกแบบมอดูโล 2 เป็น “111” นิยมเขียนเป็น 7_8 ดังตัวอย่างวงจรเข้ารหัสคอนโวลูชันแสดงในรูปที่ 2.2



รูปที่ 2.2 วงจรเข้ารหัสคอนโวลูชันที่ค่า R เท่ากับ $1/2$ และค่า K เท่ากับ 3

ส่วนค่าอัตราเข้ารหัส (R) เป็นอัตราส่วนของจำนวนข้อมูลอินพุต จากวงจรตัวอย่างการเข้ารหัสคอนโวลูชันในรูปที่ 2.10 มีค่าอัตราส่วนของการเข้ารหัสเท่ากับ $1/2$ ค่า Constraint Length เท่ากับ 3 และมีค่า Generator Polynomial G_0 เท่ากับ 7_8 และ G_1 เท่ากับ 5_8

2.6.2 การวิเคราะห์การทำงานของวงจรเข้ารหัสคอนโวลูชัน

การวิเคราะห์การทำงานของวงจรเข้ารหัสคอนโวลูชันนั้นใช้ค่า Generation Polynomial ในการคำนวณหาผลลัพธ์ของการเข้ารหัสของเอาต์พุตแต่ละค่า กรณีของวงจรเข้ารหัสในรูปที่ 2.2 นั้นค่า Generator Polynomial ในรูปแบบเลขฐานสองแสดงคือ

$$g_0 = [1\ 0\ 0]$$

$$g_1 = [1\ 1\ 1]$$

การคำนวณหาเอาต์พุตที่ได้จากการเข้ารหัสนั้นนำค่า Generator Polynomial มาคอนโวลูชัน (Convolution) กับข้อมูลอินพุตดังสมการ

$$Output_i = [Input] * g_i \quad (2.24)$$

ตัวอย่างในกรณีวงจรเข้ารหัสในรูปที่ 2.2 ข้อมูลที่เข้ารหัสมีค่าเท่ากับ “1 0 0 1” การคำนวณหาเอาต์พุตที่ได้จากการเข้ารหัสตามคำนวณได้ดังนี้

$$Output_0 = [1\ 0\ 0\ 1] * [1\ 0\ 0] = 1\ 0\ 0\ 1\ 0\ 0$$

$$Output_1 = [1\ 0\ 0\ 1] * [1\ 1\ 1] = 1\ 1\ 1\ 1\ 1\ 1$$

ค่าเอาต์พุตที่ได้จากการเข้ารหัสข้อมูลนั้นเกิดจากข้อมูลที่ได้จาก O/P_0 และ O/P_1 ซึ่งเมื่อนำมารวมกันมีค่าเท่ากับ “1 1 0 1 1 1 0 1 0 1”

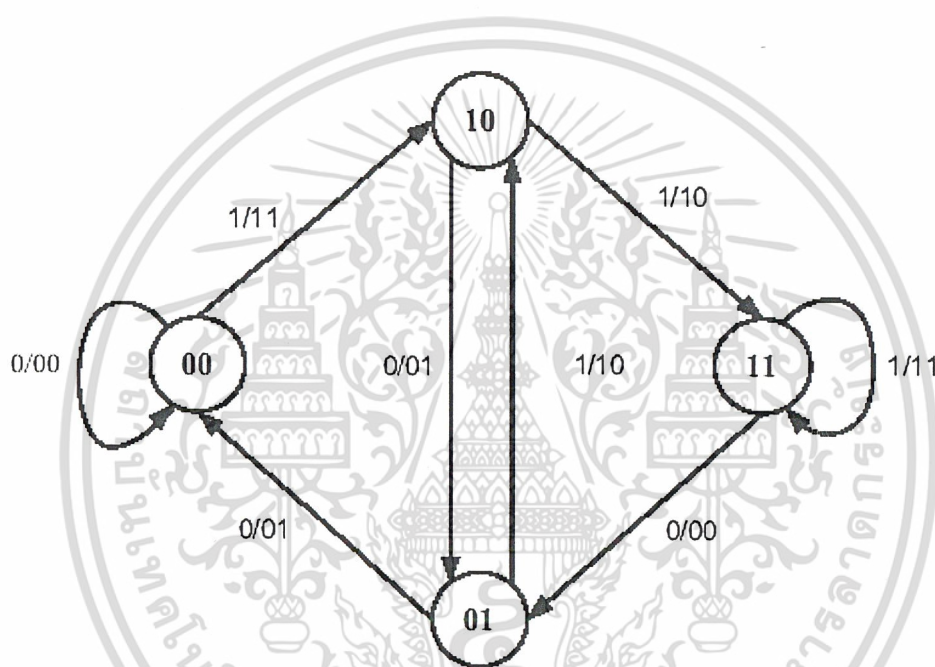
2.6.3 แผนภาพแสดงการทำงานของวงจรเข้ารหัสคอนโวลูชัน

การวิเคราะห์การทำงานของวงจรเข้ารหัสคอนโวลูชัน นอกจากการใช้สมการ Generator Polynomial ในการวิเคราะห์การทำงานแล้วยังสามารถใช้แผนภาพ (Diagram) วิเคราะห์การทำงานได้ โดยแผนภาพแสดงความสัมพันธ์ระหว่างข้อมูลที่ถูกเก็บไว้ในวงจรเข้ารหัส ข้อมูลอินพุตที่ถูกป้อนเข้ามา ณ เวลานั้น และค่าของเอาต์พุตที่ถูกส่งออกไป ซึ่งรูปแบบแผนภาพที่ใช้แสดงการทำงานของวงจรเข้ารหัสนั้นมีสามรูปแบบ คือ แผนภาพสเตต (State Diagram) แผนภาพทรี (Tree Diagram) และแผนภาพ (Trellis Diagram) ซึ่งมีรายละเอียดดังต่อไปนี้

2.6.3.1 แผนภาพสเตต (State Diagram)

แผนภาพสเตตเป็นแผนภาพแสดงค่าของข้อมูลในชิฟเรจิสเตอร์และเอาต์พุตของการเข้ารหัสคอนโวลูชันซึ่งเปลี่ยนตามข้อมูลอินพุตที่ป้อนเข้ามา ตัวอย่างของแผนภาพสเตตของวงจรเข้ารหัสในรูปที่ 2.10 เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

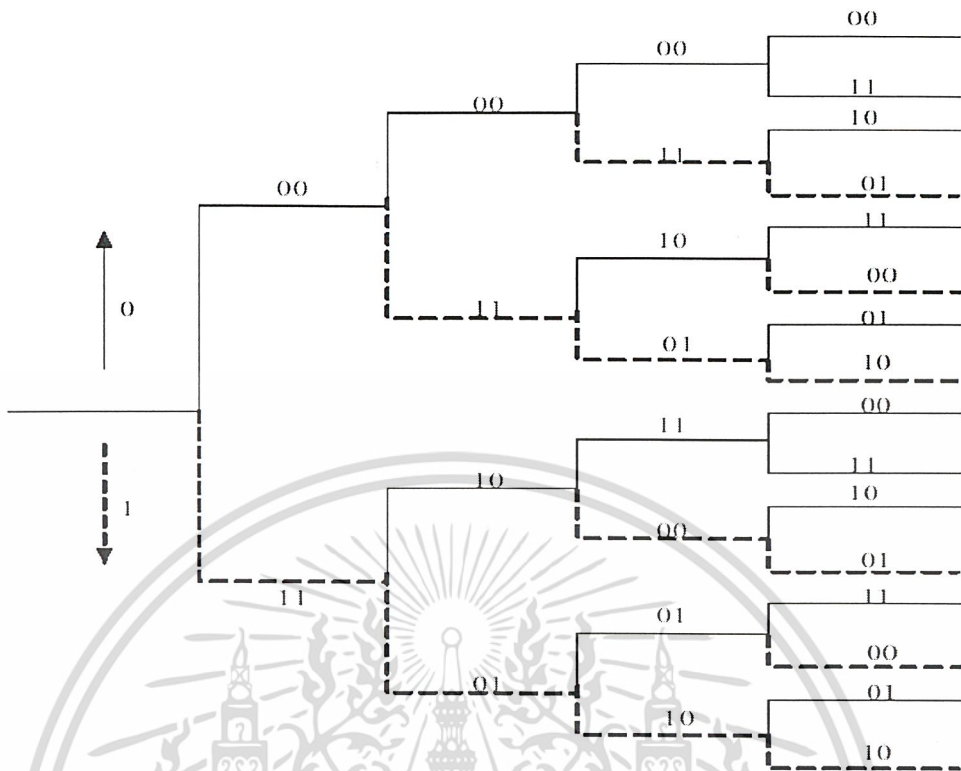
แสดงดังรูปที่ 2.11 โดยตัวเลขที่อยู่ในวงกลมแต่ละวงนั้นหมายถึงสเตต (State) หรือสถานะต่างๆ ของข้อมูลที่ถูกเก็บไว้ในรีจิสเตอร์ โดยค่าสเตตเท่ากับ 2^{k-1} สเตต ซึ่งวงจรเข้ารหัสตัวอย่างในรูปที่ 2.10 มีจำนวนสเตตทั้งหมด 4 สเตตสำหรับลูกศรที่ถูกแสดงไว้ในรูปนั้นแสดงถึงลักษณะการเปลี่ยนแปลงการทำงานจากสถานะหนึ่งไปเป็นอีกสถานะหนึ่ง ซึ่งขึ้นอยู่กับข้อมูลที่ป้อนเข้ามา ณ เวลานั้นๆ เช่น กรณีที่ข้อมูลที่เก็บไว้มีค่าเป็น “00” เมื่อมีข้อมูล “1” ป้อนเข้ามาจะมีผลทำให้ข้อมูลที่เก็บไว้ถูกเปลี่ยนไปเป็น “10” และ “11” คือ เอาต์พุตของการเข้ารหัสซึ่งแทนด้วยสัญลักษณ์ “1/11” แต่ถ้าหากว่ามี การป้อนข้อมูล “0” เข้ามา วงจรยังคงมีสถานะเป็น “00” เหมือนเดิม และ “00” คือ เอาต์พุตของการเข้ารหัสซึ่งแทนด้วยสัญลักษณ์ “0/00” ซึ่งลักษณะแผนภาพสเตตแสดงดังรูปที่ 2.3



รูปที่ 2.3 แผนภาพสเตตที่ R เท่ากับ 1/2 และ K เท่ากับ 3

2.6.3.2 แผนภาพทรี (Tree Diagram)

แผนภาพทรี (Tree Diagram) เป็นแผนภาพที่แสดงค่าเอาต์พุตของการเข้ารหัสที่ได้จากการป้อนข้อมูลอินพุต โดยแผนภาพทรีสามารถแสดงการเปลี่ยนแปลงตามเวลาได้ การพิจารณาการทำงานนั้นเริ่มจากตำแหน่งราก (Root) ของแผนภาพทรีซึ่งอยู่ในตำแหน่งซ้ายมือสุด โดยข้อมูลที่ป้อนเข้าวงจรเข้ารหัสเป็นตัวกำหนดทิศทางของข้อมูลในแผนภาพทรี โดยให้เลื่อนตำแหน่งไปข้างบนเมื่อมีการรับข้อมูล “0” และเลื่อนตำแหน่งลงล่างเมื่อมีการรับข้อมูล “1” ซึ่งหลังจากที่เลื่อนตำแหน่งแล้ว แผนภาพจะแสดงข้อมูลเอาต์พุตของข้อมูลที่ถูกเข้ารหัส ณ เวลานั้นๆ โดยข้อมูลนี้อยู่เหนือเส้นทาง ณ ตำแหน่งนั้นๆ ตัวอย่างการทำงานของแผนภาพทรีวงจรเข้ารหัสคอนโวลูชันในรูปที่ 2.3 แสดงในรูปที่ 2.4

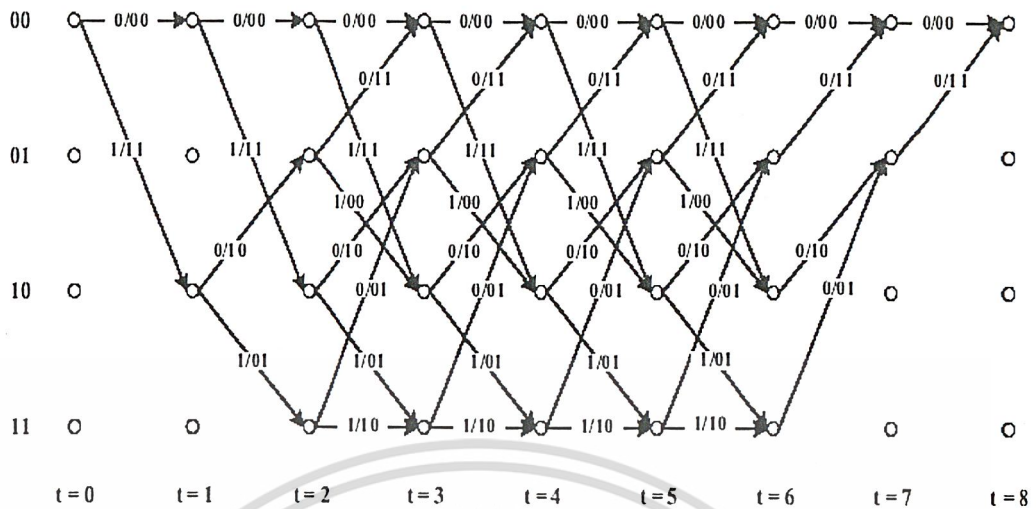


รูปที่ 2.4 แผนภาพทรีที่ R เท่ากับ 1/2 และ K เท่ากับ 3

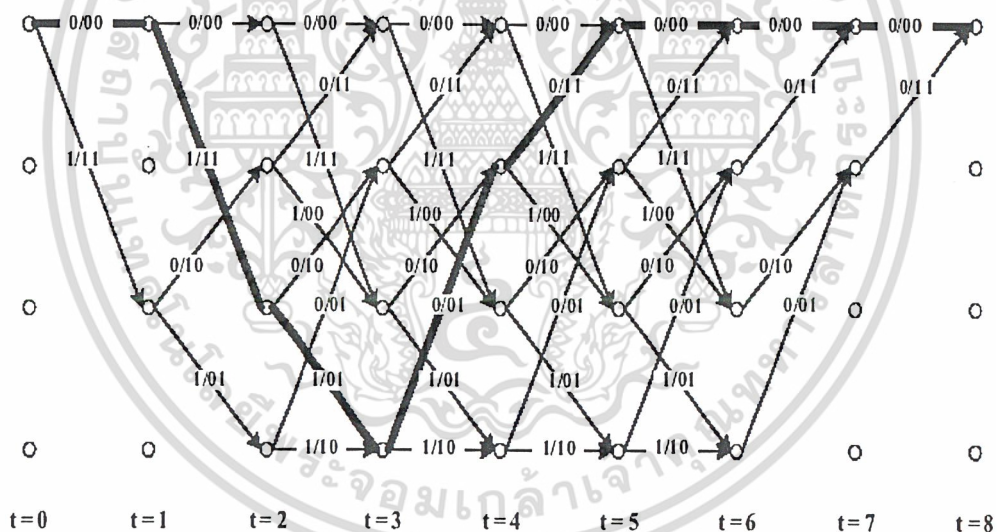
2.6.3.3 แผนภาพ Trellis (Trellis Diagram)

เมื่อมีความยาวของข้อมูลอินพุตมีความยาวมากๆ ทำให้เกิดปัญหาขึ้นกับ แผนภาพทรี จากรูปที่ 2.4 ที่เวลา t_4 แผนภาพทรีจะซ้ำแบบเดิม ซึ่งครั้งบนและครั้งล่างของแผนภาพทรีนั้นมีค่าเหมือนกัน ดังนั้นสามารถนำแผนภาพทรีนี้มาเขียนแผนภาพใหม่ซึ่งเรียกว่า แผนภาพ Trellis (Trellis Diagram) เพื่อใช้ในการวิเคราะห์การเข้ารหัสและถอดรหัสคอนไวลูชัน โดยแผนภาพ Trellis สามารถแสดงสถานะของข้อมูลหรือสเตต (State) ของการทำงานมาแสดง ณ เวลาต่างๆ ได้ ซึ่งตัวอย่างแผนภาพ Trellis ของวงจรเข้ารหัส ในรูปที่ 2.5 แสดงดังรูปที่ 2.6 โดยแนวตั้งเป็นค่าสเตตส่วนทางแนวนอนเป็นค่าเวลา

61735



รูปที่ 2.5 แผนภาพ Trellis ที่ R เท่ากับ $1/2$ และ K เท่ากับ 3



รูปที่ 2.6 ตัวอย่างการใช้งานแผนภาพ Trellis

ส่วนเส้นทางต่างๆที่อยู่ในแผนภาพ Trellis นั้นแสดงถึงลักษณะการเปลี่ยนแปลงสถานะของวงจร โดยตัวเลขที่อยู่เหนือทางเดินในแต่ละเส้นทางนั้นแสดงถึงข้อมูลที่ป้อนเข้ามา k และข้อมูลที่ส่งออกไป n ณ เวลานั้นๆ โดยค่าสเตทของแผนภาพนั้นเท่ากับ 2^{k-1} และค่าสาขา (Branch) ที่เข้ามาที่จุดเชื่อมต่อ (Node) เท่ากับ 2^k ในแต่ละเวลาลักษณะของแผนภาพ Trellis จะมีรูปแบบที่คล้ายกัน แต่แตกต่างกันเฉพาะส่วนหัวและท้ายเท่านั้น ซึ่งเป็นผลมาจากข้อมูลที่เก็บอยู่ในวงจรนั้นมีค่าเริ่มต้นจากสถานะที่มีข้อมูล “0” และจบลงที่สถานะข้อมูล “0” เช่นกัน ซึ่งขนาดของความยาวในแผนภาพ Trellis ขึ้นอยู่กับข้อมูลที่เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถูกนำมาเข้ารหัสตัวอย่างเช่นกรณีที่มีการป้อนข้อมูลอินพุต “01100000” เข้ามาภายในวงจรเข้ารหัส เมื่อใช้แผนภาพ Trellis ข้อมูลเอาต์พุตมีค่าเท่ากับ “0011100001000000” ซึ่งแสดงได้ดังรูปที่ 2.6 โดยข้อมูลที่ได้อีกหลังจากการเข้ารหัสแสดงในเส้นที่

2.6.4 การถอดรหัสคอนโวลูชัน

การถอดรหัสคอนโวลูชันนั้นมีขั้นตอนที่ซับซ้อนกว่าวิธีเข้ารหัสมาก ซึ่งวิธีที่ใช้ในการถอดรหัสคอนโวลูชันนั้นมี 3 วิธี คือ Sequential Decoding, Threshold Decoding และ Viterbi Decoding ซึ่งมีรายละเอียดดังนี้

- วิธี Sequential Decoding เป็นวิธีการถอดรหัสคอนโวลูชันแบบแรกที่น่าสนใจโดย Wozencraft ในปี 1961 วิธีการถอดรหัสใช้แผนภาพทรีมาใช้ในการงานในการค้นหาเส้นทางของข้อมูล วิธีนี้มีข้อเสียคือใช้เวลาในการถอดรหัสนานและมีการทำงานที่ซับซ้อน
- วิธี Threshold Decoding เป็นวิธีที่น่าสนใจโดย Massey ในปี 1963 การถอดรหัสด้วยวิธีนี้ใช้หน่วยตัดสินใจที่เรียกว่า Threshold Gate ในการพิจารณาตัดสินใจแก้ไขความผิดพลาด ซึ่งวิธีนี้มีความซับซ้อนน้อยแต่มีประสิทธิภาพถอดรหัสที่ต่ำ
- วิธี Viterbi Decoding เป็นวิธีที่น่าสนใจโดย A.J. Viterbi ในปี 1967 การถอดรหัสคอนโวลูชันด้วย Viterbi Decoding นิยมใช้กันมากเนื่องจากให้ประสิทธิภาพสูง แต่วิธีนี้มีความซับซ้อนในการทำงานสูงและใช้หน่วยความจำขนาดใหญ่ในการถอดรหัสวิธีนี้จะใช้อัลกอริทึมที่เรียกว่า Maximum Likelihood Decoding และแผนภาพ Trellis ในการถอดรหัส

2.7 การถอดรหัสด้วยวิธี Viterbi

การถอดรหัสคอนโวลูชันด้วยวิธี Viterbi เป็นวิธีการถอดรหัสที่ให้ประสิทธิภาพสูง ซึ่งลักษณะการทำงานของวิธี Viterbi เป็นแบบ Maximum Likelihood Decoding โดยผลลัพธ์ที่ได้จากการถอดรหัสจะเป็นเส้นทางเพียงเส้นทางเดียวที่มีความน่าจะเป็นสูงสุดจากเส้นทางทั้งหมดในแผนภาพ Trellis ซึ่งมีลักษณะคล้ายกับข้อมูลที่ถูกลบมากที่สุด โดยสำหรับกระบวนการที่จะใช้ในการค้นหาเส้นทางนั้นเรียกว่า “Viterbi Algorithm Decoding”

2.7.1 รูปแบบการทำงานของวงจรถอดรหัสด้วยวิธี Viterbi

รูปแบบการทำงานของวงจรถอดรหัสด้วยวิธี Viterbi มีสองแบบ ได้แก่ Hard Decision และ Soft Decision ซึ่งอธิบายได้ดังนี้คือ การทำงานของวงจรถอดรหัสแบบ Hard Decision นั้นข้อมูลอินพุตที่รับจากส่วนของมอดูเลชันที่ถูกจัดระดับมากกว่าสองระดับต่ออินพุตหนึ่งบิต เช่น 4 ระดับ (2 บิต) หรือ 8 ระดับ (3 บิต) เป็นต้น การทำงานแบบ Soft Decision นั้นมีการทำงานที่ซับซ้อนกว่าแบบ Hard Decision แต่ให้ค่า Coding Gain ที่มากกว่า โดยการทำงานแบบ Hard decision นั้นใช้สำหรับช่องสัญญาณ Binary Symmetric หรือ Discrete Memoryless Channel ส่วนการทำงานแบบ Soft Decision นั้นให้ค่า Coding Gain ที่มากกว่าแบบ Hard Decision ประมาณ 2 dB ที่ Soft Decision แบบ 8 ระดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.7.2 การถอดรหัสด้วย Viterbi Algorithm

การทำงานของถอดรหัสด้วย Viterbi Algorithm เป็นการหาเส้นทางในแผนภาพ Trelli ซึ่งมีลักษณะที่ใกล้เคียงกับข้อมูลที่ถูกส่งมามากที่สุด โดย Viterbi Algorithm นั้น มีขั้นตอนในการทำงานแบ่งเป็น 3 ขั้นตอน ดังต่อไปนี้

ขั้นตอนที่ 1 Branch Metric Generation

ขั้นตอนนี้คำนวณหาค่า Branch Metric จากข้อมูลอินพุตที่รับเข้ามา r กับค่าเอาต์พุตของการเข้ารหัส C การคำนวณหาค่า Branch Metric ต้องคำนวณทุกๆ สาขาหรือ Branch Metric แสดงดังสมการ

$$BM_{i,j,n} = (r_n - C_{i,j})^2 \quad (2.25)$$

โดย ค่า BM แทนค่า Branch Metric ระหว่างสเตต i ไปยังสเตต j ณ เวลา n

ค่า r แทนค่า ข้อมูลอินพุตที่รับเข้ามา ณ เวลา n

ค่า C แทนค่าเอาต์พุตของการเข้ารหัสระหว่างสเตต i ไปยังสเตต j ณ เวลา n

ขั้นตอนที่ 2 Survivor Path and Path Metric Update

ขั้นตอนนี้คำนวณหาค่า Survivor Path และ Path Metric จากจำนวนสเตตการทำงานทั้งหมดค่า Path Metric ที่เลือกไว้เพื่อใช้ในการหาค่า Path Metric ครั้งต่อไป (Update) ส่วนค่า Survivor เป็นค่าที่ใช้ในการตัดสินใจหาค่าเอาต์พุต โดยการคำนวณหาค่า Survivor Path และ Path Metric นั้นค่าของ Branch Metric และ Path Metric จะถูกเข้าด้วยกัน ซึ่งผลการบวกนั้นมีสองค่าที่เข้ามาในแต่ละจุดเชื่อมต่อ (Trellis Node) ของแผนภาพ Trellis โดยค่า Path Metric เป็นค่าที่เลือกจากค่าผลบวกที่น้อยกว่า ส่วนค่า Survivor Path เป็นสเตตการทำงานที่น้อยกว่า Path Metric ซึ่งแสดงดังสมการ

$$PM_{j,n} = \min(PM_{i,n-1} + BM_{i,j,n}, PM_{i+1,n-1} + BM_{i+1,j,n}) \quad (2.26)$$

โดย ค่า PM แทน Path metric ระหว่างสเตต i ไปยังสเตต j ณ เวลา n

ค่า BM แทนค่า Branch metric ระหว่างสเตต i ไปยังสเตต j ณ เวลา n

ขั้นตอนที่ 3 Optimum Paths Trace Back

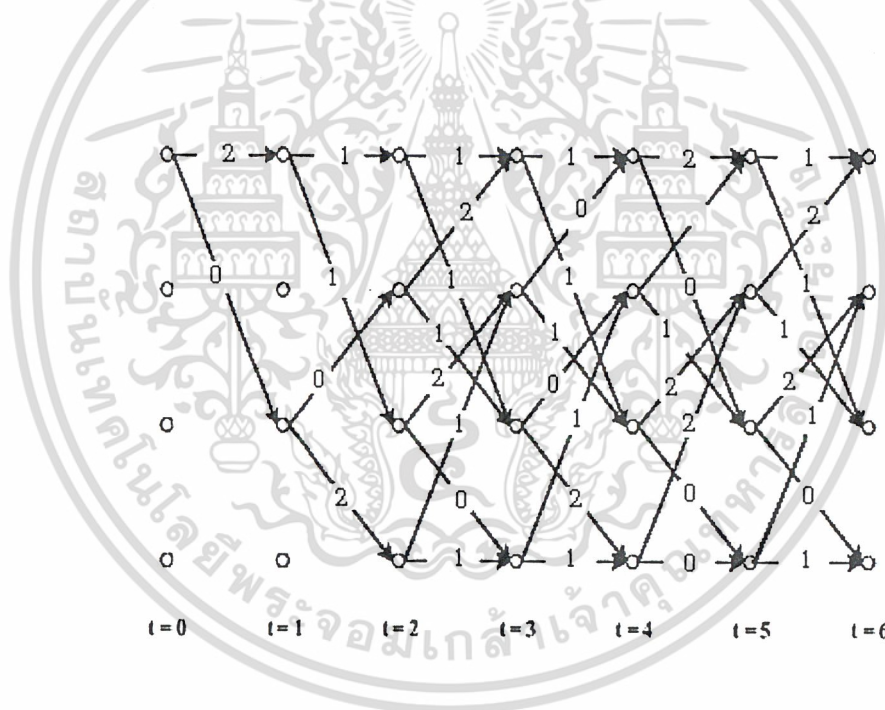
ขั้นตอนนี้เป็นขั้นตอนการตัดสินใจหาค่าเอาต์พุต โดยใช้ค่า Survivor path ในแต่ละสเตตที่บันทึกไว้มาตัดสินใจเลือกเส้นทางของข้อมูล โดยการตัดสินใจหาเส้นทางของข้อมูลจะเริ่มจาก Survivor Path ในอดีต (Trace Back) โดยในการเริ่มต้นจากเวลาผ่านไป L (Latency) โดยในทางปฏิบัติการ L ต้องมีค่ามากกว่าห้าเท่าของค่า K (Constrain Length) จึงทำให้ข้อมูลที่ถูกรหัสมีความถูกต้องสูง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

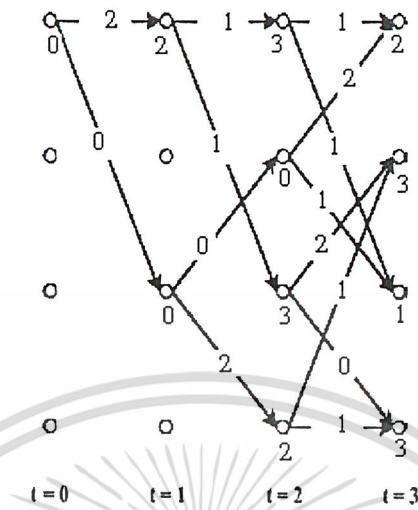
2.7.3 การถอดรหัสคอนโวลูชันที่ R เท่ากับ $1/2$ และ K เท่ากับ 3 ด้วยวิธี Viterbi

กรณีของการถอดรหัสข้อมูลที่ถูกเข้ารหัสคอนโวลูชันด้วยวงจรในรูปที่ 2.7 เมื่อป้อนข้อมูลข่าวสารอินพุตคือ “1 0 1 0 1 1” ข้อมูลเอาต์พุตคือ “11 01 10 01 11 10 00 01” ในวิธีการถอดรหัสโดยใช้วิธี Viterbi นั้นการทำงานเริ่มต้นจากการนำข้อมูลที่ได้นั้นมาแบ่งออกเป็นชุดๆ ที่มีขนาดเท่ากับจำนวนค่ารหัสซึ่งในวงจรนี้มีค่าเท่ากับสองบิต ดังนั้นข้อมูลนี้จะถูกแบ่งออกเป็น “11 01 10 01 11 10 00 01” โดยข้อมูลนี้จะถูกนำมาเปรียบเทียบกับข้อมูลที่อยู่ในเส้นทางในแผนภาพ Trellis เพื่อหาค่า Branch Metric ระหว่างข้อมูลที่ทั้งสอง ในเวลา t_1 เป็นการนำข้อมูล “11” ที่ได้รับมาเปรียบเทียบกับเส้นทางที่ลากจากสเตจเริ่มต้นที่มีข้อมูลเป็น “0” มายังสเตจ “00” ในเวลาถัดไป ซึ่งมีข้อมูลในเส้นทางเป็น “0/00” ซึ่งค่า

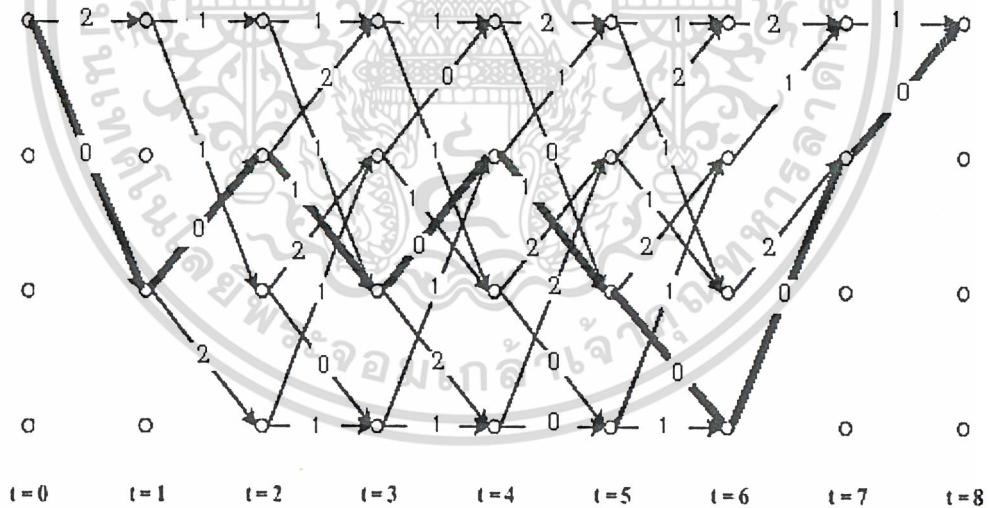
Branch Metric เท่ากับสองและเมื่อเปรียบเทียบกับเส้นทางที่ไปสเตจ “10” ในเวลาถัดไป ค่า Branch Metric เท่ากับศูนย์ ในเวลา t_2 นั้นข้อมูล “01” ที่ได้รับจะถูกนำมาเปรียบเทียบกับเส้นทางคำนวณเช่นเดียวกันนี้เรื่อยๆ จนครบทุกเส้นทาง จากนั้นในขั้นตอนต่อไปเป็นการคำนวณหาค่า Path Metric โดยในรูปที่ 2.8 แสดงการคำนวณหาค่า Branch Metric ในแผนภาพ Trellis ของวงจรเข้ารหัสในรูปที่ 2.9



รูปที่ 2.7 ค่า Branch Metric ในแผนภาพ Trellis



รูปที่ 2.8 Path Metric ในเวลา t_1 ถึง t_3



รูปที่ 2.9 เส้นทางของข้อมูลที่ได้จากการถอดรหัสด้วยวิธี Viterbi

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การคำนวณหาค่า Path Metric เริ่มต้นจากสเตต “00” ที่อยู่ซ้ายบนสุดของรูปที่ 2.8 ซึ่งกำหนดให้มีค่า Path Metric เป็นศูนย์กับสเตตที่เชื่อมต่อกับสเตตเริ่มต้นนี้ซึ่งมีสองสเตตคือ “00” และ “10” ในเวลาถัดไป การคำนวณหาค่า Path Metric จะนำ Branch Metric ที่อยู่ในเส้นทางนั้นมาบวกกับ Path Metric สะสมที่ถูกเก็บไว้ ผลลัพธ์ที่ได้จะถูกเก็บไว้สเตตที่อยู่ปลายทางของเส้นทางนั้นๆ เช่น ในการคำนวณหาค่า Path Metric สะสมที่สเตต “00” ในเวลา t_3 นั้น เป็นการตัดสินใจเลือกเส้นทางระหว่างเส้นทางที่มาจากสเตต “00” และสเตต “01” ของเวลา t_2 ซึ่งจะเลือกเส้นทางที่มีค่า Path Metric สะสมที่น้อยที่สุดโดยสำหรับเส้นทางที่มาจากสเตต “00” มีค่า Path Metric สะสมเท่ากับ 4 ($3+1$) และเส้นทางที่มาจาก สเตต “01” มีค่าเท่ากับ $2(0+2)$ ดังนั้นจึงเลือกเส้นทางที่มาจากสเตต “01” ซึ่งได้ค่า Path Metric เท่ากับ 2 การทำงานลักษณะนี้จะทำซ้ำจนครบทุกช่วงเวลาโดยในเวลา t_1 ถึง t_3 นั้นค่า Path Metric แสดงดังรูปที่ 2.8 เมื่อเวลาผ่านไปมากกว่าห้าเท่าของค่า Constrain Length จึงเริ่มการตัดสินใจเพื่อเลือกเส้นทางของข้อมูลที่ถูกถอดรหัสโดยคำนวณจากเส้นทางที่ Path Metric สะสมน้อยที่สุดเท่านั้นโดยการเริ่มต้นที่สเตต “00” ที่อยู่ขวาสุดของรูปจากนั้นจึงย้อนกลับไปในสเตตที่อยู่ซ้ายมือที่มีเส้นทางต่อกับสเตต “00” โดยเลือกเส้นทางที่ต่อกับสเตตที่มีค่า Path Metric สะสมที่มีค่าน้อยที่สุดไปยังสเตตที่อยู่ซ้ายมือจนมีการค้นหาเส้นทางครบทุกช่วงเวลา ผลลัพธ์ที่ได้จากการทำงานได้ข้อมูลที่ถอดรหัสเท่ากับ “1 0 1 0 1 1” ซึ่งแสดงดังรูปที่ 2.9

บทที่ 3

เอฟพีจีเอพื้นฐานและการออกแบบ

ปัจจุบันเทคโนโลยีวงจรรวม (Integration Circuit) แบบดิจิทัลได้พัฒนาอย่างรวดเร็ว การออกแบบวงจรรวมนั้น สามารถรวมอุปกรณ์ดิจิทัลที่มีจำนวนมากให้อยู่ในรูปของวงจรรวมตัวเดียวซึ่งสามารถลดค่าใช้จ่ายและการใช้พลังงานลง ในบทนี้กล่าวถึงความรู้พื้นฐานวงจรรวมแบบดิจิทัล โครงสร้างพื้นฐานของอุปกรณ์เอฟพีจีเอ Xilinx ตลอดจนเทคนิควิธีการออกแบบและติดตั้งวงจรดิจิทัลบนอุปกรณ์เอฟพีจีเอ ประวัติความเป็นมาและโครงสร้างของภาษาเวริล็อก (Verilog HDL)

3.1 ความรู้พื้นฐานวงจรรวมแบบดิจิทัล

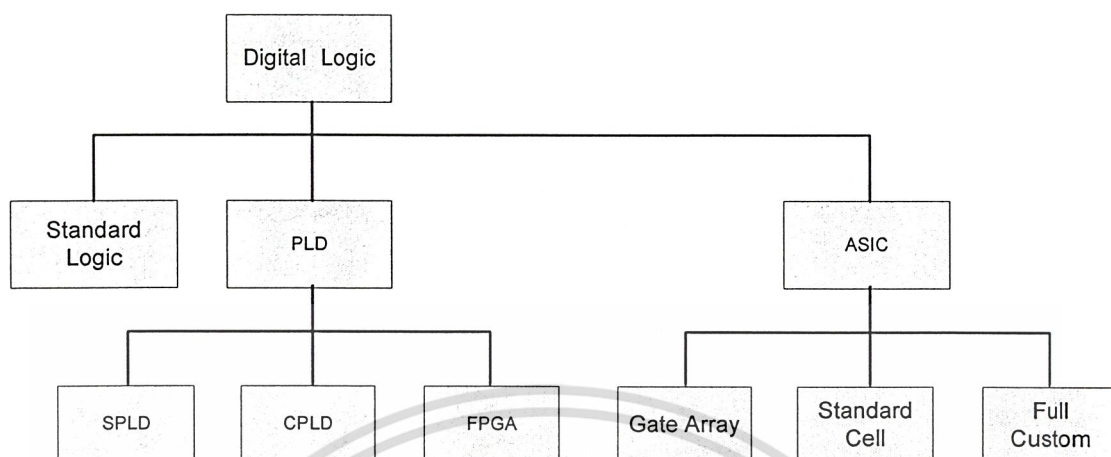
วงจรรวมแบบดิจิทัลสามารถแบ่งออกได้เป็น 3 ประเภทใหญ่ๆ คือ ลอจิกมาตรฐาน (Standard Logic) พีแอลดี (Programmable Logic Device: PLD) และเอสิก (Application Specific Integrated Circuit: ASIC) ซึ่งแสดงดังรูปที่ 3.1

3.1.1 วงจรรวมแบบลอจิกมาตรฐาน (Standard Logic)

วงจรรวมประเภทลอจิกมาตรฐานเป็นการออกแบบวงจรดิจิทัลที่ใช้วงจรรวมที่ออกแบบเป็นมาตรฐานไว้แล้ว เช่น วงจรรวมแบบทีทีแอล (TTL) หรือแบบซีเอ็มอส (CMOS) ชุดควบคุม เช่น ไมโครคอนโทรลเลอร์ หน่วยความจำต่างๆ เป็นต้น วงจรรวมประเภทนี้มีขนาดใหญ่และมีการใช้พลังงานมาก ซึ่งปัจจุบันการออกแบบวงจรดิจิทัลนิยมใช้อุปกรณ์พีแอลดีและเอสิกมากกว่าเนื่องจากมีขนาดเล็กและประสิทธิภาพสูง

3.1.2 วงจรรวมแบบพีแอลดี (Programmable Logic Device: PLD)

วงจรรวมดิจิทัลประเภทพีแอลดีเป็นอุปกรณ์ที่สามารถโปรแกรมสร้างวงจรดิจิทัลภายในได้ โดยพีแอลดีประกอบด้วยกลุ่มของลอจิกพื้นฐานทั้งวงจรคอมบิเนชัน (Combination) วงจรซีควENTIAL (Sequential) การออกแบบและโปรแกรมพีแอลดีสามารถทำได้เองโดยใช้คอมพิวเตอร์ส่วนบุคคล ซึ่งการออกแบบวงจรดิจิทัลด้วยอุปกรณ์พีแอลดีนั้นใช้เวลาน้อยกว่าการออกแบบด้วยวงจรรวมประเภทเอสิก และมีต้นทุนที่ถูกกว่าเอสิกในกรณีที่ต้องการผลิตอุปกรณ์จำนวนน้อยเพราะในการออกแบบนั้นไม่ต้องเสียค่า NRE (Non-Recurring Engineering cost)



รูปที่ 3.1 ไลอะแกรมเทคโนโลยีวงจรรวมแบบดิจิทัล

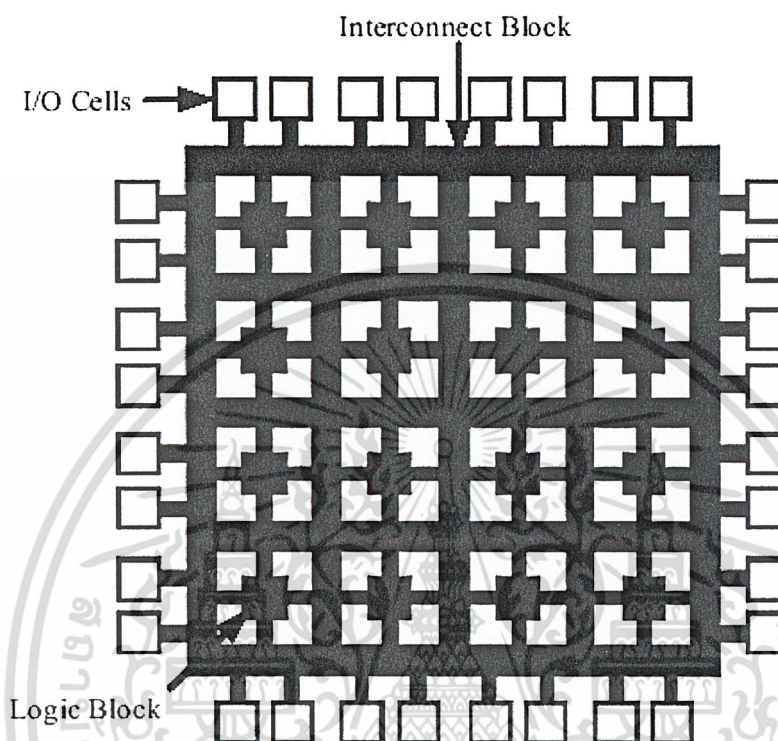
ซึ่งเป็นค่าติดตั้งเครื่องจักร (Set-up Machine) ในการผลิตวงจรรวมซึ่งมีราคาแพง และข้อดีของอุปกรณ์พีแอลดีคือ สามารถโปรแกรมเพื่อปรับเปลี่ยนวงจรในพีแอลดีได้หลายครั้งทำให้ผู้ใช้สามารถปรับปรุงความสามารถของอุปกรณ์ให้ดีขึ้นโดยไม่ต้องเปลี่ยนอุปกรณ์ใหม่ ซึ่งพีแอลดีนี้แบ่งได้เป็น 3 ประเภท คือ เอสพีแอลดี (Simple Programmable Logic Device: SPLD) ซีพีแอลดี (Complex Programmable Logic Device : CPLD) และเอฟพีจีเอ (Field Programmable Gate Array: FPGA)

3.2 ความรู้พื้นฐานอุปกรณ์เอฟพีจีเอ (Field Programmable Gate Array: FPGA)

เอฟพีจีเอหรือ FPGA(Field Programmable Gate Array) เป็นอุปกรณ์อิเล็กทรอนิกส์ที่สามารถโปรแกรมได้เพื่อให้มีฟังก์ชันการทำงานตามที่ต้องการ ซึ่งถูกพัฒนาขึ้นโดยบริษัท Xilinx ในปี 1985 เอฟพีจีเอนั้นมีปริมาณความหนาแน่นของเกตสูงกว่าอุปกรณ์ซีพีแอลดีมาก การออกแบบวงจรดิจิทัลด้วยเอฟพีจีเอนั้นเป็นที่นิยมมากในปัจจุบัน เพราะการออกแบบด้วยเอฟพีจีเอสะดวกรวดเร็วเนื่องจากการออกแบบและการโปรแกรมเอฟพีจีเอสามารถทำได้โดยใช้คอมพิวเตอร์ส่วนบุคคล การเปลี่ยนแปลงแก้ไขทำได้ง่ายและการออกแบบด้วยเอฟพีจีเอนั้นไม่ต้องเสียค่า NRE ซึ่งมีราคาแพงทำให้ประหยัดค่าใช้จ่ายและบริษัทผู้ผลิตเอฟพีจีเอ เช่น Xilinx หรือ Altera นั้นพยายามนำอุปกรณ์นี้แข่งขันด้านการตลาดกับอุปกรณ์เอสิก (ASIC) จึงส่งผลให้ราคาอุปกรณ์เอฟพีจีเอมีแนวโน้มที่ถูกลง

โครงสร้างของเอฟพีจีเอนั้นมีโครงสร้างที่ซับซ้อนมาก โดยโครงสร้างภายในของเอฟพีจีเอแบ่งได้เป็น 3 ส่วนคือ บล็อกลอจิก(Logic Block) บล็อกเชื่อมต่อภายใน (Interconnect Block) และไอโอเซลล์ (I/O Cell) โดยลอจิกบล็อกซึ่งถูกจัดอยู่ในรูปเมตริก ภายในประกอบไปด้วย วงจรเกตต่างๆ รีจิสเตอร์

หน่วยความจำ เพื่อใช้สร้างฟังก์ชันในการใช้งานโดยมีบล็อกเชื่อมต่อภายในทำหน้าที่เชื่อมต่อระหว่างลอจิกบล็อกและเชื่อมต่อกับไอโอเซลล์ (I/O Cell) ซึ่งทำหน้าที่เป็นอินพุตเพื่อติดต่อกับวงจรภายนอก



รูปที่ 3.2 โครงสร้างทั่วไปของเอฟพีจีเอ

3.2.1 โครงสร้างของเอฟพีจีเอของบริษัท Xilinx

วิทยานิพนธ์นี้ได้ใช้เอฟพีจีเอ ของบริษัท Xilinx เป็นต้นแบบสำหรับติดตั้งชุดถอดรหัส Viterbi ซึ่งเอฟพีจีเอตระกูลนี้ถูกออกแบบให้มีประสิทธิภาพและความจุของเกตสูง โดยมีความจุตั้งแต่ 40K เกต จนถึง 8M เกต ในกระบวนการผลิตเอฟพีจีเอ ของบริษัท Xilinx นี้ใช้เทคโนโลยีแบบ CMOS 0.15 μm / 0.12 μm 8-Layer Metal ทำให้เอฟพีจีเอตระกูลนี้มีความเร็วในการทำงานสูงและมีการใช้พลังงานน้อย รายละเอียดต่างๆของเอฟพีจีเอของบริษัท Xilinx มีดังนี้

3.2.2 โครงสร้างภายในเอฟพีจีเอ Xilinx

โครงสร้างของเอฟพีจีเอ Xilinx ประกอบด้วยสองส่วนใหญ่ๆ คือ บล็อก (Programmable Input/Output: I/OB) และ Internal Configurable Logic Block (ICLB) โดยบล็อกอินพุตเอาต์พุตสามารถโปรแกรมให้อินพุตหรือเอาต์พุตเพื่อติดต่อกับอุปกรณ์ภายนอกได้ ส่วน ICBL นั้นประกอบด้วย 4 อุปกรณ์หลัก คือ

- บล็อก Configurable Logic Block (CLB) ประกอบด้วยวงจรลอจิกแบบคอมบินเนชัน

(Combination) และแบบซิงโครนัส (Synchronous) รีจิสเตอร์และบัฟเฟอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- บล็อก SelectRAM เป็นหน่วยความจำแบบสองพอร์ต (Dual-port RAM) ที่สามารถเลือกขนาดได้ โดยมีขนาดสูงสุด 18 กิโลบิต
- บล็อก Multiplier เป็นวงจรคูณขนาด 18x18 บิต
- บล็อก DCM (Digital Clock Manager) เป็นบล็อกที่จัดการสัญญาณนาฬิกา ซึ่งบล็อกนี้สามารถคูณหรือหารความถี่สัญญาณนาฬิกา และนอกจากนั้นยังสามารถเลื่อนเฟส (Phase Shift) ของสัญญาณนาฬิกาได้

อุปกรณ์ใน ICLB นี้อยู่ในรูปแบบของแถวลำดับ (Array) อย่างสม่ำเสมอและอุปกรณ์เหล่านี้สามารถโปรแกรมการเชื่อมต่อกันอย่างรวดเร็วโดยส่วนของบล็อกเชื่อมต่อ (Interconnects) ซึ่งในเอฟพีจีเอของบริษัท Xilinx เรียกเทคโนโลยีการเชื่อมต่ออย่างรวดเร็วนี้ว่า “Active Interconnect technology”

3.2.3 รายละเอียดของอุปกรณ์ภายในเอฟพีจีเอ Xilinx

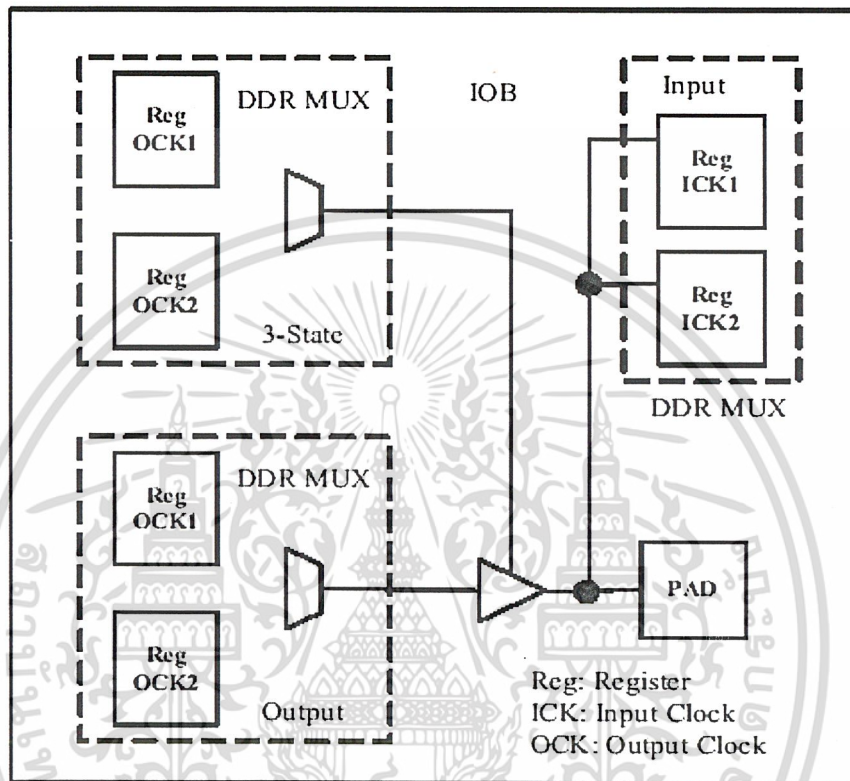
ในส่วนนี้กล่าวถึงรายละเอียดของอุปกรณ์ภายในของเอฟพีจีเอตระกูล Xilinx ซึ่งอุปกรณ์ภายในเอฟพีจีเอที่สำคัญที่ใช้ในการออกแบบในงานวิจัยมีดังนี้

3.2.3.1 บล็อกอินพุตเอาต์พุต (Input/Output Block: IOB)

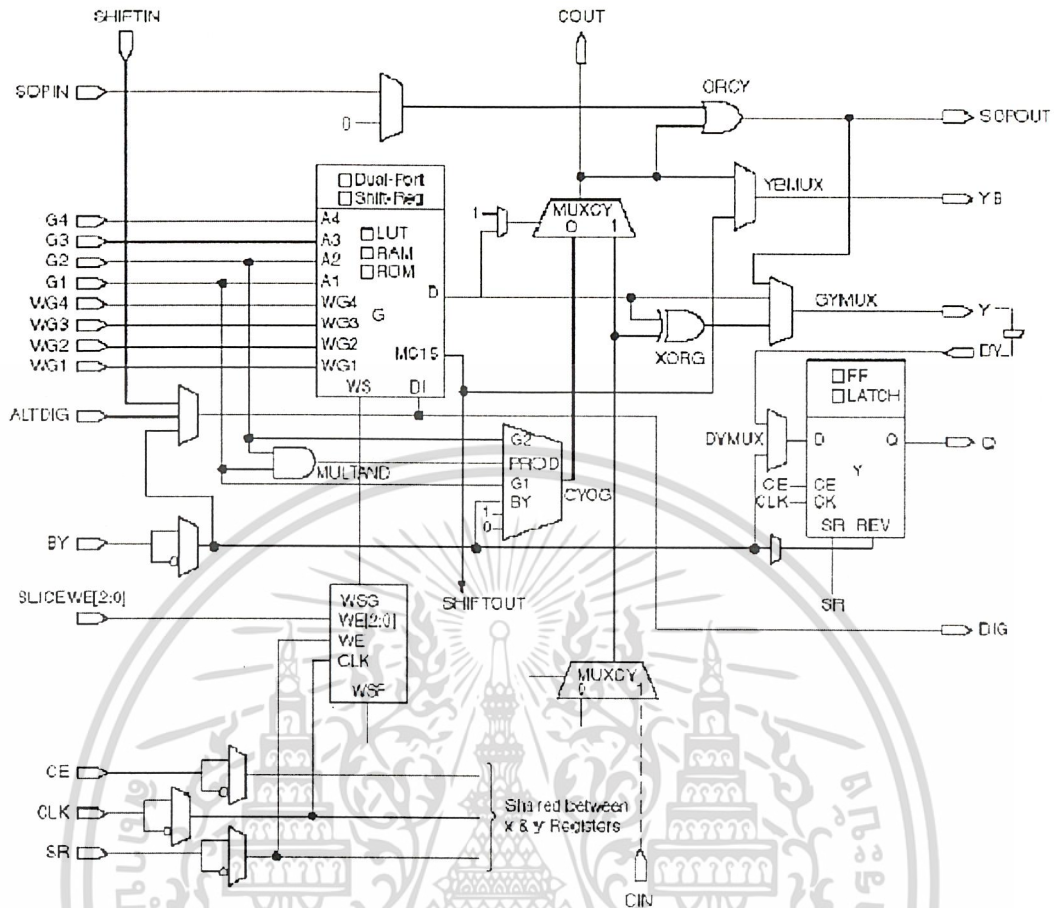
บล็อกอินพุตเอาต์พุต (IOB) เป็นส่วนที่ใช้ติดต่อกับวงจรภายนอก ซึ่งบล็อกอินพุตและเอาต์พุตสามารถโปรแกรมให้เป็นอินพุตแบบ Single Data Rate หรือแบบ Double Data Rate (DDR) ได้ ซึ่งบล็อกอินพุตเอาต์พุตนี้ประกอบด้วย 6 รีจิสเตอร์ (Register) และ 3 DDR Multiplexers ซึ่งแสดงดังรูปที่ 3.3 โดยรีจิสเตอร์นี้สามารถโปรแกรมเป็นดีฟลิปฟล็อป (D flip-flop) หรือแลตช์ (Latch) ได้ บล็อกอินพุตเอาต์พุตนี้ต่ออยู่กับส่วน Switch Matrix เพื่อเชื่อมต่อกับอุปกรณ์ภายในอื่นๆ ส่วน DDR Multiplexer ใช้ในการส่งข้อมูลแบบ DDR

3.2.3.2 บล็อก Configurable Logic Block (CLB)

บล็อก Configurable Logic Block (CLB) ประกอบด้วย วงจรลอจิกแบบคอมไบเนชัน และแบบซิงโครนัส ซึ่ง CLB ต่ออยู่กับส่วน Switch Matrix เพื่อเชื่อมต่อกับอุปกรณ์ภายในอื่นๆ โดยในแต่ละ CLB นั้นประกอบด้วย Slices จำนวน 4 Slices ซึ่งแต่ละ Slices มีลักษณะคล้ายกัน ซึ่ง Slices ประกอบด้วย อุปกรณ์ดังนี้ คือ ตัวสร้างฟังก์ชันสี่อินพุต (4-input Function Generators) จำนวน 2 ตัว วงจรลอจิกตัวทด (Carry Logic) วงจรบวก (Arithmetic Logic Gate) วงจรมัลติเพล็กซ์ขนาดใหญ่ (Wide Function Multiplexers) และหน่วยความจำ (Storage Elements) จำนวน 2 ตัว



รูปที่ 3.3 โครงสร้างของบล็อกอินพุตเอาต์พุต



รูปที่ 3.4 โครงสร้างของ Slice

ตารางที่ 3.1 ขนาดของหน่วยความจำที่สามารถโปรแกรมใช้งานได้ของบล็อกหน่วยความจำ

16K×1บิต	2K×9บิต
8K×2บิต	1K×18บิต
4K×4บิต	512K×36บิต

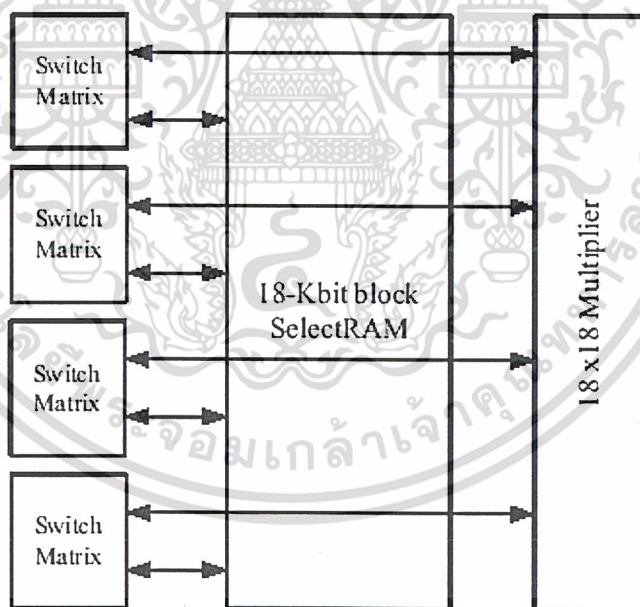
โดยตัวสร้างฟังก์ชันสี่อินพุตสามารถโปรแกรมให้เป็นตัวเปิดตารางสี่อินพุต (4-input Look-up Table) หรือโปรแกรมให้เป็นหน่วยความจำขนาด 16 บิต (16-bit of Distributed SelectRAM Memory) หรือโปรแกรมให้หิพริจิสเตอร์ 16 บิตที่ปรับเปลี่ยนได้ (16-bit Variable Tap Shift Register) โดยเอาต์พุตของตัวสร้างฟังก์ชันนี้ต่อกับหน่วยความจำ (Storage Element) ซึ่งเชื่อมต่อกับอุปกรณ์อื่นๆ โดยหน่วยความจำนี้สามารถโปรแกรมเป็นดีฟลิปฟล็อป (D Flip-Flop) หรือแลตช์ (Latch) ก็ได้ ซึ่งรูปที่ 3.4 นั้นแสดงโครงสร้างของ Slice

3.2.3.3 บล็อกหน่วยความจำ 18 กิโลบิต (18 Kbit Block SelectRAM Memory)

บล็อกหน่วยความจำ (18 Kbit Block SelectRAM Memory) เป็นหน่วยความจำที่สามารถโปรแกรมให้เป็นหน่วยความจำแบบพอร์ตเดียว (Single-port RAM) หรือหน่วยความจำแบบสองพอร์ต (Dual-port RAM) ได้ สามารถเลือกขนาดจาก 16 K x 1 บิตจนถึง 512 x 32 และสามารถเลือกให้ทำงานเป็นแบบซิงโครนัสหรืออซิงโครนัสได้ ซึ่งตารางที่ 3.1 แสดงขนาดหน่วยความจำที่สามารถเลือกโปรแกรมให้ใช้งานได้ของบล็อกหน่วยความจำนี้

3.2.4.4 บล็อกวงจรคูณขนาด 18x 18 บิต (18-Bit x 18-BIT Multiplier)

บล็อกวงจรคูณขนาด 18 x 18 บิตนี้เป็นวงจรคูณที่สามารถปรับเปลี่ยนขนาดได้ โดยมีขนาดสูงสุดคือ 18 x 18 บิต และสามารถทำการคูณแบบคิดเครื่องหมาย (2's Complement Signed Multiplier) ได้ บล็อกวงจรคูณขนาด 18 x 18 บิตนี้สามารถทำงานอย่างอิสระหรือทำงานร่วมกับบล็อกหน่วยความจำ 18 กิโลบิตในการทำงานที่เรียกว่า "Multiply-Accumulator (MAC)" ซึ่งเป็นฟังก์ชันการทำงานของการประมวลผลสัญญาณดิจิทัล (Digital Signal Processing) โดยรูปที่ 3.5 แสดงการเชื่อมต่อของบล็อกหน่วยความจำ 18 กิโลบิตและบล็อกวงจรคูณขนาด 18 x 18 บิต ซึ่งบล็อกทั้งสองจะถูกเชื่อมต่อกับ Switch Matrix



รูปที่ 3.5 การเชื่อมต่อบล็อกหน่วยความจำ 18 กิโลบิตและบล็อกวงจรคูณขนาด 18 x 18 บิต

3.2.4.5 บล็อก Digital Clock Manager (DCM)

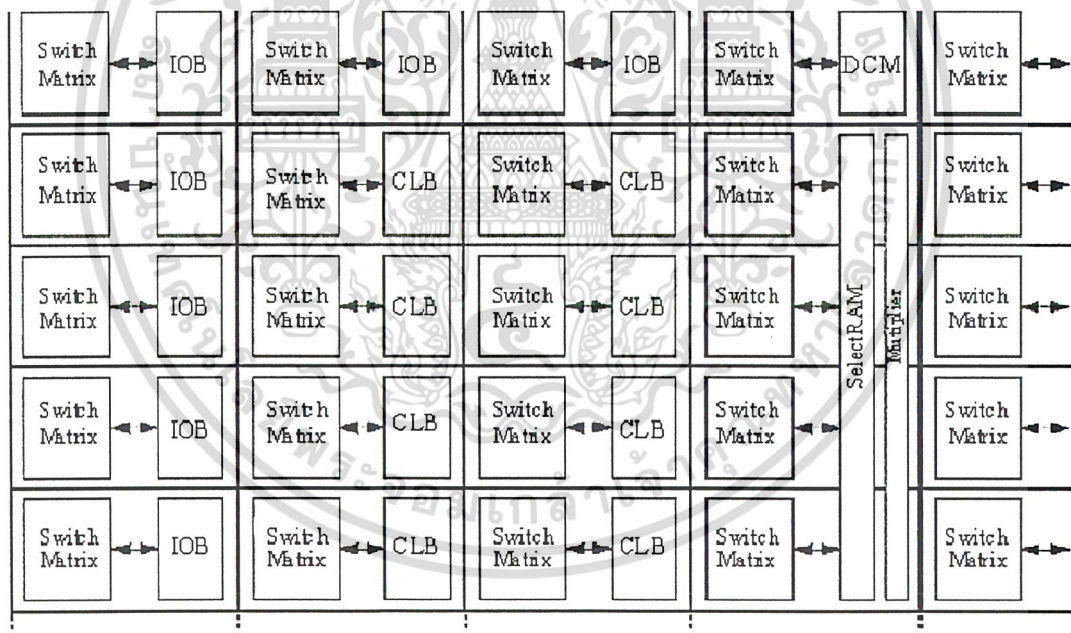
บล็อก DCM (Digital Clock Manager) เป็นบล็อกที่จัดการสัญญาณนาฬิกาของเอฟพีจีเอ โดยการทำงานใช้หลักการ Digital Delay Line และการป้อนกลับ ซึ่งมีลักษณะการทำงาน 3 แบบดังนี้

- Clock De-skew

การทำงานนี้ DCM จะกำเนิดสัญญาณนาฬิกาของระบบ ซึ่งมาจากภายในและภายนอกเอฟพีจีเอใหม่ ซึ่งมีเฟสตรงกับอินพุตของสัญญาณนาฬิกา ดังนั้นจึงสามารถกำจัดปัญหาการหน่วงเวลาของสัญญาณนาฬิกาที่มาจากขาของเอฟพีจีเอ (Pad) สามารถจับอุปกรณ์ภายในได้โดยทันที

- Frequency Synthesis

การทำงานนี้ DCM สามารถกำเนิดความถี่สัญญาณนาฬิกาใหม่ โดยใช้การคูณหรือการหารสัญญาณนาฬิกาอินพุต ในการหารความถี่นั้นสำหรับเอฟพีจีเอตระกูลนี้มีตัวหารเท่ากับ 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 9, 10, 11, 12, 13, 14, 15 และ 16 ส่วนการคูณนั้นมีตัวคูณคือ M/D โดย M และ N เป็นจำนวนเต็ม แต่ค่าความถี่ที่ได้จากการคูณต้องไม่เกิน DCM Timing Parameter ของเอฟพีจีเอแต่ละรุ่น (สูงสุดไม่เกิน 350 MHz ในเอฟพีจีเอตระกูล Xilinx)



รูปที่ 3.6 โครงสร้างของบล็อกเชื่อมต่อและหาเส้นทาง

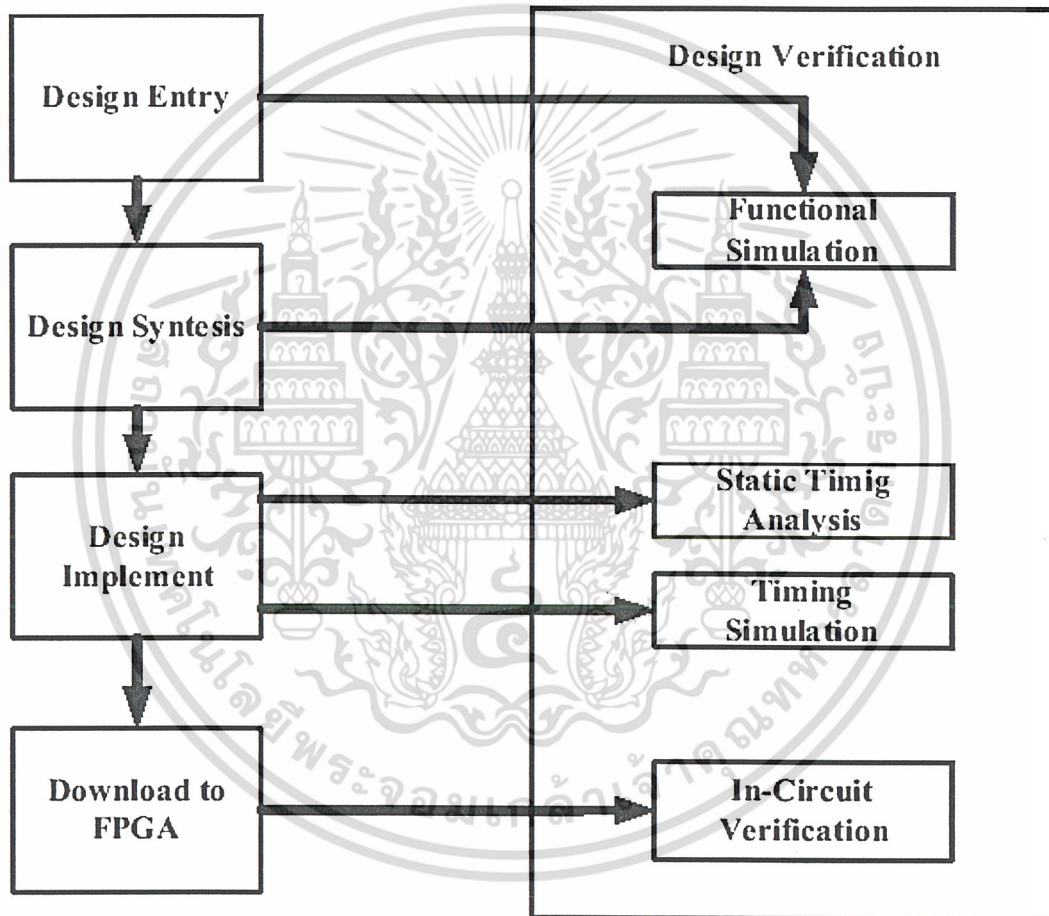
- Phase shifting

DCM สามารถเลื่อนเฟสของสัญญาณนาฬิกาได้ ซึ่งการเลื่อนเฟสของสัญญาณนาฬิกาสามารถทำได้ 90, 180 และ 270 องศา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.4.6 บล็อกเชื่อมต่อและหาเส้นทาง (Routing Resources)

การเชื่อมต่อและหาเส้นทางของอุปกรณ์ภายในเฟรียเอตระกูลนี้ใช้อุปกรณ์ Switch Matrix ในการเชื่อมต่ออุปกรณ์ภายในเช่น CLB, IOB, Block RAM, Multiplier และ DCM เข้าด้วยกัน ซึ่ง การเชื่อมต่อและหาเส้นทางใช้เทคโนโลยีที่เรียกว่า “Active Interconnect Technology” ซึ่งทำให้การเข้าถึงอุปกรณ์ต่างๆทำได้อย่างรวดเร็ว ซึ่งรูปที่ 3.6 แสดงโครงสร้างการเชื่อมต่อและหาเส้นทางของเฟรียเอตระกูล Xilinx



รูปที่ 3.7 ขั้นตอนการออกแบบและติดตั้งวงจรดิจิทัลบน FPGA

3.3 การออกแบบและติดตั้งวงจรดิจิทัลบนเอฟพีจีเอ

การออกแบบวงจรดิจิทัลเพื่อติดตั้งกับเอฟพีจีเอนั้นใช้ภาษาอธิบายฮาร์ดแวร์ (Hardware Description Language: HDL) เช่น ภาษาวีเอชดีแอล (VHDL) หรือภาษาเวรล็อก (Verilog HDL) ในการออกแบบวงจร จากนั้นนำโค้ดที่ได้ไปสังเคราะห์ (Synthesis) ให้เป็นวงจร ผลการสังเคราะห์ที่ได้คือ เน็ตลิสต์ในระดับเกต (Gate-level Netlist) ซึ่งมีรูปแบบมาตรฐาน คือ EDIF (Electronic Data Interchange Format) หรือ XNF (Xilinx Netlist Format) ซึ่งใช้สำหรับเอฟพีจีเอของบริษัท Xilinx ขั้นตอนการออกแบบและติดตั้งวงจรดิจิทัลบนเอฟพีจีเอ แสดงดังรูปที่ 3.7 ส่วนรายละเอียดขั้นตอนการออกแบบมีดังนี้

- การออกแบบ (Design Entry)

การออกแบบวงจรเริ่มต้นจาก การเขียนภาษาอธิบายฮาร์ดแวร์หรือเอชดีแอล (HDL) ในการอธิบายพฤติกรรม (Behavioral) การทำงานของวงจรหลังจากเข้าใจอัลกอริทึมการทำงานหรือมีการออกแบบวงจรไว้แล้ว ในการเขียนภาษาเอชดีแอลสามารถใช้โปรแกรม Text Editor หรือใช้ซอฟต์แวร์ที่สนับสนุนการเขียนภาษาเอชดีแอล ซึ่งทำให้ง่ายในการเขียนและสามารถตรวจสอบความถูกต้องตามหลักไวยากรณ์ได้ เช่น ModelSim ของบริษัท Model Technology เป็นต้น

หลังจากการออกแบบวงจรด้วยการเขียนภาษาเอชดีแอล ขั้นตอนที่สำคัญคือ การตรวจสอบว่าวงจรที่ออกแบบภาษาเอชดีแอลนั้นสามารถทำงานได้ตามที่ออกแบบ ซึ่งสามารถตรวจสอบได้โดยการจำลองการทำงานระดับพฤติกรรม (Behavioral Simulation) ของวงจรโดยการเขียนภาษาเอชดีแอลเพื่อทดสอบวงจรที่เรียกว่า “Testbench” ซึ่งเป็นการกำหนดรูปแบบของสัญญาณอินพุต และเมื่อจำลองการทำงานด้วยโปรแกรมซิมูเลเตอร์ (Simulator) เช่น Modelsim ก็จะได้สัญญาณเอาต์พุตทำให้สามารถตรวจสอบได้ว่าการทำงานของวงจรที่ถูกตรวจสอบ (Design Under Test : DUT) ทำงานถูกต้องหรือไม่ซึ่งถ้าไม่ถูกต้องก็สามารถกลับไปแก้ไขใหม่ได้

- การสังเคราะห์ (Design Synthesis)

เมื่อวงจรที่ออกแบบถูกตรวจสอบแล้วว่าการทำงานถูกต้อง จากนั้นโค้ดภาษาเอชดีแอล ก็จะถูกทำการสังเคราะห์เป็นวงจรเพื่อให้ได้เน็ตลิสต์ในระดับเกตออกมา แล้วตรวจสอบความถูกต้องอีกครั้ง โดยการจำลองการทำงานระดับฟังก์ชัน (Functional Simulation) ด้วยโปรแกรมซิมูเลเตอร์โดยใช้ Testbench เดิม ซึ่งถ้าการทำงานไม่ถูกต้องก็สามารถกลับไปแก้ไขโค้ดภาษาเอชดีแอลใหม่

- การติดตั้ง (Design Implement)

ในขั้นตอนการติดตั้งนี้ ผลที่ได้จากการสังเคราะห์ที่เป็นเน็ตลิสต์ในระดับเกตนั้นจะถูกแปลงให้เป็นกลุ่มของลอจิก จากนั้นจะทำการวางและเชื่อมเส้นทาง (Place & Route) ในเอฟพีจีเอโดยซอฟต์แวร์ของผู้ผลิตเอฟพีจีเอ เช่น Xilinx ISE เป็นต้น ซึ่งโปรแกรมจะรายงานเวลาการทำงานของวงจร (Static Timing) และสามารถตรวจสอบความถูกต้องโดยทำการจำลองการทำงานระดับฐานเวลา (Timing Simulation) ด้วยโปรแกรมซิมูเลเตอร์โดยใช้ Testbench เดิม การจำลองการทำงานระดับฐานเวลาถือว่าเป็นการจำลองการทำงานขั้นสุดท้ายซึ่งต้องทำหลังจากการวางและเชื่อมต่อเส้นทาง การจำลองการทำงานจะให้รายละเอียดที่แน่นอนเกี่ยวกับเรื่องของเวลา เช่น ความล่าช้าของสัญญาณไฟฟ้าตามเส้นทาง

ต่างๆ ภายในวงจร ระยะเวลาของสัญญาณนาฬิกา (Clock Period) น้อยที่สุดที่สามารถใช้วงจรได้ เป็นต้น ซึ่งถ้าการทำงานไม่ถูกต้องก็สามารถกลับไปแก้ไขโค้ดภาษาเอชดีแอลใหม่

- การโปรแกรมเอฟพีจีเอ (Download to FPGA)

เมื่อตรวจสอบด้วยการจำลองการทำงานระดับฐานเวลาแล้ว สามารถแปลงเน็ตลิสต์ในระดับเกตเป็นข้อมูลวงจร (Configuration Data) ซึ่งอยู่ในรูปของบิตสตรีม (Bit-stream) เพื่อนำบิตสตรีมนี้ไปโปรแกรมลงเอฟพีจีเอต่อไป ขั้นตอนนี้ใช้ซอฟต์แวร์ของผู้ผลิตเอฟพีจีเอ ซึ่งการตรวจสอบความถูกต้องในการทำงานลำดับสุดท้ายคือ การตรวจสอบในวงจรใช้งานจริง (In-Circuit Verification)

3.4 ภาษาเวริล็อก (Verilog HDL)

ภาษาเวริล็อก (Verilog HDL) เป็นภาษาอธิบายฮาร์ดแวร์ (Hardware Description Language : HDL) ที่ใช้ในการออกแบบวงจรระบบดิจิทัล ภาษาเวริล็อกนั้นเป็นหนึ่งในสองภาษาอธิบายฮาร์ดแวร์ที่นิยมใช้กันมากในการออกแบบวงจรระบบดิจิทัลในปัจจุบัน ซึ่งได้แก่ ภาษาวีเอชดีแอล (Veri high speed integrated circuit Hardware Description Language: VHDL) และภาษาเวริล็อก เนื่องจากภาษาเวริล็อกมีลักษณะคล้ายภาษาซี จึงทำให้วิศวกรผู้ออกแบบซึ่งส่วนใหญ่แล้วมีความรู้ด้านภาษาซี สามารถเรียนรู้และเข้าใจได้ง่าย และโครงสร้างของภาษาเวริล็อกยังมีความซับซ้อนน้อยกว่าภาษาเอชดีแอล

3.4.1 ประวัติความเป็นมา

ภาษาเวริล็อกถูกคิดค้นโดย บริษัท Gateway Design Automation ในปี 1981 ซึ่งเหตุการณ์ที่สำคัญและความเป็นมากล่าวเรียงตามลำดับได้ดังนี้

ปี 1983 บริษัท Gateway Design Automation ได้เปิดตัวภาษาอธิบายฮาร์ดแวร์ ซึ่งเรียกว่า “Verilog HDL” หรือเรียกง่าย ๆ ว่า “Verilog” พร้อมทั้งส่วนการจำลองการทำงานของภาษาเวริล็อก (Verilog Simulator)

ปี 1985 ภาษาเวริล็อกได้มีการปรับปรุงเพิ่มเติมและได้ออกเวอร์ชันใหม่ของการจำลองการทำงาน ซึ่งเรียกว่า “Verilog XL”

ปี 1987 ภาษาเวริล็อกเป็นที่นิยมใช้ ขนาดของวงจรที่ออกแบบนั้นเริ่มเกินความสามารถของชุดจำลองการทำงานของ Gateway Design System และบริษัทอื่นๆ

ปี 1989 บริษัท Cadence ซึ่งเป็นบริษัทผู้ผลิตซอฟต์แวร์ออกแบบวงจรรวมชั้นนำในปัจจุบันได้พัฒนาภาษาเวริล็อกต่อจากบริษัท Gateway Design System

ปี 1990 บริษัท Cadence ได้แยกภาษาเวริล็อกและส่วนการจำลองการทำงานออกจากกันโดยได้นำภาษาเวริล็อกเปิดเผยต่อสาธารณะและในปีนั้นผู้ใช้ภาษาเวริล็อกและบริษัทต่างๆ ได้จัดตั้งสมาคม Open Verilog International (OVI) เป็นผู้กำหนดควบคุมรายละเอียด (Specification) ของภาษาเวริล็อก และได้จัดทำคู่มืออ้างอิงสำหรับภาษาเวริล็อกเวอร์ชัน 1.0 (Verilog 1.0 Reference Manual) ขึ้นมา

ปี 1993 สมาคม OVI ได้ปรับปรุงเพิ่มเติมความสามารถของภาษาเวริล็อก เช่น ความสามารถของอาร์เรย์ และได้ออกคู่มืออ้างอิงสำหรับภาษาเวริล็อกใหม่ เป็นเวอร์ชัน 2.0 (Verilog 2.0 Reference Manual)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปี 1995 สมาคม OVL ได้ส่งภาษาเวรลล็อกเวอร์ชัน 2.0 ให้สมาคม IEEE ซึ่งสมาคม IEEE ได้ปรับปรุงและได้กำหนดเป็นมาตรฐานใหม่ที่เรียกว่า มาตรฐาน IEEE 1394-1995

ปี 2001 มีการปรับปรุงให้เพิ่มความสามารถของภาษาเวรลล็อกมาตรฐาน IEEE 1394 ทั้งด้านการสังเคราะห์วงจรและการจำลองการทำงาน อีกทั้งยังปรับปรุงให้ง่ายต่อการใช้งาน ซึ่งสมาคม IEEE กำหนดเป็นมาตรฐานใหม่ที่เรียกว่า มาตรฐาน IEEE 1394-2001

3.4.2 โครงสร้างของภาษาเวรลล็อก

ภาษาเวรลล็อกนั้นสามารถอธิบายพฤติกรรมของวงจรดิจิทัลได้หลายระดับได้แก่ การอธิบายวงจรทรานซิสเตอร์และการเชื่อมต่อ (Wire) ซึ่งเรียกว่า ระดับการสวิตช์ (Switch Level) การอธิบายการเชื่อมต่อของลอจิกเกตและฟลิปฟล็อปในวงจร ซึ่งเรียกว่า ระดับเกต (Gate Level) และการอธิบายการโอนถ่ายสัญญาณระหว่างรีจิสเตอร์ เรียกว่า ระดับ RTL (Register Transfer Level) ซึ่งการออกแบบวงจรดิจิทัลด้วยภาษาเวรลล็อกส่วนใหญ่จะใช้อธิบายพฤติกรรมของวงจรดิจิทัลในระดับ RTL นี้

ภาษาเวรลล็อกนั้นมีไวยากรณ์คล้ายภาษาซี เช่น จบประโยคด้วย ; หรือใช้ // เพื่อไม่ให้นำไปใช้ในการคอมไพล์ เป็นต้น โครงสร้างของภาษาเวรลล็อกนั้นประกอบด้วยมอดูล (Module) ซึ่งแต่ละมอดูลนั้นสามารถรวมเข้าด้วยกันเป็นมอดูลขนาดใหญ่หรือเรียกว่า ท็อปมอดูล (Top Module) โดยแต่ละมอดูลย่อยนั้นสามารถทำงานพร้อมกัน (Concurrent) ซึ่งโครงสร้างของภาษาเวรลล็อกประกอบด้วยส่วนต่างๆ ดังแสดงในตัวอย่างการอธิบายพฤติกรรมของวงจรดีฟลิปฟล็อป (D Flip-Flop) ได้ดังนี้

```
Module dff(clk,rst,din,dout);           // <ชื่อมอดูล>(<ชื่อพอร์ต>)
Input din, clk,rst;                    //<ชื่อสัญญาณอินพุต>
Output dout;                           //<ชื่อสัญญาณเอาต์พุต>
Reg dout;                              //<ชื่อสัญญาณ>
always@(posedge clk or posedge rst)    //<รายละเอียดโปรแกรม>
    if(rst)
        dout <= 1'b;
    else
        dout <= din;
endmodule                             //<จบมอดูล>
```

บทที่ 4

โครงสร้างชุดถอดรหัสและการออกแบบ

บทนี้จะกล่าวถึงหลักการทำงานและการออกแบบชุดถอดรหัส Viterbi ซึ่งเป็นชุดถอดรหัสที่ให้ประสิทธิภาพสูงในการถอดรหัสคอนวูลูชัน ในส่วนแรกกล่าวถึงโครงสร้างโดยทั่วไปและหลักการทำงานของชุดถอดรหัส Viterbi แบบต่างๆ และส่วนหลังนั้นกล่าวถึงการออกแบบชุดถอดรหัส Viterbi ที่มีโครงสร้างขนาดเล็กที่ใช้โครงสร้างขนานแบบ 4 Add-Compare-Select (ACS) ซึ่งเป็นจำนวนของ ACS ที่ใช้อุปกรณ์ในการสร้างจำนวนน้อย และให้ความเร็วในการถอดรหัสสูง นอกจากนั้นส่วน Survivor Memory Unit (SMU) ได้ปรับปรุงวิธีการทำงานแบบ Register Exchange ที่ให้ความเร็วในการทำงานสูง แต่มีโครงสร้างขนาดใหญ่ ให้มีขนาดเล็กลงและเหมาะสมกับโครงสร้างแบบ 4ACS

4.1 โครงสร้างและการทำงานของชุดถอดรหัส Viterbi

โครงสร้างโดยทั่วไปของชุดถอดรหัส Viterbi นั้น สามารถแบ่งได้เป็น 3 ส่วน ตามลักษณะการทำงานในการถอดรหัสของ Viterbi Algorithm ได้แก่ ภาควิชา Branch Metric Unit (BMU) ภาควิชา Add-Compare-Select Unit (ACSU) และภาควิชา Survivor Memory Unit (SMU) ซึ่งบล็อกไดอะแกรมแสดงการทำงานของชุดถอดรหัส Viterbi แสดงดังรูปที่ 4.1 โครงสร้างและการทำงานในส่วนต่างๆของชุดถอดรหัส Viterbi นี้ สามารถอธิบายได้ดังต่อไปนี้

4.1.1 โครงสร้างภาควิชา Branch Metric Unit (BMU)

ภาควิชา BMU เป็นส่วนที่ทำหน้าที่คำนวณค่า Branch Metric (BM) จากข้อมูลอินพุตที่รับเข้ามา ซึ่งเป็นลำดับแรกของการถอดรหัสด้วยวิธีของ Viterbi Algorithm โดยข้อมูลอินพุตนี้จะถูกจัดระดับสัญญาณ (Quantizer) และทำให้สัมพันธ์ (Synchronizer) กับการทำงานของชุดถอดรหัสก่อนการคำนวณค่า Branch Metric เป็นการคำนวณค่าความแตกต่างของข้อมูลอินพุตกับค่าเอาต์พุตของการเข้ารหัสที่กำหนดไว้ตามแผนภาพ Trellis (Branch Word) แสดงดังสมการ

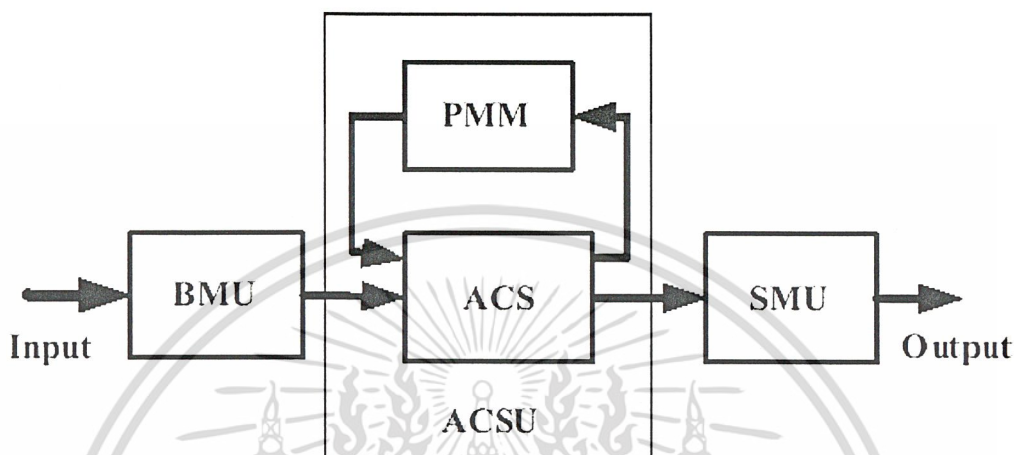
$$BM_{i,j,n} = (r_n - C_{i,j})^2 \quad (4.1)$$

โดย ค่า BM แทนค่า Branch Metric r แทนค่าข้อมูลอินพุตและ C แทนค่าเอาต์พุตของการเข้ารหัสระหว่างสเตต i ไปยังสเตต j ณ เวลา n ซึ่งค่า Branch Metric เกิดจากค่า Constraint Length ค่า Generator Polynomial และค่าอัตราการใช้รหัส (Coding Rate) การคำนวณค่า Branch Metric นี้ต้องคำนวณทุกๆ สาขา (Branch) ตามแผนภาพ Trellis การทำงานของภาควิชา BMU นั้นแบ่งตามรูปแบบในการถอดรหัสแบบ Viterbi ซึ่งมีอยู่ 2 รูปแบบ คือ Hard Decision และ Soft Decision ในการทำงานหาค่า Branch Metric แบบ Hard Decision ใช้วิธี “Hamming Distance” ซึ่งเป็นการเปรียบเทียบข้อมูลที่รับเข้ามากับค่าเอาต์พุตของการเข้ารหัสที่กำหนดไว้ในแผนภาพ Trellis (Branch Word) โดยค่า Branch Metric ซึ่งเป็นความแตกต่างข้อมูล

อินพุตกับค่า Branch Word เช่น กรณีอัตราการใช้รหัสเป็น $1/2$ ข้อมูลอินพุตเป็น “00” ค่า Branch Word เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

“11” จะได้ค่า Branch Metric เท่ากับ 2 เป็นต้น ส่วน Soft Decision ใช้วิธี “Euclidean Distance” ซึ่งเป็นการเปรียบเทียบข้อมูลที่รับเข้ามาที่ถูกจัดระดับมากกว่าสองระดับกับค่า Branch Word เช่น กรณี Soft Decision แบบ 3 บิตและอัตราการเข้ารหัส 1/2 ข้อมูลอินพุตที่รับเข้ามาเป็น



BMU: Branch Metric Unit

ACS: Add-Compare-Select

PMM: Part Metric Memory

ACSU: Add-Compare-Select-Unit

SMU: Survivor Memory Unit

รูปที่ 4.1 บล็อกไดอะแกรมของชุดถอดรหัส Viterbi

“0000” ค่า Branch Word “11” การหาค่า Branch Metric นี้จะนำค่า “000” มาเปรียบเทียบกับค่า Branch Word ที่ละบิต ซึ่งกำหนดให้เท่ากับจำนวนของ Soft Decision ในที่นี้คือ “111” ซึ่งความแตกต่างสูงสุดคือ 7 ดังนั้นค่า Branch Metric ของกรณีนี้เท่ากับ 14 โดยทั่วไปแล้วเทคนิคการออกแบบภาค BMU มีอยู่ 2 วิธี คือ

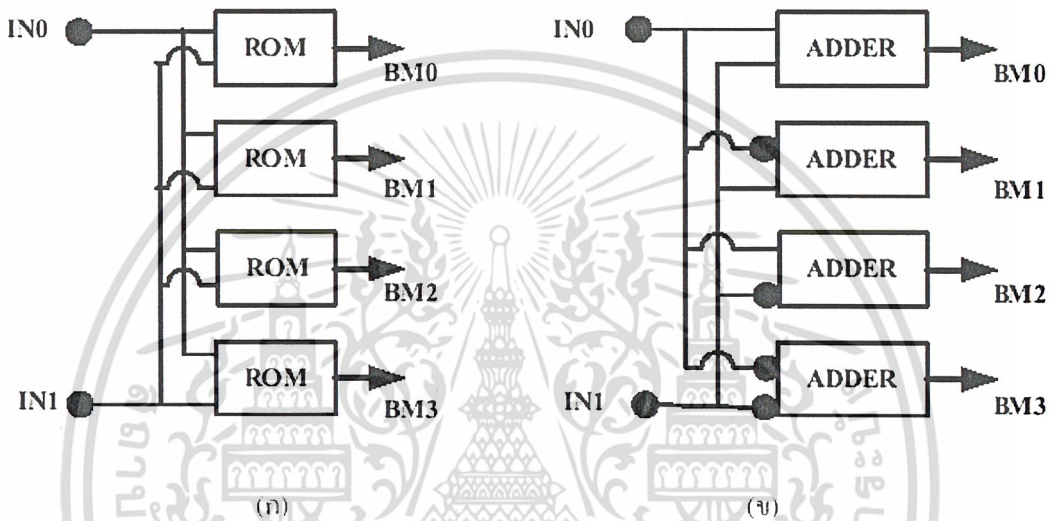
- วิธีการเปิดตาราง (ROM Look-up Table) เป็นวิธีการหาค่า Branch Metric โดยใช้ค่าอินพุตเลือกค่า Branch Metric ที่มีการกำหนดก่อนแล้วใน ROM
- วิธีการบวก (Half Adder) เป็นวิธีการหาค่า Branch Metric โดยการบวกค่าอินพุตที่รับเข้ามาเข้าด้วยกันซึ่งวงจรการบวกจะกำหนดตามแผนภาพ Trellis ซึ่งวิธีการเปิดตารางและการบวกของอัตราการเข้ารหัสเท่ากับ 1/2 แสดงดังรูปที่ 4.2

โครงสร้างวิธีการเปิดตารางนั้นให้ง่ายที่ขนาดเล็กกว่าวิธีการบวก ซึ่งวิธีการนี้เหมาะสมกับการทำงานแบบ Hard Decision เนื่องจากการใช้งานวิธีการเปิดตารางกับแบบ Soft Decision นั้นมีความซับซ้อนจึงนิยมใช้วิธีการบวกมากกว่า วิธีการเปิดตารางนั้นมีโครงสร้างที่เหมาะสมสำหรับตัวถอดรหัสแบบขนาน เนื่องจากค่า Branch Metric ที่ส่งให้ภาค ACSU นั้นจะคงที่ไม่มีเปลี่ยนแปลง ส่วนแบบเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่นิยมนำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

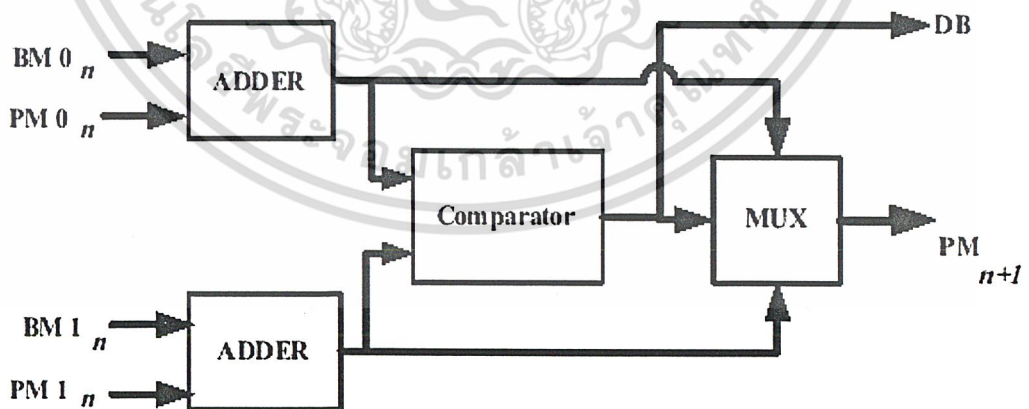
วิธีการบวกนั้นเหมาะสมสำหรับชุดถอดรหัสแบบอนุกรมเนื่องจาก Branch Metric ใน Add-Compare-Select (ACS) จะเปลี่ยนแปลงตามค่าสเตทของการทำงานของชุดถอดรหัส

4.1.2 โครงสร้างภาค Add-Compare-Select Unit (ACSU)

ภาค ACSU เป็นภาคที่ทำงานต่อจากภาค BMU ซึ่งภาค ACSU นี้ทำหน้าที่คำนวณค่า Path Metric (PM) หรือเรียกอีกอย่างหนึ่งว่าค่า State Metric (SM) และ Decision Bit (DB) ซึ่งต้องคำนวณทุกสเตทตามแผนภาพ Trellis



รูปที่ 4.2 โค้ดแแกรมภาค BMU (ก) วิธีการเปิดตาราง (ข) วิธีการบวกที่ R เท่ากับ 1/2



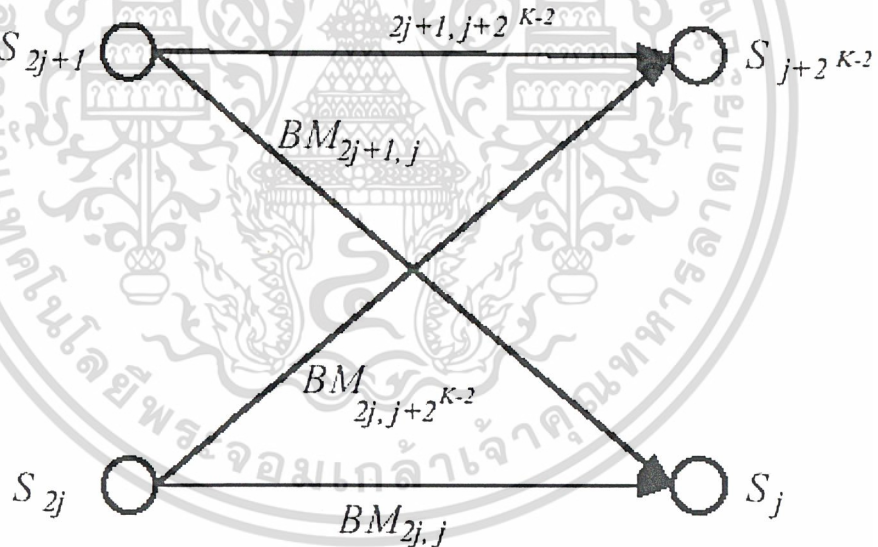
รูปที่ 4.3 โค้ดแแกรม Add-Compare-Select (ACS)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วิธีการหาค่า Path Metric และ Decision Bit แต่ละสเตทนั้นจะทำโดย Add-Compare-Select (ACS) ซึ่งนำเอาค่า Branch Metric จาก BMU และค่า Path Metric หรือค่า State Metric ที่ถูกเก็บไว้ใน Path Metric Memory (PMM) มาบวกเข้าด้วยกัน ซึ่งการบวกนั้นจะทำทั้งสองค่าตามเส้นทางของแผนภาพ Trellis โดยค่า Path Metric ใหม่ (Update Path Metric) และ Decision Bit นั้นเลือกจากผลบวกของเส้นทางที่มีค่าน้อยกว่า ซึ่งการหา Path Metric ใน ACSU แสดงดังสมการ

$$PM_{n+1}^0 = \min(PM_n^0 + BM_n^0, PM_n^1 + BM_n^1) \quad (4.2)$$

โดยที่ PM_{n+1}^i ($i = 0,1$) และ BM_{n+1}^i ($i = 0,1$) คือค่า Path Metric และ Branch Metric ที่เวลา n ตามลำดับ หลังจากนั้น Decision Bit ถูกส่งไปยัง SMU เพื่อใช้ในการถอดรหัส ส่วนค่า Path Metric ที่เลือกไว้จะเก็บไว้ใน PMM สำหรับใช้คำนวณครั้งต่อไป การทำงานของวงจรดังแสดงในรูปที่ 4.3



รูปที่ 4.4 โครงสร้าง ACS แบบ Butterfly

โครงสร้างการทำงานของ ACSU แบ่งตามโครงสร้างของชุดถอดรหัส Viterbi ซึ่งมีสองประเภท คือ โครงสร้างแบบอนุกรม (Serial Architecture) และโครงสร้างแบบขนาน (Parallel Architecture)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- โครงสร้างแบบอนุกรม (Serial Architecture)

โครงสร้างการทำงานของ ACSU นี้ประกอบด้วย ACS ตัวเดียวหรือเท่ากับ n โดย n เท่ากับ 0,1,2 โดยการเพิ่มจำนวนของ ACS จะทำให้อัตราเร็วในการถอดรหัสของชุดถอดรหัสแบบอนุกรมเพิ่มขึ้น แต่การเพิ่มจำนวนของ ACS จะต้องไม่เท่ากับจำนวนเท่ากับสเตทของการทำงาน โดยการทำงานของ ACSU แบบนี้ใช้ Random Access Memory (RAM) ในการเก็บข้อมูลต่างๆไว้โดยมีภาค Controller เป็นตัวควบคุมการทำงาน โครงสร้างแบบนี้จะให้อัตราเร็วในการถอดรหัสต่ำ แต่ข้อดีของตัวถอดรหัสจะมีขนาดเล็ก และโครงสร้างนี้เหมาะสมสำหรับการสร้างตัวถอดรหัส Viterbi ที่มีค่า Constraint Length สูง

- โครงสร้างแบบขนาน (Parallel Architecture)

โครงสร้างแบบนี้ประกอบไปด้วย ACS เท่ากับจำนวนเท่ากับสเตทของการทำงาน เช่น ถ้าค่า Constraint Length เท่ากับ 9 จะใช้ 256 ACS และนิยมใช้โครงสร้างแบบ Butterfly ซึ่งจะนำ 2 ACS มาต่อร่วมกันดังแสดงดังรูปที่ 4.4 โดยค่า Branch Metric ในโครงสร้างแบบ Butterfly จะเท่ากันซึ่งแสดงดัง

$$BM(j, 2j) = BM(j + 2^{k-2}, 2j + 1) \quad (4.3)$$

$$BM(j, 2j + 1) = BM(j + 2^{k-2}, 2j) = -BM(j, 2j) \quad (4.4)$$

โดย i และ j เป็นลำดับของสเตทการทำงานหรือลำดับของ ACS ที่นำมาต่อร่วมกัน ส่วน BM เป็นค่า Branch Metric และ K เป็นค่า Constraint Length โครงสร้างนี้จะทำให้ลดการเชื่อมต่อลงจึงทำให้พื้นที่ในการสร้างชุดถอดรหัสลดลง และทำให้การใช้พลังงานลดลงด้วย ข้อดีของแบบขนานคือให้อัตราเร็วของข้อมูลที่สูง แต่ขนาดของวงจรมีขนาดใหญ่ ซึ่งโครงสร้างแบบขนานเหมาะสำหรับการสร้างตัวถอดรหัส Viterbi ที่มีค่า Constraint Length ต่ำเท่านั้น

4.1.3 โครงสร้างภาค Survivor Memory Unit (SMU)

ภาค SMU เป็นภาคสุดท้ายในการทำงานของ Viterbi Algorithm โดยภาค SMU ทำหน้าที่ถอดรหัสข้อมูลจากข้อมูลอินพุตให้เป็นข้อมูลที่ถูกต้อง โดยการหาเส้นทางที่ของข้อมูลในแผนภาพ Trellis ในการทำงานถอดรหัสจะใช้ Decision Bit จาก ACSU เป็นตัวตัดสินใจโดยทั่วไปแล้วการทำงานของภาค SMU มี 2 วิธีคือ วิธี Register Exchange (RE) และ วิธี Trace Back (TB) ซึ่งรายละเอียดมีดังนี้

วิธีทำงานแบบ Register Exchange

การทำงานของภาค SMU วิธี Register Exchange เป็นแบบ Forward ดังนั้นจึงสามารถทำงานถอดรหัสข้อมูลได้อย่างต่อเนื่องและรวดเร็ว โดยโครงสร้างประกอบด้วยมัลติเพล็กซ์ (Multiplex) แบบ 2:1 และ รีจิสเตอร์ (Register) โดยมัลติเพล็กซ์และรีจิสเตอร์จะถูกเชื่อมต่อกันเหมือนกับแผนภาพ Trellis โดยมัลติเพล็กซ์จะถูกควบคุมการทำงานโดย Decision Bit จากภาค ACSU ซึ่งลักษณะวงจรแสดงดังรูปที่ 4.5 โดยจำนวนของ Register ต้องมากกว่า 6 เท่าของค่า Constraint Length จึงจะทำให้ข้อมูลที่ถูกละทิ้งมีความถูกต้อง ซึ่งกรณีนี้ค่า Register Constraint Length เท่ากับ 9 จำนวนของ Register ต้องมากกว่า 54

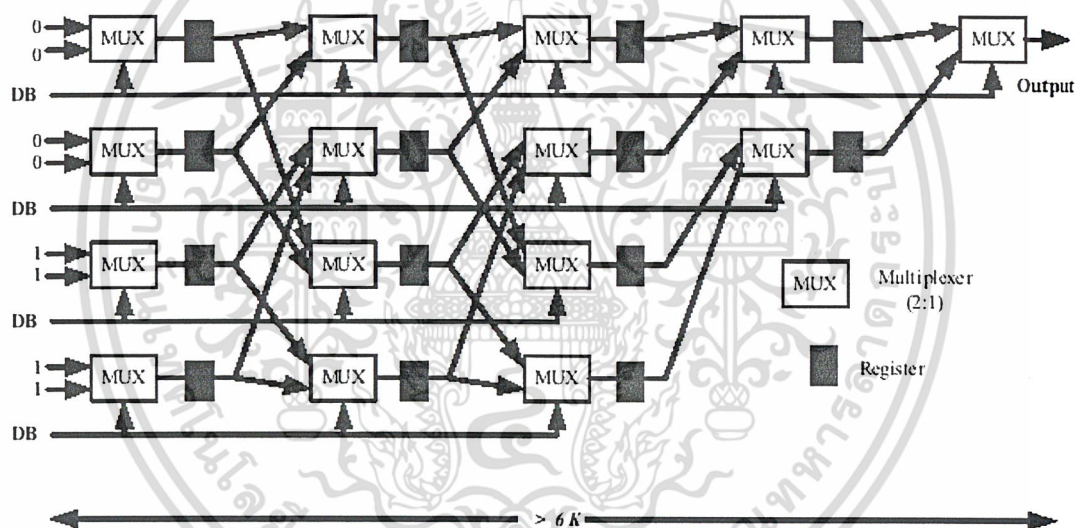
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปจะเห็นว่าวิธี Register Exchange มีโครงสร้างที่ไม่ซับซ้อน แต่มีโครงสร้างที่ใหญ่สำหรับชุดถอดรหัสที่มีค่า Constraint Length สูง และมีการใช้พลังงานมากเนื่องจากในการทำงานนั้นต้องมีการส่งข้อมูลให้กันตลอดเวลา

วิธีการทำงานแบบ Trace Back

การทำงานของภาค SMU วิธี Trace Back จะเป็นการทำงานแบบ Backward ซึ่งการถอดรหัสเริ่มต้นจากจุดที่เวลามากกว่าหกเท่าของค่า Constraint Length เหมือนกับการทำงานของวิธี Register Exchange ซึ่งเรียกค่านี้ว่า “Trace Back Depth” และการเริ่มต้นจะเริ่มจากค่าสเตทการทำงานที่มีค่า Path Metric น้อยที่สุด แล้วทำการย้อนกลับหาค่าเส้นทางของข้อมูลตามแผนภาพ Trellis โดยใช้ Decision Bit จาก ACSU ตัดสินใจในการหาเส้นทาง โดยตัวอย่างวงจรการทำงานแสดงดังรูปที่ 4.6 ซึ่งเป็นวงจร Trace Back ที่ค่า Constraint Length เท่ากับ 3 โดยวงจรประกอบด้วยหน่วยความจำขนาด 16×4 ซึ่งทำหน้าที่บันทึกค่า Decision Bit เมื่อหน่วยความจำเก็บ

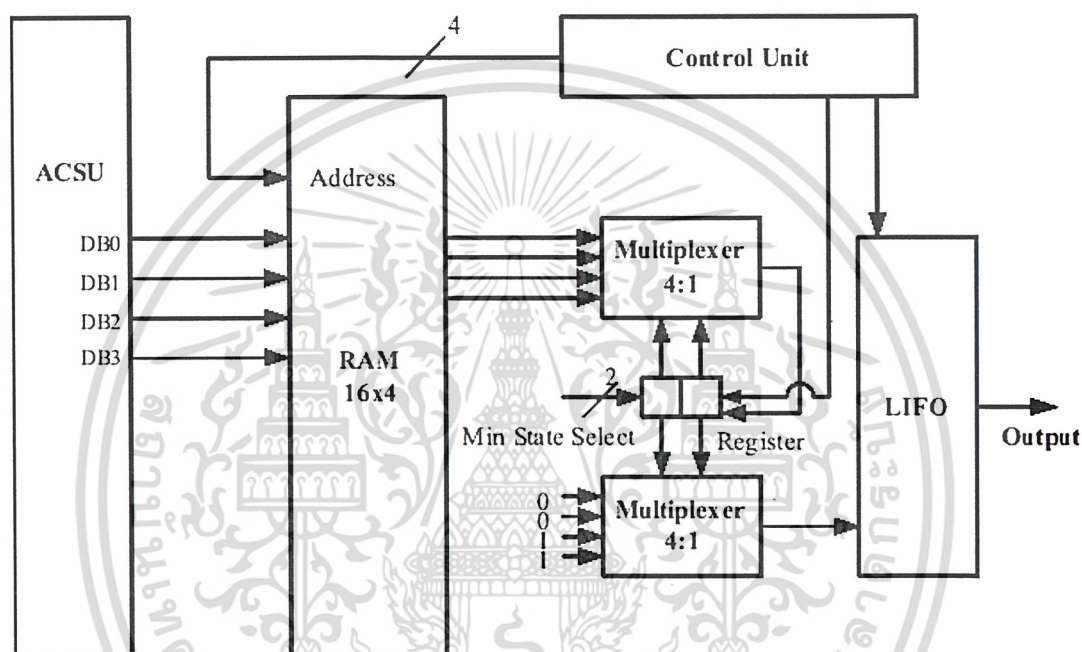


รูปที่ 4.5 วงจร SMU แบบ Register Exchange ที่ค่า K เท่ากับ 3

Decision Bit ครบ 16 ค่า ซึ่งมากกว่าหกเท่าของค่า Constraint Length ก็จะเริ่มทำการถอดรหัสหรือ Trace Back โดยเริ่มจากการหาค่า Path Metric ที่น้อยที่สุดหรือเรียกว่า “Best State” มาเป็นจุดเริ่มต้น โดยภาค Min Path Metric Selection ส่วนตัดสินใจเลือกเส้นทางจะใช้มัลติเพล็กซ์แบบ 4: 1 โดยเลือกจาก Decision Bit ที่ส่งจากหน่วยความจำ โดยสัญญาณการควบคุมมาจากข้อมูลที่ถูกเลือกโดยมัลติเพล็กซ์ในเวลาก่อนนั้น ซึ่งอยู่ในชิพรีจิสเตอร์ข้อมูลที่ถูกเลือกนี้จะเป็นข้อมูลที่ถูกถอดรหัสแต่ยังเป็นข้อมูลแบบย้อนกลับจึงจำเป็นต้องใช้ LIF (Last In First Out RAM) กลับข้อมูลให้ถูกต้อง

วิธี Trace Back นั้นเหมาะสมสำหรับชุดถอดรหัส Viterbi ที่มีค่า Constraint Length สูง เนื่องจากใช้อุปกรณ์ในการสร้างน้อย ส่วนความเร็วในการถอดรหัสจะช้ากว่า Register Exchange เนื่องจากต้องมีการหาค่า Path Metric น้อยที่สุด และต้องใช้หน่วยความจำในการทำงาน ส่วนความซับซ้อนของวงจรเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มากกว่าเนื่องจากต้องมีส่วนของภาค Controller ควบคุมการทำงาน ส่วนความเร็วในการถอดรหัสนั้นไม่สามารถทำได้ต่อเนื่องได้ เนื่องจากต้องรอ Decision Bit ที่เก็บไว้ในหน่วยความจำก่อนจึงสามารถถอดรหัสได้ โดยในการหา Path Metric ที่น้อยที่สุดนั้นเป็นวิธีการที่เหมาะสมสำหรับ ที่ชุดถอดรหัสที่มีค่า Constraint Length ขนาดเล็กเพราะการสร้าง ส่วน Min Path Metric Selection ที่มีการเปรียบเทียบจำนวนมากทำได้ยากและเกิดการซ้ำของข้อมูลมาก สำหรับการถอดรหัสแบบต่อเนื่องนั้น วงจรจะมีขนาดใหญ่และต้องใช้หน่วยความจำเพิ่มขึ้นเป็นสองเท่าเมื่อเทียบกับแบบไม่ต่อเนื่อง



รูปที่ 4.6 โค้ดแกรม SMU แบบ Trace Back ที่ค่า K เท่ากับ 3

4.1.4 การเปรียบเทียบโครงสร้างของชุดถอดรหัส Viterbi แบบต่างๆ

การสร้างชุดถอดรหัส Viterbi จากลักษณะโครงสร้างและการทำงานของชุดถอดรหัส Viterbi ที่กล่าวมาการใช้พื้นที่ (Silicon Area) การใช้พลังงาน (Power Consumption) และความเร็วในการทำงานสามารถเปรียบเทียบกันได้ดังแสดงในตาราง 4.1

ตารางที่ 4.1 ผลเปรียบเทียบข้อดีข้อเสียของโครงสร้างของชุดถอดรหัส Viterbi แบบต่างๆ

	BMU		ACSU		SMU	
	เปิดตาราง	การบวก	อนุกรม	ขนาน	Register Exchange	Trace Back
การใช้พื้นที่	น้อย	มาก	น้อย	มาก	มาก	น้อย
การใช้พลังงาน	ต่ำ	สูง	ต่ำ	สูง	สูง	ต่ำ
ความเร็ว	สูง	ต่ำ	ต่ำ	สูง	สูง	ต่ำ

4.2 การออกแบบสร้างสำหรับปริยญาณิพนธ์นี้

ชุดถอดรหัสแบบ Viterbi และ ชุดเข้ารหัส Convolution สำหรับงานวิจัยนี้ได้ออกแบบโดยรายละเอียดการออกแบบระบุไว้ดังนี้

- โครงสร้างชุดเข้ารหัสคอนโวลูชัน
มีอัตราการเข้ารหัส (R) เท่ากับ 1/2 Constraint Length เท่ากับ 3 Generator Polynomial เท่ากับ $7_8, 5_8$
- โครงสร้างภาค Branch Metric Unit (BMU)
มีการทำงานแบบ Hard Decision โดยโครงสร้างเป็นวิธีการเปิดตาราง
- โครงสร้างภาค Add-Compare-Select Unit (ACSU)
โครงสร้างเป็นแบบขนาน ACSU เท่ากับสเตทการทำงาน คือ 4 ACSU โดยใช้โครงสร้างขนานแบบ Butterfly โดยจะนำ 2 ACS มาต่อร่วมกัน
- โครงสร้างภาค Survivor Memory Unit (SMU)
ใช้วิธีทำงานแบบ Register Exchange ซึ่งสามารถทำงานถอดรหัสข้อมูลได้อย่างต่อเนื่องและรวดเร็ว

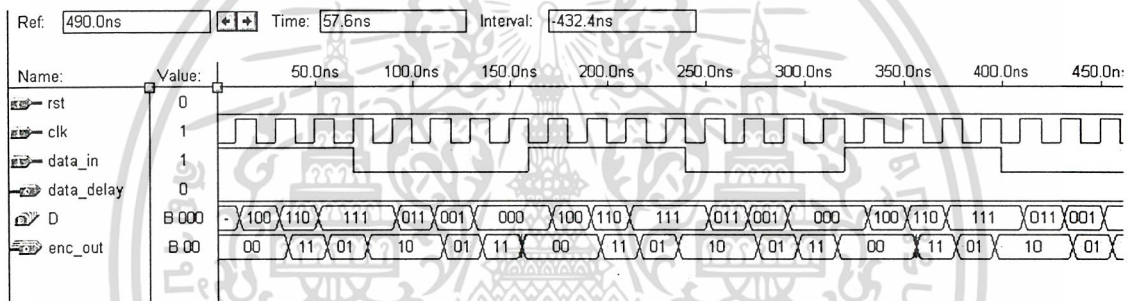
บทที่ 5

ผลการออกแบบและทดสอบการทำงาน

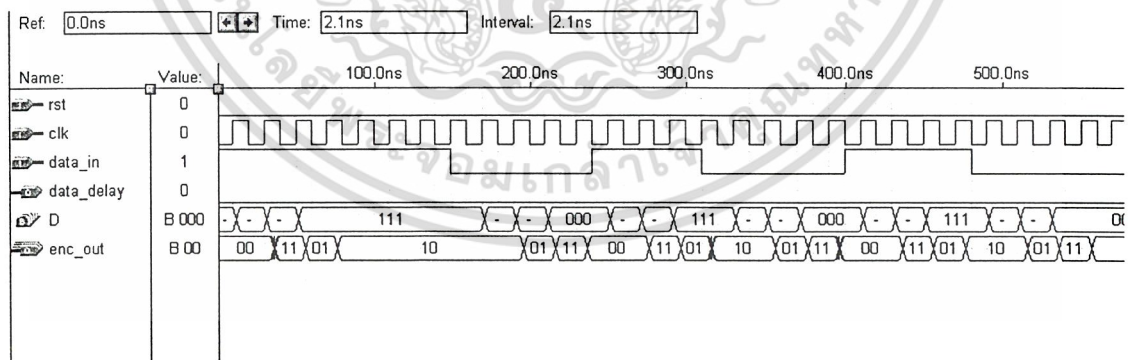
การออกแบบชุดถอดรหัส Viterbi และชุดเข้ารหัส Convolution สำหรับปริิณญาณิพนธ์นี้ใช้ภาษาเวริลล็อก (Verilog HDL) ในการบรรยายพฤติกรรมการทำงานของวงจรชุดเข้ารหัสและถอดรหัสสำหรับติดตั้งใน FPGA Xilinx โดยใช้โปรแกรม ในการออกแบบสังเคราะห์วงจร จำลองการทำงานและตรวจสอบ ในบทนี้จะแสดงถึงการออกแบบและทดสอบการทำงาน โดยแสดงผลจำลองการทำงานของชุดเข้ารหัส และ ชุดถอดรหัส โดยแบ่งตามภาคการทำงาน

5.1 มอดูลชุดเข้ารหัส

5.1.1 ผลจำลองการทำงานของมอดูลชุดเข้ารหัส



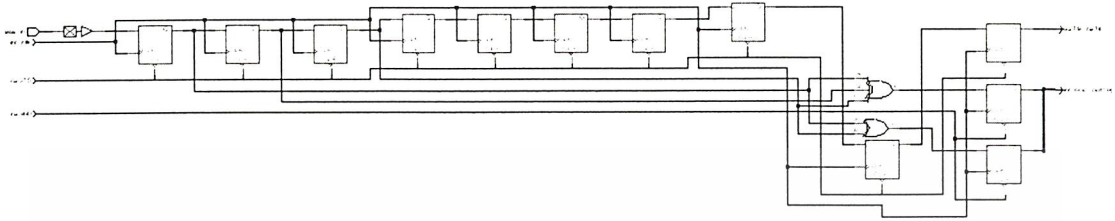
รูปที่ 5.1 แสดงการจำลองทำงานของโปรแกรมเข้ารหัสเมื่ออินพุต (11110000)



รูปที่ 5.2 แสดงการจำลองทำงานของโปรแกรมเข้ารหัสเมื่ออินพุต (111111100001111)

การจำลองการทำงานของมอดูลชุดเข้ารหัสนี้ อาศัยการ Simulate ในโปรแกรม MAX + pluse II แสดงดังรูปที่ 5.1และ 5.2 ซึ่งสัญญาณทดสอบ ประกอบด้วยสัญญาณนาฬิกาซึ่งกำหนดเป็น Clk สัญญาณรีเซต ซึ่งกำหนดเป็น rst สัญญาณอินพุตหรือข้อมูลที่ต้องการเข้ารหัสกำหนดเป็น data_in โดยการทดสอบเริ่มจากการป้อนสัญญาณนาฬิกา Clk เพื่อกำหนดการทำงาน ป้อน data_in ที่ต้องการเข้ารหัสเมื่อทำการเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

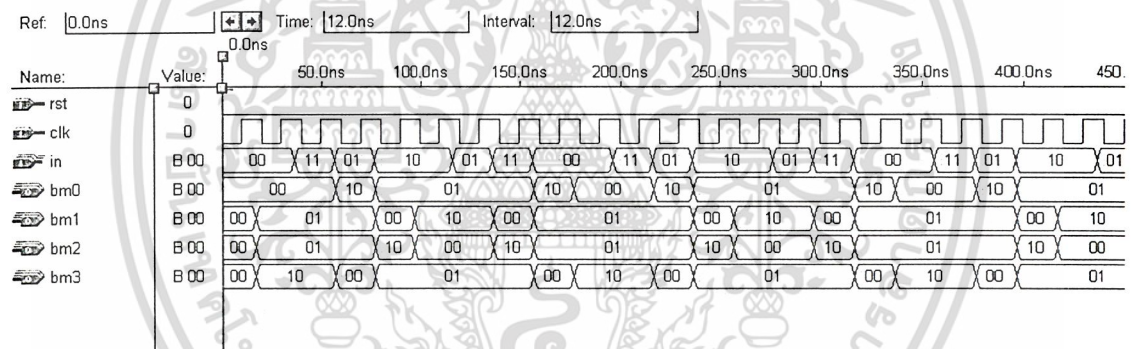
Simulate แล้ว Output ของมอดูลคือ ค่า บิตข้อมูลที่ได้จากการเข้ารหัสป้องกันการผิดพลาด คอนโวลูชัน โค้ค อัตราการเข้ารหัส 1/2 มี Constraint Length (K) เท่ากับ 3 โดยการกำหนดเป็น enc_out และข้อมูล Output ที่ถูกหน่วงเวลาไว้กำหนดเป็น data_delay โดย Output ทั้งหมดแสดงอยู่ในรูปเลขฐานสอง



รูปที่ 5.3 แสดงวงจรระดับ RTL ที่ได้จากการสังเคราะห์ชุดเข้ารหัส

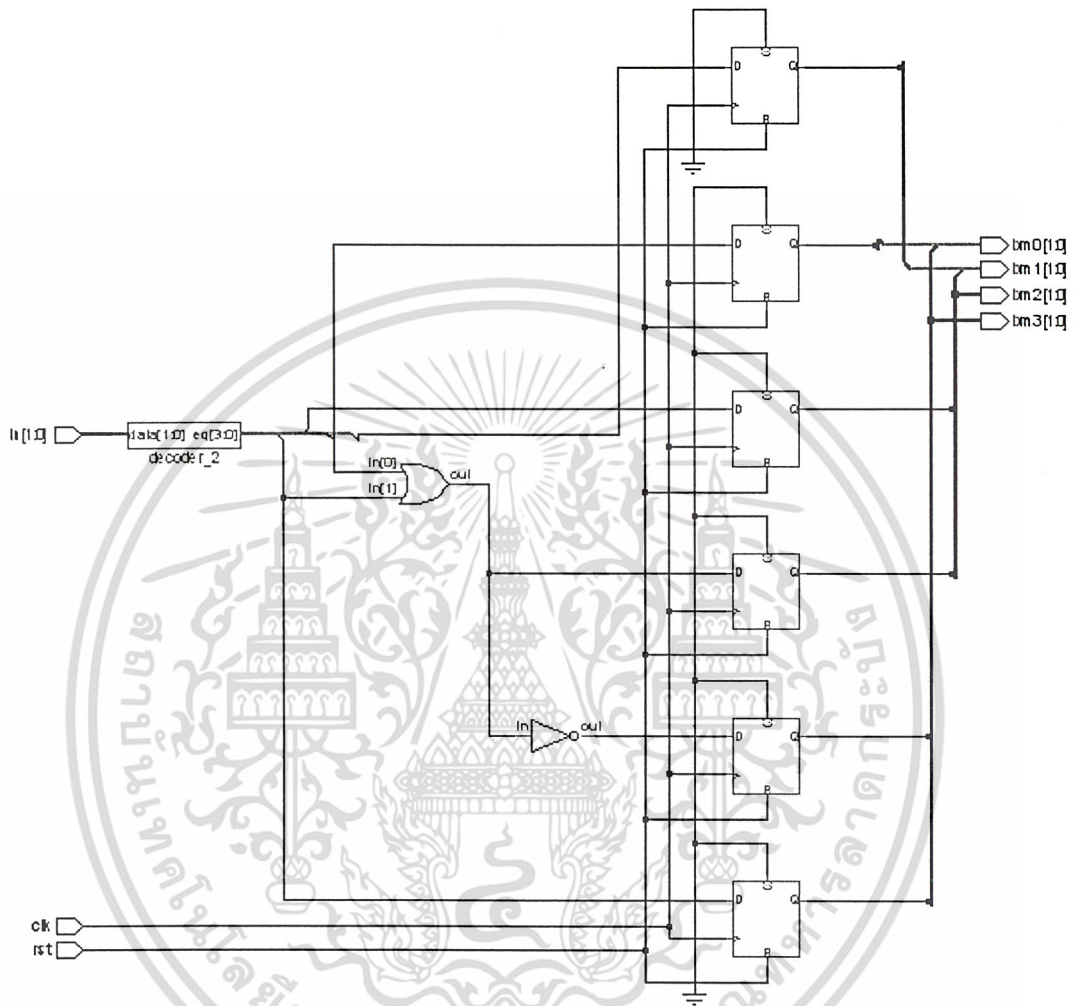
5.2 มอดูลชุดถอดรหัส

5.2.1 ผลการจำลองการทำงานของมอดูล BMU (Branch Metric Unit)



รูปที่ 5.4 แสดงจำลองการทำงานของชุด BMU

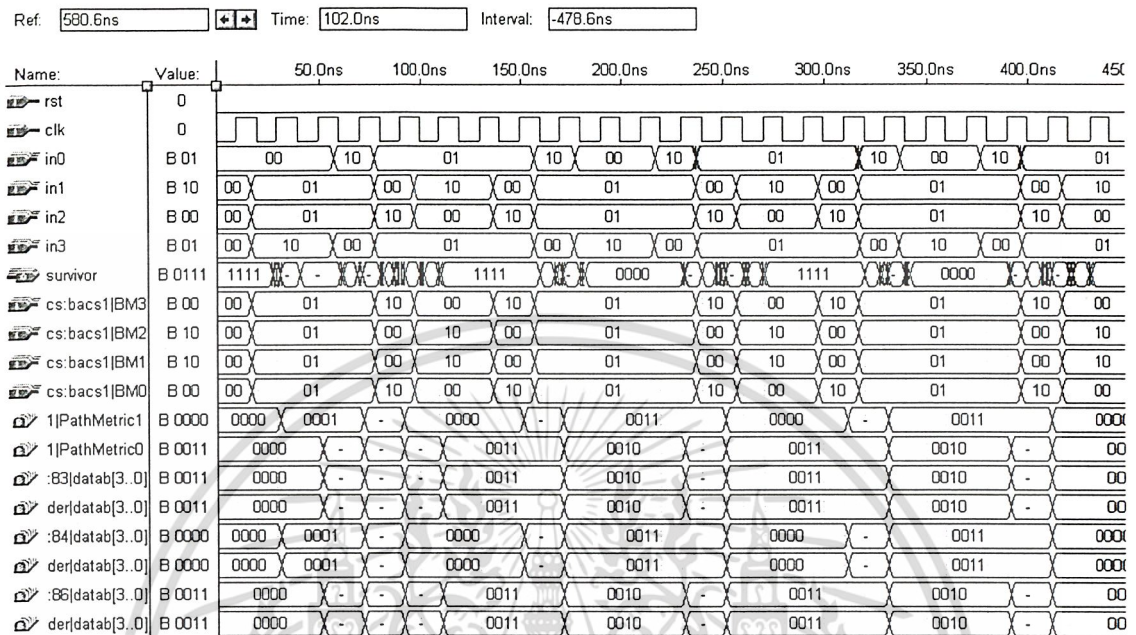
การจำลองทำงานของ BMU โดยอาศัยการ Simulate ในโปรแกรม MAX+plus II แสดง ดังรูป ที่ 5.4 ซึ่งสัญญาณทดสอบของมอดูล นี้ประกอบด้วย สัญญาณนาฬิกาที่กำหนดเป็น Clk สัญญาณรีเซ็ตซึ่ง กำหนดเป็น rst สัญญาณอินพุตที่เป็นบิตข้อมูลที่ได้จากการเข้ารหัสจากมอดูลชุดเข้ารหัสคอนโวลูชัน ที่ R=1/2 กำหนดเป็น in โดยการทดสอบเริ่มโดยการป้อนสัญญาณนาฬิกา Clk เพื่อกำหนดจังหวะการทำงานของมอดูล และป้อน in ซึ่งได้จากการเข้ารหัสและเมื่อผ่านการ Simulate แล้ว Output ของมอดูลคือค่า Branch Metric (bm0 ถึง bm3) ซึ่งแสดงเป็นเลขฐานสอง ซึ่งผลจากการจำลองการทำงานที่ได้นี้ สามารถ ตรวจสอบได้ค่าการทำงานของวงจร



รูปที่ 5.5 ฟังก์ชันแสดงวงจรระดับ RTL ที่ได้จากการสังเคราะห์ชุด BMU

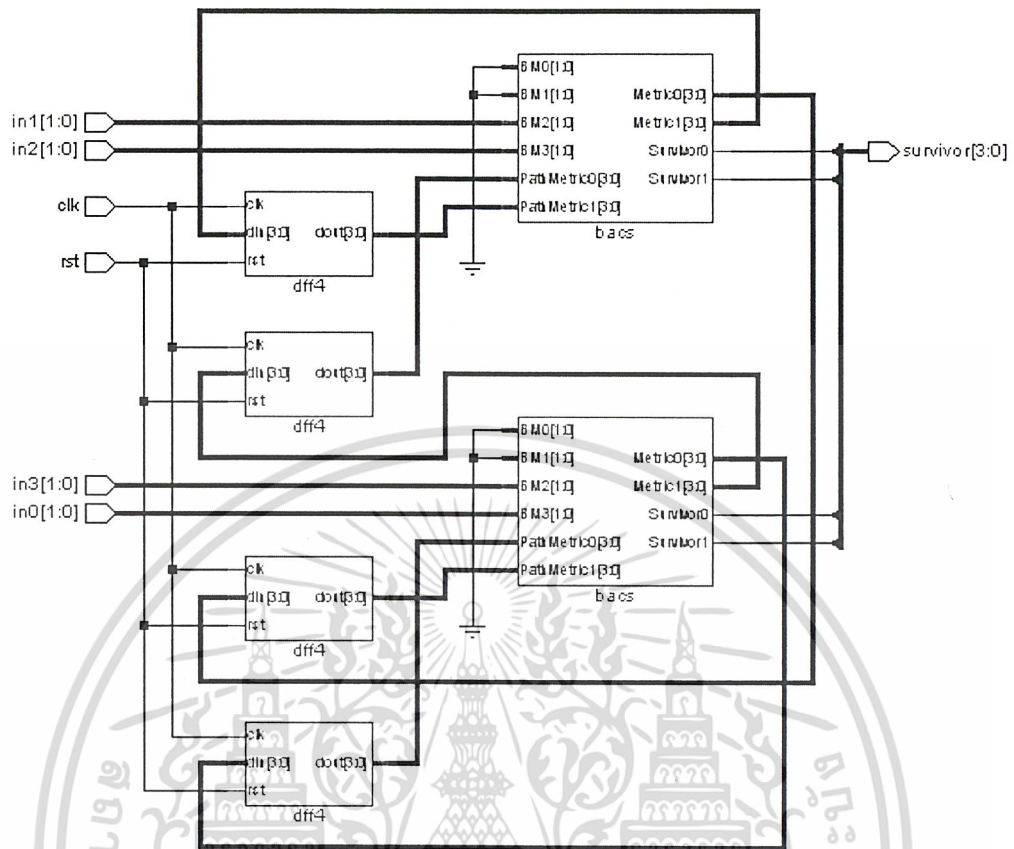
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2.2 ผลการจำลองการทำงานมอดูล ACSU (Add-Compare-Select Unit)



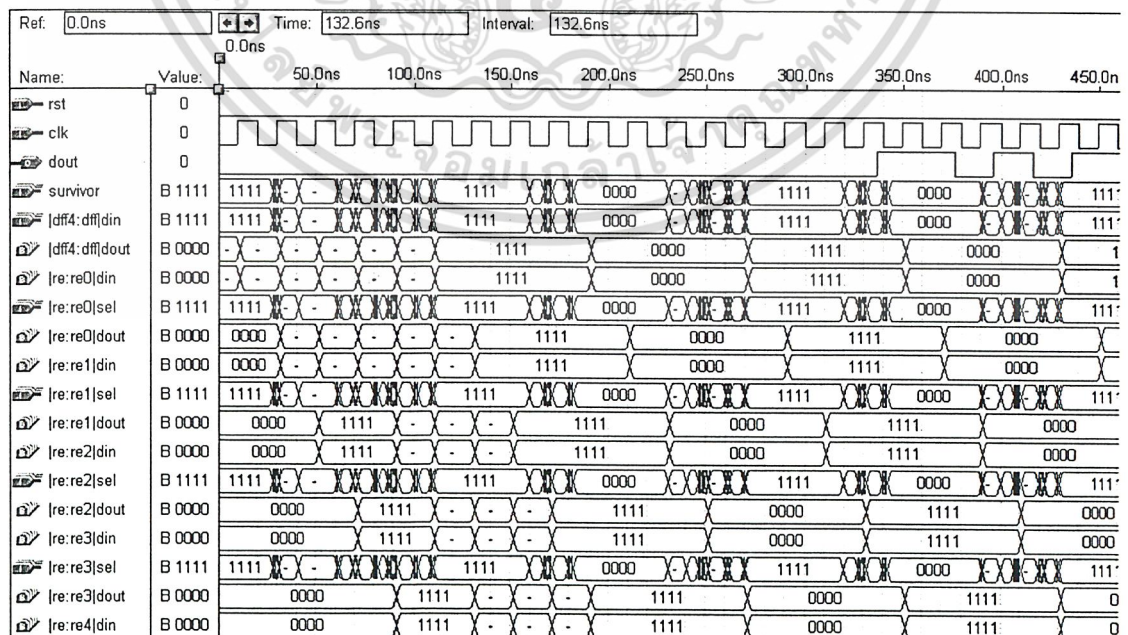
รูปที่ 5.6 แสดงจำลองการทำงานของชุด ACSU

การจำลองทำงานของ ACS โดยอาศัยการ Simulate ในโปรแกรม MAX+plus II แสดง ดังรูปที่ 5.6 ซึ่งสัญญาณทดสอบของมอดูล นี้ประกอบด้วย สัญญาณนาฬิกาที่กำหนดเป็น Clk สัญญาณรีเซ็ตซึ่งกำหนดเป็น rst สัญญาณอินพุต ที่กำหนดเป็น in0 ถึง in1 ซึ่งก็คือค่า Branch Metric (bm0 ถึง bm3) ที่ส่งต่อมาจากภาค BMU โดยการทดสอบเริ่มโดยการป้อนสัญญาณนาฬิกา Clk เพื่อกำหนดการทำงานของมอดูล และป้อนค่า Branch Metric ให้กับ in0 ถึง in3 เมื่อผ่านการ Simulate แล้ว Output ที่ได้ คือ Survivor ซึ่งแสดงเป็นเลขฐานสอง 4 bit ซึ่งก็คือค่า Decision Bit (DB) ซึ่งผลจากการจำลองการทำงานที่ได้นี้สามารถตรวจสอบได้ค่าการทำงานของวงจร



รูปที่ 5.7 แสดงวงจรระดับ RTL ที่ได้จากการสังเคราะห์ชุด ACSU

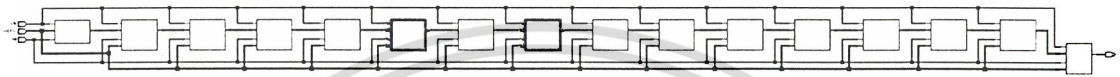
5.2.3 ผลการจำลองการทำงานมอดูล SMU(Survivor Memory Unit)



รูปที่ 5.8 แสดงจำลองการทำงานของชุด SMU

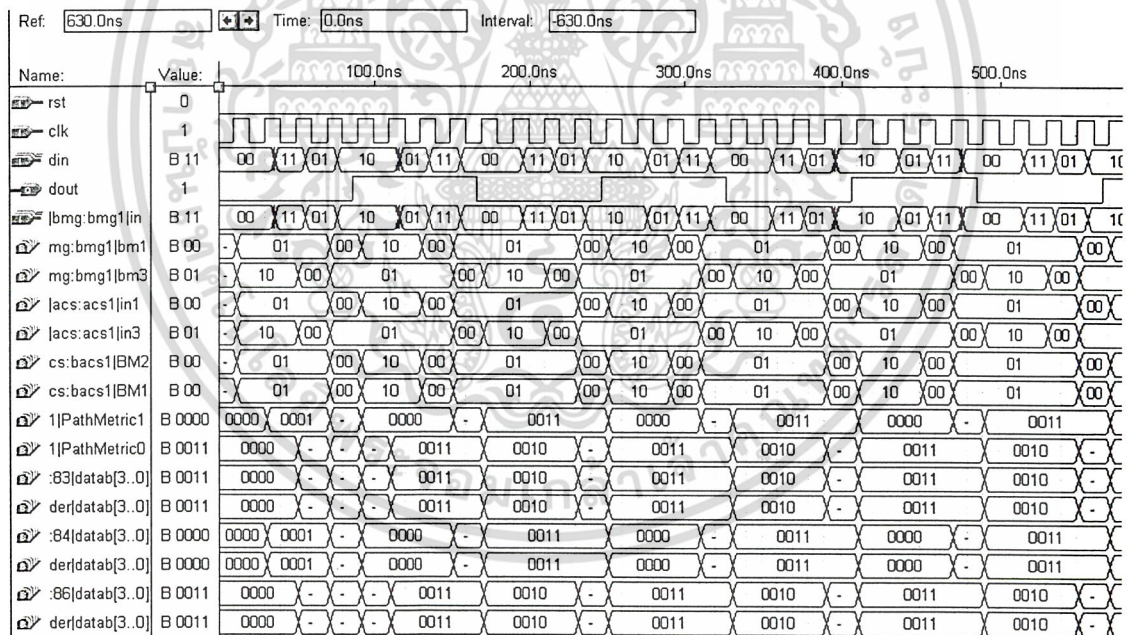
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การจำลองทำงานของ SMU โดยอาศัยการ Simulate ในโปรแกรม MAX+plus II แสดง ดังรูปที่ 5.8 ซึ่งสัญญาณทดสอบของมอดูล นี้ประกอบด้วย สัญญาณนาฬิกาที่กำหนดเป็น Clk สัญญาณรีเซ็ตซึ่งกำหนดเป็น rst สัญญาณอินพุตที่กำหนดเป็น Survivor 4 Decision Bit ซึ่งได้รับมาจาก ACSU โดยการจำลองการทำงานเริ่มจากป้อนสัญญาณนาฬิกาเพื่อกำหนดการทำงาน และป้อนค่าอินพุต Survivor เมื่อทำการ Simulate แล้ว Output ที่ได้คือ dout ซึ่งเป็นข้อมูลที่ได้จากการถอดรหัส

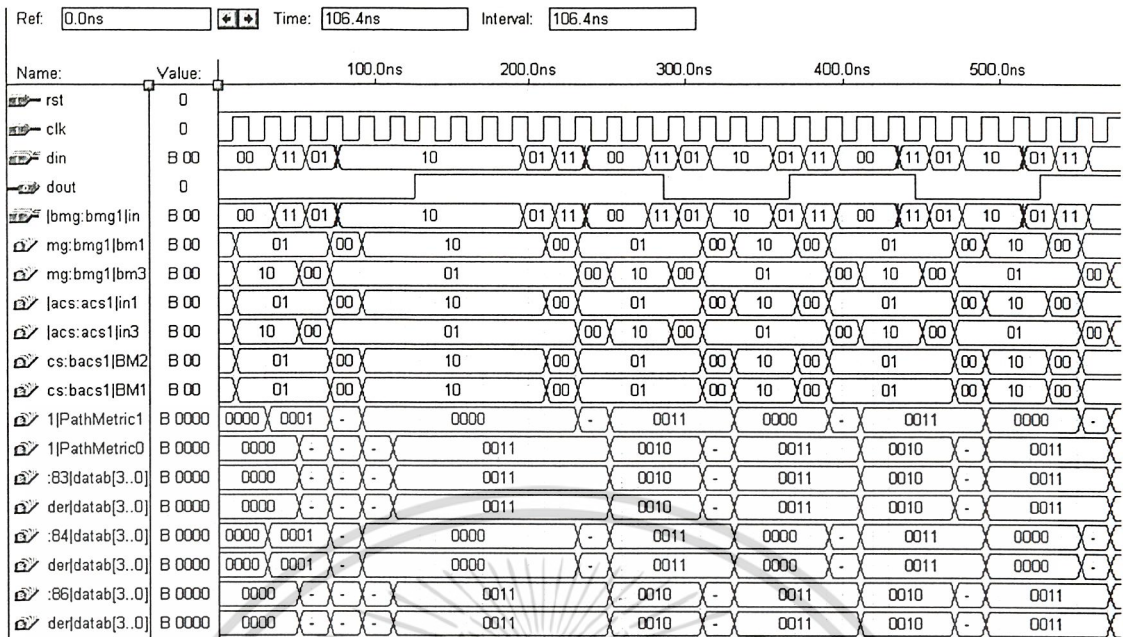


รูปที่ 5.9 ฟังก์ชันแสดงวงจรระดับ RTL ที่ได้จากการสังเคราะห์ชุด SMU

5.2.4 ผลการจำลองชุดถอดรหัส

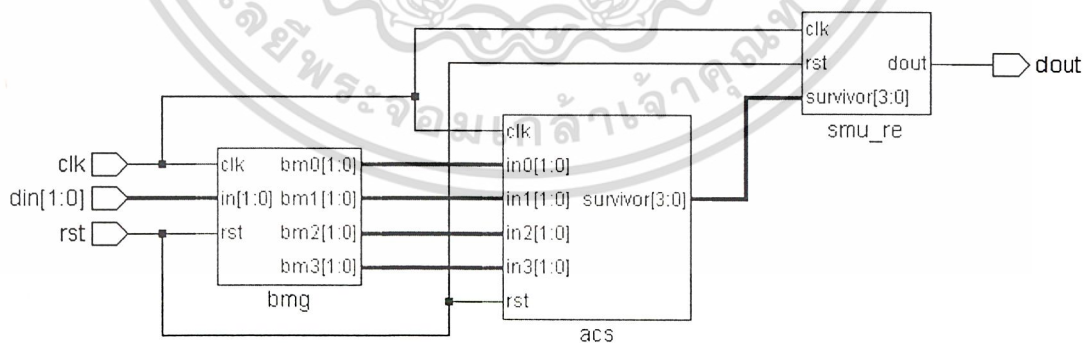


รูปที่ 5.10 แสดงการจำลองทำงานของโปรแกรมถอดรหัสเมื่อผ่านการเข้ารหัสด้วยอินพุต (11110000)



รูปที่ 5.11 แสดงการจำลองการทำงานของโปรแกรมถอดรหัสเมื่อผ่านการเข้ารหัสด้วยอินพุต (1111111100001111)

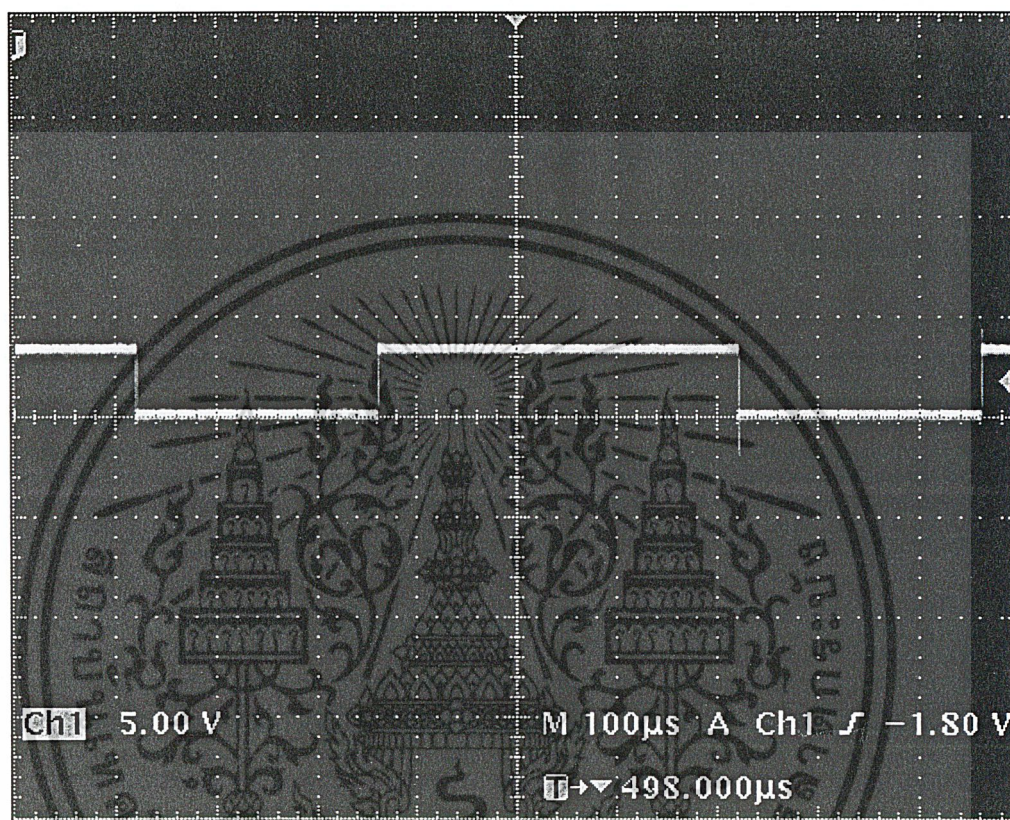
ผลจำลองการถอดรหัสที่เกิดขึ้นจากการนำหลายๆ โมดูลมาประกอบรวมกันและทำการป้อนสัญญาณอินพุตที่ได้จากการเข้ารหัสโดยชุดเข้ารหัสซึ่งจากการจำลองการเข้ารหัสผลที่ได้คือสัญญาณเอาต์พุตที่ได้มีค่าเท่ากับข้อมูลที่ป้อนให้กับชุดการเข้ารหัสจึงทำให้ผลการเข้ารหัสและถอดรหัสที่ได้ออกมาถูกต้อง



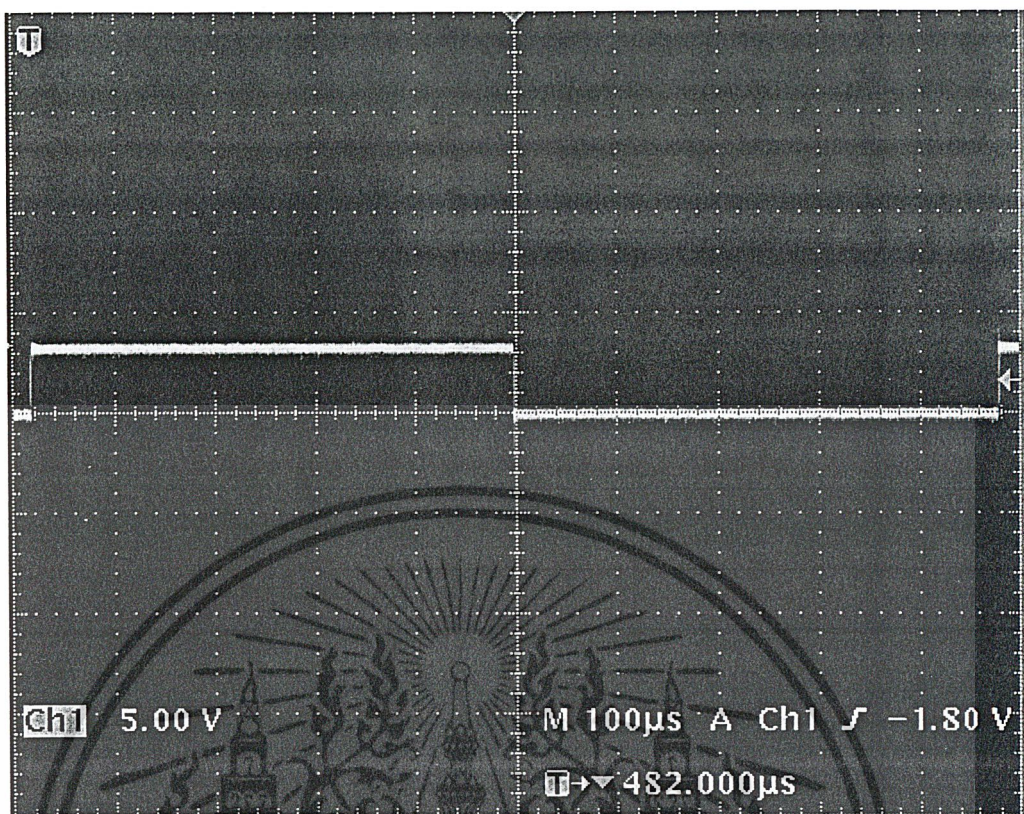
รูปที่ 5.12 ฟังก์ชันแสดงวงจรระดับ RTL ที่ได้จากการสังเคราะห์ชุดถอดรหัส

5.3 ผลการทดลองที่ได้จากการสังเคราะห์โปรแกรมการเข้ารหัสและถอดรหัสลงบนอุปกรณ์ FPGA

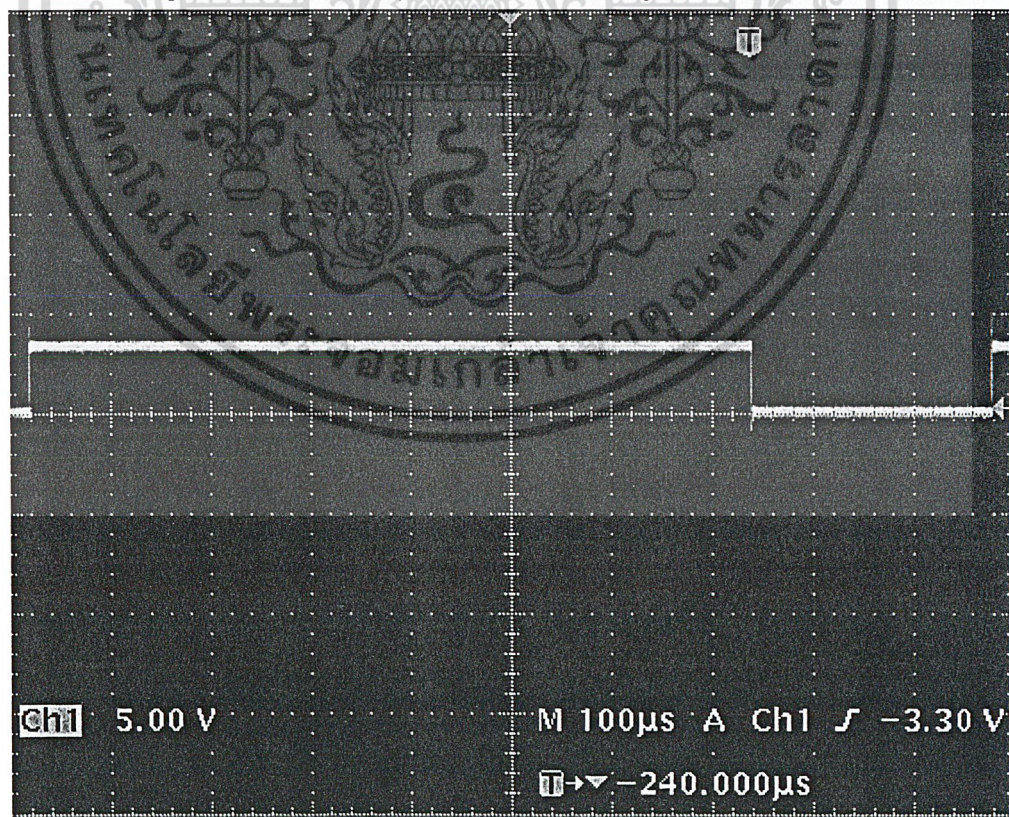
รูปแสดงผลการทดลองโดยการป้อนข้อมูลผ่านชุดเข้ารหัสและถอดรหัส ซึ่งผลที่ได้เกิดจากการป้อนข้อมูลที่ละ 8 บิต ค่าเดิมหลายๆครั้ง เพื่อที่จะทำให้สามารถแสดงผลบนสโคปได้



รูปที่ 5.13 แสดงเอาต์พุตที่ได้จากการป้อนข้อมูล (00011111)



รูปที่ 5.14 แสดงเอาต์พุตที่ได้จากการป้อนข้อมูล (11110000)

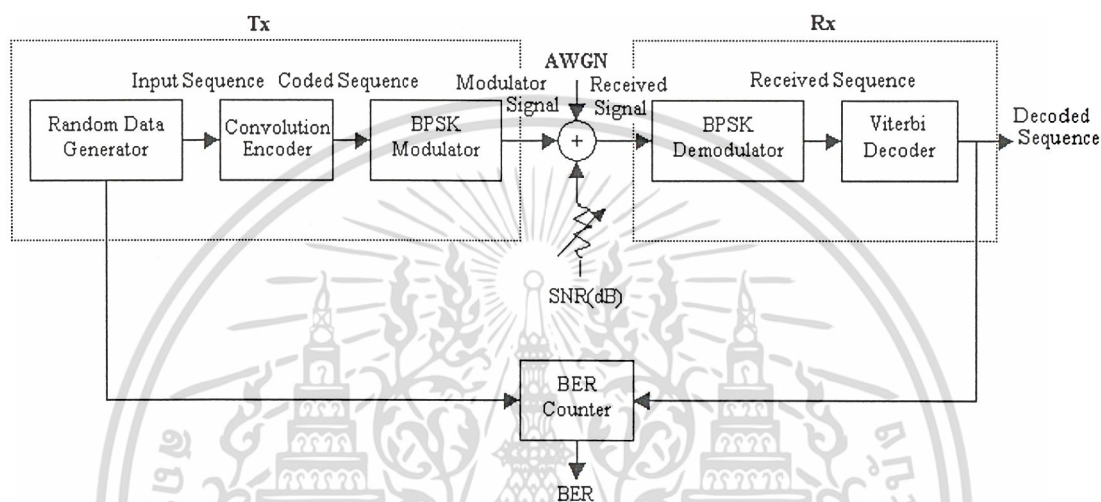


รูปที่ 5.15 แสดงเอาต์พุตที่ได้จากการป้อนข้อมูล (11111100)

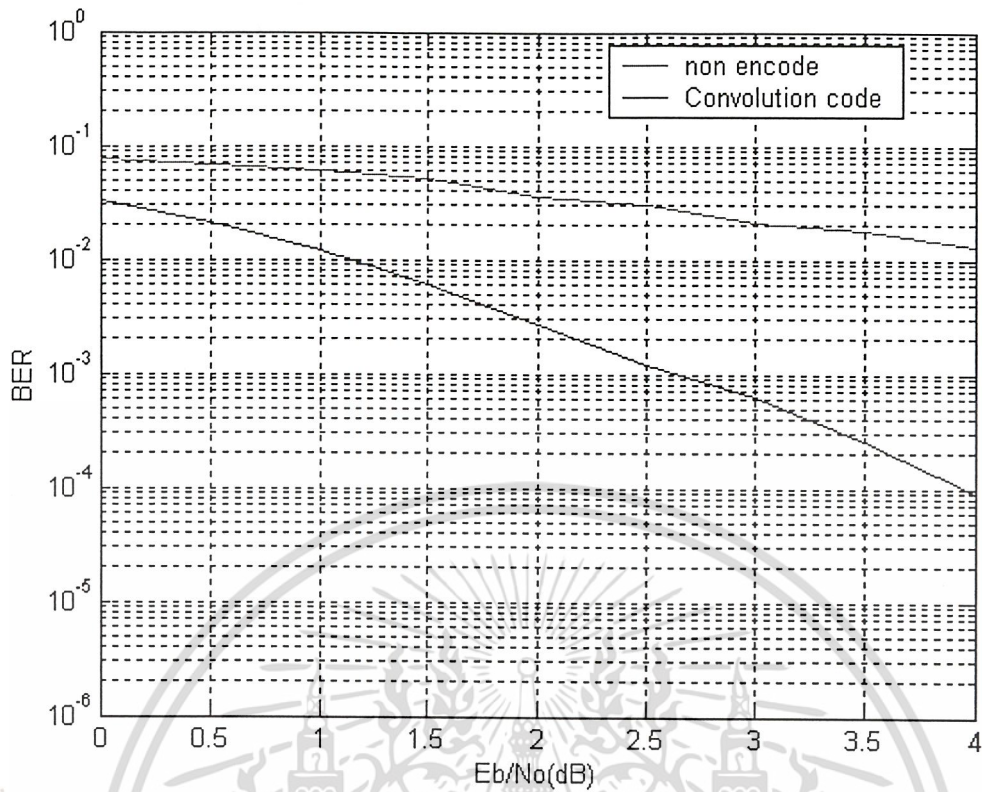
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.4 การทดสอบประสิทธิภาพของชุดเข้ารหัสและชุดถอดรหัสในช่องสัญญาณ AWGN

การทดสอบประสิทธิภาพเป็นการทดสอบหาอัตราความผิดพลาดบิต (Bit Error Rate : BER) จากผลกระทบของสัญญาณรบกวนในช่องสัญญาณการสื่อสารแบบ AWGN (Additive White Gaussian Noise) ในการทดสอบจะทำการเปลี่ยนค่าของกำลังของสัญญาณต่อสัญญาณรบกวนต่างๆ เพื่อบันทึกอัตราความผิดพลาดบิตในกรณีที่เข้ารหัสและไม่เข้ารหัส



รูปที่ 5.16 แสดงบล็อกไดอะแกรมการทดสอบประสิทธิภาพของชุดเข้ารหัสและชุดถอดรหัสในช่องสัญญาณ AWGN



รูปที่ 5.17 กราฟแสดงค่าผลของอัตราความผิดพลาดบิต (BER) กับค่ากำลังของสัญญาณต่อสัญญาณรบกวน (E_b/N_0) ที่ค่า $K=3$, $R=1/2$ เทียบกับกรณีเข้ารหัสและไม่เข้ารหัส ในช่องสัญญาณ AWGN โดยใช้โปรแกรม Matlab

บทที่ 6

สรุปผลและข้อเสนอแนะ

การเข้ารหัส คอนโวลูชันและการถอดรหัสแบบ Viterbi เป็นเทคนิคหนึ่งที่ใช้แก้ไขความผิดพลาดในการรับส่งข้อมูล ในระบบสื่อสารข้อมูลระบบดิจิทัล ซึ่งเรียกว่าการป้องกันความผิดพลาดล่วงหน้า (FEC) ซึ่งการเข้ารหัสแบบคอนโวลูชันและถอดรหัส Viterbi นี้ได้ถูกนำมาใช้งานในระบบสื่อสารดิจิทัลในปัจจุบันอย่างแพร่หลายโดยเฉพาะการสื่อสารในระบบโทรศัพท์เคลื่อนที่ ที่สามารถสื่อสารข้อมูลด้วยความเร็วสูง

สรุปผลการทดลอง

การสร้างชุดเข้ารหัสและถอดรหัสในปริิณญาณิพนธ์นี้ได้นำเสนอถึงวิธีการในการออกแบบชุดเข้ารหัส คอนโวลูชัน และการถอดรหัส Viterbi โดยในส่วนเข้ารหัสจะประกอบด้วย Shift Register และตัวมอดูโล 2 ส่วนในชุดถอดรหัสจะมีโครงสร้างแบบขนาน 4 ส่วน ACS (Add Compare Select) BMU (Branch Metric Unit) ใช้วิธีการเปิดตาราง (Rom Look Up Table) PMM (Path Metric Memory) และ SMU (Survivor Memory Unit) แบบ Register Exchange

ชุดถอดรหัส Viterbi นี้ถูกออกแบบโดยใช้ภาษา Verilog HDL และสังเคราะห์ลงบนเอฟพีจีเอตระกูล Xilinx ผลการออกแบบและจำลองการทำงานของชุดทดลองนี้ สามารถเข้ารหัสบิตข้อมูลและถอดรหัสบิตข้อมูล ได้อย่างถูกต้อง ซึ่งผลการทดลองที่ได้นี้สำเร็จตามวัตถุประสงค์ และขอบเขตของงานวิจัยที่ตั้งไว้

ปัญหาและอุปสรรค

- ขาดความรู้ความชำนาญในการใช้โปรแกรมที่จำเป็นสำหรับโครงการวิจัย
- แหล่งความรู้ข้อมูลในการค้นคว้ามีน้อย และแหล่งข้อมูลส่วนใหญ่มาจากหนังสือภาษาต่างประเทศซึ่งมีความยากในการทำความเข้าใจ
- มีเวลาน้อยในการศึกษาโปรแกรมทางภาษาที่ใช้ในการอธิบายการทำงาน โปรแกรมที่ใช้ในการตรวจสอบผลการทำงานและการทำความเข้าใจเกี่ยวกับวิธีการทำงานของชุดทดลอง

ข้อเสนอแนะในการพัฒนา

- การเข้ารหัสเพื่อป้องกันความผิดพลาดล่วงหน้า FEC แบบคอนโวลูชันโค้ด ค่า Constraint length ยิ่งสูง ประสิทธิภาพในการป้องกันก็จะดียิ่งขึ้น
- การสร้างชุดถอดรหัส Viterbi ในกรณีที่ค่า Constraint length สูงๆ โครงสร้างแบบขนานจะใช้พื้นที่และจำนวนของอุปกรณ์จำนวนมาก ทำให้ต้นทุนในการสร้างสูง

- การสร้างชุดถอดรหัส Viterbi โครงสร้างแบบอนุกรม ACS ตัวเดียวจะใช้เวลาในการทำงานเท่ากับจำนวน State คือ 2^{K-1} การเพิ่มโครงสร้าง ACS ให้มากขึ้นจะช่วยลดเวลาในการทำงานลงทำให้การทำงานในการถอดรหัสเร็วขึ้น



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก
โปรแกรมชุดเข้ารหัส

//โปรแกรมชุดเข้ารหัสที่ $R=1/2$

```
module encodeK3R12(clk,rst,data_in,enc_out,data_delay);
    input clk;
    input rst;
    input data_in;
    output data_delay;
    output [1:0] enc_out;
    reg [1:0] enc_out;
    reg [2:0] D;
    reg [36:0] SR;
    assign data_delay=SR[0];

    always@(posedge clk or posedge rst)
    begin
        if(rst)
            begin
                D<=0;
                enc_out<=0;
                SR<=0;
            end
        else
            begin
                SR<={data_in,SR[36:1]};
                D<={data_in,D[2:1]};
                enc_out[1]<=D[2]^D[1]^D[0];
                enc_out[0]<=D[2]^D[0];
            end
        end
    end
endmodule
```

โปรแกรมชุดถอดรหัส

//ท้อปมอดูลชุดถอดรหัส top_re

```
module top_re(clk,rst,din,dout);  
    input clk;  
    input rst;  
    input [1:0] din;  
    output dout;  
    wire [1:0] bm0,bm1,bm2,bm3;  
    wire [3:0] survivor;  
  
    bmg bmg1(clk,rst,din,bm0,bm1,bm2,bm3);  
    acs acs1(clk,rst,bm0,bm1,bm2,bm3,survivor);  
    smu_re smu(clk,rst,survivor,dout);  
  
endmodule
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

//ท้อปมอดูล bmg

```
module bmg(clk,rst,in,bm0,bm1,bm2,bm3);  
    input  clk;  
    input  rst;  
    input  [1:0] in;  
    output [1:0] bm0;  
    output [1:0] bm1;  
    output [1:0] bm2;  
    output [1:0] bm3;  
    reg [1:0] bm0,bm1,bm2,bm3;
```

```
always @(posedge clk or posedge rst)
```

```
if (rst)
```

```
begin
```

```
    bm0 <= 0;
```

```
    bm1 <= 0;
```

```
    bm2 <= 0;
```

```
    bm3 <= 0;
```

```
end
```

```
else
```

```
begin
```

```
    case (in)
```

```
        2'b00 : {bm0,bm1,bm2,bm3} <= 8'b000010110;
```

```
        2'b01 : {bm0,bm1,bm2,bm3} <= 8'b01001001;
```

```
        2'b10 : {bm0,bm1,bm2,bm3} <= 8'b01100001;
```

```
        2'b11 : {bm0,bm1,bm2,bm3} <= 8'b10010100;
```

```
        default : {bm0,bm1,bm2,bm3} <= 8'b0;
```

```
    endcase
```

```
end
```

```
endmodule
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

//มอดูล bacs

```
module bacs (BM3,BM2,BM1,BM0,PathMetric1,PathMetric0,Survivor1,
             Survivor0,Metric1,Metric0);
    input [1:0] BM0,BM1,BM2,BM3;
    input [3:0] PathMetric0,PathMetric1;
    output Survivor0,Survivor1;
    output [3:0] Metric0,Metric1;
    wire [3:0] ADD0, ADD1,ADD2, ADD3;

    assign ADD0 = BM0 + PathMetric0;
    assign ADD1 = BM2 + PathMetric1;
    COMPARATOR C1(Survivor0,ADD0, ADD1);
    assign Metric0 = Survivor0 ? ADD1: ADD0;
    assign ADD2 = BM1 + PathMetric0;
    assign ADD3 = BM3 + PathMetric1;
    COMPARATOR C2(Survivor1,ADD2, ADD3);
    assign Metric1 = Survivor1 ? ADD3: ADD2;

endmodule
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

//ท้อปมอดูล acs

```
module acs(clk,rst,in0,in1,in2,in3,survivor);
    input [1:0] in0;
    input [1:0] in1;
    input [1:0] in2;
    input [1:0] in3;
    output [3:0] survivor;
    input clk;
    input rst;
    wire[3:0] p0,p1,p2,p3,m0,m1,m2,m3;
    wire s0,s1,s2,s3;

    bacs bacs0(in0,in3,in3,in0,p1,p0,s2,s0,m2,m0);
    bacs bacs1(in2,in1,in1,in2,p3,p2,s3,s1,m3,m1);
    dff4 dff0(clk,rst,m0,p0);
    dff4 dff1(clk,rst,m1,p1);
    dff4 dff2(clk,rst,m2,p2);
    dff4 dff3(clk,rst,m3,p3);
    assign survivor = {s3,s2,s1,s0};

endmodule
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

//ท้อปมอดูล smu_re

```
module smu_re(clk,rst,survivor,dout);
    input clk;
    input rst;
    input [3:0] survivor;
    output dout;
    wire[3:0] din,d1,d2,d3,d4,d5,d6,d7,d8,d9,d10,d11,d12,d13,d14;
    wire[1:0] sel;

    assign sel = survivor[1:0];
    dff4 dff(clk,rst,survivor,din);
    re re0(clk,rst,din,survivor,d1);
    re re1(clk,rst,d1,survivor,d2);
    re re2(clk,rst,d2,survivor,d3);
    re re3(clk,rst,d3,survivor,d4);
    re re4(clk,rst,d4,survivor,d5);
    re re5(clk,rst,d5,survivor,d6);
    re re6(clk,rst,d6,survivor,d7);
    re re7(clk,rst,d7,survivor,d8);
    re re8(clk,rst,d8,survivor,d9);
    re re9(clk,rst,d9,survivor,d10);
    re re10(clk,rst,d10,survivor,d11);
    re re11(clk,rst,d11,survivor,d12);
    re re12(clk,rst,d12,survivor,d13);
    re re13(clk,rst,d13,survivor,d14);
    redecode redecode1(clk,rst,d14,sel,dout);

endmodule
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

//โมดูล redecode

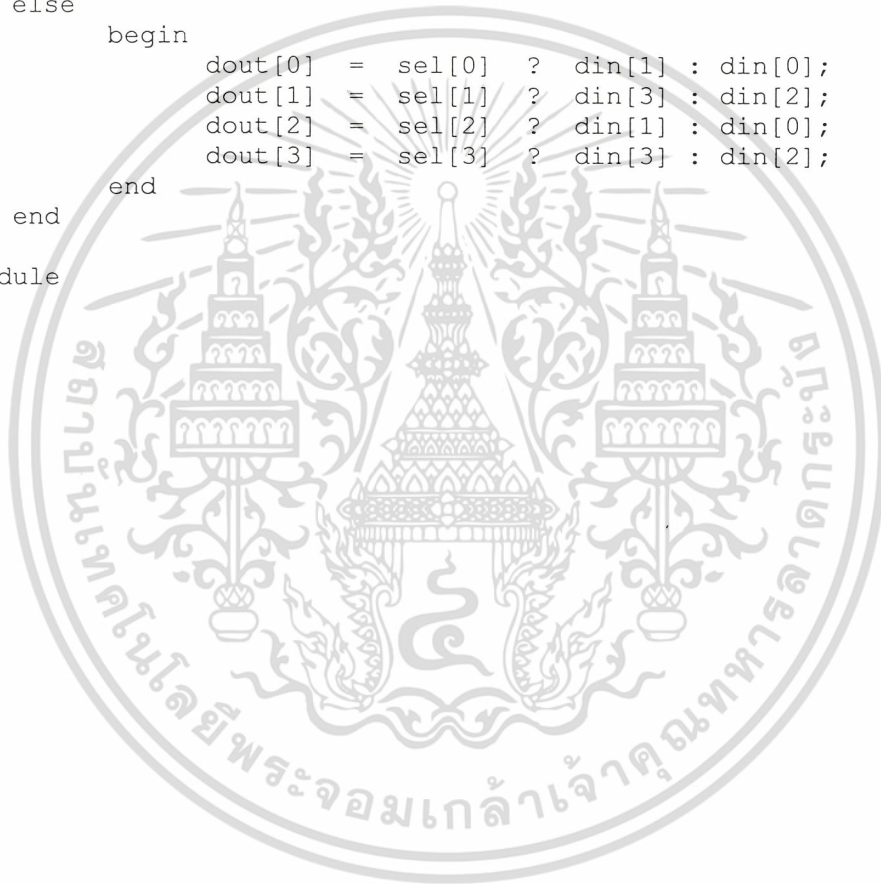
```
module redecode(clk,rst,in,sel,dout);
    input clk,rst;
    input [1:0] sel;
    input [3:0] in;
    output dout;
    reg d1,d0,dout;

    always @(posedge clk or posedge rst )
    begin
        if (rst)
            begin
                d0 <= 0;
                d1 <= 0;
                dout <= 1'b0;
            end
        else
            begin
                d0 <= sel[0] ? in[1] : in[0];
                d1 <= sel[1] ? in[3] : in[2];
                dout <= sel[0] ? d1 : d0;
            end
        end
    end
endmodule
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

//มอดูล re

```
module re (clk,rst,din,sel,dout);  
    input clk,rst;  
    input [3:0] din,sel;  
    output [3:0] dout;  
    reg [3:0] dout;  
  
    always @(posedge clk or posedge rst )  
  
begin  
    if (rst)  
        dout = 0;  
    else  
        begin  
            dout[0] = sel[0] ? din[1] : din[0];  
            dout[1] = sel[1] ? din[3] : din[2];  
            dout[2] = sel[2] ? din[1] : din[0];  
            dout[3] = sel[3] ? din[3] : din[2];  
        end  
    end  
endmodule
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

//มอดูล dff4

```
module dff4(clk,rst,din,dout);  
    input clk, rst;  
    input [3:0] din;  
    output [3:0] dout;  
    reg [3:0] dout;  
  
    always @(posedge clk or posedge rst)  
  
begin  
    if (rst)  
        dout <= 4'b0; //asynchronous RESET active High  
    else  
        dout <= din;  
    end  
endmodule
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

//มอดูล COMPARATOR

```
module COMPARATOR ( Survivor,Metric0, Metric1);
    input [3:0] Metric0,Metric1;
    output Survivor;
    wire M1msb, M0msb;
    wire [3:0] M1unsigned, M0unsigned;
    wire M1msb_xor_M0msb, M1unsignedcompM0;

    assign M0msb = Metric0 [3];
    assign M1msb = Metric1 [3];
    assign M1unsigned = {1'b0, Metric1 [2:0]};
    assign M0unsigned = {1'b0, Metric0 [2:0]};
    assign M1msb_xor_M0msb = M1msb ^ M0msb;
    assign M1unsignedcompM0 = (M1unsigned > M0unsigned)? 0:1;
    assign Survivor = M1msb_xor_M0msb ^ M1unsignedcompM0;

endmodule
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์ชิ้นนี้ สำเร็จลุล่วงได้ด้วยดี ด้วยความกรุณาเป็นอย่างดียิ่งจาก ดร. พรชัย ทรัพย์นิธิ และ อ. ศรวัฒน์ ชิวปรีชา ที่กรุณาให้ ทั้งในด้านความรู้ ความช่วยเหลือ ให้คำปรึกษาแนะนำ รวมถึง การอนุเคราะห์ เครื่องมือ และ อุปกรณ์ต่างๆ ตลอดจนตรวจสอบแก้ไขจนสำเร็จลงได้ ขอขอบคุณ คุณชัยวัฒน์ แก้วสาย ที่คอยให้ความช่วยเหลือเป็นอย่างดียิ่ง

สุดท้ายนี้ขอกราบขอบคุณ คุณพ่อ คุณแม่ และ เพื่อนทุกคน ที่คอยช่วยเหลือจนการจัดทำ ปริญญานิพนธ์ครั้งนี้สำเร็จลงได้

คณะผู้จัดทำ



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนังสืออ้างอิง

- [1] Bernard Sklar. "Digital Communications Fundamentals and Applications" New Jersey:Prentice Hall,Inc. 1998
- [2] B. Sklar, " Digital Communication Fundamentals and Applications " Communication Engineering Services,Tarzana,California and University of California,Los Angeles,2001
- [3] Daniel C. Hyde, " Handbook on Verilog HDL "Computer Science Department Bucknell University,1997
- [4] Douglas J Smith, "HDL Chip Design " Madison, AL,USA : Doone Publications,1999
- [5] John G. Proakis, Masoud Salehi, " Contemporary Communication Systems Using MATLAB " USA : BROOKS/COLE ,2000
- [6] กฤดากร กล่อมการ, "การสื่อสารข้อมูล (Data Communications)" กรุงเทพฯ : แผนกตำรา คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2545
- [7] ชัยวัฒน์ แก้วสาย, "การออกแบบสร้างชุดถอดรหัส Viterbi ขนาดเล็กสำหรับโทรศัพท์เคลื่อนที่ยุคที่ 3" วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมสารสนเทศ บัณฑิตวิทยาลัยสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง,2546
- [8] ผศ.ดร.ลัญญกร วุฒิสัทติกุลกิจ, "เทคโนโลยีโทรคมนาคม ทฤษฎีข่าวสารและการเข้ารหัส" กรุงเทพฯ : สำนักพิมพ์แห่งจุฬาลงกรณ์มหาวิทยาลัย,2546
- [9] สงคราม สุนทรานู,อังสุมารินทร์ สุพรรณรัตน์ "การเข้ารหัสควบคุมความผิดพลาดโดยใช้คอนโวลูชันเนลโค้ด" วิทยานิพนธ์คณะวิศวกรรมศาสตร์ สาขาวิศวกรรมโทรคมนาคม สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง,2543