ภาควิชาครุศาสตร์วิศวกรรม

คณะครุศาสตร์อุตสาหกรรม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อ เครื่องบันทึกเสียงพูดระบบดิจิตอล

Digital Voice Recorder

ชื่อนักศึกษา

1. นายเทพารักษ์	พุทธรักษ์	รหัสประจำตัว	43035375
2. นายปิยะ	ศรีปทุมภรณ์	รหัสประจำตัว	43035382
3. นายอนุวัฒน์	ธารเอี่ยม	รหัสประจำตัว	43035399
4. นายสุรินทร์	แท่นรัตน์	รหัสประจำตัว	43035619

หลักสูตร ครุศาสตร์อุตสาหกรรมบัณฑิต **สาขาวิชา** อิเล็กทรอนิกส์และคอมพิวเตอร์

อาจารย์ที่ปรึกษา อาจารย์อำพล ทองระอา

อาจารย์ที่ปรึกษาร่วม อาจารย์วรวิทย์ สมหา

คณะกรรมการสอบปริญญานิพนธ์		ลายมือชื่อ
1. อาจารย์สุระชัย	พิมพ์สาลี	.....
2. อาจารย์วรวิทย์	สมหา	.....
3. อาจารย์กิติพงค์	มะโน	.....
4. อาจารย์อำพล	ทองระอา	.....
5. อาจารย์พงษ์เกียรติ	เชษฐพิทักษ์สกุล	.....

วัน/เดือน/ปีที่สอบ วันจันทร์ที่ 29 เมษายน พ.ศ. 2545 เวลา 11.00 น.

สถานที่สอบ ห้อง ค.315 คณะครุศาสตร์อุตสาหกรรม สจล.

ภาควิชารับรองแล้ว

ลงนาม.....

(ผศ.วิสุทธิ์ อุทัยรัตนกิจ)

หัวหน้าภาควิชาครุศาสตร์วิศวกรรม

วันที่ 4 เดือน พ.ค. พ.ศ. 2545

<BT4402172>

เครื่องบันทึกเสียงพูดระบบดิจิตอล

ปริญญานิพนธ์

เครื่องบันทึกเสียงพูดระบบดิจิตอล

DIGITAL VOICE RECORDER

นายเทพารักษ์ พุทธรักษ์
นายปิยะ ศรีปทุมภรณ์
นายอนุวัฒน์ ธารเอี่ยม
นายสุรินทร์ แท่นรัตน์

ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรครุศาสตร์อุตสาหกรรมบัณฑิต
สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์
ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2544

ปริญญานิพนธ์

เรื่อง เครื่องบันทึกเสียงพูดระบบดิจิตอล

Digital Voice Recorder

วัตถุประสงค์

1. เพื่อศึกษาระบบบีบอัดข้อมูลเสียงพูดด้วยเวฟเลตแพคเกจ การเข้ารหัสและถอดรหัสฮัฟ แมน
2. เพื่อออกแบบวงจรเครื่องบันทึกเสียงพูดระบบดิจิตอล
3. เพื่อสร้างเครื่องบันทึกเสียงระบบดิจิตอล
4. เพื่อทดสอบเครื่องบันทึกเสียงพูดระบบดิจิตอลที่สร้างขึ้น
5. เพื่อนำไปใช้ในการบันทึกเสียงพูดได้จริง

ประโยชน์ที่คาดว่าจะได้รับ

1. เข้าใจกระบวนการทำงานของระบบการบีบอัดข้อมูลด้วยเวฟเลตเพคเกจ การเข้ารหัสและถอดรหัสฮัฟฟ์แมน
2. ได้วงจรเครื่องบันทึกเสียงพูดระบบดิจิตอล
3. ได้เครื่องต้นแบบเครื่องบันทึกเสียงพูดระบบดิจิตอล
4. สามารถนำไปปรับปรุงให้ได้เครื่องต้นแบบที่สมบูรณ์
5. นำไปใช้ในการบันทึกเสียงพูดได้จริง

ชื่อหัวข้อ	เครื่องบันทึกเสียงพูดระบบดิจิตอล
นักศึกษา	นายเทพารักษ์ พุทธรักษ์
	นายปิยะ ศรีปทุมภรณ์
	นายอนุวัฒน์ ธารเอี่ยม
	นายสุรินทร์ แท่นรัตน์
อาจารย์ที่ปรึกษา	อาจารย์อำพล ทองระอา
อาจารย์ที่ปรึกษาร่วม	อาจารย์วรวิทย์ สมหา
หลักสูตร	ครุศาสตร์อุตสาหกรรมบัณฑิต
สาขาวิชา	อิเล็กทรอนิกส์ และคอมพิวเตอร์
ปีการศึกษา	2544

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้เสนอการออกแบบ และการสร้างเครื่องบันทึกเสียงพูดระบบดิจิตอล โดยใช้หลักการบีบอัดข้อมูลของเวฟเลตเพคเกจ และทำการเข้ารหัสด้วยหลักการของฮัฟฟ์แมน ด้วยอุปกรณ์ FPGAs โครงสร้างของการแปลงเวฟเลตเพคเกจ และฮัฟฟ์แมนออกแบบให้อยู่ในรูปของฮาร์ดแวร์ โดยใช้ความถี่สุ่มของสัญญาณเสียงพูด 8 กิโลเฮิรตซ์ และทำการเก็บค่าสัมประสิทธิ์ที่ถูกแปลงโดยเวฟเลตเพคเกจ และการเข้ารหัสโดยฮัฟฟ์แมน ไว้ในหน่วยความจำภายนอกขนาด 128 กิโลไบต์

จากการทดสอบสามารถบันทึกเสียงพูดได้นานประมาณ 30 วินาที ซึ่งปริญญานิพนธ์ฉบับนี้ได้พัฒนาบนอุปกรณ์ FPGAs Xilinx Virtexe

Thesis Title	Digital Voice Recorder
Students	Mr. Teparak Puttarak
	Mr. Piya Sripatumpron
	Mr. Anuwat Tran-iam
	Mr Surin Tanrat
Advisor	Mr.Amphon Thongra-ar
Co-Advisor	Mr.Worawit Somha
Education Level	Bachelor of Science in Industrial Education
Program in	Electronics and Computer
Academic	2001

ABSTRACT

This thesis presents design and implementation of Digital Voice Recorder.It uses wavelet packet compression and huffman coding. It implement by FPGAs device. The structure of Wavelet packet and Huffman coding are design to hardware. This implementation uses sampling rates of speech has 8 kHz and keep coefficient of Wavelet packet and Huffman encoder in external memory size 128 kilobyte. This Digital Voice Recorder has been test recorder speech about 30 sec. This thesis implement on FPGAs Xilinx Virtexe.

กิตติกรรมประกาศ

การจัดทำปริญญานิพนธ์นี้สามารถสำเร็จลุล่วงไปได้ด้วยดี จากความร่วมมือของสมาชิกภายในกลุ่มทุกท่าน นอกจากนี้ยังได้รับความกรุณาจากอาจารย์อำพล ทองระอา ที่เป็นอาจารย์ที่ปรึกษาประจำโครงงาน อาจารย์วรวิทย์ สมหา ที่เป็นอาจารย์ที่ปรึกษาร่วมประจำโครงงาน รวมทั้งอาจารย์ประจำภาควิชาครุศาสตร์วิศวกรรมทุกท่าน ที่ให้คำปรึกษาแนะนำ และให้ความช่วยเหลือในด้านต่างๆ ตลอดจนให้โอกาสในการทำโครงงานอย่างเต็มที่ ทั้งด้านเวลา สถานที่เครื่องมือ และอุปกรณ์ต่างๆ ขอบพระคุณบุพการีผู้ให้กำเนิดที่ให้โอกาสในการศึกษาตลอดจนผู้ที่เกี่ยวข้องทุกคนที่ให้คำแนะนำต่างๆ และเป็นกำลังใจในการทำงานจนทำให้โครงงานนี้สำเร็จลุล่วงไปด้วยดี

สารบัญ

เรื่อง	หน้า
บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญรูป	V
สารบัญตาราง	VI
บทที่ 1 บทนำ	1
1.1 ความเป็นมา และความสำคัญของปริญญานิพนธ์	1
1.2 ขีดความสามารถของโครงงาน	1
1.3 เนื้อหาโดยสังเขป	1
บทที่ 2 ทฤษฎี และหลักการ	3
2.1 บทนำ	3
2.2 ทฤษฎีของเวฟเลต	3
2.3 พื้นฐานทฤษฎีของเวฟเลต	6
2.4 ตัวแปลงเวฟเลต และค่าจำกัดความ	7
2.5 การแปลงกลับเวฟเลต	9
2.6 การกระจายของพลังงานในเวฟเลตโดเมน	10
2.7 อนุกรมเวฟเลตต่อเนื่อง	10
2.8 อนุกรมเวฟเลตเต็มหน่วยเวลา	12
2.9 รูปแบบการแปลงเวฟเลตแม่เต็มหน่วยเวลา	12
2.10 Multiresolution Wavelet Transform	14
2.11 หลักการเข้ารหัสแบบฮัฟฟ์แมน	15
2.11.1 หลักการเข้ารหัสฮัฟฟ์แมน	16
2.11.2 การเขียน Binary tree เพื่อเข้ารหัสฮัฟฟแมน	16
2.12 การแทนรหัสฮัฟฟ์แมน	20
2.13 การถอดรหัสฮัฟฟ์แมน	20

สารบัญ (ต่อ)

เรื่อง	หน้า
2.14 ภาษา VHDL	23
2.14.1 ประวัติความเป็นมาของภาษา VHDL	23
2.14.2 Top – Down Design	23
2.14.3 Terminology and Convertions	26
2.15 Entity Design Unit	28
2.16 Architecture Design Unit	29
2.17 โครงสร้างภายในอุปกรณ์ FPGAs	30
2.18 ส่วนอินพุต และเอาต์พุตของอุปกรณ์ FPGAs	32
2.19 รายละเอียดการใช้งานอุปกรณ์ FPGAs	32
บทที่ 3 การออกแบบ การสร้าง และการทำงาน	34
3.1 ความคิดเบื้องต้นในการสร้างเครื่องบันทึกเสียงพูดระบบดิจิตอล	34
3.2 วงจรในการสร้างเครื่องบันทึกเสียงพูดระบบดิจิตอล	35
3.2.1 วงจรภายนอก	35
3.2.2 วงจรภายใน	41
3.3 การแปลงเวฟเลต	43
3.3.1 การออกแบบตารางการแปลงข้อมูล	43
3.3.2 การออกแบบวงจรแปลงเวฟเลต	53
3..3.3 การแปลงเวงเลตไป–กลับในวงจรเดียวกัน	58
3.4 การตัดระดับสัญญาณ	69
3.5 การจัดระดับข้อมูล	70
3.6 ความคิดเบื้องต้นในการสร้างวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน	76
3.7 การออกแบบวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนเพียงอย่างเดียว	78
3.8 วงจรเข้ารหัสฮัฟฟ์แมน	78
3.8.1 การออกแบบวงจรเข้ารหัสฮัฟฟ์แมน	79
3.8.2 การออกแบบวงจรเข้ารหัสฮัฟฟ์แมนบน Schemaitics Editor	81
3.8.3 วงจรบวกเลขขนาด 8 บิต	82

สารบัญ (ต่อ)

เรื่อง	หน้า
3.8.4 วงจรลบเลขขนาด 8 บิต	83
3.8.5 วงจรตัวนับ1 ถึง 255	83
3.8.6 วงจรสร้างสัญญาณนาฬิกา 9 ลูก และวงจรแทนรหัสฮัฟฟ์แมน ความยาวของรหัส 9 บิต	85
3.8.7 วงจรสร้างสัญญาณนาฬิกา 1 ลูก	88
3.8.8 วงจรมัลติเพล็กซ์ฮัฟฟ์แมน และมัลติเพล็กซ์สัญญาณนาฬิกา	90
3.8.9 วงจรควบคุมการจัดเก็บฮัฟฟ์แมนในหน่วยความจำ	90
3.9 วงจรถอดรหัสฮัฟฟ์แมน	92
3.9.1 การออกแบบวงจรถอดรหัสฮัฟฟ์แมน	94
3.9.2 การออกแบบวงจรถอดรหัสฮัฟฟ์แมนบน Schemaitics Editor	96
3.9.3 วงจรตัวนับ 00000H ถึง 1FFFFH	97
3.9.4 วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0	99
3.9.5 วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00001H ลงในบัฟเฟอร์ 1FFFFH	100
3.9.6 วงจรบัฟเฟอร์พักรหัสฮัฟฟ์แมนจากหน่วยความจำ	102
3.9.7 วงจรสร้างสัญญาณนาฬิกาเลื่อนรหัสฮัฟฟ์แมน	103
3.9.8 การออกแบบวงจรสร้างสัญญาณนาฬิกา 9 ลูก	106
3.9.9 การออกแบบวงจรสร้างสัญญาณนาฬิกา 1 ลูก	108
3.9.10 วงจรชี้ข้อมูล	110
3.10 การแปลงข้อมูลกลับ	111
3.10.1 การรีควอนไตซ์	111
3.10.2 การแปลงกลับเวฟเลต	111
บทที่ 4 การทดลอง และผลการทดลอง	112
4.1 ผลการทดลองการแปลงเวฟเลต	112
4.2 ผลการทดลองวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน	116
4.3 วงจรเข้ารหัสฮัฟฟ์แมน	116

สารบัญ (ต่อ)

เรื่อง	หน้า
4.4 วงจรถอดรหัสฮัฟฟ์แมน	119
บทที่ 5 ปัญหา และแนวทางพัฒนา	121
5.1 บทสรุป	121
5.2 ปัญหา และแนวทางการแก้ไขปัญหา	121
5.3 แนวทางในการพัฒนา	123
ภาคผนวก ก เครื่องเครื่องบันทึกเสียงพูดระบบดิจิตอล	125
ภาคผนวก ข ตารางฮัฟฟ์แมน และวงจรพิมพ์	127
บรรณานุกรม	147
ประวัติผู้แต่ง	148

สารบัญตาราง

ตาราง	หน้า
ตารางที่ 2.1 ความน่าจะเป็นของข้อมูล	15
ตารางที่ 2.2 เปรียบเทียบความน่าจะเป็น	17
ตารางที่ 2.3 การแทนรหัสฮัฟฟ์แมน	20
ตารางที่ 2.4 ตัวอย่างรหัสฮัฟฟ์แมนที่นำมาถอดรหัส	21
ตารางที่ 2.5 คุณสมบัติของ FPGAs ตระกูลต่างๆ	30
ตารางที่ 2.6 ประมาณการนับเกตพื้นฐาน	30
ตารางที่ 2.7 การแสดงโหมดต่างๆ ของคอนฟิกกูเรชัน	33
ตารางที่ 3.1 การเก็บสัมประสิทธิ์เวฟเลตแม่สามระดับ	43
ตารางที่ 3.2 ลักษณะการทำงานแบบ Pipeline ของการแปลงเวฟเลตไป–กลับ 2 ระดับ	61
ตารางที่ 3.3 การจัดระดับของข้อมูล 16 บิตไปเป็น 4 บิต	73
ตารางที่ 3.4 ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 9 ลูก	86
ตารางที่ 3.5 ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 1 ลูก	86
ตารางที่ 3.6 ฟังก์ชันการทำงานของวงจรควบคุมการโหลดข้อมูล ตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0	99
ตารางที่ 3.7 ฟังก์ชันการทำงานของวงจรควบคุมการโหลดข้อมูล ตำแหน่ง 00000H ลงในบัฟเฟอร์ 1	101
ตารางที่ 3.8 ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 9 ลูก	106
ตารางที่ 3.9 ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 1 ลูก	108

สารบัญรูป

รูป	หน้า
รูปที่ 2.1 เวฟเลตแม่	4
รูปที่ 2.2 สัญญาณ y = sine (5x)	4
รูปที่ 2.3 สัญญาณ y = Exp (-0.5^2)	5
รูปที่ 2.4 การสเกล และการเปลี่ยนแปลงของเวฟเลตแม่	6
รูปที่ 2.5 การแปลงเวฟเลตของสัญญาณ *f(x)* ให้อยู่ในโดเมนของเวฟเลต	7
รูปที่ 2.6 ขบวนการแปลงเวฟเลต	8
รูปที่ 2.7 ขบวนการแปลงกลับเวฟเลต	9
รูปที่ 2.8 Time Scale Resolution ของอนุกรมเวฟเลต	11
รูปที่ 2.9 กระบวนการแปลงแบบ DWTS ที่มีการ Scaleing เท่ากับสอง	13
รูปที่ 2.10 การแตกกระจายเวฟเลตด้วย DWTS block	13
รูปที่ 2.11 กระบวนการแปลงแบบ DWTS ที่มีการ Scaling เท่ากับสอง	14
รูปที่ 2.12 การรวมกลับของเวฟเลตแบบ IDTWS block	14
รูปที่ 2.13 การจับคู่ระหว่าง S_6 และ S_7	16
รูปที่ 2.14 การจับคู่ระหว่าง A_1 และ S_5	17
รูปที่ 2.15 การจับคู่ระหว่าง S_3 และ S_4	17
รูปที่ 2.16 การจับคู่ระหว่าง B_1 และ S_2	18
รูปที่ 2.17 การจับคู่ระหว่าง C_1 และ D_1	18
รูปที่ 2.18 การจับคู่ระหว่าง S_1 และ E_1	18
รูปที่ 2.19 การจับคู่ระหว่าง S_0 และ F_1	19
รูปที่ 2.20 การจัดโครงสร้างข้อมูลแบบฮัฟฟ์แมนทรี	19
รูปที่ 2.21 VHDL Statement สำหรับ Multiply Accumulate Algorithm	24
รูปที่ 2.22 โครงสร้าง Multiply – Accumulate Unit	24
รูปที่ 2.23 โครงสร้างอย่างง่ายๆ ของ Entity Design Unit	28
รูปที่ 2.24 รูป Multiplexer, (a) VHDL Entity Design (b) มุมมอง Interface	28
รูปที่ 2.25 โครงสร้างอย่าง่ายๆ ของ Architecture Design Unit	29
รูปที่ 3.1 ผังการทำงานของวงจรเครื่องบันทึกเสียงพูดระบบดิจิตอล	34

สารบัญรูป (ต่อ)

รูป	หน้า
รูปที่ 3.2 วงจรปรีไมค์โครโฟน	35
รูปที่ 3.3 ตำแหน่งขาต่างๆ ของ ไอซี DAC 0808	36
รูปที่ 3.4 วงจรใช้งานจริงของไอซี DAC 0808	37
รูปที่ 3.5 ตำแหน่งขาต่างๆ ของไอซีเบอร์ ADC 0804	38
รูปที่ 3.6 วงจรใช้งานจริงของไอซี DAC 0804	38
รูปที่ 3.7 วงจรที่ติดต่อระหว่างอุปกรณ์ FPGAs และหน่วยความจำภายนอก	39
รูปที่ 3.8 วงจรติดต่อกับอุปกรณ์ภายนอก	40
รูปที่ 3.9 แผนผังการทำงานภายในอุปกรณ์ FPGAs	41
รูปที่ 3.10 ขั้นตอนการทำเวฟเลตเพคเกจ	44
รูปที่ 3.11 ลักษณะของการทำประสาน (Convolution) ในระดับที่ A	45
รูปที่ 3.12 ลักษณะของการทำประสาน (Convolution) ในระดับที่ B	46
รูปที่ 3.13 ลักษณะของการทำประสาน (Convolution) ในระดับที่ C	48
รูปที่ 3.14 ข้อมูลในแต่ละระดับการทำเวฟเล็ฟเพคเกจ 3 ระดับ	52
รูปที่ 3.15 การเลื่อข้อมูลเพื่อทำการคูณกับตัวกรองตัวแรก	54
รูปที่ 3.16 การเลื่อข้อมูลเพื่อทำการคูณกับตัวกรองตัวที่สอง	54
รูปที่ 3.17 ขั้นตอนในการแปลงเวฟเลตเฟคเกจ	55
รูปที่ 3.18 วงจรแปลงเวฟเลต	57
รูปที่ 3.19 ลักษณะการเรียงข้อมูลภายใน ดี ฟลิป-ฟลอป ของวงจร SHF2	57
รูปที่ 3.20 เปรียบเทียบการแปลงเวฟเลตธรรมดา กับการแปลงเวฟเลตแบบใหม่	59
รูปที่ 3.21 การทำงานของวงจรเลื่อนข้อมูลของการแปลงเวฟเลตระดับที่ 1	62
รูปที่ 3.22 แผนผังลำดับการทำงานของวงจรเลื่อนข้อมูล	62
รูปที่ 3.23 วงจรแปลงเวฟเลตระดับที่ 1	63
รูปที่ 3.24 การทำงานของวงจรเลื่อนข้อมูลในการแปลงเวฟเลตระดับที่ 2 Stage 1 ถึง Stage 2	65
รูปที่ 3.25 การทำงานของวงจรเลื่อนข้อมูลในการแปลงเวฟเลตระดับที่ 2 Stage 3 ถึง Stage 6	65

สารบัญรูป (ต่อ)

รูป	หน้า
รูปที่ 3.26 การทำงานของวงจรเลื่อนข้อมูลในการแปลงเวฟเลตระดับที่ 2 Stage 7 ถึง Stage 10	66
รูปที่ 3.27 การทำงานของวงจรเลื่อนข้อมูลในการแปลงเวฟเลตระดับที่ 2 Stage 11 ถึง Stage 14	66
รูปที่ 3.28 แผนผังการเลื่อนข้อมูลของการแปลงเวฟเลตระดับที่ 2	67
รูปที่ 3.29 วงจรแปลงเวฟเลตระดับที่ 2	68
รูปที่ 3.30 การตัดระดับสัญญาณ	69
รูปที่ 3.31 วงจรตัดระดับสัญญาณ	70
รูปที่ 3.32 การหาจุดตัดระดับสัญยาณของข้อมูล	71
รูปที่ 3.33 ลักษณะการจัดระดับของข้อมูลจาก 16 บิต ไปเป็น 4 บิต	72
รูปที่ 3.34 ผังการทำงานของวงจรจัดระดับสัญญาณ (Quantizer)	76
รูปที่ 3.35 แผนผังการทำงานของวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน	77
รูปที่ 3.36 วงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนโดยใช้ฮาร์ดแวร์เพียงอย่างเดียว	78
รูปที่ 3.37 แผนผังการทำงานของวงจรเข้ารหัสฮัฟฟ์แมน	79
รูปที่ 3.38 วงจรเข้ารหัสฮัฟฟ์แมนบน Schematic Editor	81
รูปที่ 3.39 วงจรบวกเลข ขนาด 8 บิต	82
รูปที่ 3.40 วงจรลบเลข ขนาด 8 บิต	83
รูปที่ 3.41 แผนผังสถานะของวงจรตัวนับ 1 ถึง 255	84
รูปที่ 3.42 วงจรตัวนับ 1 ถึง 255 เมื่อแปลงเป็นแมโคร	84
รูปที่ 3.43 แผนผังสถานะของวงจรสร้างสัญญาณนาฬิกา 9 ลูก	86
รูปที่ 3.44 วงจรแทนรหัสฮัฟฟ์แมนความยาว 9 บิต	87
รูปที่ 3.45 วงจรสร้างสัญญาณนาฬิกา 9 ลูก และวงจรแทนรหัสฮัฟฟ์แมน 9 บิต เมื่อแปลงเป็นแมโคร	87
รูปที่ 3.46 แผนผังสถานะของวงรสร้างสัญญาณนาฬิกา 1 ลูก	89
รูปที่ 3.47 วงจรสร้างสัญญาณนาฬิกา 1 ลูกเมื่อแปลงเป็นแมโคร	89
รูปที่ 3.48 วงจรมัลติเพล็กซ์รหัสฮัฟฟ์แมน และสัญญาณนาฬิกา	90

สารบัญรูป (ต่อ)

รูป	หน้า
รูปที่ 3.49 วงจรควบคุมการจัดเก็บรหัสฮัฟฟ์แมนลงในหน่วยความจำ	90
รูปที่ 3.50 ตัวอย่างรหัสฮัฟฟ์แมนที่ได้แทนข้อมูลในหน่วยความจำขนาด 8 บิต	92
รูปที่ 3.51 ตัวอย่างรหัสฮัฟฟ์แมนที่จัดเก็บไว้ในหน่วยความจำขนาด 8 บิต	93
รูปที่ 3.52 แผนผังการทำงานของวงจรถอดรหัสฮัฟฟ์แมน	94
รูปที่ 3.53 วงจรถอดรหัสฮัฟฟ์แมนบน Schematic Editor	96
รูปที่ 3.54 แผนผังสถานะของวงจรตัวนับ 0 ถึง 255	98
รูปที่ 3.55 วงจรตัวนับ 0 ถึง 255 เมื่อแปลงเป็นแมโคร	98
รูปที่ 3.56 วงจรตัวนับ 00000H ถึง 1FFFFH สร้างจากวงจรตัวนับ 0 ถึง 255	98
รูปที่ 3.57 แผนผังสถานะของวงจรควบคุมการโหลดข้อมูลที่ตำแหน่ง 00000H ลงในบัฟเฟอร์ 0	99
รูปที่ 3.58 วงจรควบคุมการโหลดข้อมูลที่ตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0 เมื่อแปลงเป็นแมโคร	100
รูปที่ 3.59 แผนผังสถานะวงจรควบคุมการโหลดข้อมูลตำแหน่ง 00001H ถึง 1FFFFH ลงในบัฟเฟอร์ 1	101
รูปที่ 3.60 วงจรควบคุมการโหลดข้อมูลที่ตำแหน่งที่ 00001H ถึง 1FFFFH ลงในบัฟเฟอร์ 1 เมื่อแปลงเป็นแมโคร	101
รูปที 3.61 วงจรบัฟเฟอร์พักรหัสฮัฟฟ์แมนจากหน่วยความจำ	102
รูปที่ 3.62 วงจรสร้างสัญญาณนาฬิกาเลื่อนรหัสฮัฟฟ์แมน	104
รูปที่ 3.63 แผนผังสถานะวงจรสร้างสัญญาณนาฬิกา 9 ลูก	107
รูปที่ 3.64 วงจรสร้างสัญญาณนาฬิกา 9 ลูกเมื่อแปลงเป็นแมโคร	107
รูปที่ 3.65 แผนผังสถานะของวงจรสร้างสัญญาณนาฬิกา 1 ลูก	109
รูปที่ 3.66 วงจรสร้างสัญญาณนาฬิกา 1 ลูกเมื่อแปลงเป็นแมโคร	109
รูปที่ 3.67 วงจรชี้ข้อมูล	110
รูปที่ 4.1 การ Latch ข้อมูลพร้อมกับสัญญาณควบคุม	113
รูปที่ 4.2 การนำค่าที่ Latch ทำการคำนวณ	114
รูปที่ 4.3 ผลลัพธ์ของการคำนวณในแต่ละครั้ง	115
รูปที่ 4.4 ผลจำลองการทำงานของวงจรเข้ารหัสฮัฟฟ์แมน	117

สารบัญรูป (ต่อ)

รูป	หน้า
รูปที่ 4.5 จำลองจัดเก็บฮัฟฟ์แมนที่แทนข้อมูล S1, S2, S3 ตามลำดับ	118
รูปที่ 4.6 ตำแหน่ง และข้อมูลในหน่วยความจำเพื่อทดสอบการดีโค้ดหัสฮัฟฟ์แมน	119
รูปที่ 4.7 ผลจำลองการทำงานของวงจรถอดรหัสฮัฟฟ์แมน	119

บทที่ 1
บทนำ

1.1 ความเป็นมา และความสำคัญของปริญญานิพนธ์

ในปัจจุบันโลกได้มีการพัฒนาทางด้านของเทคโนโลยีไปอย่างรวดเร็ว ไม่ว่าจะเป็นด้านการสื่อสารข้อมูล อิเล็กทรอนิกส์ เครื่องใช้อำนวยความสะดวกต่างๆ ล้วนแต่มีการพัฒนามาจากเทคโนโลยีโดยทั้งสิ้น และก็เช่นเดียวกันกับการบันทึกเสียงก็จำเป็นที่จะต้องใช้เทคโนโลยีเข้ามาช่วยในการบันทึกเสียงพูด โดยที่เครื่องบันทึกเสียงพูด ก็จะทำการบันทึกเสียงโดยที่ไม่ผ่านตัวกระทำใดๆ ในการลดขนาดของข้อมูล ซึ่งจะทำให้เปลืองพื้นที่ในการบันทึกข้อมูลเสียง

ดังนั้นจึงได้ทำการศึกษากระบวนการลดขนาดของข้อมูลโดยใช้เวฟเลตเพคเกจซึ่งวิเคราะห์ความซับซ้อน และลดขนาดของข้อมูลโดยใช้การเข้ารหัสฮัฟฟ์แมน การเข้ารหัสเพื่อลดขนาดของบิต ข้อมูล

1.2 ขีดความสามารถของโครงงาน

1) บีบอัดข้อมูลเสียงพูดโดยใช้เวฟเลตเพคเกจ
2) มีหน่วยความจำ 128 กิโลไบต์
3) มีอัตราการสุ่มสัญญาณเสียงพูด 8 กิโลเฮิรตซ์
4) สามารถบันทึกเสียงพูด และเล่นกลับเสียงพูดได้

1.3 เนื้อหาโดยสังเขป

เนื้อหาในปริญญานิพนธ์ได้แบ่งออกเป็นส่วนต่างๆ เพื่อสะดวกต่อการศึกษาและค้นคว้าทำความเข้าใจแต่ละบทจะประกอบไปด้วยเนื้อหาดังต่อไปนี้

บทที่ 2 ทฤษฎี และหลักการประกอบด้วยเนื้อหาดังนี้ คือ เวฟเลตเพคเกจการเขียนโปรแกรมโดยใช้ภาษา VHDL การเข้ารหัส และถอดรหัสฮัฟฟ์แมน

บทที่ 3 การออกแบบ และการสร้างวงจรแอนะลอก วงจรดิจิตอลภายนอกอุปกรณ์ FPGAs และวงจรดิจิตอลที่ออกแบบ ภายในอุปกรณ์ FPGAs ลำดับขั้นตอนการออกแบบวงจร รวมถึงการทำงานในส่วนต่างๆ ซึ่งจะทำให้ผู้อ่านมีความเข้าใจในขั้นตอนการทำงานโดยรวม และส่วนย่อยของโครงงาน

บทที่ 4 การทดลอง และผลการทดลองนั้นได้กล่าวถึงขั้นตอนการทดลอง การทดสอบประสิทธิภาพในการทำงานของเครื่องบันทึกเสียงพูดระบบดิจิตอลเพื่อที่จะตรวจสอบเครื่องบันทึกเสียงพูดระบบดิจิตอลทำงานได้ตามวัตถุประสงค์หรือไม่

บทที่ 5 บทสรุป ปัญหา แนวทางในการแก้ไข และการพัฒนา เพื่อเป็นการสรุปผลในการทำโครงงาน ปัญหาที่เกิดขึ้น และได้เสนอแนวทางในการแก้ไขปัญหา รวมทั้งแนวทางในการพัฒนาเพื่อให้โครงงานมีประสิทธิภาพมากยิ่งขึ้น

ภาคผนวก ก เครื่องบันทึกเสียงพูดต้นแบบ

ภาคผนวก ข ตารางฮัฟฟ์แมน และวงจรพิมพ์

บทที่ 2
ทฤษฎี และหลักการ

2.1 บทนำ

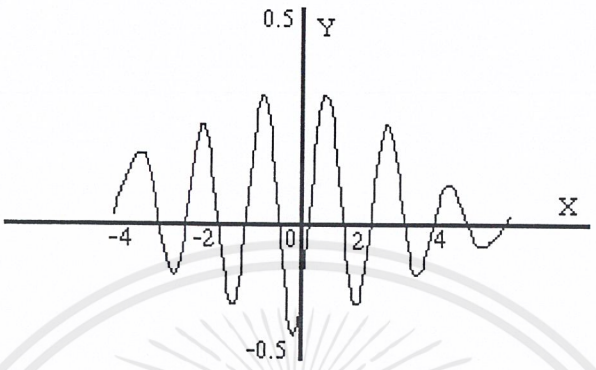
สัญญาณ หรือฟังก์ชันใดๆ ก็ตามสามารถที่จะแสดงให้อยู่ในรูปของการรวมเชิงเส้นของฟังก์ชันมูลฐานต่างๆ ได้ซึ่งทำให้สามารถทำการวิเคราะห์ หรือแก้ปัญหาได้อย่างสะดวกสามารถยกตัวอย่างได้ เช่น การใช้ฟังก์ชันไซน์ และโคไซน์เป็นฟังก์ชันมูลฐานในการวิเคราะห์ในระบบฟูเรียร์ เป็นต้น

การวิเคราะห์เวฟเลตก็เป็นการแสดงฟังก์ชันใดๆ ให้อยู่ในรูปของการรวมเชิงเส้นของฟังก์ชันมูลฐานเช่นเดียวกัน ด้วยลักษณะพิเศษของฟังก์ชันมูลฐานเวฟเลตซึ่งมีลักษณะของการออสซิลเลตตามแกนแนวนอน และมีแอมปลิจูดที่ลดลงสู่ศูนย์ทั้งด้านบวก และด้านลบอย่างรวดเร็ว ต่างกับฟังก์ชันไซน์ และโคไซน์ซึ่งแผ่ตามแนวนอนไปสู่อนันต์จึงทำให้เหมาะที่จะใช้ในการวิเคราะห์สัญญาณที่มีลักษณะไม่คงที่ ลักษณะของความไม่ต่อเนื่อง และลักษณะของปลายแหลมคมได้เป็นอย่างดี เนื่องจากในโดเมนของเวฟเลตจะประกอบไปด้วยสัมประสิทธิ์ของการกระจาย หรือตัวคูณของฟังก์ชันมูลฐานเวฟเลตที่มีขนาดเล็ก และลู่เข้าสู่ศูนย์อย่างรวดเร็ว ซึ่งในรูปโดเมนของฟูเรียร์ของฟังก์ชันดังกล่าวจะต้องประกอบด้วยสัมประสิทธ์ของการกระจาย หรือตัวคูณของฟังก์ชันไซน์ และโคไซน์เป็นจำนวนมากถึงอนันต์ แต่เมื่อมองในแง่ของการแปลงเวฟเลตเพื่อเข้ารหัสแล้ว การแปลงเวฟเลตจึงเป็นการแปลงที่มีประสิทธิภาพสูง เมื่อใช้กับสัญญาณที่มีลักษณะไม่คงที่ ลักษณะของความไม่ต่อเนื่อง และลักษณะของปลายแหลมคม ประโยชน์ที่สำคัญอีกประการหนึ่งของการแปลงเวฟเลต คือ ความเปลี่ยนแปลงที่เกิดขึ้นภายในโดเมนของเวฟเลตซึ่งจะมีผลทำให้เกิดการเปลี่ยนแปลงในโดเมนเป็นช่วงเท่านั้น ซึ่งเป็นผลมาจากการวิเคราะห์แบบหลายระดับความละเอียด

2.2 ทฤษฎีเวฟเลต

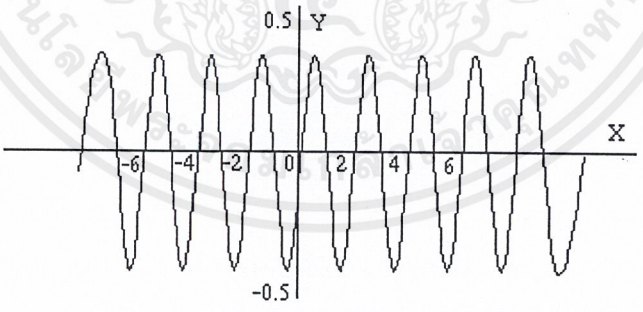
ทฤษฎีเวฟเลตเป็นกระบวนการทางคณิตศาสตร์ที่ใช้อธิบายถึงแบบจำลองของสัญญาณระบบ หรือกระบวนการแล้วกำหนดให้เซตของสัญญาณที่มีลักษณะเฉพาะเป็นองค์ประกอบพื้นฐานโดยจะเรียกว่า "เซตของเวฟเลต" สมาชิกของเวฟเลตจะมีคุณสมบัติที่สำคัญ คือ เป็นสัญญาณที่จะเกิดขึ้นมาจากต้นแบบเดียวกัน ต้นแบบของสัญญาณในเซตของเวฟเลตนี้เรียกว่า "เวฟเลตแม่"

เวฟเลตจะมีคุณสมบัติของสัญญาณที่เกิดในช่วงเวลาที่สั้นๆ หรือกล่าว คือ เวฟเลตเป็นสัญญาณที่เกิดขึ้นอย่างต่อเนื่อง และมีขนาดลดลงสู่ศูนย์ทั้งสองด้าน



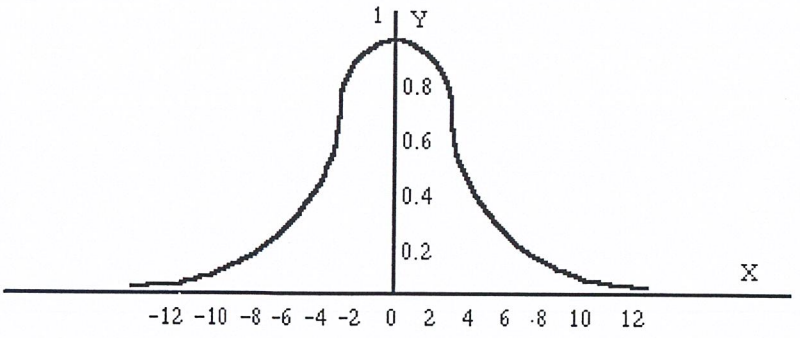
รูปที่ 2.1 เวฟเลตแม่

สัญญาณที่ 2.1 จะประกอบด้วยลักษณะทั้งสองด้าน คือ การเกิดขึ้นอย่างต่อเนื่อง ซึ่งเกิดจากสัญญาณไซน์ดังในรูปที่ 2.2 และการลู่เข้าสู่ศูนย์ของสัญญาณทั้งสองด้านจากหน้าต่างฟังก์ชันซึ่งแสดงได้ ดังรูปที่ 2.3 เมื่อนำเอาฟังก์ชันทั้งสองมาคูณเข้าด้วยกันจะได้เป็นสัญญาณของเวฟเลต ดังรูปที่ 2.3



รูปที่ 2.2 สัญญาณของ Y = Sin5x

จากเวฟเลตแม่ที่ได้จะถูกนำไปสร้างสมาชิกตัวอื่นๆ ในเซตของเวฟเลตที่ต้องการนำไปอธิบายถึงสัญญาณกระบวนการ หรือระบบต่างๆ โดยสมาชิกอื่นๆ ได้จากเวฟเลตแม่ที่ได้ถูกทำการ



รูปที่ 2.3 สัญญาณ $Y = Exp(-0.5x^2)$

สเกล (Scaling) : a และเลื่อนตำแหน่ง (Translator) : b ไปตามแกนของเวลา ถ้ากำหนด $g(t)$ เป็นฟังก์ชันของเวฟเลตแม่ แล้วการสเกลด้วนพารามิเตอร์ของ "a" และการเลื่อนตำแหน่งด้วยพารามิเตอร์ของ "b" จะสามารถหาได้จากสมการที่ 2.1

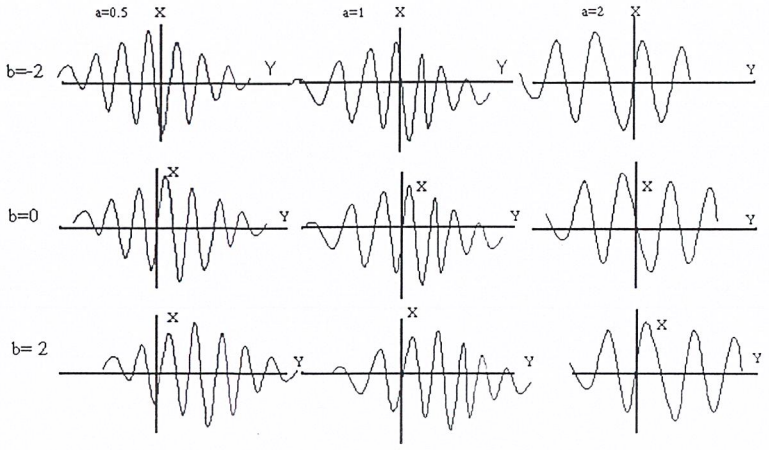
$$g_{a.b}(t) = \frac{g(t-b)}{a} \tag{2.1}$$

เมื่อ a คือ อัตราของการสเกล

b คือ การเลื่อนตำแหน่ง

ซึ่งทำการสเกล และเลื่อนตำแหน่งตามแกนของเวลานี้ทั้งหมดจะรวมเรียกว่า Affine Operation(การทำ Linear apping ร่วมกับการทำ Translator) และเพื่อสัญญาณที่ถูกสเกล และมีค่าของสัญญาณที่เท่าเดิม จึงทำการ Normalization สัญญาณที่ได้ดังรูปที่ 2.4

จากรูปที่ 2.4 จะสามารถสังเกตเห็นว่า ลักษณะจำนวนลูกคลื่นสัญญาณนั้นจะเหมือนกันทั้งหมด สมาชิกแต่ละตัวจะแตกต่างกันที่ขนาด และตำแหน่งแกนเวลาเท่านั้น โดยสัญญาณนั้นจะมีลักษณะลู่เข้าหาศูนย์ทั้งสองข้าง เมื่อทำการสเกลและเลื่อนตำแหน่งไป และค่าทางด้านแอมปลิจูดก็จะเปลี่ยนไปด้วย ซึ่งระดับของแอมปลิจูดนั้นจะขึ้นอยู่กับข้อมูลที่เข้ามาและค่าของการสเกล และเลื่อนตำแหน่ง



รูปที่ 2.4 การสเกล และการเปลี่ยนแปลงของเวฟเลตแม่

2.3 พื้นฐานทฤษฎีของเวฟเลต

ทฤษฎีเวฟเลตเป็นวิธีการอย่างหนึ่งเช่นเดียวกับระบบตัวเลขที่ใช้แทนจำนวนสิ่งของต่างๆ ได้ เช่น ผลเฉลยสมการดิฟเฟอเรนเชียลการเต้นของหัวใจ สภาพภูมิอากาศ เป็นต้น

ทฤษฎีการแปลงเวฟเลต เป็นเครื่องมือทางด้านคณิตศาสตร์อย่างหนึ่งที่สามารถนำไปประยุกต์ใช้งานได้กับสิ่งต่างๆ โดยมีวัตถุประสงค์ที่จะใช้อธิบายระบบ หรือสัญญาณเหล่านั้นได้ดีกว่าวิธีการอื่นๆ การที่จะนำทฤษฎีของเวฟเลตไปใช้งาน จึงอยู่บนพื้นฐานที่ว่า "การนำมาใช้งานมีประสิทธิภาพเพียงใด" ซึ่งจะอธิบายสิ่งใดสิ่งหนึ่งได้ โดยการแยกสิ่งเหล่านั้นออกเป็นส่วนย่อยๆ ที่มีความสัมพันธ์กันโดยส่วนย่อยๆ เหล่านั้น คือ เวฟเลตแม่ที่ถูกสเกล แล้วทำการเลื่อนตำแหน่ง

ขบวนการที่ทำการแยกสัญญาณออกเป็นส่วนๆ หรือการหาค่าสัมประสิทธิ์นี้เราจะเรียกว่า "การกระจายเวฟเลต" หรือ "การแปลงเวฟเลต" และในทางกลับกันก็ต้องมีการรวมกลับของสัญญาณ เพื่อให้ได้สัญญาณเดิมเราจะเรียกว่า " การแปลงกลับเวฟเลต"

การแปลงเวฟเลตอย่างต่อเนื่องของฟังก์ชัน $f(x)$ จะสามารถแสดงด้วยสมการทางคณิตศาสตร์ คือ

$$w_g[f(x)](a,b) = |a|^{-1/2} \int f(x) g^*\left(\frac{x-b}{a}\right) dx \qquad (2.2)$$

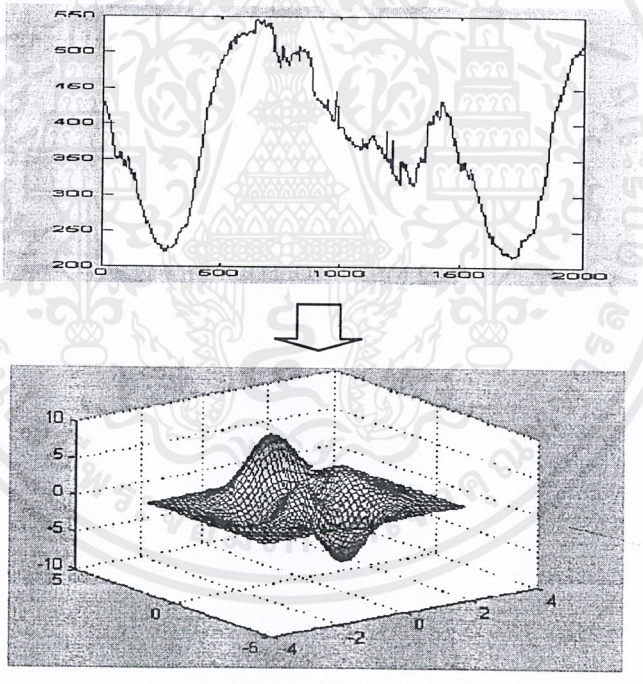
เมื่อ $g(x)$ เป็นฟังก์ชันเวฟเลตแม่ และค่า $g^*(x)$ คือ Complex conjugate ของ $g(x)$

a คือ Scaling

b คือ Translation

$w_g f(a,b)$ แทนค่าของสัมประสิทธิ์ที่ได้

จะเห็นว่าการแปลงเวฟเลตจะเป็นการส่งผ่านฟังก์ชันที่มีตัวแปรอิสระ x ไปยังปฐมภูมิของฟังก์ชันที่เป็น 2 มิติของตัวแปรอิสระ คือ a และ b เรียงตามลำดับ ค่าสัมประสิทธิ์ของเวฟเลตที่ a,b นี้จะแสดงถึงค่าที่สัมพันธ์กัน ระหว่างเวฟเลตแม่ที่สเกล a และตำแหน่ง b กับสัญญาณอินพุตที่ตำแหน่งเดียวกันโดยถ้าสัญญาณที่ได้ทั้งสองนี้มีความหมายที่เหมือนกัน หรือใกล้เคียงกันมากค่าสัมประสิทธิ์ที่ได้ก็จะมีค่ามากไปด้วยสัมประสิทธิ์ที่ได้ $w_g[f(x)](a,b)]$ จะเรียกว่าโดเมนของเวฟเลต (Wavelet Domain)



รูปที่ 2.5 การแปลงเวฟเลตของสัญญาณ $f(x)$ ให้อยู่ในโดเมนของเวฟเลต

2.4 ตัวแปลงเวฟเลต และค่าจำกัดความ

ในการกำหนดฟังก์ชันที่จะสามารถนำไปใช้เป็นเวฟเลตแม่ได้นั้นจะต้องมีคุณสมบัติของการแกว่งไปมา และการลู่เข้าสู่ศูนย์ โดยจะเป็นฟังก์ชันที่สามารถหาค่าพลังงานเวฟเลตได้เมื่อ

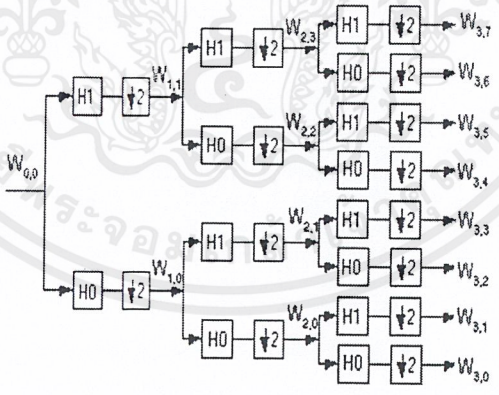
$$c_g = \int_{-\infty}^{\infty} \frac{|G(w)|^2}{|w|} dw < \infty \qquad (2.3)$$

เมื่อ $G(w)$ Furier transform ของ g

C_g ค่าพลังงานของ g

การที่ค่าพลังงานของเวฟเลตแม่ที่มีค่าเป็นหนึ่งหน่วยจะทำให้เกิดคุณสมบัติที่สำคัญอย่างหนึ่ง คือ ค่าพลังงานของสัมประสิทธิ์ที่ได้ในเวฟเลตโดเมนจะเท่ากับค่าพลังงานของสัญญาณที่อยู่ในโดเมนเวลา หรือปฐมภูมิ และยังทำให้ค่าสัมประสิทธิ์เวฟเลต a,b ใดๆ ที่สามารถที่จะนำมาเปรียบเทียบกันได้ สำหรับการแปลงเวฟเลตอย่างต่อเนื่องนั้นคุณสมบัติที่ต้องการฟังก์ชันที่จะนำมาเป็นเวฟเลตแม่จะมีเพียงดังที่กล่าวข้างต้น ซึ่งการเลือกใช้เวฟเลตจะขึ้นอยู่กับการประยุกต์ใช้งาน

เซตของค่าการแปลงเวฟเลตนี้จะสามารถเป็นตัวแทนของสัญญาณ และการแปลงกลับเวฟเลตแม่ตัวเดิมก็สามารถใช้ในการแปลงกลับได้ค่าสัมประสิทธิ์เวฟเลตเปรียบเสมือนตัวประกอบย่อยของสัญญาณในการหาค่าสัมประสิทธิ์ของสัญญาณส่วนประกอบนั้นจะถูกฉายไปยังส่วนย่อย ซึ่งผลลัพธ์ที่ได้ก็ คือ ค่าจำนวนจริง หรือจำนวนเชิงซ้อนเรียกว่าค่าสัมประสิทธิ์เวฟเลตของสัญญาณหรือฟังก์ชันที่มีความสัมพันธ์กับเวฟเลตแม่



รูปที่ 2.6 ขบวนการแปลงเวฟเลต

นอกจากที่กล่าวมาแล้วมีการเปรียบเทียบสัญญาณที่เรียกว่า Correlation เป็นการเปรียบเทียบว่าสัญญาณทั้งสองมีความเหมือนกันมากเพียงใดสัญญาณจะถูกนำมาเปรียบเทียบกับสัญญาณเวฟเลตแม่ g ที่ถูกสเกล และเลื่อนตำแหน่งออกไป แสดงตามรูปที่ 2.6 Correlation จะแสดงด้วย I ที่

อยู่ในวงกลม ในการแปลงตามรูปที่ 2.6 นั้น เรียกอีกอย่างหนึ่งว่า Analysis Filter โดยกระบวนการดังกล่าวจะทำการแตกกระจายสัญญาณออกเป็นส่วนย่อย ในการแปลงกลับเวฟเลตนั้นสัญญาณเหล่านี้จะถูกนำกลับมารวมกันอีกครั้งหนึ่ง

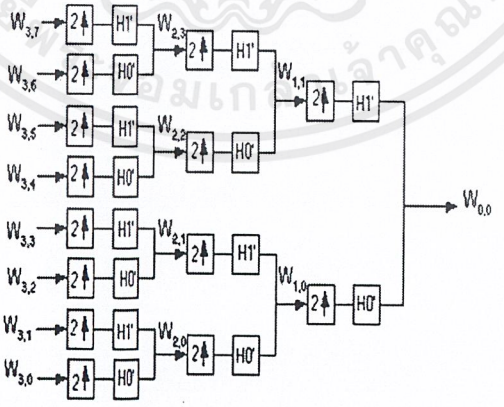
2.5 การแปลงกลับเวฟเลต

การแปลงกลับเวฟเลตเป็นการทำสัมประสิทธิ์เวฟเลตแปลงกลับให้เป็นฟังก์ชันเดิม ซึ่งจะแสดงได้ดังรูปที่ 2.7 และในการแปลงกลับจะแสดงได้ดังสมการ 2.5

$$w_g^{-1} : w_g f(a,b) \longrightarrow f(x) \tag{2.4}$$

$$f(x) = \frac{1}{c} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} w_g f(a,b) \frac{1}{\sqrt{a}} g\left(\frac{x-b}{a}\right) \frac{dbd}{a^2} \tag{2.5}$$

การแปลงกลับเวฟเลตจะสร้างฟังก์ชันสัญญาณต้นฉบับขึ้นมาจากการรวมกันของสัญญาณเวฟเลตแม่ g คูณกับค่าถ่วงน้ำหนัก (ค่าสัมประสิทธิ์เวฟเลต) กรณีที่การแปลงเวฟเลตนี้ใช้เวฟเลตแตกต่างไปจากตัวของเวฟเลตแม่ที่ใช้ตอนการแปลงเวฟเลต ผลลัพธ์ที่ได้จะเป็นสัญญาณ หรือฟังก์ชันที่มีความแตกต่างกันออกไปจาก ฟังก์ชันของต้นฉบับเดิม ดังรูปที่ 2.7



รูปที่ 2.7 การแปลงกลับเวฟเลต

2.6 การกระจายของพลังงานในเวฟเลตโดเมน

จากการที่สัมประสิทธิ์ของเวฟเลตเป็นตัวแทนของสัญญาณได้รูปแบบหนึ่งแล้วจะมีคุณสมบัติเฉพาะตัวเช่นเดียวกับสัญญาณนั้นคุณสมบัติที่สำคัญของสัญญาณ คือ การหาค่าพลังงานของสัญญาณนั้นๆ นั่นเอง ค่าพลังงานของสัญญาณสามารถจะคำนวณได้จากสมการ

$$E_f = \int_{-\infty}^{\infty} |f(t)|^2 \, dt \tag{2.6}$$

เมื่อ E_f คือ ค่าพลังงานของสัญญาณ

$f(t)$ คือ สัญญาณ

ใน Wavelet Transform Domain ค่าความถี่ซึ่ง หมายถึงการสเกลค่าพลังงานไปตามสมการและค่าพลังงานรวมเป็น

$$\Delta E_f = \left| w_g f(a,b) \right|^2 \frac{dad}{a^2} \tag{2.7}$$

$$E_f = \frac{1}{c} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left| w_g f(a,b) \right|^2 \frac{dad}{a^2} \tag{2.8}$$

2.7 อนุกรมเวฟเลตต่อเนื่อง (Continuous Time Wavelet Series)

ในอนุกรมเวฟเลตต่อเนื่องจำนวนค่าสัมประสิทธิ์ที่ได้จะมีจำนวนจำกัด (ในกรณีของเวฟเลตต่อเนื่องจะมีค่าเป็นอนันต์) ค่าสัมประสิทธิ์จะแบ่งออกเป็นส่วนๆ ตามค่าเชิงสเกลที่กำหนดโดยตัวเลขจำนวนเต็ม m และค่าของ Translator จะถูกกำหนดโดยตัวเลขจำนวนเต็ม n ดังนั้นค่า Scale (a) และค่า Translator (b) จะเป็น

$$a = a_0^m \tag{2.9}$$

$$b = nb_0 a_0^m \tag{2.10}$$

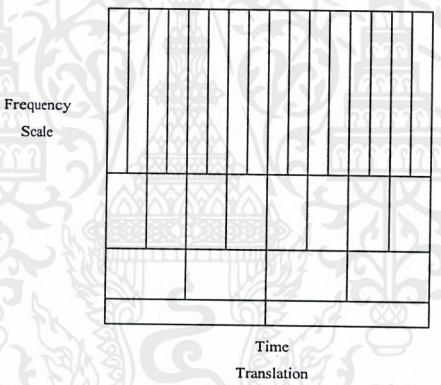
เมื่อ a คือ ช่วงของ Scale

b คือ ช่วงของ Translator

เมื่อนำมาแทนสมการ 2.3 แล้ว สมการของอนุกรมเวฟเลตต่อเนื่อง จะได้เป็น

$$w_g f(m,n) = a_0^{-m/2} \int_{-\infty}^{\infty} f(x) g(a_0^{-m} x - nb_0) dx \qquad (2.11)$$

ซึ่งสมการข้างต้นยังใช้การ Integrate อยู่เนื่องมาจากสัญญาณ $f(x)$ และเวฟเลตแม่ $g(x)$ นั้นยังเป็นเวลาแบบต่อเนื่อง และเช่นเดียวกับการแปลงเวฟเลตต่อเนื่อง ค่าของสัมประสิทธิ์ก็จะเป็นตัวแทนของสัญญาณ Wavelet Domain แต่ในเวฟเลตจะใช้แต่ค่าของสเกลที่อยู่ในช่วงบวกเท่านั้น



รูปที่ 2.8 Time Scale Resolution ของอนุกรมเวฟเลต

จากตัวอย่างรูปที่ 2.4 ที่ผ่านมาค่าของสเกลมีค่ามาก เวฟเลตที่ได้จะเป็นเวฟเลตแม่ที่ถูกยืดออกตามแกนเวลา ซึ่งเวลาที่นานขึ้นจะทำความละเอียดตามแกนเวลาลดลงดังแสดงตามรูปที่ 2.8 และค่าของสเกลมีค่าน้อย เวฟเลตที่ได้จะมีขนาดลดลงตามแกนของเวลาคาบเวลาที่ลดลงมา มีผลทำให้ความละเอียดตามแกนของเวลามีมากขึ้น แต่ความละเอียดทางด้านความถี่จะลดลงดังแสดงในรูปที่ 2.8 และนอกจากนี้รูปที่ 2.8 ยังแสดงถึงการเกิด และลำดับค่าของสัมประสิทธิ์เวฟเลตด้วยจากที่ได้กล่าวมาก่อนแล้ว ช่องสี่เหลี่ยมที่แสดงดังรูปข้างบนจะเป็นตัวแทนของเวฟเลตแม่ที่ Scale และ Translation ต่างๆ กันซึ่งจะนำไปสู่ค่าของสัมประสิทธิ์เวฟเลตนั้นๆ ค่าของเวฟเลตที่ค่าสัมประสิทธิ์เวฟเลตระดับของสเกลที่มีค่ามากก็จะใช้เวลาในการประมวลผลนานจากเวฟเลตแม่ที่ยืดออกนั้นเอง

ดังนั้นค่าสัมประสิทธิ์ที่สัมพันธ์กันกับระดับของสเกลที่มากขึ้น จะได้จำนวนสัมประสิทธิ์ต่อหน่วยเวลาที่น้อยกว่าสัมประสิทธิ์ที่ค่าของสเกล (Scale) น้อยลงไป เนื่องจากขนาดของเวฟเลตแม่ที่สั้นเข้า จึงทำให้เวลาในการประมวลผลน้อยลง และเมื่อพิจารณาตามแกนของเวลาจะพบว่าค่าของสัมประสิทธิ์ที่ระดับสเกลที่มากจะมีค่าซ้ำกันในขณะที่สัมประสิทธิ์ที่ระดับสเกลน้อยๆ จะเปลี่ยนไปเรื่อยๆ ดังนั้นเพื่อเป็นการลดความซ้ำกันของสัมประสิทธิ์ที่เกิดขึ้น จึงต้องตัดส่วนที่ซ้ำกันออกไป ซึ่งเรียกว่า Sub Sampling

2.8 อนุกรมเวฟเลตเต็มหน่วยเวลา (Discreat Time Wavelet Series)

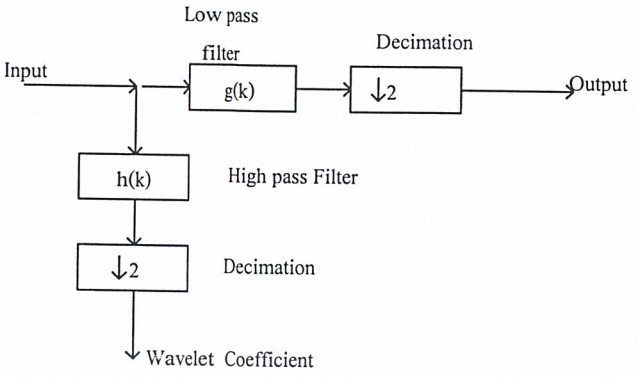
นอกเหนือจากการแปลงเวฟเลตที่มีค่าของ Scale และ Translation แบบเต็มหน่วยแล้วยังมีค่าของแกนเวลาที่สามารถเป็นแบบเต็มหน่วยได้เช่นกัน เช่น ลำดับของตัวเลขสามารถนำมาใช้กับเวฟเลตได้ ซึ่งเรียกว่าอนุกรมเวฟเลตเต็มหน่วยเวลา (DWTS) การแปลงแบบ DWTS มีสามการของการแปลงดัง นี้

$$w_h f(m,n) = a_0^{-m/2} \sum_{k=-\infty}^{\infty} f(k) h(a_0^{-m} k - n b_0) \qquad (2.12)$$

สมการข้างต้น $h(x)$ เป็นแบบเต็มหน่วยเวลา ดังนั้นที่ค่าของ k ที่ไม่เป็นจำนวนเต็ม ค่าของ $h(k)$จะได้โดยการประมาณค่า

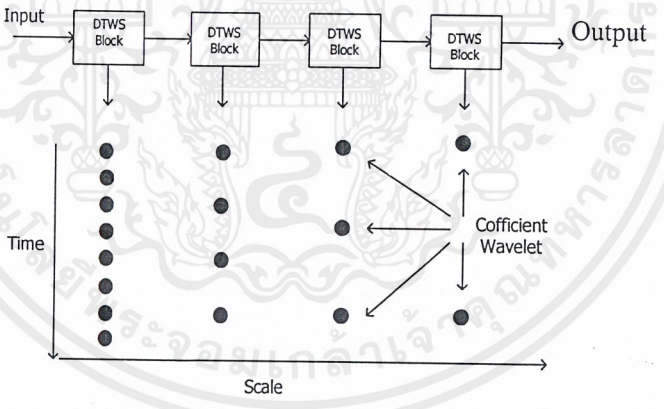
2.9 รูปแบบการแปลงเวฟเลตเต็มหน่วยเวลา

การแปลงเวฟเลตเต็มหน่วยเวลาประกอบด้วยค่าในโดเมนเวลา และ Scale Translation Domain (Wavelet Domain) จะเป็นจำนวนเต็มหน่วยที่ค่าอัตราส่วนย่อยของแกนเวลา (Scale) เท่ากับสอง แล้วทำการ Scaling ด้วยค่าอัตราส่วนสอง สามารถที่จะทำได้ง่าย และมีประสิทธิภาพโดยการตัด Sampling ของข้อมูลนั้นออกไปครึ่งหนึ่ง วิธีการนี้เรียกว่า Decimation หรือ Subsampling ด้วยสอง รูปแบบทั่วไปของการแปลงเวฟเลตที่มีค่าของการย่อขนาดเป็นค่ากำลังสองดังรูปที่ 2.9 จากกระบวนการตามรูปที่ 2.9 สามารถนำไปใช้สำหรับการแปลงเวฟเลตแบบ Multiresolution, Orthogonal และ Perfect Recontruction – Quadature Mirror Filter (PRQMF) ซึ่งในแต่ละแบบจะมีคุณสมบัติของการกรองที่แตกต่างกัน



รูปที่ 2.9 กระบวนการแปลงแบบ DWTS ที่มีการ Scaling เท่ากับสอง

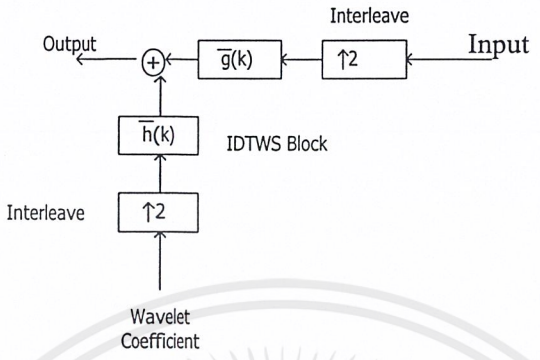
ตัวกรองจะถูกแบ่งเป็นตัวกรองความถี่ต่ำผ่าน และตัวกรองความถี่สูงผ่านที่กำหนดด้วยค่าของสัมประสิทธิ์ *g(k)* และ *h(k)* ตามลำดับ ซึ่งค่าสัมประสิทธิ์ของตัวกรองนี้ก็จะสัมพันธ์กับเวฟเลตแม่ที่ใช้ร่วมกับการแปลงเวฟเลต ขบวนการของการแตกกระจายเวฟเลตจะแสดงตามรูปที่ 2.10



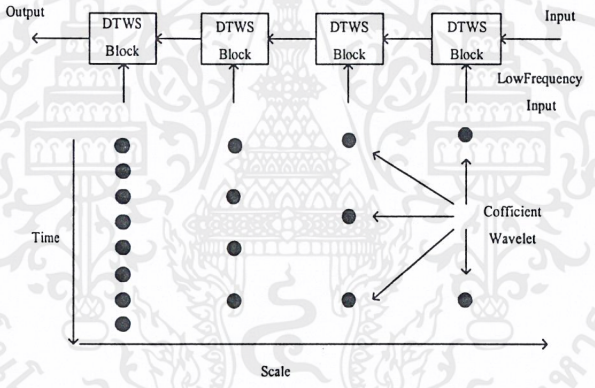
รูปที่ 2.10 การแตกกระจายเวฟเลตด้วย DWTS Block

ขบวนการทั้งหมดจะเกิดขึ้นมาจากการนำเอาบล็อกย่อยๆ ของ DWTS ที่มีค่าของการ Scaling เท่ากับสองมาต่อเข้าด้วยกัน สัญญาณเอาต์พุตที่ได้จากบล็อกแรก ในส่วนของตัวกรองความถี่ต่ำผ่านจะไปเป็นสัญญาณอินพุตของส่วนต่อไป จากรูปที่ 2.10 จะเห็นว่าสัญญาณอินพุตถูกป้อนเข้าทางซ้ายมือ ค่าสัมประสิทธิ์ที่ได้ของเวฟเลตจะออกมาทางด้านล่าง ซึ่งมีจำนวนลดลงเรื่อยๆ และสุดท้ายส่วนของสัญญาณความถี่ต่ำที่เหลือออกมาทางด้านขวามือนั่นเอง สัญญาณ และค่า

สัมประสิทธิ์ที่ได้ออกมาจะเป็นตัวแทนของสัญญาณในโดเมนเวลา นั่นเอง การแปลงกลับเวฟเลต หรือ "การรวมกลับเวฟเลต" แบบเต็มหน่วยเวลา สามารถทำตามวิธีการดังรูปที่ 2.11 และ 2.12



รูปที่ 2.11 กระบวนการแปลงแบบ IDWTS ที่มีการ Scaling เท่ากับสอง



รูปที่ 2.12 แสดงการรวมกลับของเวฟเลตแบบ IDTWS Block

จากรูปที่ 2.12 จะมีการทำงานที่กลับกันกับการทำงานของ DWTS โดยจะรับสัญญาณความถี่ต่ำ และสัมประสิทธิ์เวฟเลตเพื่อนำมาสังเคราะห์หรือสร้างสัญญาณในโดเมนเวลาสำหรับการ Decimation หรือ Sup Sampling ด้วยอัตราส่วนสองเสร็จแล้ว จะถูกแทนด้วยกระบวนการที่กลับกัน คือ กระบวนการ Interleave หรือ Up Sampling ที่จะแทรกข้อมูล 0 เข้าไปในแต่ละ Sampling

2.10 Multiresolution Wavelet Transform

ในการทำ Multiresolution Wavelet Transform นั้นฟังก์ชันที่นำมาเป็นเวฟเลตแม่อาจจะมีคุณสมบัติแบบ Non Orthogonal หรืออื่นๆ ก็ได้ โดยที่ข้อกำหนดเบื้องต้นของตัวเวฟเลตแม่ หรือตัวกรองความถี่สูงผ่านจะต้องมีความสัมพันธ์กับฟังก์ชันที่เป็น Scaling Function หรือตัวกรองความถี่ต่ำผ่านด้วยการทำ Multiresolution Wavelet Transform นี้มีคุณสมบัติของ ค่าสัมประสิทธิ์ที่ได้ออกแบบมาเป็นแบบ Pyramidal Structure อยู่ในตัวซึ่งคุณสมบัตินี้ไม่จำเป็นสำหรับการแปลงเวฟเลต โดยทั่วไปแต่ด้วยคุณสมบัติ Pyramidal Structure ทำให้การแปลง เวฟเลตที่ระดับสเกลต่างๆ สามารถที่จะทำได้โดยใช้ Scaling Function (Low Pass Fillter) และ Wavelet Function (High Pass Fillter) เพียงตัวเดียว ซึ่งสามารถทำได้ง่ายในทางปฏิบัติ โดยการทำงานซ้ำ

2.11 หลักการเข้ารหัสแบบฮัฟฟ์แมน

การเข้ารหัสแบบฮัฟฟ์แมนจะใช้ทฤษฎีความน่าจะเป็นในการเข้ารหัสความน่าจะเป็นที่กล่าวถึงนี้ คือ ความซ้ำซ้อนของข้อมูลที่จะเกิดขึ้นในข้อมูลชุดใหญ่ จะทำการเข้ารหัสฮัฟฟ์แมนด้วยข้อมูลบิตสั้นลง ตามความน่าจะเป็นของข้อมูลที่จะเกิดขึ้น ถ้าความน่าจะเป็นเกิดขึ้นมากจะแทนข้อมูลด้วยรหัสฮัฟฟ์แมนบิตสั้น และถ้าความน่าจะเป็นเกิดขึ้นน้อยก็แทนด้วยรหัสที่มีจำนวนบิตยาวซึ่งจะแสดงการเข้ารหัสฮัฟฟ์แมนโดยการสมมุติมีข้อมูลอยู่ชุดหนึ่งมีความน่าจะเป็นดังตารางที่ 2.1

ตารางที่ 2.1 ความน่าจะเป็นของข้อมูล

Symbols	Probability
S0	40%
S1	24%
S2	10%
S3	9%
S4	7%
S5	5%
S6	3%
S7	2%
Total	100%

2.11.1 หลักการเข้ารหัสฮัฟฟ์แมน

การเข้ารหัสฮัฟฟ์แมนเมื่อได้ความน่าจะเป็นของข้อมูลจะต้องจัดโครงสร้างข้อมูลให้อยู่ในรูปแบบของฮัฟฟ์แมนทรีสามารถกระทำเป็นขั้นตอนได้ดังนี้ คือ

1) เริ่มจับคู่จากข้อมูลที่มีความน่าจะเป็นน้อยที่สุดไปยังข้อมูลที่มีความน่าจะเป็นมากที่สุด

2) นำข้อมูลที่มีความน่าจะเป็นน้อยกว่าไว้ทางขวามือ

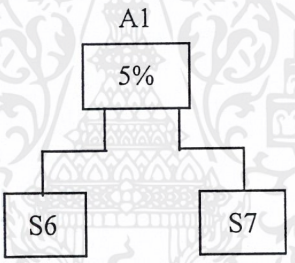
3) เมื่อทำการจับคู่ข้อมูลแล้วจะได้ความน่าจะเป็นของข้อมูลเพิ่มขึ้นให้ทำการเลื่อนความน่าจะ เป็นเพิ่มขึ้นไปเพื่อที่จะจับคู่กับข้อมูลตัวที่มีความน่าจะเป็นรียงลำดับกัน

4) ทำการจับคู่ชุดข้อมูลจนกว่าจะมีความน่าจะเป็นร้อยละ100

5) จัดโครงสร้างข้อมูลเป็นฮัฟฟ์แมนทรี

6) แทนรหัสฮัฟฟ์แมนตามโครงสร้างข้อมูลแบบฮัฟฟ์แมนทรี

2.11.2 การจัดโครงสร้างข้อมูลแบบฮัฟฟ์แมนทรี



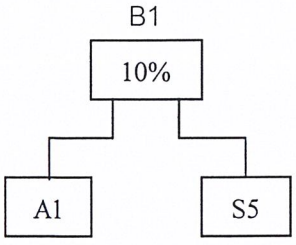
รูปที่ 2.13 การจับคู่ระหว่าง S6 และ S7

จากรูปที่ 2.13 สามารถอธิบายได้ว่า

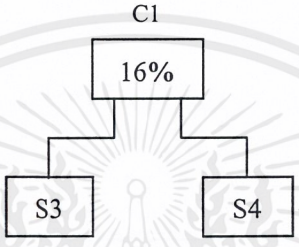
A1 เป็นรหัสเพื่อให้การจับคู่ดูง่ายในการที่จะไปจับคู่กับตัวถัดไป

S7 อยู่ทางขวามือเพราะความน่าจะเป็นต่ำกว่า S6

การจับคู่ระหว่าง S6 และ S7 จะได้ความน่าจะเป็นเพิ่มขึ้นแล้วนำไปจับกับ S5 แสดงได้ดังรูปที่ 2.14 B1 เกิดจากการรวมของ S5 และ A1 ซึ่งเมื่อทั้งสองรวมกันได้ความน่าจะเป็นร้อยละ 10 จะได้ความน่าจะเป็นไปอยู่ต้นๆ จากนั้นทำการจับคู่ S3, S4 ซึ่งมีความน่าจะเป็นร้อยละ 9 และร้อยละ 7 ดังรูปที่ 2.15 ซึ่งสามารถสรุปข้อมูลได้ดังตารางที่ 2.2



รูปที่ 2.14 การจับคู่ระหว่าง A1 และ S5

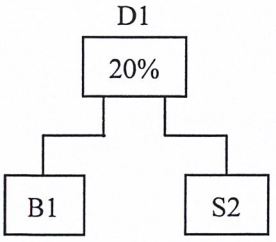


รูปที่ 2.15 การจับคู่ระหว่าง S3 และ S4

ตารางที่ 2.2 การเปรียบเทียบความน่าจะเป็นของชุดข้อมูลกับข้อมูล

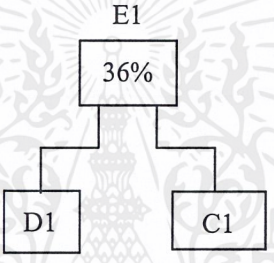
Symbols	Probability
S0	40%
S1	24%
C1	16%
B1	10%
S2	10%

จับคู่ระหว่าง S2, B1 โดยเอา S2 ไว้ทางขวามือเพราะความน่าจะเป็นน้อยกว่า ดังรูปที่ 2.16



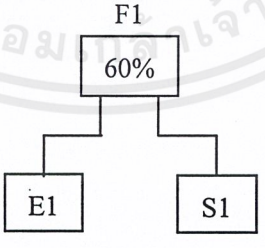
รูปที่ 2.16 การจับคู่ระหว่างS2 และ B1

จับคู่ตามลำดับความน่าจะเป็นระหว่าง C1, D1 โดยเอา C1 ไว้ทางขวามือเพราะความน่าจะเป็นน้อยกว่า ดังรูปที่ 2.17



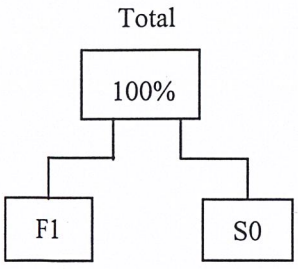
รูปที่ 2.17 การจับคู่ระหว่าง C1 และ D1

จับคู่ตามลำดับความน่าจะเป็นระหว่าง S1, E1 โดยเอา S1 ไว้ทางขวามือดังรูปที่ 2.18



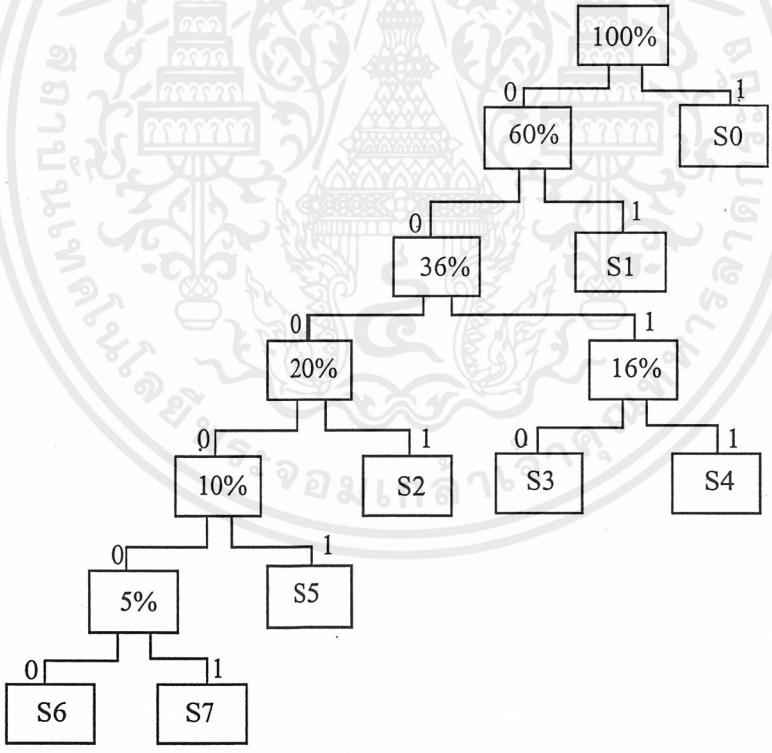
รูปที่ 2.18 การจับคู่ระหว่าง S1 และ E1

สุดท้ายในการจับคู่ คือ S0, F1 โดยเอา S0 ทั้งคู่รวมความน่าจะเป็นได้ร้อยละ 100 ดังรูปที่ 2.19



รูปที่ 2.19 การจับคู่ระหว่าง S0 และ F1

นำผลการจับคู่อันดับสุดท้ายมาขยายส่วนย่อยเพื่อจัดโครงสร้างข้อมูลแบบฮัฟฟ์แมนทรี ซึ่งชุดข้อมูล F1 เกิดจาก S1, E1 ชุดข้อมูล E1 เกิดจาก D1, C1 ชุดข้อมูล C1 เกิดจาก S4, S3 ชุดข้อมูล D1 เกิดจาก S2, B1 ชุดข้อมูล B1 เกิดจาก S5, A1 และชุดข้อมูล A1 เกิดจาก S7, S6 สามารถเขียนเป็นฮัฟฟ์แมนทรีได้ดังนี้



รูปที่ 2.20 การจัดโครงสร้างข้อมูลเป็นฮัฟฟ์แมนทรี

2.12 การแทนรหัสฮัฟฟ์แมน

จากโครงสร้างข้อมูลแบบฮัฟฟ์แมนทรีสามารถแทนรหัสฮัฟฟ์แมน ตามตารางที่ 2.3

ตารางที่ 2.3 การแทนรหัสฮัฟฟ์แมน

Symbols	Probability	Huffman Codes
S0	40%	1
S1	24%	01
S2	10%	0001
S3	9%	0010
S4	7%	0011
S5	5%	00001
S6	3%	000000
S7	2%	000001
Total	100%	

การเก็บข้อมูลที่เป็นรหัสฮัฟฟ์แมนจะทำการจัดเก็บข้อมูลแบบอนุกรมดังนั้นในแต่ละตำแหน่งของหน่วยความจำ สำหรับแต่ละตำแหน่งของหน่วยความจำจะมีส่วนของรหัสแบบปะปนกันอยู่ ซึ่งการถอดรหัสข้อมูลก็จะมีกรรมวิธีที่เป็นเฉพาะของฮัฟฟ์แมน ซึ่งจะได้กล่าวโดยสังเขปต่อไป

2.13 การถอดรหัสฮัฟฟ์แมน

สำหรับหลักการถอดรหัสฮัฟฟ์แมนนั้นหลักการถอดรหัสฮัฟฟ์แมนที่ได้จัดเก็บในหน่วยความจำจะเรียงต่อกันอยู่แบบอนุกรม สำหรับรหัสบางรหัสอาจมีบางส่วนที่อยู่ต่างตำแหน่งของหน่วยความจำ เพราะการจัดเก็บรหัสฮัฟฟ์แมนจะทำการแบบเรียงต่อในหน่วยความจำกันไปดังนั้นส่วนการถอดรหัสจึงจำเป็นต้องดึงข้อมูลออกมาเป็นอนุกรมแล้วทำการตรวจสอบข้อมูลที่เป็นบิต 0 ที่อยู่หน้ารหัสฮัฟฟ์แมน แล้วทำการเปรียบเทียบกับตารางที่ 2.4 เปรียบเทียบรหัสที่จัดเก็บเอาไว้ในหน่วยความจำในหน่วยความถาวร เมื่อได้รหัสออกมาแล้วก็ทำการถอดรหัสฮัฟฟ์แมนเป็นข้อมูล

ตารางที่ 2.4 ตัวอย่างรหัสฮัฟฟ์แมนที่นำมาถอดรหัส

Code length	Symbols	Huffman Code	Code1	Code2
1	S1	1	-	1
3	S2	010	0	10
3	S3	011	0	11
4	S4	0010	00	10
4	S5	0011	00	11
5	S6	00011	000	11
6	S7	0001 00	000	100
6	S8	0001 01	000	101
7	S9	0000 001	0000	1
7	S10	0000 100	0000	100
7	S11	0000 101	0000	101
7	S12	0000 110	0000	110
7	S13	0000 111	0000	111
8	S14	0000 0011	0000 00	11
8	S15	0000 0100	0000 0	100
8	S16	0000 0101	0000 0	101
9	S17	0000 0000 1	0000 0000	1
9	S18	0000 0001 0	0000 000	10
9	S19	0000 0001 1	0000 000	11
9	S20	0000 0010 0	0000 00	100
9	S21	0000 0010 1	0000 00	101

จากตาราง 2.4 เราสามารถนำมาถอดรหัสข้อมูลได้ดังนี้ คือ จะใช้วงจรตรวจสอบบิตที่เป็น 0 ก่อนหรือดูที่โค้ด 1 ก่อนที่จะถึงข้อมูลโค้ด 2 นั่นเองจะวิเคราะห์จากโค้ด 1 มี 0 กี่ตัวแล้วนำเอาโค้ด 2 ไปถอดรหัสเป็นข้อมูลต่อไป

สมมุติข้อมูลที่ได้จากหน่วยความจำมีดังนี้ คือ 00111000111000101111000011 ทำการตรวจสอบบิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 0011 ตรงกับ S5 ดึงข้อมูลนี้ออกไปข้อมูลที่ถอดรหัสต่อไป คือ

10001110001011110000111

ทำการตรวจสอบบิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 1 ตรงกับ S1 เลื่อนรหัสออกไปส่วนของรหัสฮัฟฟ์แมนที่เหลือต้องถอดรหัสต่อไป คือ

0001110001011110000111

ทำการตรวจสอบบิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 00011 ตรงกับ S6 เลื่อนรหัสออกไปส่วนของรหัสฮัฟฟ์แมนที่ต้องถอดรหัสต่อไป คือ

10001011110000111

ทำการตรวจสอบบิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 1 ตรงกับ S1 เลื่อนรหัสออกไปส่วนของรหัสฮัฟฟ์แมนที่ต้องถอดรหัสต่อไป คือ

0001011110000111

ทำการตรวจสอบบิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 000101 ตรงกับ S8 เลื่อนรหัสออกไปส่วนของรหัสฮัฟฟ์แมนที่ต้องถอดรหัสต่อไป คือ

1110000111

ทำการตรวจสอบบิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 1 ตรงกับ S1 เลื่อนรหัสออกไปส่วนของรหัสฮัฟฟ์แมนที่ต้องถอดรหัสต่อไป คือ

110000111

ทำการตรวจสอบบิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 1 ตรงกับ S1 เลื่อนรหัสออกไปส่วนของรหัสฮัฟฟ์แมนที่ต้องถอดรหัสต่อไป คือ

10000111

ทำการตรวจสอบบิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 1 ตรงกับ S1 เลื่อนรหัสออกไปส่วนของรหัสฮัฟฟ์แมนที่ต้องถอดรหัสต่อไป คือ

0000111

บิต 0 ทางซ้ายมือจะได้รหัสดังนี้ 0000111 ตรงกับ S13

ดังนั้น ข้อมูล (Symbol) ที่ได้ถอดรหัสตามลำดับ คือ

S5, S1, S6, S1, S8, S1, S1, S1, S13

2.14 ภาษา VHDL

2.14.1 ประวัติความเป็นมาภาษา VHDL

วิวัฒนาการของภาษา VHDL นั้นเริ่มต้นประมาณปี ค.ศ. 1981 โดยที่กระทรวงกลาโหมสหรัฐอเมริกา หรือ Department of Defense (DOD) มองเห็นว่าอุปกรณ์อิเล็กทรอนิกส์ และคอมพิวเตอร์ที่ใช้ในกิจการทหารเป็นอุปกรณ์ที่ได้รับการพัฒนามาเมื่อ 20 ปีก่อน เพราะเทคโนโลยีในขณะนั้นทำให้การพัฒนาทางอุปกรณ์อิเล็กทรอนิกส์เป็นไปอย่างช้า แต่อุปกรณ์ทางด้านไมโครอิเล็กทรอนิกส์ ได้รับการพัฒนาไปอย่างรวดเร็วซึ่งจะเห็นได้จากวงจรอิเล็กทรอนิกส์หลายวงจรในปัจจุบันสามารถที่จะใช้เทคโนโลยีการออกแบบ และผลิตวงจรรวมขนาดใหญ่มาก หรือ VLSI ต่างๆ เหล่านั้นให้อยู่บนอุปกรณ์สารกึ่งตัวนำ

ฉะนั้นภาษาดังกล่าวจึงจัดเป็นภาษาโปรแกรมระดับสูง (High Level Language) เช่นเดียวกับภาษา Pascal, Fortran, ADA ซึ่งในทางวิศวกรรมการออกแบบเรียกว่า Hardware Description Language หรือ HDL ดังนั้นภาษามาตราฐานนี้จึงมีชื่อว่า VHSIC – HDL หรือ VHDL

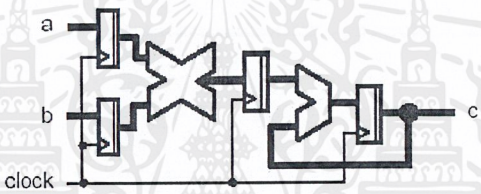
2.14.2 Top – Down Design

ภาษา VHDL เป็นภาษาที่นำมาใช้ในการกำหนด และบรรยายพฤติกรรมการทำงานของฮาร์ดแวร์ในระบบดิจิตอลนั้น คือ ความอ่อนตัวของภาษาที่สามารถจำลองการทำงานจากหลักการของรูปแบบ (Simulate Conceptual Designs) แต่ในขณะเดียวกันก็สามารถจำลองการทำงานของฮาร์ดแวร์ ที่ให้รายละเอียดเกี่ยวกับเวลาได้อย่างถูกต้อง (Timing Base Simulation) และจากโครงสร้างของภาษายังสามารถจำลองการทำงานในรูปของลำดับชั้น (Hierarchy of Simulation Levels) ความสามารถดังกล่าวจึงช่วยให้วิศวกรออกแบบสามารถที่จะเขียนรูปแบบการบรรยาย จากระดับบนสุดของวงจรที่อยู่ในรูปสังเขป (High Level of Abstraction) ลงสู่รายละเอียดในระดับล่างของวงจรได้ เช่น Gate Level เป็นต้น เพราะฉะนั้นภาษา VHDL จึงเป็นภาษาที่สนับสนุนการเขียนในรูปแบบในทุกๆ ลักษณะ และวิธีการดังเช่น ตัวอย่างที่แสดงในรูปที่ 1.1 คือ รูปแบบ (Model) ของลำดับขั้นตอนการคูณ และหาผลรวม (Multiply Accumulate Algorithm) ของตัวแปรสองตัว a และ b ส่วนผลลัพธ์ คือ c ในลักษณะของเลขฐานสอง (Binary Number) รูปแบบนี้แสดงในระดับที่สังเขปที่สุดของแนวความคิดที่จะแก้ปัญหาเพื่อหาผลลัพธ์ โดยไม่ได้คำนึงถึงโครงสร้างของวงจรอย่างที่เคยชิน เช่น อุปกรณ์วงจรรวม Arithmetic and Logic Unit (ALU)

```
FOR i IN 1 TO 1024 LOOP
        result := result + a(I) * b(I);
END LOOP;
C<= result;
```

รูปที่ 2.21 VHDL Statement สำหรับ Multiply Accumulate Algorithm

ในขณะอีกด้านหนึ่งของมุมมองในปัญหาเดียวกันนี้ ภาษา VHDL ก็สามารถใช้บรรยายลำดับขั้นตอนของการคูณ และหาผลรวม (Multiply Accumulate Algorithm) โดยแสดงได้ในรายละเอียดของการสร้างวงจรดังกล่าวจริงๆ ตามที่เห็นได้จากรูป 2.21



รูปที่ 2.22 โครงสร้าง Multiply – Accumulate Unit

ขั้นตอนของขบวนการออกแบบโดยใช้วิธี Top – Down Design มีรายละเอียดดังนี้

1) System Specification and Analysis ขั้นตอนของการสร้างข้อกำหนดความต้องการ (Specification) และวิเคราะห์ระบบ เพื่อหาความคิด และหลักการ (Idea and Concept) ในการแก้ปัญหา

2) Modeling and Simulation การเขียนรูปแบบของระบบที่ต้องการออกแบบโดยภาษา VHDL หรือภาษา HDL อื่นๆ จากแนวความคิดอย่างสังเขปที่ได้ สำหรับบรรยายพฤติกรรมการทำงานพร้อมทั้งจำลองการทำงาน เพื่อเปรียบเทียบ และตรวจสอบความถูกต้องกับข้อกำหนด (Specification)

3) Logic and Test Synthesis หลังจากที่ได้หลักการขั้นต้นพร้อมกับแนวความคิดที่ผ่านการตรวจสอบแล้วหลักการนี้จะถูกเพิ่มเติมในรายละเอียดลงมาเป็นลำดับขั้นตอนที่เหมือนกัน คือ Modeling and Simulation จนกระทั่งอยู่ในระดับที่จะนำไปผลิตวงจร หรือสังเคราะห์ (Synthesis) ในขั้นตอนนี้เองเทคโนโลยีที่จะมารองรับวงจรออกแบบจะถูกกำหนดขึ้นอยู่ในรูปของวงจรที่ประกอบด้วยอุปกรณ์อิเล็กทรอนิกส์ (Gate Level) และการเชื่อมต่อระหว่างกันของอุปกรณ์เหล่านั้น

หรือไม่ก็อยู่ในรูปของ nettles ที่สามารถนำไปผลิตลงบนอุปกรณ์อื่นได้ นอกจากนั้นการผลิตบางเทคโนโลยีอาทิ เช่น Gate Array หรือ Standard Cell และ Full Custom IC อาจจะต้องสร้างโครงสร้างของวงจรใหม่หลังจากที่สังเคราะห์ครั้งแรกแล้ว เพื่อความสะดวกต่อการตรวจสอบการทำงานหลังจากที่ผลิตเป็นวงจรต้นแบบแล้ว หรือที่เรียกว่า "Design For Test " (DFT) พร้อมทั้งข้อมูลในการตรวจสอบ (Test Pattern) จะถูกกำหนดในขั้นตอนนี้

4) Pre-Timing Verification หลังจากการสังเคราะห์วงจรให้อยู่ในรูป Gate – Level หรือ Nettles แล้วข้อมูลที่ได้จากผู้ผลิตอุปกรณ์วงจรนั้น นอกจากจะเป็นข้อมูลสำหรับจำลองการทำงานในเรื่องของความถูกต้องของฟังก์ชัน (Functional Simulation) แล้ว ยังมีข้อมูลเกี่ยวกับเวลาด้วย ซึ่งเป็นความจริงที่ว่าอุปกรณ์ทางอิเล็กทรอนิกส์ทุกชิ้นจะมี Propagation delay เสมอ ถึงแม้ว่าจะเป็นเวลาที่น้อยมากในระดับ Nanosecond (10^{-9}Second) แต่ภายในวงจรหนึ่งประกอบด้วย Gate ของฟังก์ชันต่างๆ จำนวน 10,000 Gate ขึ้นไป เวลาดังกล่าวนี้จะสะสมกันมากขึ้น จนอาจจะทำให้การทำงานของวงจรรวมทั้งหมดผิดไป หรือไม่สามารถทำงานในย่านความถี่สัญญาณนาฬิกาที่สูงได้

5) Physical Design and Analysis คือ ขั้นตอนของการผลิตวงจรจริง (Technology and Device Mapping) โดยนำข้อมูลที่ได้จากการสังเคราะห์มาผลิต ซึ่งอาจอยู่ในรูปของแผงวงจรไฟฟ้า (Printed Circuit Board: PCB) ที่ประกอบด้วยอุปกรณ์หลายชิ้น หรืออยู่ในรูปของวงจรรวมเฉพาะงาน (ASIC)

6) Post – Timing Verification หลังจากที่ได้วงจรจริงมาแล้ว ยังต้องมีความจำเป็นที่ต้องตรวจสอบการทำงานที่คำนึงถึงเวลาด้วย เพื่อความถูกต้องของวงจรครั้งสุดท้ายก่อนที่จะนำไปรวมเข้ากับอุปกรณ์อื่นๆ ให้เป็นระบบดิจิตอล เพราะในขั้นตอนนี้วงจรที่ออกแบบ จะประกอบด้วย Input Pad และ Output Pad ซึ่งเป็นจุดต่อสำหรับรับ และส่งสัญญาณกับภายนอก

7) System Level Verification หลังจากที่นำวงจรที่ออกแบบรวมเข้ากับอุปกรณ์อื่นๆ ให้เป็นระบบดิจิตอลแล้ว จะต้องทดสอบการทำงาน รวมทั้งระบบร่วมกับอุปกรณ์อื่นๆ อีกครั้ง เป็นการควบคุมคุณภาพของผลิตภัณฑ์

จากความอ่อนตัวของภาษา และความสามารถที่ใช้ในการเขียนรูปแบบได้หลายลักษณะนี้เอง VHDL จึงเป็นเครื่องมือที่ใช้สำหรับออกแบบตั้งแต่ขั้นตอนบนสุด คือ แนวความคิดที่จะแก้ปัญหาลงไปทีละขั้นตอนของการผลิตวงจรจริง (Form Idea to Implementation) ข้อดีที่เห็นได้ชัดของการนำ VHDL มาใช้ในการออกแบบลักษณะ Top – Down นี้ คือ วิศวกรสามารถที่จะสร้างรูปแบบ และจำลองการทำงานเพื่อตรวจสอบความถูกต้องกับข้อกำหนด (Specification) จากการจำลองการทำงานในระยะต้นๆ ของการออกแบบนั้น หลักการต่างๆ ที่ถูกกำหนดขึ้นใช้ในการแก้ปัญหาจะถูกตรวจสอบด้วยทุกครั้ง ก่อนที่จะมีการลงทุนในขั้นตอนสุดท้ายของการออกแบบ หรือการ

สร้างวงจรนั้น ซึ่งหมายความว่าข้อผิดพลาดที่อาจเกิดขึ้นได้จากหลักการที่กำหนดขึ้น จะถูกตรวจพบ และจัดการแก้ไขให้ถูกต้องได้ ก่อนที่จะทำงานในขั้นตอนต่อๆ ไปของขบวนการการออกแบบ

2.14.3 Terminology and Conventions

ในการเขียนโปรแกรมภาษาบรรยาย VHDL นั้นจะมีคำศัพท์เทคนิคเฉพาะสำหรับในการเขียนโปรแกรม สามารถอธิบายได้ดังนี้

ลักษณะของรูปแบบ (Model Styles): ลักษณะของการเขียนรูปแบบ (Model) ด้วยภาษา VHDL สามารถแบ่งได้เป็น

Behavioral Model : หรือเรียกอีกอย่างหนึ่งได้ว่า Algorithmic Description เป็นรูปแบบที่ใช้บรรยายพฤติกรรมของระบบดิจิตอล ในส่วนที่บรรยายมีโครงสร้างคล้ายกับภาษาชั้นสูง (High Level Language) ทั่วๆ ไป เช่น Pascal หรือ C เป็นต้น สำหรับในการจำลองการทำงาน (Simulation) คำสั่ง แต่ละคำสั่ง (Statement) จะถูกประเมินผลเป็นไปตามลำดับ (Sequential) จากบนลงล่าง ยกเว้นในกรณีคำสั่ง Loop หรือการเรียกใช้โปรแกรมย่อย รูปแบบลักษณะนี้จะไม่ให้รายละเอียดที่เกี่ยวกับผลิต หรือโครงสร้างของ ฮาร์ดแวร์ ตรงกันข้ามจะให้รายละเอียดเกี่ยวกับความสัมพันธ์ระหว่าง อินพุตกับเอาต์พุต

Data Flow Model : เรียกอีกอย่างหนึ่งได้ว่า “Register Transfer Level” (RTL) เป็น รูปแบบที่ถูกเขียนขึ้นเพื่อจุดประสงค์ที่จะใช้เครื่องมือสำหรับสังเคราะห์วงจรอัตโนมัติ รูปแบบลักษณะนี้ส่วนใหญ่จะเป็น Procedural Constructs และ Functional Operators

Structural Model : เป็นรูปแบบที่แสดงการเชื่อมต่อกันระหว่างอุปกรณ์ต่างๆ ที่ประกอบกันขึ้นเป็นวงจรระบบดิจิตอล และสามารถเรียกอีกอย่างหนึ่งได้ว่า “Nettles Representation” เป็นการเขียนที่แสดงให้เห็นโครงสร้างของ Hardware

Mixed – Level Model : จากคุณสมบัติที่อ่อนตัวของภาษา VHDL จึงสามารถที่จะเขียนรูปแบบ โดยใช้ลักษณะต่างๆ ที่กล่าวมาแล้วข้างต้น บรรยายวงจร หรือระบบดิจิตอลเดียวกันได้ ฉะนั้นรูปแบบนี้จึงมีการผสม

Concurrency : ในภาษา VHDL นั้นชุดคำสั่ง แต่ละชุดจะทำงานในเวลาเดียวกัน และอิสระต่อกัน ลักษณะเช่นนี้เป็นคุณสมบัติที่เป็นความจริงทางฟิสิกส์ ของวงจรอิเล็กทรอนิกส์ ชุดคำสั่งนี้เรียกว่า “Concurrent Statement” และจะทำงานก็ต่อเมื่อมีการเปลี่ยนแปลงค่าของสัญญาณ

Sequential : นอกจากความสามารถที่ชุดคำสั่งจะทำงานแบบ Concurrent แล้วบางครั้งการเขียนรูปแบบในลักษณะที่บรรยายพฤติกรรมของวงจรมีความจำเป็นที่จะต้องให้ชุดคำสั่งทำงานเป็นลำดับขั้นเรียงกันจากบนลงล่าง เช่นการเขียนแบบ Behavior Model เป็นต้น ชุดคำสั่งที่เป็นแบบ Sequential นี้จะใช้ในโปรแกรมย่อย และ Process Statement

Driver : สัญญาณต่างๆ ในVHDL นั้นจะถูกควบคุมด้วยตัวขับ หรือ “Driver” สัญญาณเหล่านี้จะรับค่าใหม่ (ระดับของสัญญาณ) ได้ด้วยตัวขับนี้เอง

Transaction : การเกิด Transaction กับ Signal นั้นจะเกิดขึ้นเมื่อมีการกำหนดค่าๆ หนึ่งให้กับ Signal นั้นค่าใหม่ที่ Signal ได้รับอาจจะมีผล หรือไม่มีผลทำให้เกิดการเปลี่ยนแปลงของระดับสัญญาณ เช่น การเปลี่ยนค่าจากระดับลอจิก 0 ไปเป็น ระดับลอจิก 1

Event : คือ การเปลี่ยนระดับลอจิกของ Signal จากระดับหนึ่งไปสู่ระดับอื่น อย่างเช่นในระบบดิจิตอลการเปลี่ยนจากระดับลอจิก 0 ไปเป็นระดับลอจิก 1 หรือในทางตรงกันข้ามถือว่าสัญญาณ นั้นเกิด “Event” ฉะนั้นจะเห็นได้ว่าการที่จะเกิด Event ได้นั้นก็จะต้องเกิด Transaction ด้วย แต่ในทางตรงกันข้ามการเกิด Transaction ไม่จำเป็นต้องเกิด Event ทุกครั้ง

Sensitivity List : คือ รายชื่อของ Signal ต่างๆ ที่มีผลให้เกิดการทำงานของ Concurrent Statement เมื่อเกิด Event ขึ้นกับ Signal ตัวใดตัวหนึ่ง หรือหลายตัวพร้อมกันในรายชื่อนั้น

Objects : ในภาษา VHDL นั้นคำว่า Object ใช้เขียนเพื่อบ่งบอกถึงองค์ประกอบส่วนหนึ่งของรูปแบบ ซึ่งเปรียบได้เหมือนกับภาชนะที่มีไว้สำหรับบรรจุค่าต่างๆ สามารถแบ่งออกได้เป็น 3 ชั้นด้วยกัน คือ

CONSTANT : ได้แก่ Object ประเภทหนึ่งที่เมื่อกำหนดค่าเริ่มต้นให้แล้วจะคงค่านั้นไว้ตลอดไม่สามารถที่จะดัดแปลง หรือแก้ไขได้

SIGNAL : หมายถึง Object ประเภทหนึ่งที่สามารถกำหนดค่าที่สัมพันธ์กับเวลาให้ได้ ซึ่งหมายความว่า Signal สามารถที่จะปรับค่าได้เพียงค่าเดียวเท่านั้น ซึ่ง Driver จะเก็บค่าในอนาคตสำหรับ Signal ไว้ด้วย Signal สามารถประกาศใช้ได้ในส่วน ที่เป็นเนื้อที่ของ Concurrent Body นั้น

VARIABLE : ได้แก่ Object ที่สามารถกำหนดค่าใดๆ ได้ และสามารถที่เปลี่ยนแปลงค่าได้ตลอดเวลาการจำลองการทำงานแต่จะเก็บค่าเพียงค่าเดียวเท่านั้นในเวลาหนึ่งเพราะเนื่องจาก Variable สามารถประกาศใช้ได้ในส่วนของ Sequential Body เท่านั้น อันได้แก่ส่วนประกาศของ Process Function หรือ Produce ดังนั้นตัวแปรจึงสามารถนำไปใช้ได้เฉพาะในขอบเขตที่ถูกประกาศ ใช้ VHDL นั้นประกอบด้วยส่วนต่างๆ ที่สำคัญ และเป็นพื้นฐานของการเขียนรูปแบบระบบดิจิตอลที่สำคัญ 4 หน่วย คือ

1) Entity Design Unit

2) Architecture Design Unit

3) Package Design Unit

4) Configuration Design Unit

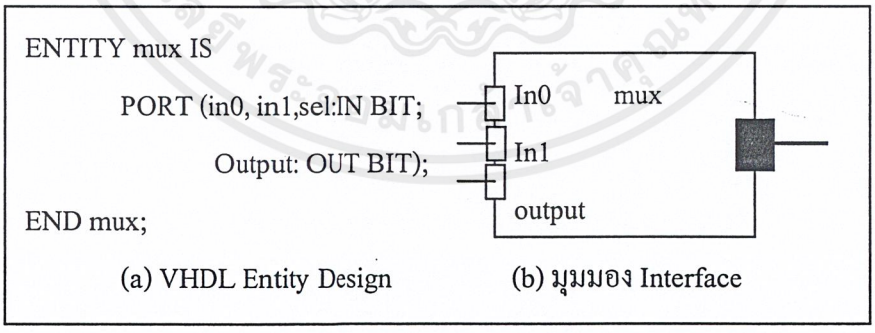
2.15 Entity Design Unit

หน่วยของแบบ Design unit ใช้ติดต่อระหว่างภายนอกกับรูปแบบที่จะเขียนขึ้นเรียกว่า " Entity Design Unit " ในส่วนนี้ใช้กำหนดจุดต่อ (Connection Point) ของรูปแบบกำหนดการไหลผ่านจุดต่อเหล่านั้น ดังรูปที่ 2.23 แสดงโครงสร้างอย่างง่ายๆ ของ Entity Design Unit

```
ENTITY component_name IS
        input and output ports
    Physical and other parameters
END [component_name ];
```

รูปที่ 2.23 โครงสร้างอย่างง่ายของ Entity Design Unit

ส่วนนี้จะขึ้นต้นด้วยคำว่า ENTITY และ IS ซึ่งระหว่างคำทั้งสองนี้เป็นส่วนสำหรับชื่อของรูปแบบที่ต้องการจะเขียน (Component_name) สำหรับกฎการตั้งชื่อนั้นจะต้องเป็นไปตามกฎเกณฑ์ของภาษาหลังจากนั้นจะตามด้วยส่วนที่ใช้กำหนดช่องทางเข้า–ออกของข้อมูล (Input - Output) รวมทั้งพารามิเตอร์อื่นๆ ส่วนนี้เรียกว่าส่วนหัว (Entity Header) และที่สำคัญ คือ Entity Design Unit จะต้องลงท้ายด้วยคำว่า END และเครื่องหมายอัฒภาค หรือ Semicolon (;)



รูปที่ 2.24 Multiplexer

ในรูปที่ 2.24 เป็นการบรรยายอุปกรณ์ที่มีชื่อว่า Mux ในส่วนหัวของ Entity (Header) มีการกำหนดจุดต่อ 4 จุดภายใต้ชุดคำสั่ง PORT โดยที่ 3 จุดแรกเป็นจุดให้ข้อมูลไหลผ่านเข้า (Input) ได้

แก่ in0, in1 และ sel ซึ่งกำหนดด้วยทิศทางกาติดต่อกับภายนอกเป็นการไหลเข้า (IN) ส่วนจุด Output เป็นจุดที่ข้อมูลไหลออก (OUT) ส่วนประเภท (Type) ของข้อมูลที่จะไหลเข้าออกนั้นเป็นประเภทบิต

2.16 Architecture Design Unit

ใช้เขียนบรรยายกำหนดพฤติกรรมของการจำลองการทำงานพฤติกรรมต่างๆ ที่ขึ้นอยู่กับข้อมูลที่ผ่าน เข้า–ออก ตรงช่องทางตลอดจนพารามิเตอร์ต่างๆ ที่กำหนดใน Entity Design Unit ดังแสดงในรูปที่ 2.25 Architecture Design Unit

```
ARCHITECTURE identifier OF component_name IS
        [declaration]
BEGIN
        Specification of the functionnality of the
        Component in terms of its input lines and as
        Influenced by physical and other parameters
END [identifier ] ;
```

รูปที่ 2.25 โครงสร้างอย่างง่าย ของ Architecture Design Unit

ส่วนของ Architecture Design Unit นั้น เริ่มต้นด้วยคำว่า Architecture และตามด้วยชื่อสิ่งที่แสดงให้เห็น Architecture นั้นใช้บรรยาย Entity Design Unit ใด (OF < Entity Design Unit) ส่วนที่อยู่ระหว่าง Architecture และ Begin เป็นส่วนประกาศกำหนด Architecture Declarative Area ที่เป็นส่วนเผื่อเลือก (Option) ในบริเวณนี้สามารถใช้เขียนประกาศกำหนดค่าต่างๆ ที่จะนำไปใช้ภายใน Architecture นั้นได้ อาทิ เช่น ประเภทต่างๆ ตัวอย่างเช่น BIT, BIT_Vector, สัญญาณ (Signal), ตัวคงที่ (Contant) , โปรแกรมย่อย และอุปกรณ์ ส่วนที่ใช้บรรยายความสัมพันธ์ระหว่างข้อมูลที่ไหลเข้า และไหลออกของรูปแบบนั้นจะถูกบรรยายในบริเวณเนื้อที่ระหว่างคำว่า Begin กับ End ของ Architecture Design Unit และนอกจากนั้นชุดคำสั่งทุกคำสั่งที่อยู่ภายในบริเวณนี้จะเป็นชุดคำสั่งที่อยู่ภายในบริเวณนี้จะเป็นชุดคำสั่งแบบแข่งขนานเท่านั้น Architecture Design Unit จะต้องปิดท้ายด้วยคำสั่ง End และชื่อของ Architecture นั้นๆ ที่เป็นส่วนเผื่อเลือกการเขียนรูปแบบระบบดิจิตอลด้วยภาษา VHDL สามารถเขียนได้ในลักษณะต่างๆ ดังนี้

1) Dataflow Description

2) Behavioral Description

3) Structure Description

4) Mixed Model Description

2.17 โครงสร้างภายในของอุปกรณ์ FPGAs

FPGA จัดเป็นวงจรรวมเฉพาะกิจชนิดหนึ่งที่สามารถจะโปรแกรมเป็นวงจรเชิงเลขใดๆ ก็ได้ เช่นเดียวกับ EPLD ต่างกันที่ EPLD โปรแกรมลงบน EPROM ภายใน และสามารถโปรแกรมใหม่ได้หลังจากนำไปลบด้วยแสง UV แต่ใน FPGAs สามารถโปรแกรมลงบนสแตติกแรมภายในด้วยข้อมูลที่อยู่ภายนอก จะทำให้สามารถโปรแกรมใหม่ได้โดยการรีเซตด้วยสัญญาณจากภายนอก และนอกจากนั้น FPGAs ยังประหยัด และมีความจุวงจรสูง

วงจรรวมชนิดที่ใช้ในโครงงานนี้ผลิตโดยบริษัทไซลิงค์ (Xilinx) ซึ่งเป็นบริษัทที่ทำการค้นคว้าร่วมกับบริษัทเอ็มเอ็มไอสร้างกลุ่มของเกตไว้จำนวน 600 – 25,000 เกต ดังในตารางที่ 2.5 การที่ต้องการบอกขนาดของวงจรรวมเป็นจำนวนเกตเพราะจะได้รู้ว่าขนาดของ วงจรที่ได้ออกแบบไว้สามารถโปรแกรมลงบน FPGAs ได้หรือไม่ FPGAs มีโครงสร้างภายในใกล้เคียงกับสถาปัตยกรรมเกตอะเรย์ (GAL, Gate Array Logic) มากสามารถโปรแกรม และลบคอนฟิกกูเรชั่น (Configuration) ภายใน สแตติกแรม (Static RAM) ได้โดยใช้กระแสไฟฟ้าซึ่งทำการโปรแกรมได้ โดยดึงข้อมูลฐานสิบหกมาจากภายนอก FPGAs ตัวแรกผลิตโดยบริษัท ไซลิงค์ คือ เบอร์ XC2064 (2000 Family) ประกอบด้วยเซลล์เรียงกันเป็นเมตริกซ์ (Matrix) เป็นจำนวน 64 เซลล์ หลังจากนั้นผลิต FPGAs ตระกูล 3000 และ แต่ละเซลล์เรียกว่า CLB (Configuration Logic Block)

ตารางที่ 2.5 คุณสมบัติของ FPGAs ตระกูลต่างๆ

FPGAs	APP Gate Count	Max I/O	Flip-Flop	RAM Bits	Available CLBs
XC2064	1,000	58	122	0	64
XC2018	1,500	74	174	0	100
XC3020/3120	1,800	64	256	0	64
XC3030/3130	2,700	80	360	0	100
XC3042/3142	3,700	96	480	0	144

ตารางที่ 2.5 (ต่อ) คุณสมบัติของ FPGAs ตระกูลต่างๆ

FPGAs	APP Gate Count	Max I/O	Flip-Flop	RAM Bits	Available CLBs
XC3064/3164	5,500	120	688	0	244
XC3091/3190	7,500	144	982	0	320
XC3195	9,000	176	1320	0	48
XC4002A	2,000	64	256	2048	64
XC3003/4003A	3,000	80	360	3200	100
XC4000H	3,000	160	200	3200	100
XC4004A	4,000	960	480	4608	144
XC4005/4005A	5,000	122	616	6072	196
XC4005H	5,000	192	392	6272	196
XC4006	6,000	128	768	8192	256
XC4008	8,000	144	936	10368	324
XC4010	10,000	160	1120	12800	400
XC4013	13,000	192	1536	18432	576
XC4052	25,000	256	2560	32768	1024

ตารางที่ 2.6 ประมาณการนับเกตพื้นฐาน

Gate	Equivalent gate Count	Gate	Equivalent gate Count
INV	1	RS Latch	3
NAIINR2	1	D Latch	4
NAIINR3	2	D Latch with CLR	5
NAIINR4	2	D Latch with PRE	5
NAIINR6	5	D Latch with PRE/CLR	6
NAIINR8	6	DF/F	6
NAIINR9	7	DF/F with CLR	7

ตารางที่ 2.6 (ต่อ) ประมาณการนับเกตพื้นฐาน

Gate	Equivalent gate Count	Gate	Equivalent gate Count
NAIINR12	8	DF/F with PRE	7
NAIINR16	11	DF/F with PRE/CLR	8
BUFF	2	JK F/F with CLR	9
ANIIOR2	2	JK F/F with PRE	12
ANIIOR3	2	JK F/F with PRE/CLR	13
ANIIOR4	3	TF/F with CLR	8
XOR2	3	TF/F with PRE	8
XNOR2	3	TF/F with PRE/CL	9

2.18 ส่วนอินพุต และเอาต์พุตของอุปกรณ์ FPGAs

รอบนอกของ FPGAs จะประกอบไปด้วย IOBs ประมาณ 64 ถึง 144 ตัว ซึ่งขึ้นอยู่กับตระกูลของ FPGAs ซึ่ง IOBs จะเป็นตัวเชื่อมต่อระหว่างภายในกับภายนอกของวงจรลอจิกของ FPGAs ลักษณะของ IOBs จะมีลักษณะ 2 ทิศทาง สามารถโปรแกรมให้เป็นอินพุต หรือเอาต์พุตก็ได้

2.19 รายละเอียดการใช้งานอุปกรณ์ FPGAs

FPGAs สามารถทำงานได้หลายลักษณะ โดยกำหนดได้ที่ขาสัญญาณ M0, M1 และ M2 ดังในตาราง 2.7 แสดงโหมดต่างๆ ของการคอนฟิกูเรชันในลักษณะมาสเตอร์พาราแรลสำหรับโปรแกรมคอนฟิกทีละ 1 ไบต์ (Byte) จากหน่วยความจำภายนอกที่เป็นแบบขนาน โดยสามารถรับโปรแกรมคอนฟิก (Config) จากแอดเดรส (Address) ต่ำ หรือสูงก่อนก็ได้ สำหรับการต่อใช้งานในลักษณะเพอริเฟอรัล (Peripheral) จะรับโปรแกรมคอนฟิกทีละ 1 ไบต์ จากไมโครโปรเซสเซอร์ โดยสามารถโต้ตอบกันได้ว่าพร้อมที่จะรับข้อมูล หรือไม่ การต่อลักษณะสเลฟซีเรียล (Slave Serial) รับโปรแกรม คอนฟิกทีละ 1 บิตจากไมโครโปรเซสเซอร์ตามสัญญาณนาฬิกา ส่วนการต่อลักษณะมาสเตอร์ซีเรียล (Master Serial) จะรับโปรแกรมทีละ 1 บิต จากหน่วยความจำภายนอกที่เป็นแบบอนุกรม

จากความต้องการในการสร้างให้สามารถใช้กระแสไฟฟ้าต่ำจากลักษณะการต่อใช้งาน 5 แบบ จึงมีเพียง 2 แบบเท่านั้นที่เหมาะสม คือ แบบมาสเตอร์ซีเรียล และแบบสเลฟซีเรียลส่วนแบบมาสเตอร์พาราแรลต้องใช้ EPROM 27CXXX ซึ่งกินกระแสมากกว่า PROM XC17XXX เหมาะในการทดสอบต้นแบบก่อน เมื่อวงจรต้นแบบทำงานได้ถูกต้องแล้วจึงทำการอัดโปรแกรมลง PROM อีกทีหนึ่งเพราะว่าในแบบพาราแรลนั้น EPROM สามารถโปรแกรมได้ใหม่ต่างกับ PROM ที่โปรแกรมลงครั้งเดียว

ตารางที่ 2.7 โหมดต่างๆ ของการคอนฟิกกูเรชัน

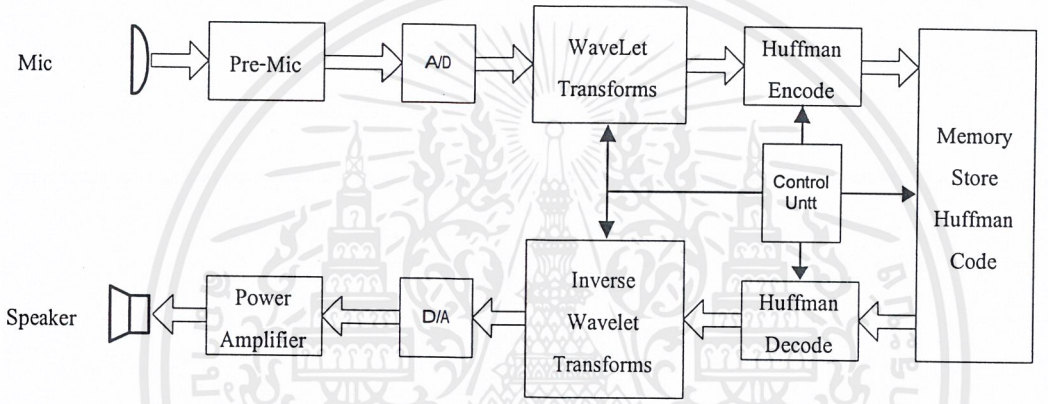
Mode	M2	M1	M1	M0	Data
Master Serial	0	0	0	Output	Bit-Serial
Slave Serial	1	1	1	Input	Bit-Serial
Master Parallel Up	1	0	0	Output	Byte-Wide,00000up
Master Parallel Down	1	1	0	Output	Byte-Wide,3FFF Down
Peripheral Synchr	0	1	1	Input	Byte-Wide
Peripheral Asynchr	1	0	1	Output	Byte-Wide
Reserved	0	1	0		
Reserved	0	0	1		

การใช้งาน FPGAs ด้วยการต่อในลักษณะสเลฟซีเรียล และมาสเตอร์ซีเรียล เมื่อเริ่มทำการจ่ายไฟเข้าตัว FPGAs จะทำการลบข้อมูลภายในหน่วยความจำที่ใช้ในคอนฟิกว่าเป็นลักษณะใดในตารางที่ 7 เป็นแบบอนุกรม หรือแบบขนานหลังจากนั้นจะเริ่มทำการโปรแกรมคอนฟิกสัญญาณ DONE/PROGRAM มีระดับลอจิกเป็น 0 ซึ่งอยู่ในระหว่างโปรแกรม และเมื่อข้อมูลในคอนฟิกที่รับมาจากภายนอกเต็มหน่วยความจำที่ใช้งานอยู่ในคอนฟิก และความยาวของข้อมูลตรงกับส่วนหัวของข้อมูลในคอนฟิกก็จะปรากฏสัญญาณ DONE /PROGRAM จะมีระดับลอจิกเป็น 1 ซึ่งหมายถึงโปรแกรมทำการคอนฟิกเสร็จเรียบร้อยแล้ว

บทที่ 3
การออกแบบ การสร้าง และการทำงาน

3.1 ความคิดเบื้องต้นในการสร้างเครื่องบันทึกเสียงพูดระบบดิจิตอล

เครื่องบันทึกเสียงพูดระบบดิจิตอลนั้นสามารถออกแบบได้โดยใช้ทั้งส่วนที่เป็นฮาร์ดแวร์และส่วนที่ซอฟต์แวร์ร่วมกัน โดยมีหลักการทำงานดังนี้



รูปที่ 3.1 ผังการทำงานของวงจรเครื่องบันทึกเสียงพูดระบบดิจิตอล

จากผังการงานรูปที่ 3.1 สัญญาณเสียงจะถูกแปลงให้เป็นสัญญาณไฟฟ้าโดยไมโครโฟน ผ่าน วงจรภาคปรีไมโครโฟนสำหรับขยายสัญญาณให้มีความแรงของสัญญาณเพียงพอที่จะส่งต่อไปยัง ภาค Analog to Digital (A/D) สำหรับแปลงสัญญาณเสียงที่ได้เป็นสัญญาณดิจิตอลขนาด 8 บิต ทำการประมวลผลลดรูปขนาดสัญญาณเสียงที่รับเข้ามาโดยสมการเวฟเลต ซึ่งเป็นการนำสัญญาณที่ได้มาแยกความแตกต่างทางด้านความถี่ด้วยสมการเวฟเลต เพื่อให้เกิดความซ้ำซ้อนของข้อมูลได้เพิ่มขึ้นทำการจัดระดับสัญญาณแล้วส่งต่อไปยังภาคของฮัฟฟ์แมน เพื่อลดบิตข้อมูลจากความซ้ำซ้อนซึ่งได้จากคุณสมบัติจากเวฟเลต และนำรหัสฮัฟฟ์แมนเข้าไปเก็บไว้ในหน่วยความจำขนาด 8 บิต 128 กิโลไบต์

เมื่อต้องการที่จะอ่านสัญญาณเสียงจะนำรหัสฮัฟฟ์แมนที่ได้จัดเก็บไว้ในหน่วยความจำโดยการอ่านข้อมูลจากหน่วยความจำไปยังภาคถอดรหัสฮัฟฟ์แมน เพื่อคืนค่าข้อมูลที่ได้จากการจัด

ระดับแล้ว ส่งต่อไปยังภาคของ Inverse Wavelet Transforms สำหรับแปลงกลับข้อมูลจากการแปลงเวฟเลตเป็นสัญญาณเสียง โดยกระบวนการแปลงกลับเวฟเลต โดยผ่านกระบวนการ IDWT จากนั้นจึงส่งต่อไปยังภาค Digital to Analog Converter (D/A) ซึ่งจะเป็นส่วนที่ทำการแปลงข้อมูลขนาด 8 บิตที่ได้จากการประมวลผลมาแปลงเป็นสัญญาณแอนะลอก หรือสัญญาณเสียงตามเดิม แล้วจึงส่งออกไปยังภาคของ Power Amplifier สำหรับทำการขยายสัญญาณให้มีขนาดของสัญญาณเสียงแรงพอที่จะไปขับลำโพง (Speaker)

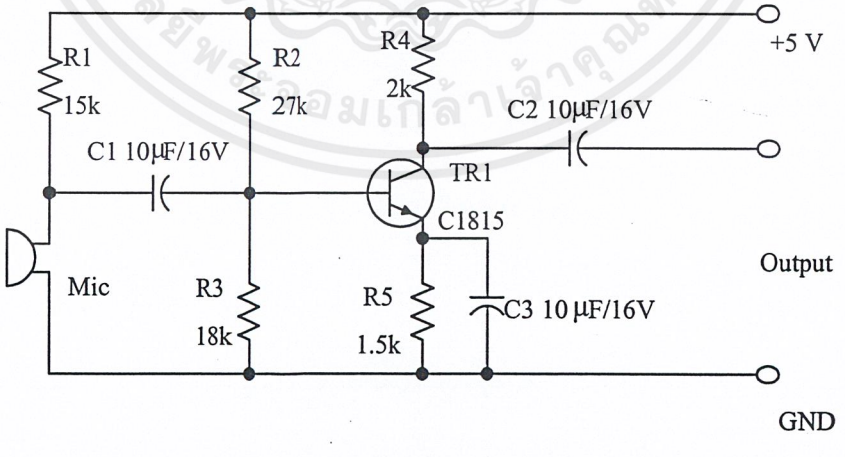
3.2 วงจรในการสร้างเครื่องบันทึกเสียงพูดระบบดิจิตอล

วงจรทั้งหมดนี้มีทั้งส่วนที่เป็นวงจรภายนอก และวงจรภายในที่เป็นอุปกรณ์ FPGAs ซึ่งจะเลือกใช้ตามความเหมาะสม และความสะดวกในการใช้งาน เพื่อเป็นการประหยัดทรัพยากร และอุปกรณ์ทางฮาร์ดแวร์ นอกจากนั้นยังลดความยุ่งยากในการต่อวงจรรวมทั้งเป็นการลดขนาดของวงจรอีกด้วย ดังนั้นวงจรส่วนใหญ่จึงถูกสร้างเป็นวงจรภายใน และวงจรภายนอกเพียงบางส่วนเท่านั้น

3.2.1 วงจรภายนอก

1) วงจรปรีไมโครโฟน

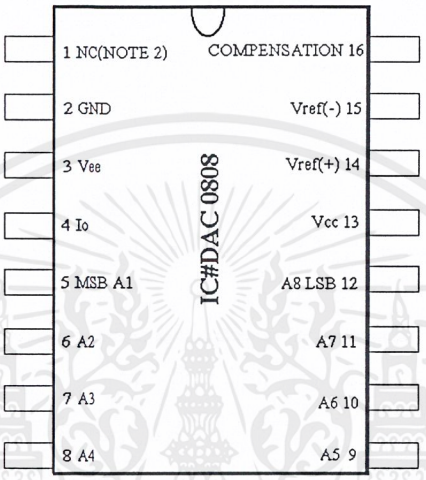
จากวงจรรูปที่ 3.2 จะเป็นวงจรขยายสัญญาณเสียงไมโครโฟน หรือวงจรภาคปรีไมโครโฟนที่ทำหน้าที่ขยายขนาดสัญญาณเสียงให้มีขนาดเพิ่มขึ้น โดยภายในวงจรจะเห็นว่าทรานซิสเตอร์ TR1 จะเป็นอุปกรณ์ขยายสัญญาณเสียงแบบคลาส A ดังรูป 3.2



รูปที่ 3.2 วงจรปรีไมค์โครโฟน

ซึ่งสัญญาณอินพุตที่ได้จาก Mic จะผ่านเข้ามาทาง C1 แล้วเป็นอินพุตให้กับวงจรขยายสัญญาณโดยใช้ทรานซิสเตอร์เบอร์ C1815 สัญญาณเอาต์พุตจากวงจรขยายสัญญาณเป็นสัญญาณแอนะลอกที่จะต้องนำไปแปลงเป็นสัญญาณดิจิตอลในภาคต่อไป

2) วงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนะลอกโดยใช้ IC#DAC0808



รูปที่ 3.3 ตำแหน่งขาต่างๆ ของไอซี DAC 0808

จากวงจรรูปที่ 3.3 DAC0808 จะเป็นไอซีแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนะลอกซึ่งมีแหล่งกำเนิดกระแส อ้างอิงอยู่ภายในตัวไอซี โดยในรูปเป็นการแสดงขาของ DAC0808

ขา 5 ถึงขา 12 (A1 – A8) เป็นขาอินพุตที่รับข้อมูลแบบดิจิตอลขนาด 8 บิต

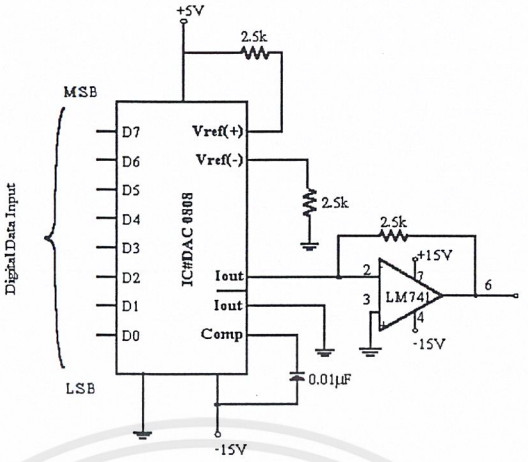
ขา 3 (Vee) เป็นแหล่งจ่ายไฟลบ 12 โวลท์

ขา 4 (Io) เป็นเอาต์พุตโดยจะส่งสัญญาณออกไปเป็นแอนะลอก

ขา 14 (Vref(+)) เป็นขาที่ใช้เทียบแรงดันไฟบวกโดยต้องต่อกับไฟบวก

ขา 15 (Vref(+)) เป็นขาที่ใช้เทียบแรงดันไฟลบโดยต้องต่อกับกราวด์

ขา 13 (Vcc) แหล่งจ่ายไฟบวก 5 โวลท์



รูปที่ 3.4 วงจรใช้งานจริงของไอซี DAC0808

จากรูปที่ 3.4 แสดงวงจรใช้งานจริงของ DAC0808 ที่ต่อเป็นตัวแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนะลอก ตัวความต้านทานขา Vref (+) คือ ตัวต้านทานที่ปรับค่าได้ เพื่อกำเนิดค่าอ้างอิงเท่ากับ 2 มิลลิแอมป์ และตัวต้านทาน ขา Vref (-) จะมีค่าเท่ากับตัวต้านทานตัวแรก คือ 2.5 k Ω ต่อเพื่อชดเชยกระแสอินพุตภายในไอซี กระแสเอาต์พุตออกที่ขา 4 นำไปแปลงเป็นแรงดันด้วยออปแอมป์

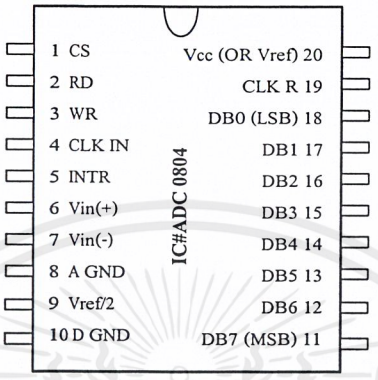
3) วงจรแปลงสัญญาณแอนะลอกเป็นดิจิตอล โดยใช้ IC# ADC 0804

การเปลี่ยนสัญญาณแรงดันแอนะลอกให้เป็นสัญญาณดิจิตอลมีหลักการ คือ สร้างวงจรแบ่งแรงดันด้วยตัวต้านทานเป็นแรงดันอ้างอิง (V_{REF}) ซึ่งทำหน้าที่เปรียบเทียบสัญญาณระหว่างแรงดันแอนะลอกอินพุตกับแรงดันอ้างอิงที่สร้างขึ้น และส่งสัญญาณแรงดันที่กำหนดให้ระดับลอจิกเป็น 0 และ 1 (“L” และ “H”) เข้าสู่อินพุตของวงจรเข้ารหัส เช่น ตัวเข้ารหัส เบอร์ 74147 และวงจรเข้ารหัส จะทำหน้าที่แปลงแรงดันสูง และต่ำจากเอาต์พุตของออปแอมป์เป็นสัญญาณดิจิตอลแบบบีซีดี 8421 เป็นสัญญาณดิจิตอลขนาด 3 บิต ซึ่งเป็นวงจรรวมที่ทำหน้าที่แปลงสัญญาณแอนะลอกให้เป็นสัญญาณดิจิตอล ซึ่งแต่ละเบอร์จะมีคุณสมบัติที่แตกต่างกันไป และเราสามารถแบ่งวงจรรวมชนิดดังกล่าวออกได้เป็น 2 กลุ่ม คือ

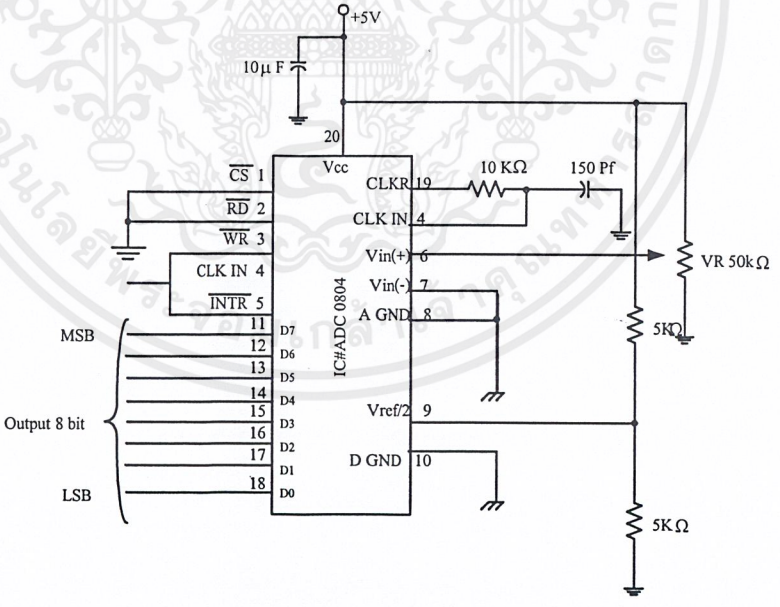
1) ชนิดเอาต์พุตที่เป็นแบบบีซีดี

2) ชนิดเอาต์พุตที่เป็นเลขฐานสอง

ในที่นี้เราจะใช้วงจรรวม หรือไอซีที่ใช้สำหรับแปลงสัญญาณแอนะลอกเป็นสัญญาณดิจิตอลที่มีเอาต์พุตเป็นเลขฐานสอง คือ วงจรรวมเบอร์ ADC 0804 มีเอาต์พุตเป็นข้อมูลดิจิตอลขนาด 8 บิต เป็นเลขฐานสอง มีขา 20 ขา เป็นแบบ DIP มีลักษณะการวางขา แสดงในรูปที่ 3.5



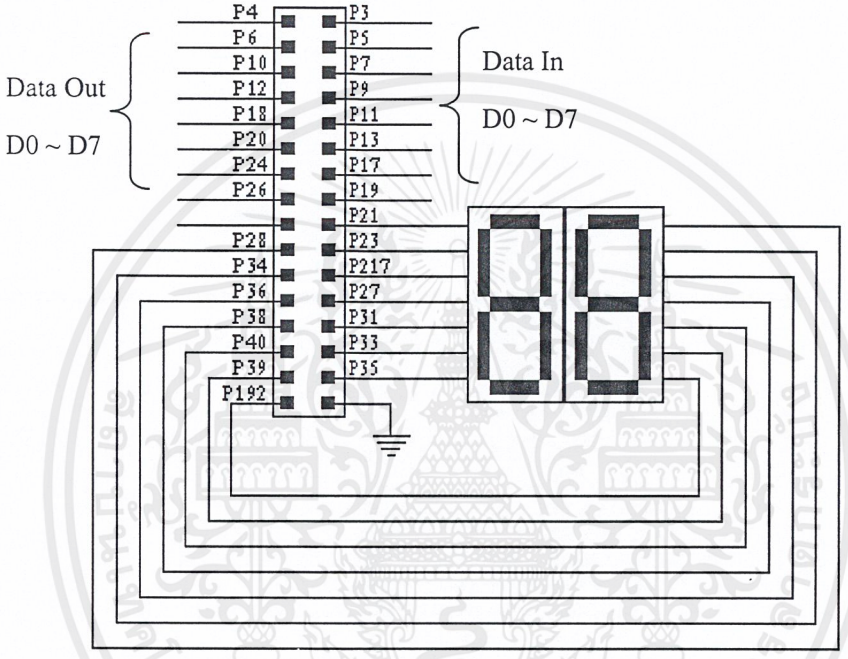
รูปที่ 3.5 การจัดวางตำแหน่งขาของไอซีเบอร์ADC 0804



รูปที่ 3.6 แสดงลักษณะการใช้งานของวงจร ADC0804

ADC0804 เป็นวงจรการทำงานที่อาศัยแรงดันที่ป้อนเข้ามา โดยเมื่อมีการเพิ่มระดับของแรงดันจะทำให้ค่าของเอาต์พุตที่แสดงออกเปลี่ยนไปตามระดับของแรงดันที่เพิ่มขึ้น และมีลักษณะการเพิ่มเป็น 2n เช่น 1, 2, 4, 8, 16 และ 32 เป็นต้น ซึ่งสามารถแสดงลักษณะลักษณะการใช้งานเบื้องต้นได้ ดังรูปที่ 3.6

4) วงจรอินเตอร์เฟสแรมกับอุปกรณ์โปรแกรมได้ (FPGAs)



รูปที่ 3.7 วงจรที่ใช้ในการติดต่อระหว่างอุปกรณ์ FPGA กับอุปกรณ์ และหน่วยความจำภายนอก

จากรูปที่ 3.7 เป็นการติดต่อวงจรภายนอกกับอุปกรณ์ FPGAs หรืออุปกรณ์ภายในนั่นเอง ขา P209, P215, P205, P203, P201, P199, P195, P193, P194, P216, P220, P191, P208, P189, P202 และ P187 ซึ่งต่อกับส่วนต่างๆ ดังนี้

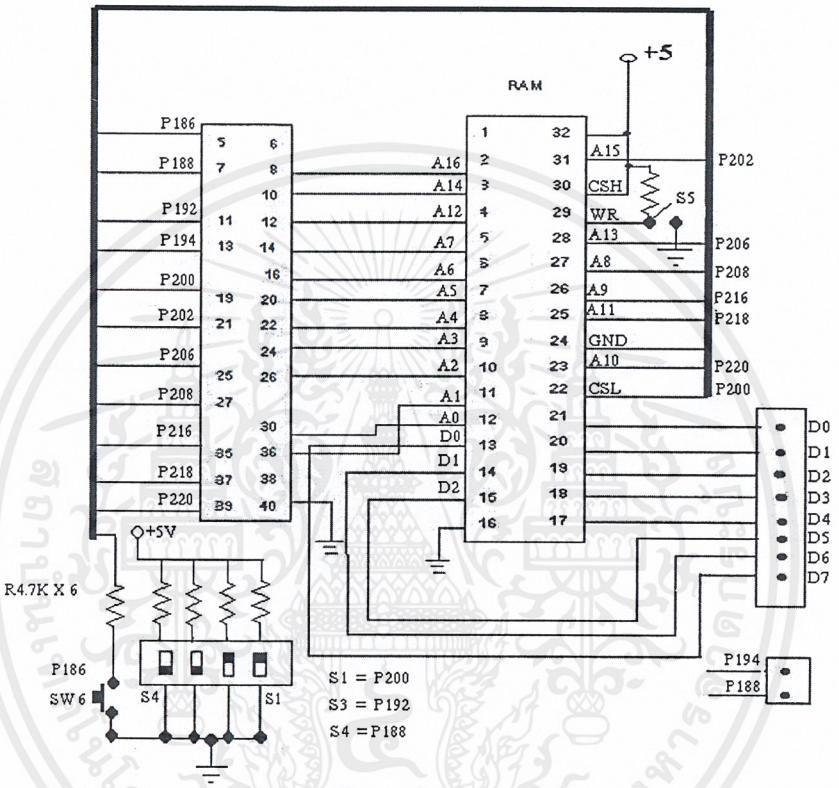
P200 : จะต่อกับขา Cs ของหน่วยความจำเพื่อใช้สำหรับการควบคุม เมื่อ Cs ของหน่วยความจำมีระดับลอจิกเป็น 0 จะทำให้หน่วยความจำพร้อมที่จะทำงาน ถ้ามีระดับลอจิกเป็น 1 ทำให้หน่วยความจำไม่ทำงาน

P206 : จะต่อกับขา WR ของหน่วยความจำซึ่งทำหน้าที่ควบคุมการอ่าน และการเขียนข้อมูลจากหน่วยความจำภายนอก โดยที่ WR ระดับลอจิกเป็น 0 จะทำการเขียนข้อมูลลงไปในหน่วยความจำ และถ้า WR ระดับลอจิกเป็น 1 จะทำการอ่านข้อมูลจากหน่วยความจำภายนอก

P186 : จะต่ออยู่กับสวิตช์ Reset เพื่อทำการรีเซตระบบ

P192 : จะต่ออยู่กับขา Cs ของ IC0804 (A/D) ถ้า Cs ระดับลอจิกเป็น 0 จะทำให้ IC0804 ทำงาน ถ้าระดับลอจิกเป็น 1 IC0804 จะไม่ทำงาน

Data : จะต่ออยู่กับขาที่ 13 ถึง 21 ของหน่วยความจำชนิดชั่วคราว (RAM)



รูปที่ 3.8 วงจรติดต่อกับอุปกรณ์ภายนอก

ในการเชื่อมต่อกับอุปกรณ์ภายนอกนั้น เพื่อที่จะใช้ในการรับข้อมูลจากวงจรที่อยู่ภายนอกอุปกรณ์โปรแกรมได้ ซึ่งจะต่อผ่านขาของตัวเชื่อมต่อ (Connector) โดยมีรายละเอียดดังต่อไปนี้

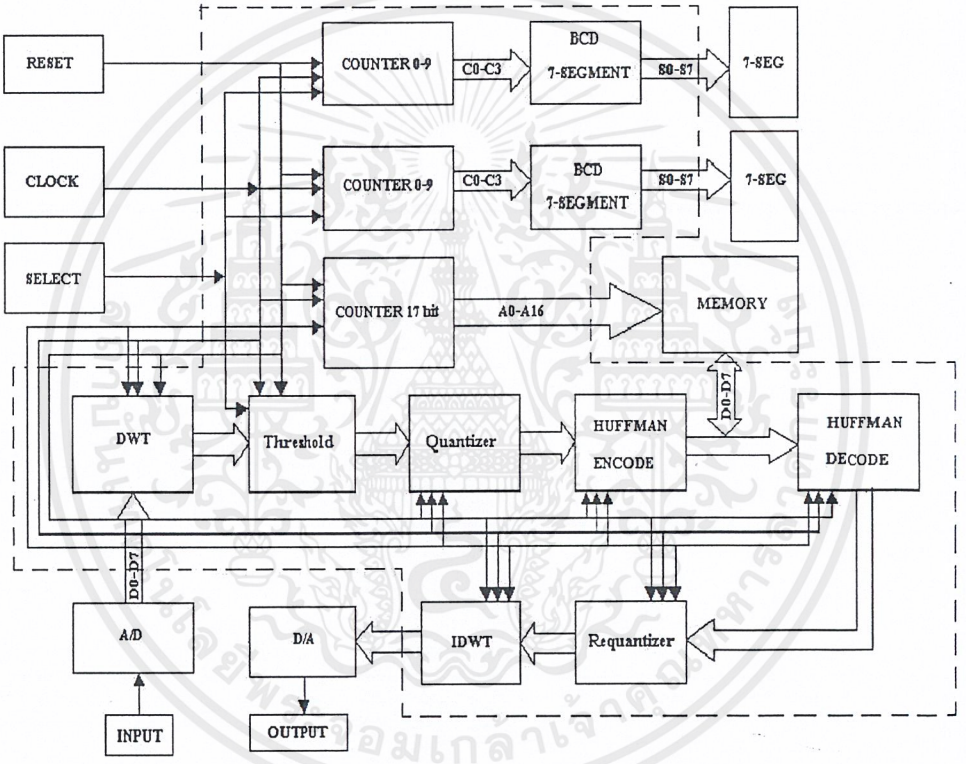
(P3, P5, P7, P9, P11, P13, P17 และ P19) : เป็นขาที่ใช้ในการรับข้อมูลจาก A/D ขนาด 8 บิต โดยจะเรียงลำดับจาก D7 ถึง D0 ตามลำดับ

(P4, P6, P10, P12, P18, P20, P24 และ P26) : เป็นขาที่ใช้ในการเชื่อมต่อสัญญาณจากขาข้อมูล D7 ถึง D0 ที่รับเข้ามา เพื่อส่งต่อไปเก็บยังหน่วยความจำ

(P21, P23, P217, P27, P31, P33 และ P35) : ต่ออยู่กับส่วนแสดงผลเจ็ดส่วนหลักที่ 1 และ P28, P34, P36, P38, P40, P39, P188 ต่ออยู่กับส่วนแสดงผลเจ็ดส่วนหลักที่ 2 ซึ่งส่วนแสดงผลเจ็ดส่วนทั้งสองตัวนี้จะเป็นตัวแสดงผลระยะเวลาการเก็บข้อมูลเสียงในหน่วยความจำ

3.2.2 วงจรภายใน

ส่วนของวงจรภายในอุปกรณ์ FPGAs นั้นจะมีทั้งสองส่วน คือ ส่วนที่เป็น Schematics EditorและVHDL ซึ่งสามารถทำการสร้างขึ้นได้จากภายในอุปกรณ์ FPGAs ซึ่งการทำงานของวงจรภายในสามารถอธิบายการทำงาน ดังรูปที่ 3.9



หมายเหตุ ส่วนที่อยู่ในเส้นประเป็นวงจรที่ออกแบบภายในอุปกรณ์ FPGAs

รูปที่ 3.9 แผนผังการทำงานของวงจรภายในอุปกรณ์ FPGAs

จากรูปที่ 3.9 วงจรภายในอุปกรณ์ FPGAs จะทำการรับข้อมูลมาจากอินพุต โดยผ่านอุปกรณ์แปลงสัญญาณแอนะลอกเป็นสัญญาณดิจิตอลจากวงจรที่ได้ออกแบบบนอุปกรณ์ FPGAs จะรับข้อมูลเข้ามาผ่านทางบัส (Bus) อินพุต และส่งข้อมูลไปยังภาคต่อไปเพื่อทำการแปลงเวฟเลต (DWT)โดยสัญญาณที่ได้จากทางด้านความถี่ต่ำจะมีลักษณะรูปคลื่นคงรูปสัญญาณตัวเดิมมากที่สุด ส่วนทางด้านความถี่สูงนั้นจะมีลักษณะเป็นสัญญาณรบกวนที่แทรกมา กับสัญญาณเสียงโดยข้อมูล

ที่ถูกแยกออกนั้นจะมีขนาดเพียง 64 ข้อมูล ของแต่ละระดับความถี่ ซึ่งจะเรียกว่าค่าสัมประสิทธิ์ของสัญญาณเสียง หลังจากได้ข้อมูลที่ผ่านขั้นตอนการแปลงเวฟเลตแล้ว ก็จะต้องนำไปทำการตัดระดับ (Threshold) ข้อมูลโดยจะกำหนดช่วงค่าที่มีช่วงอยู่ใกล้ศูนย์ทั้งทางด้านบวก และด้านลบโดยจะให้เป็นศูนย์ ซึ่งค่าที่ตัดทิ้งนี้จะเป็นข้อมูลที่มีลักษณะเป็นสัญญาณรบกวนสามารถที่จะตัดทิ้งได้จากข้อมูลที่ผ่านการตัดระดับ จะถูกนำไปกระทำการจัดระดับ (Quantization) ข้อมูลที่เข้าส่งมา โดยจะจัดระดับข้อมูลจาก 32 บิต ให้เหลือเพียง 8 บิต เมื่อได้ข้อมูลที่มีขนาด 8 บิต แล้วจะส่งต่อไปทำการเข้ารหัสฮัฟฟ์แมนซึ่งจะเป็นการเข้ารหัสแบบ อาศัยความซ้ำซ้อนของข้อมูลที่เกิดจากการตัดระดับสัญญาณ ซึ่งมีแอมปลิจูดต่ำ โดยข้อมูลที่มีความซ้ำซ้อนมากที่สุดจะถูกแทนด้วยข้อมูลเพียง 1 บิต และข้อมูลอื่นๆ จะถูกแทนด้วยข้อมูล 9 บิต จากนั้น ข้อมูลที่ถูกเข้ารหัสโดย Huffman Encoder จะนำไปทำการจัดเก็บลงในหน่วยความจำขนาด 128 กิโลไบต์ ซึ่งสามารถเก็บข้อมูลได้ 00000H ถึง 1FFFFH โดยตำแหน่งของการจัดเก็บนั้นจะถูกควบคุม โดยวงจรตัวนับขนาด 17 บิตใช้สำหรับชี้ตำแหน่งหน่วยความจำดังกล่าว

ในส่วนของการแปลงกลับรหัสฮัฟฟ์แมนที่จัดเก็บไว้ในหน่วยความจำจะถูกอ่านออกมาจากหน่วยความจำ โดยผ่านวงจรถอดรหัสฮัฟฟ์แมนเมื่อทำการถอดรหัสฮัฟฟ์แมนจะได้ข้อมูลขนาด 8 บิต ข้อมูลขนาด 8 บิต ที่ได้นี้จะถูกไปทำการ Dequantization โดยจะนำข้อมูล 8 บิต นี้ไปทำการชี้ตาราง ซึ่งเป็นหน่วยความจำสำหรับโปรแกรมเพื่อให้ได้ข้อมูลที่มีขนาด 32 บิต เพื่อที่จะส่งต่อไปทำการแปลงกลับเวฟเลต (Inverse Wavelet Transform) แล้ว จะส่งออกไปทำการแปลงสัญญาณดิจิตอลให้เป็น สัญญาณแอนะลอก แล้วนำไปแสดงผลที่ลำโพง ซึ่งข้อมูลที่ออกได้ออกมานั้นจะมีลักษณะของข้อมูล ที่ใกล้เคียงกับข้อมูลเดิมที่เข้ามา โดยจะมีส่วนที่ผิดเพี้ยนไปจากเดิม ซึ่งก็อาจจะเกิดขึ้นจากส่วนต่างๆ ของการลดขนาดข้อมูล และส่วนต่างๆ ของการแปลงกลับ แต่ในการนำข้อมูลออกมาจากการแปลเวฟเลตนั้นจะไม่จำเป็นที่จะต้องใช้งานข้อมูลที่เหมือนกับข้อมูลเดิม เพราะเป็นข้อมูลที่อาศัยลักษณะของความใกล้เคียงกับข้อมูลเดิมเท่านั้น โดยให้ข้อมูลที่ได้ออกมานั้นสามารถที่จะคงรูปของข้อมูลเดิม

3.3 การแปลงเวฟเลต

3.3.1 การออกแบบตารางการแปลงข้อมูล

ในการออกแบบตารางที่ใช้ในการเก็บค่าของข้อมูลสำหรับการแปลงค่าสัมประสิทธิ์เวฟเลตสามารถที่จะกระทำได้ โดยการใช้ตารางเข้ามาช่วยในการแปลงค่าสัมประสิทธิ์ให้อยู่ในรูปของตัวเลขฐานสิบแปลงเป็นเลขฐานสองเลขฐานสองแปลงเป็นเลขฐานสิบหก หรืออาจจะกำหนดเป็นเลข

ฐานสิบแปลงเป็นเลขฐานสิบหก และจากเลขฐานสิบหกแปลงเป็นเลขฐานสอง แต่ในที่นี้เราจะใช้ตัวเลขที่ได้จากเลขฐานสิบแปลงเป็นเลขฐานสิบหก แล้วแปลงให้อยู่ในรูปของเลขฐานสองอีกครั้ง เพื่อให้ง่ายในการจัดเก็บข้อมูลลงในหน่วยที่ได้กำหนดไว้ ขนาดของหน่วยความจำที่เก็บข้อมูลขนาด 16 บิต 256 ตำแหน่ง คือ เริ่มที่ตำแหน่ง 0 ถึง 255

ซึ่งการออกแบบตารางการแปลงข้อมูลนี้จะช่วยลดลำดับขั้นตอนในการแปลง ซึ่งจะมีผลต่อการออกแบบวงจร และจะต้องกระทำด้วยกระบวนการที่ยุ่งยาก โดยตารางการแปลงข้อมูลนี้จะหาค่าได้จากการนำสมการของแต่ละระดับของข้อมูลเข้ามาแทนในสมการระดับถัดไปเรื่อยๆ จนครบ 3 ระดับ ตามที่ต้องการซึ่งค่าที่หาได้จะเกิดจากค่าของตัวกรองที่ใช้ในแต่ละทิศทาง โดยสามารถที่จะแยกออกเป็นพจน์ แล้วนำกลับมาทำการคำนวณหาค่าสัมประสิทธิ์เพื่อให้เกิดเป็นค่าคงที่ก่อนที่จะแยกพจน์เมื่อมี ข้อมูลเข้ามาก็นำค่าของข้อมูลมากระทำกับค่าคงที่ในตารางที่คำนวณไว้ ผลที่ได้ก็คือ ค่าที่ได้จากการแปลง เวฟเลตเพคเกจในการหาค่าที่เก็บในตารางนั้นจะต้องหาทั้งหมด 8 รูปแบบ คือ LLL, LLH, LHL, LHH, HLL, HLH, HHL และ HHH ดังตารางที่ 3.1

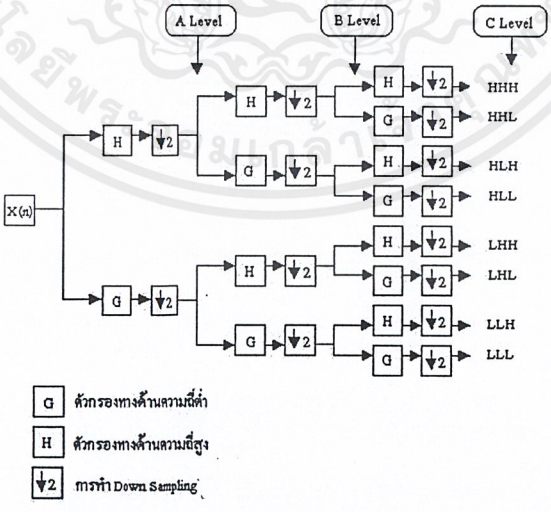
จากตาราง 3.1 ค่าที่ได้ในตารางเป็นค่าที่ได้มาจากการคูณกันของสัมประสิทธิ์เวฟเลตเพคเกจ ในแต่ละระดับ และในแต่ระดับที่นำมาคูณกันก็จะต้องเป็นค่าสัมประสิทธิ์ของระดับนั้นๆ ด้วยซึ่งจะสังเกตได้จากรูปที่ 3.10 จะมีการทำ Down Sampling ซึ่งเป็นการลดอัตราการสุ่ม โดยในที่นี้จะกระทำกระบวนการดังกล่าวไปด้วย เพื่อเป็นการลดวงจรที่จะต้องออกแบบ โดยค่าสัมประสิทธ์ที่ได้นี้จะถูกนำไปจัดเก็บลงใน ROM เพื่อที่จะนำไปทำการคูณกับค่าของข้อมูลที่เข้ามาจากค่าในตาราง 3.1 นั้นจะเป็นค่าที่เกิดจากการนำเอาค่าของเวฟเลตแม่ Darbechie 4 ซึ่งจะมีอยู่ด้วยกัน 4 ค่าด้วยกัน เมื่อนำไปแทนลงในสมการตัวกรอง FIR ซึ่งจะทำการแยกข้อมูลทางด้านสูงออกจากข้อมูลทางด้านต่ำออกจากกัน โดยถ้ากระทำทั้งหมด 3 ระดับนั้น ค่าที่ได้จากการแปลงเวฟเลตนั้นจะถูกแยกออกไปทั้งหมด 8 ทิศทางด้วยกัน ซึ่งก็จะเป็นความถี่ที่แตกต่างกันไปในแต่ละทิศทาง

ตารางที่ 3.1 การเก็บค่าสัมประสิทธิ์เวฟเลตแม่ 3 ระดับ

DN	LLL	LLH	LHL	LHH	HLL	HLH	HHL	HHH
D4n-21	- 0.00216	- 0.00808	- 0.00808	- 0.03018	- 0.00808	- 0.03018	- 0.03018	- 0.03018
D4n-20	0.00375	0.01400	0.01400	0.05228	0.01400	0.05228	0.05228	0.19514
D4n-19	0.01775	0.06628	0.06628	0.24742	0.01025	0.03627	0.03827	0.14286
D4n-18	0.00158	0.00593	0.00593	0.02213	- 0.02642	- 0.09863	- 0.09863	- 0.36815
D4n-17	-0.00649	-0.02425	- 0.08029	- 0.29969	0.07278	0.27168	0.06253	0.23340
D4n-16	-0.04476	-0.16708	- 0.07003	- 0.26143	- 0.11105	- 0.41451	- 0.05227	- 0.19514

ตารางที่ 3.1 (ต่อ) การเก็บค่าสัมประสิทธิ์เวฟเลตแม่ 3 ระดับ

DN	LLL	LLH	LHL	LHH	HLL	HLH	HHL	HHH
D4n-15	-0.11321	-0.42258	- 0.09710	- 0.34606	0.03668	0.13693	- 0.08087	- 0.30187
D4n-14	-0.06904	-0.25769	0.00748	0.02795	0.00748	0.02795	0.18107	0.87588
D4n-13	-0.02502	-0.13263	0 .22959	0.64785	- 0.03551	- 0.34170	0.17364	- 0.13255
D4n-12	0.01182	0.14117	- 0.00316	0.35040	0.06785	0.61551	- 0.40640	- 0.16490
D4n-11	0.02800	0.5638	- 0.46677	- 0.02800	- 0.16057	- 0.33419	- 0.22216	0.16457
D4n-10	0.10452	0.43126	- 0.06909	- 0.10453	0.23710	0.20166	0.61984	- 0.23713
D4n-9	0.16489	0.31805	0.61551	- 0.26194	- 0.34704	0.00316	0.05291	0.01182
D4n-8	0.28562	0.13263	0.22967	- 0.11756	0.49478	- 0.18858	- 0.43874	0.13255
D4n-7	0.46675	-0.12504	- 0.39023	0.10454	- 0.39023	0.10454	- 0.04857	0 .01301
D4n-6	0.42258	-0.11321	- 0.13693	0.03668	0.34605	- 0.09271	0.30187	-0.08087
D4n-5	0.37623	-0.10079	0.26143	- 0.07004	- 0.30994	0.08303	- 0.01400	0.00375
D4n-4	0.33796	-0.09054	0.11860	- 0.03177	0.27168	- 0.07278	- 0.12881	0.03451
D4n-3	0.30778	-0.08248	- 0.08245	0.02209	- 0.20322	0.05444	0.05444	- 0.1458
D4n-2	0.24742	-0.06628	- 0.06628	0.01775	0.14286	- 0.03827	- 0.03827	-.01025
D4n-1	0.19514	-0.05228	- 0.05228	0.01400	- 0.05228	0.01400	0.01400	- 0.00375
D4n-0	0.11267	-0.03018	- 0.03018	0.00800	0.03018	0.00808	0.00800	- 0.00216



รูปที่ 3.10 ขั้นตอนในการทำเวฟเลตเพคเกจ

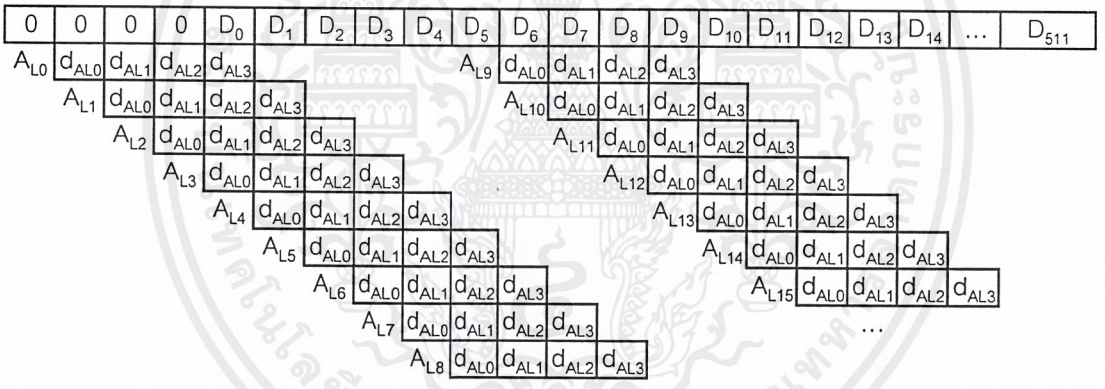
ในการแปลงเวฟเลตแบบเวฟเลตเพคเกจนั้นจะมีขั้นตอนดังรูปที่ 3.10 ซึ่งจะเห็นได้ว่าเมื่อมีข้อมูลเข้ามากระทำในระดับที่หนึ่ง และจะมีการทำการประสานกัน ระหว่างข้อมูลที่เข้ามากับเวฟเลตแม่ ซึ่งผลที่ได้จะเหมือนกับการป้อนสัญญาณผ่านตัวกรองความถี่ผลที่ได้ คือ จะกรองความถี่ต่ำ หรือสูงผ่านไป ซึ่งก็จะขึ้นอยู่กับเวฟเลตแม่ที่ใช้ด้วย เนื่องจากในที่นี้จะหา ค่าของตัวกรองของระบบโดยจะกำหนดให้ D_n เป็นข้อมูลที่รับเข้ามา

D_n คือ ข้อมูลที่รับเข้ามา

D_{ALn} คือ เวฟเลตแม่ที่จะใช้ซึ่งเป็นทางด้านความถี่ต่ำ

A_{Ln}, B_{Ln}, C_{Ln} คือ ผลที่ได้จากการกระทำในแต่ละระดับ

สำหรับในการทำการประสานเพื่อหาค่าตัวกรองของระบบทางด้าน LLL นั้นจะมีขั้นตอนต่อไปนี้คือ



รูปที่ 3.11 ลักษณะของการทำการประสานในระดับที่ A

จากรูปที่ 3.11 เป็นการประสานซึ่งผลที่ได้ คือ ค่าในระดับ A Level ทำให้ได้สมการดังต่อไปนี้

$$A_{L0} = D_0\, d_{AL3}$$

$$A_{L1} = D_0\, d_{AL2} + D_1\, d_{AL3}$$

$$A_{L2} = D_0\, d_{AL1} + D_1\, d_{AL2} + D_2\, d_{AL3}$$

$$A_{L3} = D_0\, d_{AL0} + D_1\, d_{AL1} + D_2\, d_{AL2} + D_3\, d_{AL3}$$

$$A_{L4} = D_1\, d_{AL0} + D_2\, d_{AL1} + D_3\, d_{AL2} + D_4\, d_{AL3}$$

$$A_{L5} = D_2\, d_{AL0} + D_3\, d_{AL1} + D_4\, d_{AL2} + D_5\, d_{AL3}$$

$$A_{L6} = D_3\, d_{AL0} + D_4\, d_{AL1} + D_5\, d_{AL2} + D_6\, d_{AL3}$$

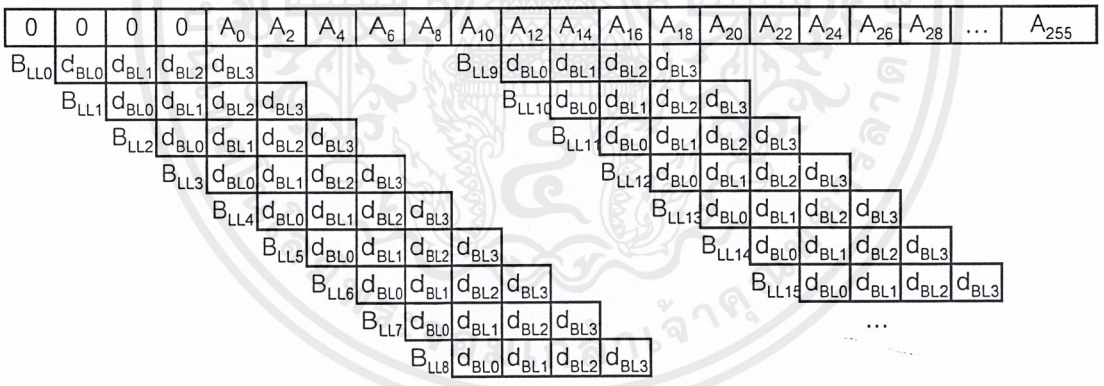
$$A_{L8} = D_5\, d_{AL0} + D_6\, d_{AL1} + D_7\, d_{AL2} + D_8\, d_{AL3}$$

$$A_{L10} = D_7\, d_{AL0} + D_8\, d_{AL1} + D_9\, d_{AL2} + D_{10}\, d_{AL3}$$

ดังนั้น สามารถที่จะสรุปสมการได้ คือ

$$A_{Ln} = D_{n-3}\, d_{AL0} + D_{n-2}\, d_{AL1} + D_{n-1}\, d_{AL2} + D_n\, d_{AL3} \qquad (3.1)$$

เมื่อผ่านการประสานแล้วจะทำการลดขนาดของข้อมูล โดยการลดอัตราการสุ่มข้อมูล ซึ่งในการลดอัตราการสุ่มของข้อมูลนี้จะทำให้ข้อมูลที่ได้มีจำนวนลดลงครึ่งหนึ่งจากข้อมูลที่ได้จากการประสาน ในตัวอย่างนี้จะทำการลดอัตราการสุ่มข้อมูล โดยจะตัดข้อมูลที่เป็นเลขคี่ออกไปเพราะเนื่องจากไม่จำเป็นต้องคำนวณหาค่าที่เป็นเลขคี่จากข้อมูลที่ผ่านการประสานในระดับที่ 1 และเมื่อทำการลดอัตราการสุ่มข้อมูลเสร็จ ก็จะทำการประสานในระดับที่ 2 ซึ่งจะได้ดังนี้



รูปที่ 3.12 การทำการประสาน ในระดับที่ 2

$$B_{LL0} = A_{L0}\, d_{BL3}$$
$$= D_0\, d_{AL3}\, d_{BL3}$$

$$B_{LL1} = A_{L0}\, d_{BL2} + A_{L2}\, d_{BL3}$$
$$= D_0\, d_{AL3}\, d_{BL2} + (D_0\, d_{AL1} + D_1\, d_{AL2} + D_2\, d_{AL3})\, d_{BL3}$$
$$= D_0\, d_{AL3}\, d_{BL2} + D_0\, d_{AL1}\, d_{BL3} + D_1\, d_{AL2}\, d_{BL3} + D_2\, d_{AL3}\, d_{BL3}$$

$$B_{LL2} = A_{L0}\, d_{BL1} + A_{L2}\, d_{BL2} + A_{L4}\, d_{BL3}$$

$= D_0\, d_{AL3}\, d_{BL1} + (D_0\, d_{AL1} + D_1\, d_{AL2} + D_2\, d_{AL3})\, d_{BL2} + (D_1\, d_{AL0} + D_2\, d_{AL1} + D_3\, d_{AL2} + D_4\, d_{AL3})\, d_{BL3}$

$= D_0\, d_{AL3}\, d_{BL1} + D_0\, d_{AL1}\, d_{BL2} + D_1\, d_{AL2}\, d_{BL2} + D_2\, d_{AL3}\, d_{BL2} + D_1\, d_{AL0}\, d_{BL3} + D_2\, d_{AL1}\, d_{BL3} + D_3\, d_{AL2}\, d_{BL3} + D_4\, d_{AL3}\, d_{BL3}$

$B_{LL3} = A_{L0}\, d_{BL0} + A_{L2}\, d_{BL1} + A_{L4}\, d_{BL2} + A_{L6}\, d_{BL3}$

$= D_0\, d_{AL3}\, d_{BL0} + (D_0\, d_{AL1} + D_1\, d_{AL2} + D_2\, d_{AL3})\, d_{BL1} + (D_1\, d_{AL0} + D_2\, d_{AL1} + D_3\, d_{AL2} + D_4\, d_{AL3})\, d_{BL2} + (D_3\, d_{AL0} + D_4\, d_{AL1} + D_5\, d_{AL2} + D_6\, d_{AL3})\, d_{BL3}$

$= D_0\, d_{AL3}\, d_{BL0} + D_0\, d_{AL1}\, d_{BL1} + D_1\, d_{AL2}\, d_{BL1} + D_2\, d_{AL3}\, d_{BL1} + D_1\, d_{AL0}\, d_{BL2} + D_2\, d_{AL1}\, d_{BL2} + D_3\, d_{AL2}\, d_{BL2} + D_4\, d_{AL3}\, d_{BL2} + D_3\, d_{AL0}\, d_{BL3} + D_4\, d_{AL1}\, d_{BL3} + D_5\, d_{AL2}\, d_{BL3} + D_6\, d_{AL3}\, d_{BL3}$

$B_{LL4} = A_{L2}\, d_{BL0} + A_{L4}\, d_{BL1} + A_{L6}\, d_{BL2} + A_{L8}\, d_{BL3}$

$= (D_0\, d_{AL1} + D_1\, d_{AL2} + D_2\, d_{AL3})\, d_{BL0} + (D_1\, d_{AL0} + D_2\, d_{AL1} + D_3\, d_{AL2} + D_4\, d_{AL3})\, d_{BL1} + (D_3\, d_{AL0} + D_4\, d_{AL1} + D_5\, d_{AL2} + D_6\, d_{AL3})\, d_{BL2} + (D_4\, d_{AL0} + D_5\, d_{AL1} + D_6\, d_{AL2} + D_7\, d_{AL3})\, d_{BL3}$

$= D_0\, d_{AL1}\, d_{BL0} + D_1\, d_{AL2}\, d_{BL0} + D_2\, d_{AL3}\, d_{BL0} + D_1\, d_{AL0}\, d_{BL1} + D_2\, d_{AL1}\, d_{BL1} + D_3\, d_{AL2}\, d_{BL1} + D_4\, d_{AL3}\, d_{BL1} + D_3\, d_{AL0}\, d_{BL2} + D_4\, d_{AL1}\, d_{BL2} + D_5\, d_{AL2}\, d_{BL2} + D_6\, d_{AL3}\, d_{BL2} + D_4\, d_{AL0}\, d_{BL3} + D_5\, d_{AL1}\, d_{BL3} + D_6\, d_{AL2}\, d_{BL3} + D_7\, d_{AL3}\, d_{BL3}$

$B_{LL5} = A_{L4}\, d_{BL0} + A_{L6}\, d_{BL1} + A_{L8}\, d_{BL2} + A_{L10}\, d_{BL3}$

$= (D_1\, d_{AL0} + D_2\, d_{AL1} + D_3\, d_{AL2} + D_4\, d_{AL3})\, d_{BL0} + (D_3\, d_{AL0} + D_4\, d_{AL1} + D_5\, d_{AL2} + D_6\, d_{AL3})\, d_{BL1} + (D_4\, d_{AL0} + D_5\, d_{AL1} + D_6\, d_{AL2} + D_7\, d_{AL3})\, d_{BL2} + (D_7\, d_{AL0} + D_8\, d_{AL1} + D_9\, d_{AL2} + D_{10}\, d_{AL3})\, d_{BL3}$

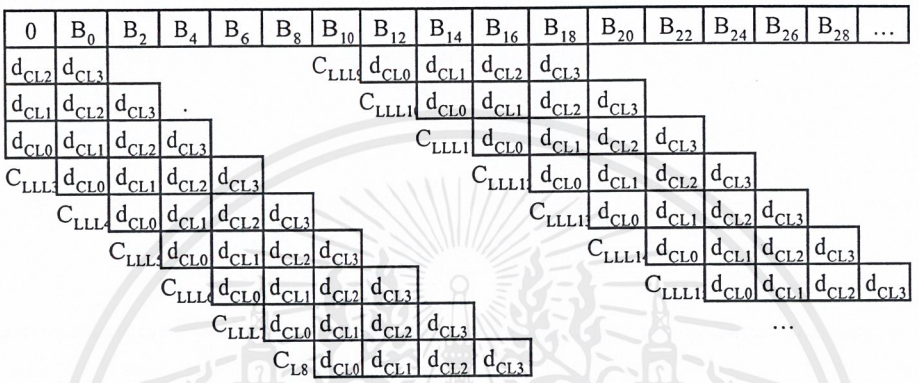
$= D_1\, d_{AL0}\, d_{BL0} + D_2\, d_{AL1}\, d_{BL0} + D_3\, d_{AL2}\, d_{BL0} + D_4\, d_{AL3}\, d_{BL0} + D_3\, d_{AL0}\, d_{BL1} + D_4\, d_{AL1}\, d_{BL1} + D_5\, d_{AL2}\, d_{BL1} + D_6\, d_{AL3}\, d_{BL1} + D_4\, d_{AL0}\, d_{BL2} + D_5\, d_{AL1}\, d_{BL2} + D_6\, d_{AL2}\, d_{BL2} + D_7\, d_{AL3}\, d_{BL2} + D_7\, d_{AL0}\, d_{BL3} + D_8\, d_{AL1}\, d_{BL3} + D_9\, d_{AL2}\, d_{BL3} + D_{10}\, d_{AL3}\, d_{BL3}$

สามารถสรุปเป็นสมการได้คือ

$$B_{LLn} = A_{L(2n-6)}\, d_{BL0} + A_{L(2n-4)}\, d_{BL1} + A_{L(2n-2)}\, d_{BL2} + A_{L2n}\, d_{BL3} \qquad (3.2)$$

หลังจากทำการประสาน B Level จะทำการลดอัตราการสุ่มข้อมูล (Down Sampling) ซึ่งมี

ลักษณะการกระทำเหมือนกันกับในระดับที่ 1 ขั้นตอนสุดท้ายคือการทำการประสานเพื่อให้ได้ค่าสัมประสิทธิ์ที่ต้องการแล้วทำการลดอัตราการสุ่มข้อมูล โดยการเลือกเอาเฉพาะข้อมูลในลำดับที่เป็นเลขคู่เท่านั้น ส่วนข้อมูลที่เป็นเลขคี่ก็จะถูกแทนที่ด้วยค่า 0 ซึ่งในระดับสุดท้ายนี้จะทำให้ได้ข้อมูลออกมาเพียง 64 ข้อมูล ในการทำระดับ LLL ซึ่งจะมีอยู่ทั้งหมด 8 รูปแบบค่าที่ได้ในระดับสุดท้ายนี้จะมี การแยกเอาความถี่ที่แตกต่างกันออกไป เรื่อยๆ



รูปที่ 3.13 การทำการประสานในระดับที่ 3

$$C_{LLL0} = B_{LL0}\, d_{CL3}$$

$$C_{LLL1} = B_{LL0}\, d_{CL2} + B_{LL2}\, d_{CL3}$$

$$C_{LLL2} = B_{LL0}\, d_{CL1} + B_{LL2}\, d_{CL2} + B_{LL4}\, d_{CL3}$$

$$C_{LLL3} = B_{LL0}\, d_{CL0} + B_{LL2}\, d_{CL1} + B_{LL4}\, d_{CL2} + B_{LL6}\, d_{CL3}$$

$$C_{LLL4} = B_{LL2}\, d_{CL0} + B_{LL4}\, d_{CL1} + B_{LL6}\, d_{CL2} + B_{LL8}\, d_{CL3}$$

$$C_{LLL5} = B_{LL4}\, d_{CL0} + B_{LL6}\, d_{CL1} + B_{LL8}\, d_{CL2} + B_{LL10}\, d_{CL3}$$

$$C_{LLL6} = B_{LL6}\, d_{CL0} + B_{LL8}\, d_{CL1} + B_{LL10}\, d_{CL2} + B_{LL12}\, d_{CL3}$$

ซึ่งใน C_{LLL6} ขึ้นไปจะทำให้สามารถสรุปสมการอื่นๆ ได้ ดังนั้นจาก

$$\begin{aligned} C_{LLL6} &= B_{LL6}\, d_{CL0} + B_{LL8}\, d_{CL1} + B_{LL10}\, d_{CL2} + B_{LL12}\, d_{CL3} \\ &= (A_{L6}\, d_{BL0} + A_{L8}\, d_{BL1} + A_{L10}\, d_{BL2} + A_{L12}\, d_{BL3})\, d_{CL0} + (A_{L10}\, d_{BL0} + A_{L12}\, d_{BL1} + A_{L14}\, d_{BL2} + A_{L16}\, d_{BL3})\, d_{CL1} + (A_{L14}\, d_{BL0} + A_{L16}\, d_{BL1} + A_{L18}\, d_{BL2} + A_{L20}\, d_{BL3})\, d_{CL2} + (A_{L18}\, d_{BL0} + A_{L20}\, d_{BL1} + A_{L22}\, d_{BL2} + A_{L24}\, d_{BL3})\, d_{CL3} \end{aligned}$$

$$= [(D_3\, d_{AL0} + D_4\, d_{AL1} + D_5\, d_{AL2} + D_6\, d_{AL3})\, d_{BL0} + (D_5\, d_{AL0} + D_6\, d_{AL1} + D_7\, d_{AL2} + D_8\, d_{AL3})\, d_{BL1} + (D_7\, d_{AL0} + D_8\, d_{AL1} + D_9\, d_{AL2} + D_{10}\, d_{AL3})\, d_{BL2} + (D_9\, d_{AL0} + D_{10}\, d_{AL1} + D_{11}\, d_{AL2} + D_{12}\, d_{AL3})\, d_{BL3}]\, d_{CL0} + [(D_7\, d_{AL0} + D_8\, d_{AL1} + D_9\, d_{AL2} + D_{10}\, d_{AL3})\, d_{BL0} + (D_9\, d_{AL0} + D_{10}\, d_{AL1} + D_{11}\, d_{AL2} + D_{12}\, d_{AL3})\, d_{BL1} + (D_{11}\, d_{AL0} + D_{12}\, d_{AL1} + D_{13}\, d_{AL2} + D_{14}\, d_{AL3})\, d_{BL2} + (D_{12}\, d_{AL0} + D_{14}\, d_{AL1} + D_{15}\, d_{AL2} + D_{16}\, d_{AL3})\, d_{BL3}]\, d_{CL1} + [(D_{11}\, d_{AL0} + D_{12}\, d_{AL1} + D_{13}\, d_{AL2} + D_{14}\, d_{AL3})\, d_{BL0} + (D_{13}\, d_{AL0} + D_{14}\, d_{AL1} + D_{15}\, d_{AL2} + D_{16}\, d_{AL3})\, d_{BL1} + (D_{15}\, d_{AL0} + D_{16}\, d_{AL1} + D_{17}\, d_{AL2} + D_{18}\, d_{AL3})\, d_{BL2} + (D_{17}\, d_{AL0} + D_{18}\, d_{AL1} + D_{19}\, d_{AL2} + D_{20}\, d_{AL3})\, d_{BL3}]\, d_{CL2} + [(D_{15}\, d_{AL0} + D_{16}\, d_{AL1} + D_{17}\, d_{AL2} + D_{18}\, d_{AL3})\, d_{BL0} + (D_{17}\, d_{AL0} + D_{18}\, d_{AL1} + D_{19}\, d_{AL2} + D_{20}\, d_{AL3})\, d_{BL1} + (D_{19}\, d_{AL0} + D_{20}\, d_{AL1} + D_{21}\, d_{AL2} + D_{22}\, d_{AL3})\, d_{BL2} + (D_{21}\, d_{AL0} + D_{22}\, d_{AL1} + D_{23}\, d_{AL2} + D_{24}\, d_{AL3})\, d_{BL3}]\, d_{CL3}$$

$$C_{LLL6} = D_3\, d_{AL0}\, d_{BL0}\, d_{CL0} + D_4\, d_{AL1}\, d_{BL0}\, d_{CL0} + D_5\, d_{AL2}\, d_{BL0}\, d_{CL0} + D_6\, d_{AL3}\, d_{BL0}\, d_{CL0} + D_5\, d_{AL0}\, d_{BL1}\, d_{CL0} + D_6\, d_{AL1}\, d_{BL1}\, d_{CL0} + D_7\, d_{AL2}\, d_{BL1}\, d_{CL0} + D_8\, d_{AL3}\, d_{BL1}\, d_{CL0} + D_7\, d_{AL0}\, d_{BL2}\, d_{CL0} + D_8\, d_{AL1}\, d_{BL2}\, d_{CL0} + D_9\, d_{AL2}\, d_{BL2}\, d_{CL0} + D_{10}\, d_{AL3}\, d_{BL2}\, d_{CL0} + D_9\, d_{AL0}\, d_{BL3}\, d_{CL0} + D_{10}\, d_{AL1}\, d_{BL3}\, d_{CL0} + D_{11}\, d_{AL2}\, d_{BL3}\, d_{CL0} + D_{12}\, d_{AL3}\, d_{BL3}\, d_{CL0} + D_7\, d_{AL0}\, d_{BL0}\, d_{CL1} + D_8\, d_{AL1}\, d_{BL0}\, d_{CL1} + D_9\, d_{AL2}\, d_{BL0}\, d_{CL1} + D_{10}\, d_{AL3}\, d_{BL0}\, d_{CL1} + D_9\, d_{AL0}\, d_{BL1}\, d_{CL1} + D_{10}\, d_{AL1}\, d_{BL1}\, d_{CL1} + D_{11}\, d_{AL2}\, d_{BL1}\, d_{CL1} + D_{12}\, d_{AL3}\, d_{BL1}\, d_{CL1} + D_{11}\, d_{AL0}\, d_{BL2}\, d_{CL1} + D_{12}\, d_{AL1}\, d_{BL2}\, d_{CL1} + D_{13}\, d_{AL2}\, d_{BL2}\, d_{CL1} + D_{14}\, d_{AL3}\, d_{BL2}\, d_{CL1} + D_{13}\, d_{AL0}\, d_{BL3}\, d_{CL1} + D_{14}\, d_{AL1}\, d_{BL3}\, d_{CL1} + D_{15}\, d_{AL2}\, d_{BL3}\, d_{CL1} + D_{16}\, d_{AL3}\, d_{BL3}\, d_{CL1} + D_{11}\, d_{AL0}\, d_{BL0}\, d_{CL2} + D_{12}\, d_{AL1}\, d_{BL0}\, d_{CL2} + D_{13}\, d_{AL2}\, d_{BL0}\, d_{CL2} + D_{14}\, d_{AL3}\, d_{BL0}\, d_{CL2} + D_{13}\, d_{AL0}\, d_{BL1}\, d_{CL2} + D_{14}\, d_{AL1}\, d_{BL1}\, d_{CL2} + D_{15}\, d_{AL2}\, d_{BL1}\, d_{CL2} + D_{16}\, d_{AL3}\, d_{BL1}\, d_{CL2} + D_{15}\, d_{AL0}\, d_{BL2}\, d_{CL2} + D_{16}\, d_{AL1}\, d_{BL2}\, d_{CL2} + D_{17}\, d_{AL2}\, d_{BL2}\, d_{CL2} + D_{18}\, d_{AL3}\, d_{BL2}\, d_{CL2} + D_{17}\, d_{AL0}\, d_{BL3}\, d_{CL2} + D_{18}\, d_{AL1}\, d_{BL3}\, d_{CL2} + D_{19}\, d_{AL2}\, d_{BL3}\, d_{CL2} + D_{20}\, d_{AL3}\, d_{BL3}\, d_{CL2} + D_{15}\, d_{AL0}\, d_{BL0}\, d_{CL3} + D_{16}\, d_{AL1}\, d_{BL0}\, d_{CL3} + D_{17}\, d_{AL2}\, d_{BL0}\, d_{CL3} + D_{18}\, d_{AL3}\, d_{BL0}\, d_{CL3} + D_{17}\, d_{AL0}\, d_{BL1}\, d_{CL3} + D_{18}\, d_{AL1}\, d_{BL1}\, d_{CL3} + D_{19}\, d_{AL2}\, d_{BL1}\, d_{CL3} + D_{20}\, d_{AL3}\, d_{BL1}\, d_{CL3} + D_{19}\, d_{AL0}\, d_{BL2}\, d_{CL3} + D_{20}\, d_{AL1}\, d_{BL2}\, d_{CL3} + D_{21}\, d_{AL2}\, d_{BL2}\, d_{CL3} + D_{22}\, d_{AL3}\, d_{BL2}\, d_{CL3} + D_{21}\, d_{AL0}\, d_{BL3}\, d_{CL3} + D_{22}\, d_{AL1}\, d_{BL3}\, d_{CL3} + D_{23}\, d_{AL2}\, d_{BL3}\, d_{CL3} + D_{24}\, d_{AL3}\, d_{BL3}\, d_{CL3}$$

ดังนั้นสมการที่ได้คือ

$$
\begin{aligned}
C_{LLL6} = \; & D_3 [d_{AL0}\, d_{BL0}\, d_{CL0}] + \\
& D_4 [d_{AL1}\, d_{BL0}\, d_{CL0}] + \\
& D_5 [d_{AL2}\, d_{BL0}\, d_{CL0} + d_{AL0}\, d_{BL1}\, d_{CL0}] + \\
& D_6 [d_{AL3}\, d_{BL0}\, d_{CL0} + d_{AL1}\, d_{BL1}\, d_{CL0}] + \\
& D_7 [d_{AL2}\, d_{BL1}\, d_{CL0} + d_{AL0}\, d_{BL2}\, d_{CL0} + d_{AL0}\, d_{BL0}\, d_{CL1}] + \\
& D_8 [d_{AL3}\, d_{BL1}\, d_{CL0} + d_{AL1}\, d_{BL2}\, d_{CL0} + d_{AL1}\, d_{BL0}\, d_{CL1}] + \\
& D_9 [d_{AL2}\, d_{BL2}\, d_{CL0} + d_{AL0}\, d_{BL3}\, d_{CL0} + d_{AL2}\, d_{BL0}\, d_{CL1} + d_{AL0}\, d_{BL1}\, d_{CL1}] + \\
& D_{10} [d_{AL3}\, d_{BL2}\, d_{CL0} + d_{AL1}\, d_{BL3}\, d_{CL0} + d_{AL3}\, d_{BL0}\, d_{CL1} + d_{AL1}\, d_{BL1}\, d_{CL1}] + \\
& D_{11} [d_{AL2}\, d_{BL3}\, d_{CL0} + d_{AL2}\, d_{BL1}\, d_{CL1} + d_{AL0}\, d_{BL2}\, d_{CL1} + d_{AL0}\, d_{BL0}\, d_{CL2}] + \\
& D_{12} [d_{AL3}\, d_{BL3}\, d_{CL0} + d_{AL3}\, d_{BL1}\, d_{CL1} + d_{AL1}\, d_{BL2}\, d_{CL1} + d_{AL1}\, d_{BL0}\, d_{CL}] + \\
& D_{13} [d_{AL2}\, d_{BL2}\, d_{CL1} + d_{AL2}\, d_{BL0}\, d_{CL2} + d_{AL0}\, d_{BL3}\, d_{CL1} + d_{AL0}\, d_{BL1}\, d_{CL2}] + \\
& D_{14} [d_{AL3}\, d_{BL2}\, d_{CL1} + d_{AL1}\, d_{BL3}\, d_{CL1} + d_{AL3}\, d_{BL0}\, d_{CL2} + d_{AL1}\, d_{BL1}\, d_{CL2}] + \\
& D_{15} [d_{AL2}\, d_{BL3}\, d_{CL1} + d_{AL2}\, d_{BL1}\, d_{CL2} + d_{AL0}\, d_{BL2}\, d_{CL2} + d_{AL0}\, d_{BL0}\, d_{CL}] + \\
& D_{16} [d_{AL3}\, d_{BL3}\, d_{CL1} + d_{AL3}\, d_{BL1}\, d_{CL2} + d_{AL1}\, d_{BL2}\, d_{CL2} + d_{AL1}\, d_{BL0}\, d_{CL3}] + \\
& D_{17} [d_{AL2}\, d_{BL2}\, d_{CL2} + d_{AL0}\, d_{BL3}\, d_{CL2} + d_{AL2}\, d_{BL0}\, d_{CL3} + d_{AL0}\, d_{BL1}\, d_{CL3}] + \\
& D_{18} [d_{AL3}\, d_{BL2}\, d_{CL2} + d_{AL1}\, d_{BL3}\, d_{CL2} + d_{AL3}\, d_{BL0}\, d_{CL3} + d_{AL1}\, d_{BL1}\, d_{CL3}] + \\
& D_{19} [d_{AL2}\, d_{BL3}\, d_{CL2} + d_{AL2}\, d_{BL1}\, d_{CL3} + d_{AL0}\, d_{BL2}\, d_{CL3}] + \\
& D_{20} [d_{AL3}\, d_{BL3}\, d_{CL2} + d_{AL3}\, d_{BL1}\, d_{CL3} + d_{AL1}\, d_{BL2}\, d_{CL3}] + \\
& D_{21} [d_{AL2}\, d_{BL2}\, d_{CL3} + d_{AL0}\, d_{BL3}\, d_{CL3}] + \\
& D_{22} [d_{AL3}\, d_{BL2}\, d_{CL3} + d_{AL1}\, d_{BL3}\, d_{CL3}] + \\
& D_{23} [d_{AL2}\, d_{BL3}\, d_{CL3}] + \\
& D_{24} [d_{AL3}\, d_{BL3}\, d_{CL3}]
\end{aligned}
$$

จากขั้นตอนการคูณสัมประสิทธิ์ดังกล่าวสามารถสรุปได้ดังสมการที่ 3.3 เมื่อทำการแปลงสมการเวฟเลตแม่ในแต่ละระดับที่ต้องการแล้ว ก็จะทำการเปลี่ยนค่าตัวคูณในตำแหน่งที่ต้องการเพื่อให้ได้ค่าตัวกรองของเวฟเลตเพคเกจ 3 ระดับซึ่งในแต่ละพจน์ ค่าของตัวกรองสามารถที่จะคำนวณหาค่าคงที่ได้ เมื่อนำค่าคงที่มาออกแบบวงจรจะช่วยลดความซับซ้อนของวงจรได้

$$
\begin{aligned}
C_{LLLn} = \; & D_{4n} [d_{AL3}\, d_{BL3}\, d_{CL3}] + \\
& D_{4n-1} [d_{AL2}\, d_{BL3}\, d_{CL3}] +
\end{aligned}
$$

$$D_{4n-2}\,[d_{AL3}\,d_{BL2}\,d_{CL3} + d_{AL1}\,d_{BL3}\,d_{CL3}] +$$

$$D_{4n-3}\,[d_{AL2}\,d_{BL2}\,d_{CL3} + d_{AL0}\,d_{BL3}\,d_{CL3}] +$$

$$D_{4n-4}\,[d_{AL3}\,d_{BL3}\,d_{CL2} + d_{AL3}\,d_{BL1}\,d_{CL3} + d_{AL1}\,d_{BL2}\,d_{CL3}] +$$

$$D_{4n-5}\,[d_{AL2}\,d_{BL3}\,d_{CL2} + d_{AL2}\,d_{BL1}\,d_{CL3} + d_{AL0}\,d_{BL2}\,d_{CL3}] +$$

$$D_{4n-6}\,[d_{AL3}\,d_{BL2}\,d_{CL2} + d_{AL1}\,d_{BL3}\,d_{CL2} + d_{AL3}\,d_{BL0}\,d_{CL3} + d_{AL1}\,d_{BL1}\,d_{CL3}] +$$

$$D_{4n-7}\,[d_{AL2}\,d_{BL2}\,d_{CL2} + d_{AL0}\,d_{BL3}\,d_{CL2} + d_{AL2}\,d_{BL0}\,d_{CL3} + d_{AL0}\,d_{BL1}\,d_{CL3}] +$$

$$D_{4n-8}\,[d_{AL3}\,d_{BL3}\,d_{CL1} + d_{AL3}\,d_{BL1}\,d_{CL2} + d_{AL1}\,d_{BL2}\,d_{CL2} + d_{AL1}\,d_{BL0}\,d_{CL3}\,] +$$

$$D_{4n-9}\,[d_{AL2}\,d_{BL3}\,d_{CL1} + d_{AL2}\,d_{BL1}\,d_{CL2} + d_{AL0}\,d_{BL2}\,d_{CL2} + d_{AL0}\,d_{BL0}\,d_{CL3}] +$$

$$D_{4n-10}\,[d_{AL3}\,d_{BL2}\,d_{CL1} + d_{AL1}\,d_{BL3}\,d_{CL1} + d_{AL3}\,d_{BL0}\,d_{CL2} + d_{AL1}\,d_{BL1}\,d_{CL2}] +$$

$$D_{4n-11}\,[d_{AL2}\,d_{BL2}\,d_{CL1} + d_{AL2}\,d_{BL0}\,d_{CL2} + d_{AL0}\,d_{BL3}\,d_{CL1} + d_{AL0}\,d_{BL1}\,d_{CL2}] +$$

$$D_{4n-12}\,[d_{AL3}\,d_{BL3}\,d_{CL0} + d_{AL3}\,d_{BL1}\,d_{CL1} + d_{AL1}\,d_{BL2}\,d_{CL1} + d_{AL1}\,d_{BL0}\,d_{CL2}] +$$

$$D_{4n-13}\,[d_{AL2}\,d_{BL3}\,d_{CL0} + d_{AL2}\,d_{BL1}\,d_{CL1} + d_{AL0}\,d_{BL2}\,d_{CL1} + d_{AL0}\,d_{BL0}\,d_{CL2}] +$$

$$D_{4n-14}\,[d_{AL3}\,d_{BL2}\,d_{CL0} + d_{AL1}\,d_{BL3}\,d_{CL0} + d_{AL3}\,d_{BL0}\,d_{CL1} + d_{AL1}\,d_{BL1}\,d_{CL1}] +$$

$$D_{4n-15}\,[d_{AL2}\,d_{BL2}\,d_{CL0} + d_{AL0}\,d_{BL3}\,d_{CL0} + d_{AL2}\,d_{BL0}\,d_{CL1} + d_{AL0}\,d_{BL1}\,d_{CL1}] +$$

$$D_{4n-16}\,[d_{AL3}\,d_{BL1}\,d_{CL0} + d_{AL1}\,d_{BL2}\,d_{CL0} + d_{AL1}\,d_{BL0}\,d_{CL1}] +$$

$$D_{4n-17}\,[d_{AL2}\,d_{BL1}\,d_{CL0} + d_{AL0}\,d_{BL2}\,d_{CL0} + d_{AL0}\,d_{BL0}\,d_{CL1}] +$$

$$D_{4n-18}\,[d_{AL3}\,d_{BL0}\,d_{CL0} + d_{AL1}\,d_{BL1}\,d_{CL0}] +$$

$$D_{4n-19}\,[d_{AL2}\,d_{BL0}\,d_{CL0} + d_{AL0}\,d_{BL1}\,d_{CL0}] +$$

$$D_{4n-20}\,[d_{AL1}\,d_{BL0}\,d_{CL0}] +$$

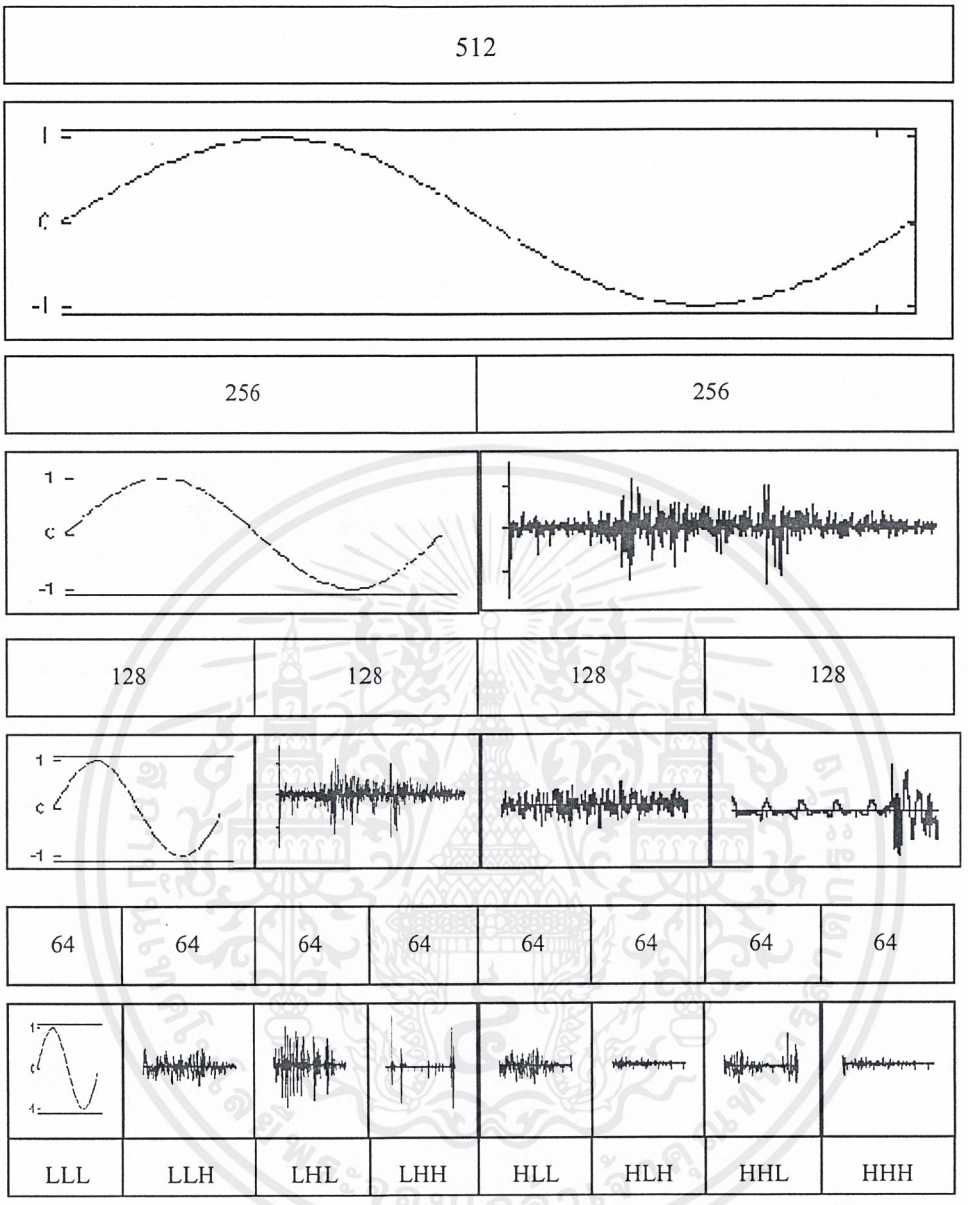
$$D_{4n-21}\,[d_{AL0}\,d_{BL0}\,d_{CL0}] \qquad (3.3)$$

ซึ่งค่าของ d นั้นเป็นเวฟเลตแม่ได้เลือกใช้ Darbechie 4 ซึ่งจะใช้ค่าตัวกรองสัมประสิทธิ์ดังนี้คือ

dbl = {0.4830, 0.830, 0.2241, - 0.1294}

dbh = {- 0.1294, - 0.2241, 0.8365, - 0.4830}

จึงสามารถที่จะจัดให้อยู่ในรูปค่าคงที่ได้ดังตารางที่ 3.1



รูปที่ 3.14 ข้อมูลในแต่ละระดับการทำเวฟเลตเพคเกจ 3 ระดับ

สำหรับค่าของ Cn ที่ได้นั้นในขั้นตอนสุดท้ายนี้จะเลือกใช้เพียงค่าที่เป็นลำดับข้อมูลเลขคู่เท่านั้น เพราะจะเป็นการลดอัตราการสุ่มข้อมูลในระดับสุดท้าย ซึ่งจะทำให้ได้ข้อมูลทางด้าน LLL ในระดับสุดท้ายมีเพียง 64 ข้อมูล ซึ่งในการกระทำดังกล่าวเมื่อต้องการค่าในทิศทางอื่นต้องเปลี่ยนค่า ตัวกรองในสมการ 3.1 จึงจะได้ค่าดังตารางที่ 3.1 ซึ่งจะทำให้ได้ค่าที่ต่างกันออกไป โดยค่าที่ได้ทั้งหมดใน C Level นั้นจะมีการแยกช่วงของความถี่ที่ต่างกันออกไป โดยแต่ละช่วงของความถี่นั้นจะมีค่าเพียง 64 ข้อมูล ซึ่งในกระบวนการกระทำดังกล่าวจะทำให้สามารถลดกระบวนการที่ซับซ้อน

ของการแปลงสัญญาณโดยเวฟเลตเพคเกจได้ ในการนำค่าที่อยู่ในตารางมาใช้งานจะทำการคูณด้วยค่า 10,000 เข้าไปเพื่อให้ได้ค่าของทศนิยม 4 ตำแหน่งในการแปลงเวฟเลต ซึ่งตัวกรองนี้จะทำให้ได้สัมประสิทธิ์ทางด้านต่างๆ ที่ต่างกัน ซึ่งก็ขึ้นอยู่กับตัวกรองที่เลือกใช้ในตารางโดยการทำงานของเวฟเลตสามารถอธิบายได้ดังนี้

สัญญาณที่ได้จากการแปลงเวฟเลตเพคเกจนั้นจะมีลักษณะดังรูปที่ 3.14 สังเกตข้อมูลในแต่ละระดับจะมีขนาดลดลงโดยกระบวนการของการทำลดอัตราสุ่มข้อมูล โดยข้อมูลจะมีความแตกต่างกันไปตามความถี่ ซึ่งข้อมูลทางด้าน LLL จะมีลักษณะคล้ายกับสัญญาณรูปเดิมมาก คือสัมประสิทธิ์ทางด้านความถี่ต่ำ และในแต่ละทิศทางของการแปลงสัญญาณในระดับความถี่ที่ต่างกัน ซึ่งจะแยกความถีที่สูง และต่ำออกไปสังเกตการแปลงความถี่ทางด้านทางด้าน LLL จะเป็นสัมประสิทธิ์ทางด้านความ ถี่ต่ำ ซึ่งจะคงลักษณะของสัญญาณรูปเดิมไว้ได้มากที่สุด

3.3.2 การออกแบบวงจรแปลงเวฟเลต

จากสมการที่ 3.1 จะเห็นได้ว่าต้องมีทั้งการคูณ และการบวกซึ่งในการหาค่าสัมประสิทธิ์ของสัญญาณในแต่ละระดับความถี่นั้นจะใช้ตัวกรองที่ต่างกันออกไปดังในตารางที่ 3.1 ซึ่งค่าสัมประสิทธิ์ที่ได้แต่ละค่าจากสัญญาณเดิมนั้นจะต้องมีการนำข้อมูลเข้ามาจัดการครั้งละ 1 ชุดข้อมูล ในหนึ่งชุดข้อมูลจะมีข้อมูลได้ 2^n ข้อมูล ซึ่งในวงจรที่ออกแบบนั้นจะใช้ข้อมูลในการแปลงแต่ละชุดคือ $2^9 = 512$ ข้อมูล โดยทำการแปลงเวฟเลต 3 ระดับจะมีค่าอยู่ที่ 64 ข้อมูล ซึ่งผลที่ได้ก็ คือข้อมูลจะถูกแยกออกในทางด้านความถี่ต่ำ และความถี่สูง จะได้ข้อมูลทั้งหมดคือ 512 ข้อมูล แต่ในการบีบอัดข้อมูลนั้นจำเป็นที่จะต้องมีขั้นตอนของการเข้ารหัส ซึ่งจะเป็นการจัดการกับความซ้ำซ้อนของข้อมูลที่เกิดจากการตัดระดับ (Threshold) ที่ไม่ต้องการออกไป โดยในส่วนของการแปลงเวฟเลตนั้นจะเป็นการแยกเอาความถี่แต่ละช่วงออกจากกัน แล้วตัดระดับความถี่ที่มีค่าเป็นความถี่สูงออกไป โดยทำการจัดระดับที่มีค่าแอมปลิจูดต่ำกว่าระดับของการ Threshold ให้มีค่าเป็นศูนย์ และเป็นผลทำให้ค่าที่เกิดขึ้นทางด้านความถี่สูงนั้นถูกตัดให้เป็นศูนย์เป็นจำนวนมากจะเหลือไว้เฉพาะคุณสมบัติทางด้านความถี่ต่ำ ซึ่งก็คือ สัญญาณจะยังคงมีลักษณะรูปร่างของสัญญาณรูปเดิม เพียงแต่จะมีขนาดของข้อมูลที่ต่างไป

ในช่วงแรกของการทำงานของวงจรจะมีการอ่านค่าจากวงจรแปลงสัญญาณแอนะลอกเป็นสัญญาณดิจิตอลจัดเก็บในวงจรพักข้อมูลที่สามารถเลื่อนข้อมูลได้ (วงจรเลื่อนข้อมูล) โดยจะทำการเลื่อนข้อมูลเข้ามาครั้งละ 8 ข้อมูล แล้วทำการคูณกับตัวกรองระบบจากตารางในทิศทางต่างๆ ซึ่งจะทำการคูณการคูณเกิดขึ้นทั้งหมด 22 ครั้ง ซึ่งในแต่ละรอบของการคูณนั้นจะมีการเก็บค่าไว้ในหน่วยความจำเพื่อทำการบวกค่าที่คูณเข้ามาใหม่จนครบ 22 ค่า โดยจะมีวงจรตัวนับ 22 เป็นตัวควบคุมให้เกิดการคูณ และบวกจำนวน 22 รอบ (จำนวนรอบที่ได้เกิดจากจำนวนของค่าตัวกรองที่คำนวณได้

ซึ่งจะเป็นจำนวนพจน์ของสมการ ที่ 3.1) เมื่อทำการคูณกับค่าตัวกรองแล้วทำการบวกกันครบ 22 ครั้ง จะทำให้ได้ค่าที่ผ่านการแปลงเวฟเลตทั้งหมด 3 ระดับ โดยในการทำการเลื่อนข้อมูลเข้าแต่ละครั้งจะเลื่อนเข้ามาครั้งละ 8 ข้อมูล (ให้สังเกตจากสมการ 3.1) ซึ่งใน 8 ข้อมูลนี้จะต้องกระทำกับตัวกรองทุกระดับ ซึ่งจะทำให้ได้ค่า C_{LLL0}, C_{LLH0}, C_{LHL0}, C_{LHH0}, C_{HLL0}, C_{HLH0}, C_{HHL0}, และ C_{HHH0} ออกมาก่อนแล้วค่อยทำการเลื่อนข้อมูลเข้ามาในหน่วยความจำที่สามารถเลื่อนข้อมูลได้ (วงจรเลื่อนข้อมูล) อีก 8 ข้อมูล โดยข้อมูลชุดเดิมจะต้องเลื่อนไปจนกว่าจะถึงรีจีสเตอร์ตัวสุดท้าย ในการเลื่อนข้อมูลครั้งต่อไปถ้าค่าที่เกินรีจีสเตอร์ตัวสุดท้ายจะถูกตัดทิ้งไปไม่นำมาคูณ และบวกอีก (เพราะในสมการ 3.1 จะมีการนำค่าของผลคูณมาบวกกันทั้งหมดเพียง 22 พจน์เท่านั้น) ซึ่งในการทำงานจะมีลักษณะดังรูป 3.15 ต่อไปนี้

db_0	db_1	db_2	db_3	db_4	db_5	db_6	db_7	db_8	db_9	db_{10}	db_{11}	db_{12}	db_{13}	db_{14}	db_{15}	db_{16}	db_{17}	db_{18}	db_{19}	db_{20}	db_{21}
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

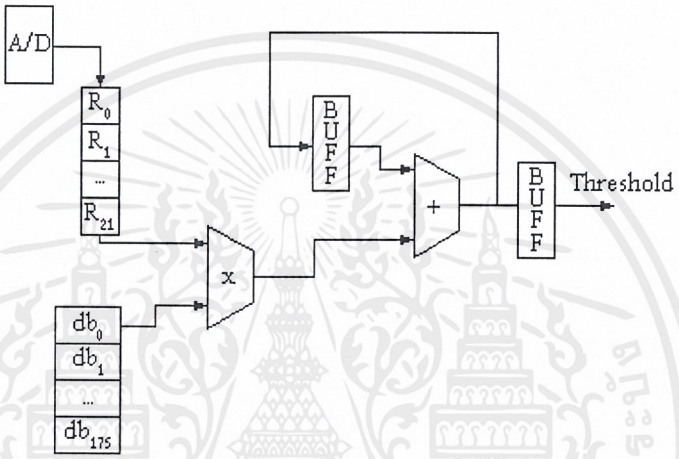
รูปที่ 3.15 การเลื่อนข้อมูลสำหรับทำการคูณกับตัวกรองในครั้งแรก

การอ่านข้อมูล 8 ข้อมูลเข้ามาในหน่วยความจำเลื่อนข้อมูลได้ชุดแรก แล้วทำการคูณกับตัวกรองในทิศทางต่างๆ พร้อมกับบวกกันในแต่ละพจน์ของการคูณ เมื่อเปลี่ยน (การเปลี่ยนค่าตัวกรองจะกระทำโดยการอ่านค่าตัวกรองที่เรียงกันไปเรื่อยๆ โดยค่าดังกล่าวจะเป็นค่าตัวกรองทุกระดับซึ่งจะมีทั้งหมด 176 ค่า เมื่อทำการคูณครบทุกค่าจะวนกลับมาชี้ที่ค่าเริ่มต้นใหม่ ซึ่งก็จะทำให้ได้ค่าต่างๆ ในการแปลงทุกระดับในการเลื่อนข้อมูลแต่ละครั้ง) ค่าตัวกรองในระดับต่างๆ จะทำให้ได้ค่าที่ทำการแปลง เวฟเลตเพคเกจ 3 ระดับของข้อมูลตัวแรก (C_0) ดังรูป 3.16

db_0	db_1	db_2	db_3	db_4	db_5	db_6	db_7	db_8	db_9	db_{10}	db_{11}	db_{12}	db_{13}	db_{14}	db_{15}	db_{16}	db_{17}	db_{18}	db_{19}	db_{20}	db_{21}
D_{15}	D_{14}	D_{13}	D_{12}	D_{11}	D_{10}	D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	0	0	0	0	0	0

รูปที่ 3.16 การเลื่อนข้อมูลเพื่อทำการคูณกับตัวกรองในครั้งที่สอง

เมื่อทำการคูณ และการบวกครบทั้ง 22 พจน์แล้วก็จะทำการรับข้อมูลเข้ามาใหม่อีก 8 ข้อมูล เพื่อหาค่าสัมประสิทธิ์ในการแปลงเวฟเลตของสัญญาณ ซึ่งจะได้เป็นข้อมูลตัวที่สอง (C1) ดังรูปที่ 3.16 ซึ่งจะกระทำในลักษณะนี้ไปเรื่อยๆ จนครบ 64 ครั้ง ซึ่งจะทำให้ได้สัมประสิทธิ์ในแต่ละระดับของการสัญญาณที่เข้ามา โดยที่ค่าที่ได้จากกระบวนการดังกล่าวจะมีขนาด 32 บิต สำหรับบิตที่ 32 จะเป็นบิตเครื่องหมาย ซึ่งจะเป็นตัวควบคุมการทำงานของวงจรว่าให้ทำการบวกค่า หรือลบค่าที่ผ่านการคูณเข้ามา ดังรูปที่ 3.17 เป็นการแสดงการทำงานทั้งหมดของการแปลงเวฟเลตเพคเกจ



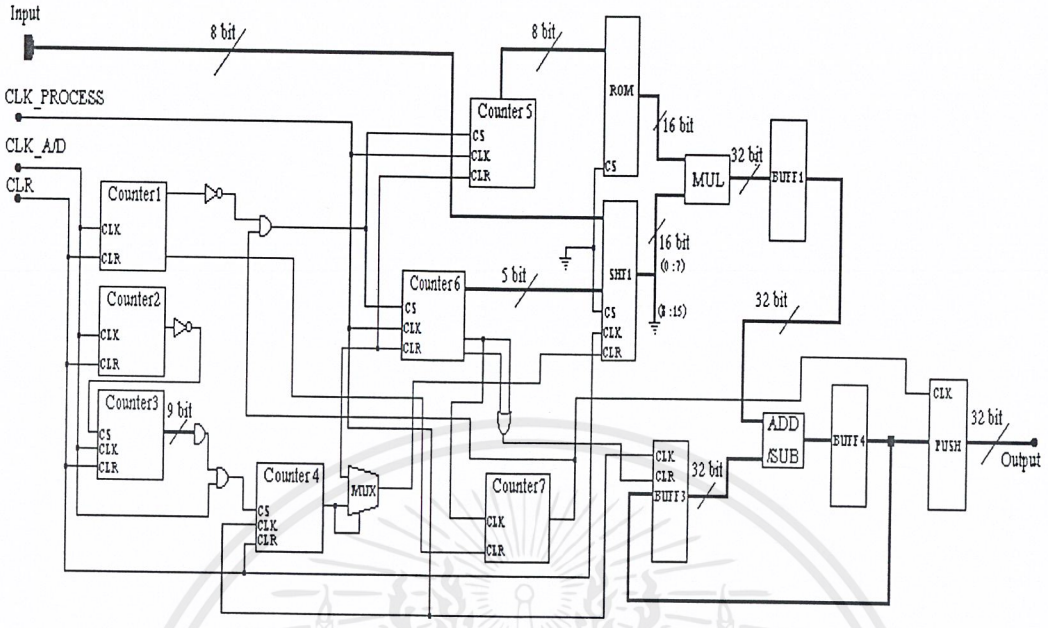
รูปที่ 3.17 ขั้นตอนในการแปลงเวฟเลตเพคเกจ

ในการรับข้อมูลเข้ามาเพื่อทำการแปลงเวฟเลตเพคเกจนั้นจะต้องรับค่าเข้ามาครั้งละ 512 ข้อมูล ซึ่งในการออกแบบจำเป็นที่จะต้องใช้เกตจำนวนมากในการใช้งานเพื่อสร้างวงจรสำหรับเก็บข้อมูลขนาด 8 บิต จำนวน 512 ตำแหน่ง ดังนั้นจึงใช้วงจรเลื่อนข้อมูล (วงจรเลื่อนข้อมูล) แทน ซึ่งเป็นวงจรเลื่อนข้อมูลขนาด 8 บิตมีลักษณะการทำงานดังในสมการที่ 3.1 ซึ่งผลที่ได้คือ จะมีการเลื่อนข้อมูลครั้งละ 8 ข้อมูลเพื่อทำการคูณกับค่าของตัวกรองที่ได้คำนวณไว้ ดังที่ได้อธิบายไว้ในตอนแรก จึงทำการคูณกับค่าของตัวกรองในตาราง แล้วทำการบวกกันจนครบ 22 ครั้ง ซึ่งจะถือว่าได้ค่าที่ผ่านการแปลงเวฟเลตออกมา 1 ค่าทางด้าน LLL ซึ่งเป็นค่าแรก แล้วก็ทำการเลื่อนข้อมูลในวงจรเลื่อนข้อมูลจนครบ 8 รอบ

ในการเลื่อนข้อมูลเข้ามาในแต่ละครั้งจะมีวงจรตัวนับตัวที่ 1 เป็นตัวควบคุมการรับค่าเข้ามา โดยจากวงจรในรูปที่ 3.18 เมื่อมีสัญญาณนาฬิกา Clk_A/D ซึ่งจะเป็นสัญญาณนาฬิกาตัวเดียวกันกับสัญญาณนาฬิกาในการสุ่มตัวอย่าง (Sampling) สัญญาณ โดยวงจรดังกล่าวจะมีลักษณะการทำงานแบบมีเงื่อนไข ซึ่งถ้าวงจรตัวนับตัวที่ 1 ทำการนับข้อมูลที่อ่านเข้ามาได้ครบ 8 ข้อมูล จะทำการส่ง

ค่าออกมาเป็นระดับลอจิก 0 ไปกระทำกับลอจิก NOT ซึ่งจะเปลี่ยนเป็นระดับลอจิก 1 ไปทำการ AND กับ เงื่อนไขที่ถูกส่งออกมาจาก วงจรตัวนับตัวที่ 7 โดยในครั้งแรกค่าที่ถูกส่งออกมาจะเป็นระดับลอจิก 1 ซึ่งเมื่อทำการ กับตัวกระทำทางลอจิกแอนด์ จะทำให้ผลลัพธ์ที่ได้ออกมาเป็นระดับลอจิก 1 ไปควบคุมให้ วงจรตัวนับตัวที่ 6 และ วงจรตัวนับตัวที่ 7 เริ่มนับ โดยวงจร วงจรตัวนับตัวที่ 5 จะส่งค่าที่นับ 1 ถึง 176 ไป ชี้ตำแหน่งค่าที่ถูกจัดเก็บไว้ในหน่วยความจำถาวร ซึ่งในขณะเดียวกัน วงจรตัวนับตัวที่ 6 ก็จะทำการส่งค่าที่ทำการนับ(นับ 1 ถึง 22) ไปถอดรหัสเพื่ออ่านข้อมูลออกจาก ดีฟลิป-ฟลอป แต่ละตัวใน SHF1 เพื่อไปทำการคูณกันในส่วนของ MUL แล้วทำการบวก หรือลบค่าของวงจร (Add/Sub) ในการบวก และลบค่าแต่ละครั้งจะนำผลลัพธ์ที่ได้ไปเก็บไว้ใน BUF4 แล้วส่งไปเก็บยัง BUF3 เพื่อรอทำการบวกรอบถัดไป ซึ่งผลที่ได้ตรงนี้คือ วงจรตัวนับตัวที่ 5 และ 6 จะทำการชี้ข้อมูลไปพร้อมๆ กัน พร้อมทั้งทำการคูณ และบวก ซึ่งจะกระทำการนับค่าเพิ่มขึ้นเรื่อยๆ จนกระทั่ง วงจรตัวนับตัวที่ 6 ทำการนับจนครบ 22 จะทำการส่งระดับลอจิก 0 ควบคุมวงจรเลื่อนข้อมูล SHF2 เพื่อนำค่าที่ได้จากการบวกจำนวน 22 รอบซึ่งนำมากเก็บมาเก็บไว้ใน ดี ฟลิป – ฟลอปตัวแรก (วงจรภายในออกแบบแล้วใช้ร่วมกันกับส่วนของเข้ารหัสและถอดรหัสฮัฟฟ์แมน) ในขณะเดียวกันค่าที่ถูกส่งออกมาจากวงจรตัวนับที่6 ส่งค่าไปทำการลบข้อมูลใน BUF3 เพื่อให้ค่าที่อยู่ภายในเป็น 0 ทั้ง 8 บิต พร้อมส่งข้อมูลไปให้วงจร วงจรตัวนับตัวที่ 7 ซึ่งเป็น วงจรตัวนับ 1 ถึง 8 เพื่อควบคุม ข้อมูลให้ได้ทุกค่าในแต่ละทิศทาง (LLL, LLH, LHL, LHH, ... , HHH) ซึ่งก็คือ 8 ทิศทางที่ต้องกระทำกับข้อมูลในแต่ละครั้งที่ทำการเลื่อนข้อมูลเข้ามา ซึ่งจากการที่ วงจรตัวนับตัวที่ 6 ทำการนับครบ 22 นั้น แสดงว่าวงจร วงจรตัวนับตัวที่ 5 จะไป ชี้ที่ตำแหน่งที่ 22 ด้วย พร้อมทั้ง MUL และ Add/Sub ได้ทำงานทั้งหมด 22 ครั้ง ซึ่งผลที่ได้ก็คือ จะได้ค่าที่ผ่านการแปลงเวฟเลตออกมา 1 ค่า ในทิศทาง LLL หรือ LLL_0

สำหรับสัมประสิทธิ์ที่ได้จากการแปลงเวฟเลต จะมีคาบเวลาค่อนข้างต่ำหรือความถี่สูง ดังนั้นไม่สามารถที่จะนำไปเข้าสู่ภาคควอนไตธ์ได้ในทันที เพราะการควอนไตธ์มีขีดจำกัดของระยะเวลาที่จะต้องประมวลผลการควอนไตธ์ ซึ่งสัญญาณนาฬิกาของภาคควอนไตธ์จะไม่สามารถจัดระดับข้อมูลทัน และในอีกกรณีหนึ่งคือจะต้องคำนึงถึงส่วนของวงจรเข้ารหัสฮัฟฟ์แมนซึ่งได้ออกแบบเอาไว้ให้นำสัมประสิทธิ์เวฟเลต มาเข้ารหัสฮัฟฟ์แมนในช่วงระยะเวลาทุก 2 เท่าของสัญญาณนาฬิกาที่นำไปอ่านข้อมูลมาจาก A/D (Clock A/D) จึงต้องมีการจัดเก็บพักข้อมูลเอาไว้ในวงจรเลื่อนข้อมูลอีกชุดหนึ่งซึ่งมีขนาด 32 บิต ทั้งหมด 8 ชุด แล้วทำการควอนไตธ์ในช่วงระยะเวลา 7 สัญญาณนาฬิกาซึ่งอ่านข้อมูลในช่วงระยะเวลาถัดไปจาก A/D ที่รับข้อมูลเข้ามาใหม่ดังได้ที่กล่าวมาแล้วข้างต้น



รูปที่ 3.18 วงจรแปลงเวฟเลต

หลังจากที่ วงจรตัวนับตัวที่ 6 ทำการส่งสัญญาณไปควบคุมส่วนอื่นๆ แล้วก็จะกลับมาเริ่มนับที่ 1 ใหม่ พร้อมทั้ง วงจรตัวนับตัวที่ 5 ก็จะทำการนับต่อไปจาก 23, 24, ..., 44 ซึ่งจะตรงกับวงจรตัวนับ ตัวที่ 6 นับครบ 22 วงจรตัวนับตัวที่ 6 ก็จะส่งสัญญาณมาให้ วงจรตัวนับตัวที่ 7 เพื่อทำการเพิ่มค่า และในส่วนอื่นก็จะมีการทำงานเหมือนกัน ซึ่งวงจรจะทำงานลักษณะนี้ไปจนกระทั่งวงจรตัวนับตัวที่ 7 นับถึง 8 ซึ่ง วงจรตัวนับตัวที่ 5 ก็จะนับถึง 176 พร้อมกับ วงจรตัวนับตัวที่ 6 นับครบ 22 เป็นจำนวน 8 รอบ และส่วนของ SHF2 ก็จะเก็บค่าของการแปลงเวฟเลตไว้ภายในหน่วยความจำ จำนวน 8 ค่า ซึ่งเป็น ข้อมูลขนาด 32 บิต โดยจะมีลักษณะการวางตัวของข้อมูล ดังรูปที่ 3.19 เมื่อ วงจรตัวนับตัวที่ 7 ทำการ นับครบ 8 จะทำการส่งระดับ ลอจิก 0 ควบคุมวงจรตัวนับตัวที่ 5 และวงจรตัวนับตัวที่ 6 โดยจะไปทำการ AND กับค่าค่าที่ส่งออกมาจาก วงจรตัวนับตัวที่ 1

F-F7	F-F6	F-F5	F-F4	F-F3	F-F2	F-F1	F-F0
LLL_0	LLH_0	LHL_0	LHH_0	HLL_0	HLH_0	HHL_0	HHH_0

รูปที่ 3.19 ลักษณะการเรียงของข้อมูลภายใน ดี ฟลิปฟลอป ของวงจร SHF2

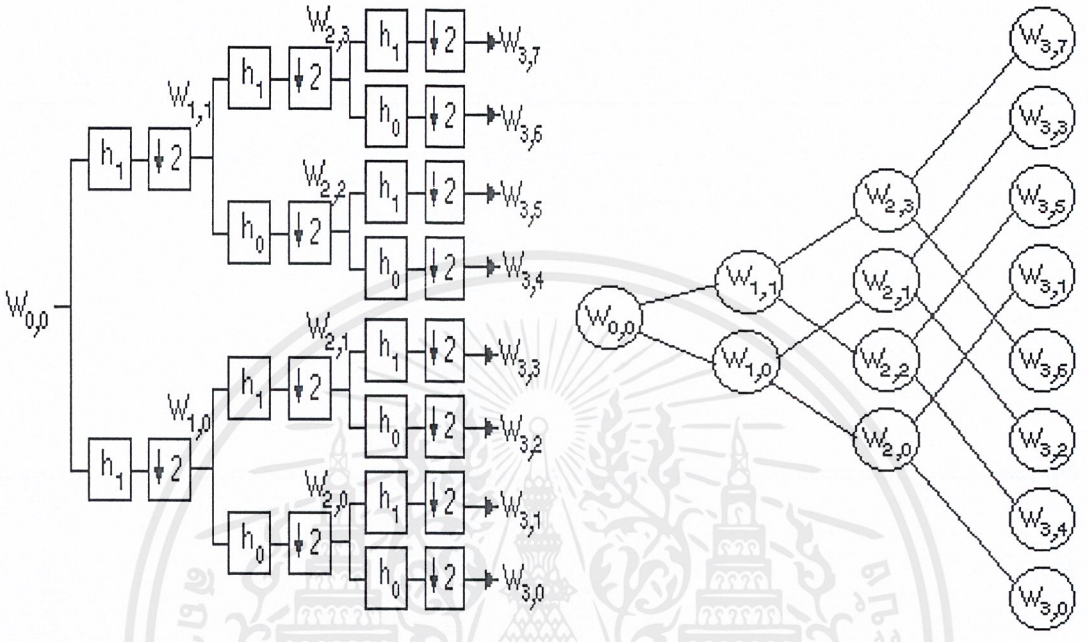
จากขบวนการทำงานของการแปลงเวฟเลตนั้นจะใช้สัญญาณนาฬิกาที่ต่างกับสัญญาณนาฬิกาของการอ่านข้อมูลเข้ามา ซึ่งสัญญาณนาฬิกาจะต้องมีความเร็วกว่าสัญญาณนาฬิกาของข้อมูลเข้ามา

เมื่อขบวนการดังกล่าวเสร็จสิ้นก็จะทำการอ่านข้อมูลเข้ามาอีก 8 ข้อมูล ซึ่งเมื่อ วงจรตัวนับตัวที่ 1 ทำการนับครบ 8 สร้างสัญญาณ 0 ออกไปควบคุม วงจรตัวนับตัวที่ 5 และวงจรตัวนับตัวที่ 6 พร้อมทั้งส่ง สัญญาณควบคุมออกไปทำการควบคุมให้ วงจรตัวนับตัวที่ 7 กลับไปนับค่าเริ่มที่ศูนย์ ขณะเดียวกันข้อมูล ที่มีอยู่ใน SHF1 จะถูกเลื่อนไปยัง ดีฟลิป-ฟลอป ตัวถัดไปภายใน SHF1 และในส่วนของวงจร ตัวนับตัวที่ 2 จะทำการนับ 1, 2, 4, 6 ซึ่งในการนับครั้งแรกนั้นจะทำการส่งสัญญาณระดับลอจิก 0 ออกไปเพื่อเป็นเงื่อนไขอินพุตทำให้กับวงจรตัวนับตัวที่ 3 ทำการนับค่าซ้ำในเพื่อให้ข้อมูลที่อ่านเข้าไปมีจำนวนครบ 512 (ถ้าไม่มีการหลอกให้วงจรทำการนับซ้ำ จะทำให้ข้อมูลที่อ่านเข้าไปมีเพียง 511 ข้อมูล เพราะในช่วงแรกสัญญาณนาฬิกาลูกแรกจะทำให้ วงจรตัวนับตัวที่ 3 นับขึ้นไป 1 ซึ่งในการนับครั้งแรกควรจะเริ่มนับจาก 0) เมื่อ วงจรตัวนับตัวที่ 3 นับครบ 511 ก็จะได้ข้อมูลครบ 512 พอดีทุกบิตของ วงจรตัวนับจะส่งค่าออกมา โดยจะไปทำการ AND แล้วจะส่งสัญญาณ ไปทำการรีเซตวงจรเลื่อนข้อมูลสำหรับที่จะรอรับค่าเข้ามาใหม่เข้ามาแปลงเวฟเลตใหม่

3.2.3 การแปลงเวฟเลตไป - กลับในวงจรเดียวกัน

ในการลดจำนวนข้อมูลโดยการใช้เวฟเลตเป็นตัวแตกกระจายข้อมูล และเข้ารหัสด้วยฮัฟฟ์แมนนั้นถ้าทำกระบวนการดังกล่าวตามขั้นตอนปกติจะทำให้เกิดปัญหาในส่วนของการแปลงกลับเวฟเลต (Inverse Wavelet Transform) ซึ่งในการออกแบบวงจรนั้นจำเป็นต้องสร้างวงจรขนาดใหญ่ ต้องมีการทำงานเป็นแบบขนาน กระทำทุกส่วนพร้อมๆ กัน ซึ่งการกระทำกระบวนการดังกล่าวนี้จะเป็นการสิ้นเปลืองเกตภายในอุปกรณ์ FPGA พร้อมทั้งมีการออกแบบวงจรที่ยุ่งยากตามไปด้วย ดังนั้นจึงมีการนำเอาการแปลงเวฟเลตมาคิดใหม่ เพื่อให้มีลักษณะการทำงานของวงจรที่ง่ายขึ้นเพื่อลดขนาดของวงจรแปลงเวฟเลต ซึ่งวงจรดังกล่าวนี้จะสามารถที่จะนำไปใช้งานในขั้นตอนการแปลงไป และแปลงกลับได้ด้วย ลักษณะของวงจรแปลงเวฟเลตไปกลับนี้จะมีการทำงานแบบขนาน Pipeline โดยจะมีการส่งข้อมูลเข้าไปทำการแปลงแบบอนุกรม ซึ่งจะขึ้นกับจังหวะของการอ่านข้อมูลเข้ามาของ A/D โดยจะกระทำกระบวนการต่างๆ เสร็จสิ้นภายในสัญญาณนาฬิกาแต่ละลูกเมื่อทำเสร็จในระดับแรกก็จะถูกส่งไปกระทำในระดับถัดไปเรื่อยๆ ซึ่งการแปลงเวฟเลตแบบนี้จะมีกระบวนการเหมือนปกติแต่จะแตกต่างที่มีการทำงานแบบขนาน และในการแปลงเวฟเลตแต่ละระดับนั้นจะต้องมีการจัดเรียงข้อมูลภายในหน่วยความจำให้ถูกต้อง เพื่อที่จะนำข้อมูลที่ถูกจัดเรียงอย่างถูกต้องนั้นไปทำการแปลงกลับเวฟเลตโดยอาศัยวงจรแปลงกลับเวฟเลตตัวเดียวกันกับ

วงจรแปลงเวฟเลต ซึ่งในการแปลงกลับนั้นจะเปลี่ยนเพียงค่าของเวฟเลตแม่ที่ใช้ในการแปลงกลับเวฟเลต



รูปที่ 3.20 เปรียบเทียบการแปลงเวฟเลตธรรมดากับการแปลงเวฟเลตแบบใหม่

จากรูปที่ 3.20 จะเห็นว่าลักษณะของทรีที่ใช้ในการแปลงเวฟเลตนั้นยังเหมือนเดิมแต่จะแตกต่างกันทีมีการจัดเรียงข้อมูลที่ได้ในหน่วยความจำให้ถูกต้อง โดยค่าดังกล่าวนี้จะเป็นค่าที่ได้จากการแปลงเวฟเลต 3 ระดับ คือ $W_{3,j}$ ซึ่งการแปลงเวฟเลตดังรูปที่ 3.20 นั้นจะใช้สมการ 3.4 ช่วยในการแปลง ซึ่งค่าที่หาได้แต่ละระดับนั้นจะถูกส่งไปกระทำในระดับถัดไปเรื่อยๆ จนครบทุกระดับ

$$w_{i,j}(n) = \sum_{k=0}^{L-1} h_{i\%2}(k) w_{i-1,[j/2]}(n - 2^{i-1}k) \qquad (3.4)$$

โดยที่ค่าของ

w คือ ค่าของข้อมูลแต่ละระดับ

h คือ ค่าของเวฟเลตแม่โดยจะใช้ Daubechies 4

i คือ ระดับของการแปลงโดย i < I -1

j คือ ทิศทางของการแปลงโดย j < J^i

n คือ ลำดับของข้อมูลที่ต้องการของแต่ละระดับโดย

k คือ ค่าของนิพจน์แต่ละพจน์ ซึ่งจะเพิ่มขึ้นตามค่าของการบวกโดย k<L-1

L คือ จำนวนค่าของตัวกรอง

I คือ จำนวนของระดับการแปลงเวฟเลต

J คือ จำนวนแถบความถี่ของการกระจายเวฟเลต

h_0 = [0.483, 0.8365, 0.2241, -0.1294]

h_1 = [- 0.1294, -0.2241, 0.8365, -0.483]

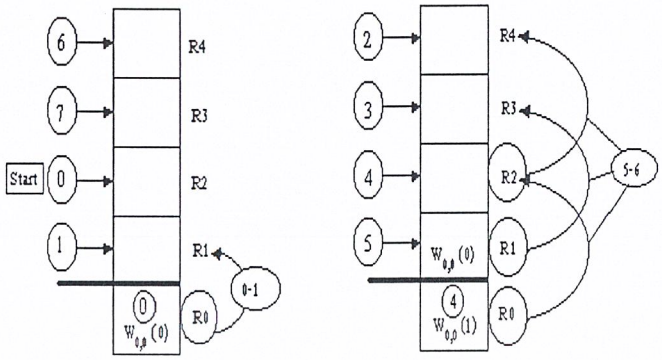
จากสมการที่ 3.4 สังเกตว่า ค่าของ $(n\text{-}2^{j-1}k)$ นั้นจะเป็นวิธีการกำหนดการทำ Down Sampling ไปในสมการ ซึ่งค่าที่ได้ออกมาในแต่ละระดับนั้นจะมีการทำ Down Sampling ไปในตัวด้วย ค่าที่ได้ จะมีลักษณะดังตารางที่ 3.2

จากตารางที่ 3.2 จะสังเกตเห็นว่าการทำกระบวนการแปลงเวฟเลตแต่ละระดับนั้นจะต้องอาศัยสัญญาณนาฬิกาทั้งหมด 4 ลูก จะได้ค่าของการแปลงในแต่ละระดับ โดยที่ค่าที่กระทำนั้นจะแบ่งออกเป็น h_0 และ h_1 ตามสมการที่ 3.2 ซึ่งในการกระทำแต่ละครั้งจะมีการคูณค่าระหว่างเวฟเลตแม่กลับสัญญาณทางอินพุตแล้วทำการบวกกันในแต่ละพจน์ของการทำงาน โดยเมื่อทำงานครบ 4 รอบจะได้ ค่าของการแปลงเวฟเลต 1 ระดับ ในขณะเดียวกันในระดับที่ 2 ของการแปลงเวฟเลต ก็จะทำการคำนวณ โดยผลที่ได้ของแต่ละระดับนั้นจะถูกนำไปทำการคำนวณต่อในระดับถัดไป โดยในระดับที่ 1 นั้นจะมีการจัดเรียงค่าที่ได้เป็นทางด้านความถี่สูงสลับกับค่าทางด้านความถี่ต่ำ โดยในการจัดเรียงนั้นค่าของตัวกรองที่เป็นตัวกำหนดการทำงานนั้นจะขึ้นอยู่กับค่าของ i โดยจะได้จากการมอดเอาค่า ซึ่งค่าที่ได้นั้นจะเกิดขึ้นได้เพียง 0 และ 1 เท่านั้น และค่าของ W นั้นจะถูกควบคุมโดยค่าของ i ซึ่งก็คือระดับของการแปลง และ j จะเป็นตัวกำหนดทิศทางของข้อมูลทางด้านเอาต์พุตของแต่ละระดับว่าจะให้ใช้ W ตัวใดไปใช้ในการคำนวณ เมื่อกระทำในระดับแรกเสร็จผลที่ได้จะถูกนำไปกระทำในระดับที่ 2 โดยจะมีลักษณะของการจัดเรียงข้อมูลดังในรูปที่ 3.20 โดยจะมีการนำข้อมูลที่ถูกสลับกันระหว่างข้อมูลทางด้านความถี่สูง กับข้อมูลทางด้านความถี่ต่ำ ไปทำการคำนวณกับค่าตัวกรองตัวเดิม โดยจะถูกควบคุมโดยค่าของ i เหมือนเดิม ซึ่งผลของระดับถัดมานั้นจะมีการจัดกลุ่มของข้อมูลที่ได้จากทางด้านความถี่สูงกับความถี่ต่ำเช่นกัน แต่จะถูกจัดในลักษณะแบบไขว้ โดยจะสามารถสังเกตได้จากทรีที่ใช้เปรียบเทียบระหว่างเวฟเลตแบบเดิม กับเวฟเลตแบบใหม่ ผลที่ได้คือจะทำให้สามารถใช้งานวงจรเดียวกันในการแปลงไป และแปลงกลับได้ภายในตัวเดียวกัน

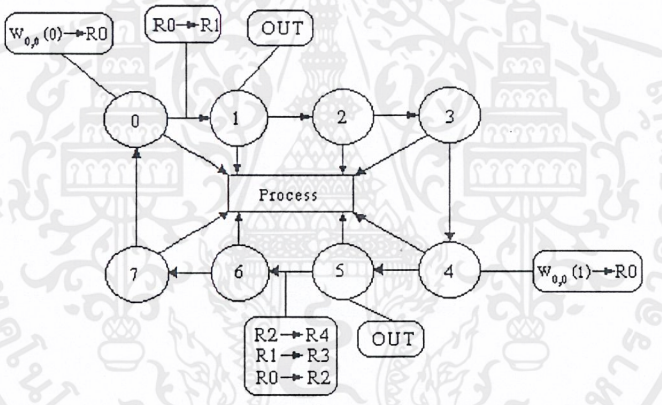
ตารางที่ 3.2 ลักษณะการทำงานแบบ Pipeline ของการแปลงเวฟเลตไป – กลับ 2 ระดับ

Cycle	Input L0	Process L0		Input L1	Process L1		Out L2
0	$W_{0,0}(0)$	$W_{0,0}(-1)$	$h_1 1$		$W_{1,1}(-4)$	$h_1 2$	
1		$W_{0,0}(0)$	$h_1 0$	$W_{1,1}(0)$	$W_{1,1}(-2)$	$h_1 1$	
2		$W_{0,0}(-3)$	$h_0 3$		$W_{1,1}(0)$	$h_1 0$	$W_{2,3}(0)$
3		$W_{0,0}(-2)$	$h_0 2$		$W_{1,0}(-6)$	$h_1 3$	
4	$W_{0,0}(1)$	$W_{0,0}(-1)$	$h_0 1$		$W_{1,0}(-4)$	$h_1 2$	
5		$W_{0,0}(0)$	$h_0 0$	$W_{1,0}(0)$	$W_{1,0}(-2)$	$h_1 1$	
6		$W_{0,0}(-1)$	$h_1 3$		$W_{1,0}(0)$	$h_1 0$	$W_{2,1}(0)$
7		$W_{0,0}(0)$	$h_1 2$		$W_{1,1}(-6)$	$h_0 3$	
8	$W_{0,0}(2)$	$W_{0,0}(1)$	$h_1 1$		$W_{1,1}(-4)$	$h_0 2$	
9		$W_{0,0}(2)$	$h_1 0$	$W_{1,1}(2)$	$W_{1,1}(-2)$	$h_0 1$	
10		$W_{0,0}(-1)$	$h_0 3$		$W_{1,1}(0)$	$h_0 0$	$W_{2,2}(0)$
11		$W_{0,0}(0)$	$h_0 2$		$W_{1,0}(-6)$	$h_0 3$	
12	$W_{0,0}(3)$	$W_{0,0}(1)$	$h_0 1$		$W_{1,0}(-4)$	$h_0 2$	
13		$W_{0,0}(2)$	$h_0 0$	$W_{1,0}(2)$	$W_{1,0}(-2)$	$h_0 1$	
14		$W_{0,0}(1)$	$h_1 3$		$W_{1,0}(0)$	$h_0 0$	$W_{2,0}(0)$
15		$W_{0,0}(2)$	$h_1 2$		$W_{1,1}(-6)$	$h_1 3$	
16	$W_{0,0}(4)$	$W_{0,0}(3)$	$h_1 1$		$W_{1,1}(0)$	$h_1 2$	
17		$W_{0,0}(4)$	$h_1 0$	$W_{1,1}(4)$	$W_{1,1}(2)$	$h_1 1$	
18		$W_{0,0}(1)$	$h_0 3$		$W_{1,1}(4)$	$h_1 0$	$W_{2,3}(4)$
19		$W_{0,0}(2)$	$h_0 2$		$W_{1,0}(-2)$	$h_1 3$	
20	$W_{0,0}(5)$	$W_{0,0}(3)$	$h_0 1$		$W_{1,0}(0)$	$h_1 2$	
21		$W_{0,0}(4)$	$h_0 0$	$W_{1,0}(4)$	$W_{1,0}(2)$	$h_1 1$	
22		$W_{0,0}(3)$	$h_1 3$		$W_{1,0}(4)$	$h_1 0$	$W_{2,1}(4)$
23		$W_{0,0}(4)$	$h_1 2$		$W_{1,1}(-2)$	$h_0 3$	
24	$W_{0,0}(6)$	$W_{0,0}(5)$	$h_1 1$		$W_{1,1}(0)$	$h_0 2$	
25		$W_{0,0}(6)$	$h_1 0$	$W_{1,1}(6)$	$W_{1,1}(2)$	$h_0 1$	
26		$W_{0,0}(3)$	$h_0 3$		$W_{1,1}(4)$	$h_0 0$	$W_{2,2}(4)$
27		$W_{0,0}(4)$	$h_0 2$		$W_{1,0}(-2)$	$h_0 3$	
28	$W_{0,0}(7)$	$W_{0,0}(5)$	$h_0 1$		$W_{1,0}(0)$	$h_0 2$	
29		$W_{0,0}(6)$	$h_0 0$	$W_{1,0}(6)$	$W_{1,0}(2)$	$h_0 1$	
30		$W_{0,0}(5)$	$h_1 3$		$W_{1,0}(4)$	$h_0 0$	$W_{2,0}(4)$
31		$W_{0,0}(6)$	$h_1 2$		$W_{1,1}(2)$	$h_1 3$	

ดังนั้นจากตารางที่ 3.2 สามารถที่จะนำมาทำการออกแบบวงจรให้มีการทำงานตามการคำนวณของตารางที่ 3.2 โดยจะต้องมีการกำหนดให้มี วงจรเลื่อนข้อมูล ทั้งหมด 5 ตัวในระดับที่ 1 ซึ่งจะต้องมีการเลื่อนข้อมูลตามเงื่อนไข โดยจะต้องมีวงจรตัวนับเป็นตัวควบคุมการทำงานว่าจะให้ทำการเลื่อนตำแหน่งของข้อมูล และต้องนำข้อมูลใดออกไปทำการคำนวณ โดยจะมีสัญญาณนาฬิกาตัวเดียวกันในการควบคุมการทำงานของวงจรแปลงเวฟเลตในทุกระดับของการแปลง สามารถที่จะอธิบายลักษณะการทำงานของวงจรเลื่อนข้อมูล ดังรูปที่ 3.21



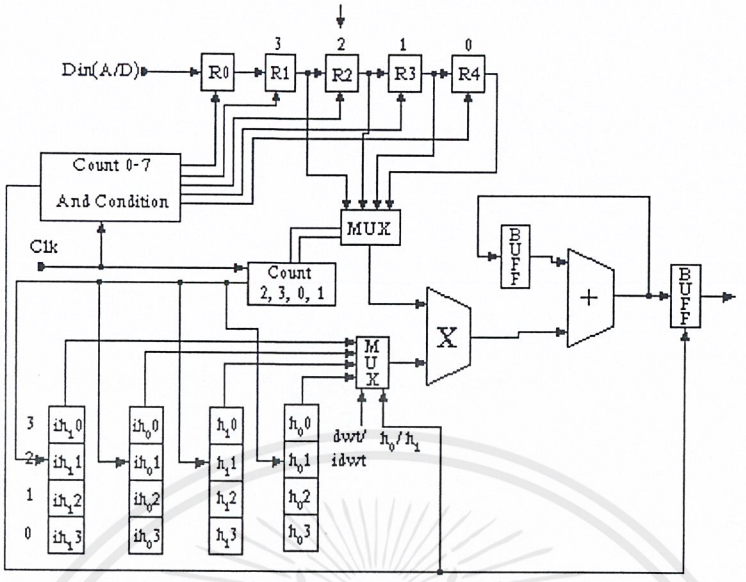
รูปที่ 3.21 การทำงานของวงจรเลื่อนข้อมูลของการแปลงเวฟเลตระดับที่ 1



รูปที่ 3.22 แผนผังลำดับการทำงานของวงจรเลื่อนข้อมูล

1) การแปลงเวฟเลตในระดับที่ 1

จากรูปที่ 3.21 จะเห็นว่าในการควบคุมการเลื่อนข้อมูลภายในหน่วยความจำนั้นจะต้องมีเงื่อนไขเข้ามาเกี่ยวข้อง คือ ในแต่ละลูกของสัญญาณนาฬิกาจะต้องมีการส่งค่าออกเพื่อนำไปทำการคำนวณ โดยค่าที่อ่านออกไปนั้นจะได้จากค่าที่รับเข้ามาผ่านวงจร A/D และจะมีการทำงานที่มีลักษณะเป็นแบบวงรอบ ซึ่งเมื่อทำการนับจนครบ 8 จะมีการทำงานที่กลับมาซ้ำแบบเดิมจึงทำให้การออกแบบจำเป็นต้องมีวงจรตัวนับมอดูลัส 0 ถึง 7 โดยจะวนกลับมานับที่เดิม



รูปที่ 3.23 วงจรแปลงเวฟเลตระดับที่ 1

ในช่วงเริ่มต้นของการทำงานของวงจรนั้นจะถือเป็นสัญญาณนาฬิกาลูกแรก คือ Stage 0 จะมีการอ่านค่าเข้ามาทำการเก็บใน R0 โดยค่าที่อ่านเข้ามาคือ $W_{0,0}(0)$ ขณะเดียวกันในช่วงของ Stage 0 ถึง Stage 1 จะต้องทำการเลื่อนข้อมูลจาก R0 ไปยัง R1 เพื่อที่จะทำการชี้เอาค่าใน R1 ซึ่ง R1 เก็บ ค่าของ $W_{0,0}(0)$ ไว้นำไปทำการคูณกับค่าของเวฟเลตแม่ โดยการชี้ตำแหน่ง R1 ของ Stage 1 ขณะเดียวกันใน Stage 1 จะต้องมีการสั่งให้มีการส่งค่าไปควบคุมให้บัฟเฟอร์ ส่งค่าที่ได้จากการแปลงเวฟเลตค่าแรกของระดับแรกออกไปยังระดับที่สอง แล้วทำการกลับไปชี้ข้อมูลใน R4 ซึ่งในช่วงเริ่มต้นจะมีค่าเป็น 0 ออกไปทำการคำนวณใน Stage 2 แล้วกระทำใน Stage 3 ซึ่งจะชี้ค่าที่ R3 ออกไปโดยในช่วงเริ่มต้น R3 จะมีค่าเป็น 0 เมื่อเข้าสู่ Stage 4 จะต้องมีการอ่านข้อมูลเข้ามาอีก 1 ค่า โดยค่าที่อ่านเข้ามาจะต้องนำไปเก็บไว้ใน R0 ในช่วงนี้จะยังไม่มีการเลื่อนข้อมูลจนกระทั้งเสร็จสิ้น Stage 5 จะต้องมีการเลื่อนค่าใน R2 ซึ่งในช่วงเริ่มต้นจะมีค่าเป็น 0 ไปเก็บใน R4 เลื่อนค่าใน R1 ซึ่งเก็บค่าของ $W_{0,0}(0)$ ไปไว้ใน R3 และเลื่อนค่าที่เก็บใน R0 ซึ่งเก็บค่าของ $W_{0,0}(1)$ ที่เพิ่งทำการอ่านเข้ามาไปเก็บใน R0 โดยขั้นตอนดังกล่าวจะต้องกระทำในช่วง Stage 5 ถึง Stage 6 โดยจะต้องทำให้เสร็จก่อน Stage 6 เมื่อเข้าสู่ Stage 6 จะต้องมีการ ชี้ค่าที่ตำแหน่ง R0 ออกไปทำการคูณกับค่าของเวฟเลตแม่ที่เปลี่ยนค่าตามการชี้ของวงจรตัวนับโดยใน Stage 5 นี้จะต้องมีการส่งค่าไปบอกให้บัฟเฟอร์ ส่งค่าที่ได้จากการแปลงเวฟเลตระดับแรก ค่าที่ 2 ออกไปให้ส่วนของการแปลงเวฟเลตระดับที่ 2 จาก Stage 6 และ 7 จะเป็นการส่งค่าออกไปทำการคูณกับค่าของเวฟเลตแม่ จะเห็นได้ว่า

การทำงานครบ 8 Stage (0-7) เมื่อเข้าสู่ Stage ถัดไปนั้นจะมีลักษณะการทำงานแบบเดียวกับ Stage 0 ซึ่งสามารถที่จะสรุปการทำงานได้ดังแผนผังการทำงานของวงจรเลื่อนข้อมูล

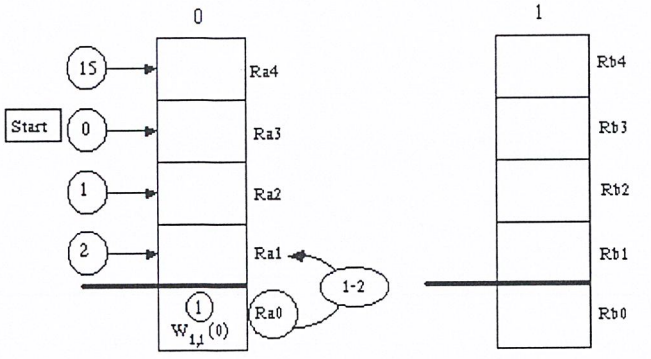
จากลักษณะการทำงานของวงจรเลื่อนข้อมูลดังรูปที่ 3.21 สามารถที่จะออกแบบวงจรให้มีลักษณะการทำงานดังรูปที่ 3.23 ได้โดยในการควบคุมการทำงานนั้นจะต้องอาศัยการทำงานของวงจรตัวนับมอดูลัส 0 ถึง 7 ซึ่งในแต่ละครั้งของการนับนั้นจะมีการนำเงื่อนไขไปควบคุมการเลื่อนข้อมูล พร้อมชี้ข้อมูลออกมาทำการคูณกับค่าของเวฟเลตแม่ พร้อมทั้งควบคุมส่วนอื่นๆ ส่วนของบัฟเฟอร์ในส่วนต่างๆ ด้วยเพื่อที่จะให้มีการทำงานเป็นไปตามที่ลำดับที่กำหนดไว้ซึ่งจะมีการทำงานดังนี้

จากวงจรแปลงเวฟเลตระดับที่ 1 ในรูป 3.23 จะมี วงจรเลื่อนข้อมูล เพื่อใช้เป็นส่วนเก็บและ เลื่อนข้อมูลเพื่อให้ได้ข้อมูลถูกต้องเพื่อนำไปคำนวณตามที่ต้องการ โดยใช้มัลติเพล็กซ์เลือกข้อมูล เพื่อส่งออกไปทำการคูณสำหรับการคูณนั้นจะกระทำกับค่าของเวฟเลตแม่ซึ่งจะมีทั้งทางด้านต่ำ และด้านสูง การเลือกค่าของเวฟเลตแม่เพื่อนำไปทำการคูณนั้นจะใช้วงจรตัวนับเดียวกันกับส่วนที่ใช้ในการเลือกข้อมูลจาก วงจรเลื่อนข้อมูล โดยวงจรเลื่อนข้อมูลนั้นจะทำการเลื่อนข้อมูลโดยมีเงื่อนไขจากวงจรตัวนับมอดูลัส 0 ถึง 7 สำหรับให้การทำงานเป็นแบบวงรอบลักษณะเดียวกับแผนผังการทำงานของวงจรในรูปที่ 3.22

2) การแปลงเวฟเลตระดับที่ 2

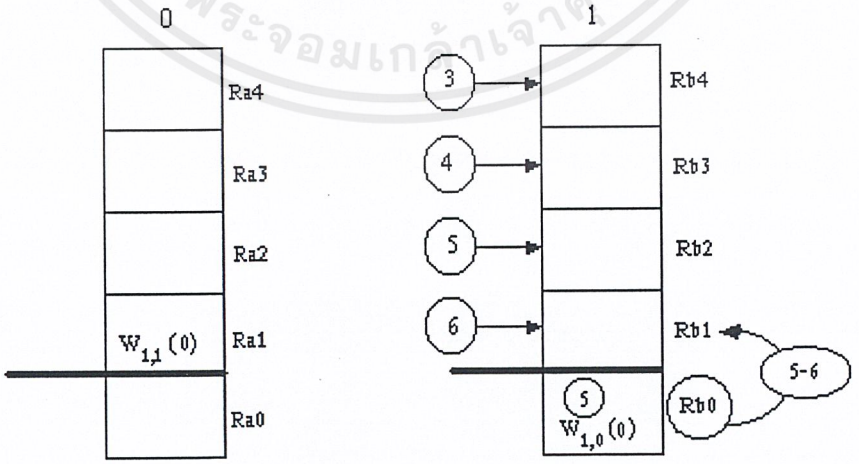
จากกระบวนการแปลงเวฟเลตในระดับที่ 1 นั้น กระบวนการทุกอย่างจะต้องกระทำพร้อมกันกับระดับที่ 2 ด้วย ซึ่งโดยจะใช้ฐานเวลาตัวเดียวกันในการควบคุมการอ่านค่าในวงจรเลื่อนข้อมูลออกไปทำการคูณกับค่าของเวฟเลตแม่ ซึ่งในระดับที่สองของการแปลงเวฟเลตนั้นก็จะมีลักษณะการทำงานคล้ายกันกับการแปลงเวฟเลตในระดับที่ 1 เพียงแต่จะต้องมีการเพิ่มจำนวนของวงจรเลื่อนข้อมูลขึ้นอีก และมีการควบคุมการอ่านค่าที่แตกต่างกันไปจากเดิม และมีการออกแบบวงจรตัวนับแตกต่างกันโดยจะมีการนับเพิ่มขึ้นอีก ซึ่งในระดับที่สองนี้จะมีการจัดเรียงค่า และลำดับของการทำงานดังในตารางที่ 3.2 โดยจะมีการทำงานดังรูปที่ 3.24

จากรูปที่ 3.24 จะเห็นว่ามี วงจรเลื่อนข้อมูล เพิ่มขึ้นมาอีกชุด โดย 2 ชุดนี้จะแยกออกจากกันซึ่งจะมี มัลติเพล็กซ์ เป็นตัวเลือกข้อมูลชุดใดในรีจิสเตอร์ส่งออกไป โดยจะกระทำตามเงื่อนไขดังต่อไปนี้ คือการทำงานของวงจรเลื่อนข้อมูลนี้จะมีลักษณะดังรูปที่ 3.24 ซึ่งการอ่านค่าเข้ามาเก็บในวงจรเลื่อน ข้อมูลจะมีการอ่านค่าที่เก็บออกไปทำการคำนวณเป็นจำนวน 2 รอบด้วยกัน ซึ่งก็เพื่อหาค่าผลลัพธ์ทางด้านสูง และด้านต่ำด้วย

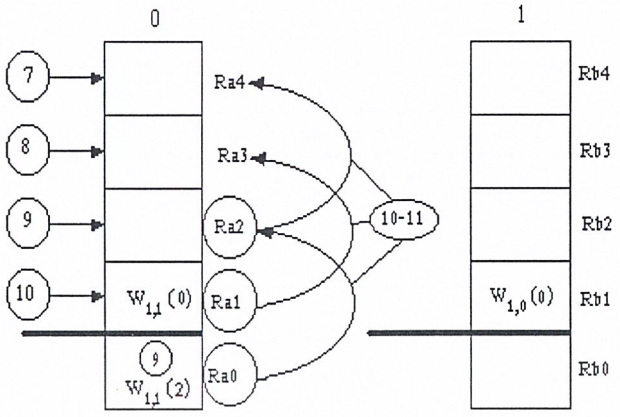


รูปที่ 3.24 การทำงานของวงจรเลื่อนข้อมูลในการแปลงเวฟเลตระดับที่ 2 Stage 0 ถึง Stage 2

ในช่วงเริ่มต้นของการทำงานในส่วนของการแปลงเวฟเลตระดับที่ 2 นี้จะมีการส่งค่าที่เก็บใน วงจรเลื่อนข้อมูล ออกไปทำการคูณกับค่าของเวฟเลตแม่ โดยในสัญญาณนาฬิกาลูกที่ 0 ส่วนของการ แปลงเวฟเลตระดับที่ 2 จะเริ่มนับเป็น Stage 0 จะทำการชี้เอาค่าที่เก็บใน Ra3 โดยจะมีค่าเป็น 0 เพราะยังไม่มีการอ่านค่าเข้ามาจากระดับที่ 1 เมื่อถึง Stage 1 จะต้องมีการอ่านค่าเข้ามาจากระดับที่หนึ่ง ซึ่งค่าที่อ่านเข้ามานี้คือค่าที่ได้จากการแปลงเวฟเลตในระดับที่ 1 ซึ่งก็คือ $W_{1,1}(0)$ โดยจะนำไปเก็บใน R0แต่ค่าที่ถูกชี้โดย Stage 1 จะยังคงชี้ค่าที่เป็น 0 ออกไป ขณะเดียวกันในช่วง Stage 1 ก่อนถึง Stage 2 จะต้องทำการเลื่อนข้อมูลจาก Ra0 ไปเก็บไว้ใน Ra1 เมื่อถึง Stage 2 จะทำการชี้เอาค่า $W_{1,1}(0)$ ออกไปทำการคูณกับค่าของเวฟเลตแม่ทางด้าน h_2 พร้อมทั้งทำการส่งค่าที่ที่ได้จากการคำนวนแต่ละรอบไปยังระดับที่ 3 ด้วย ดังรูปที่ 3.24

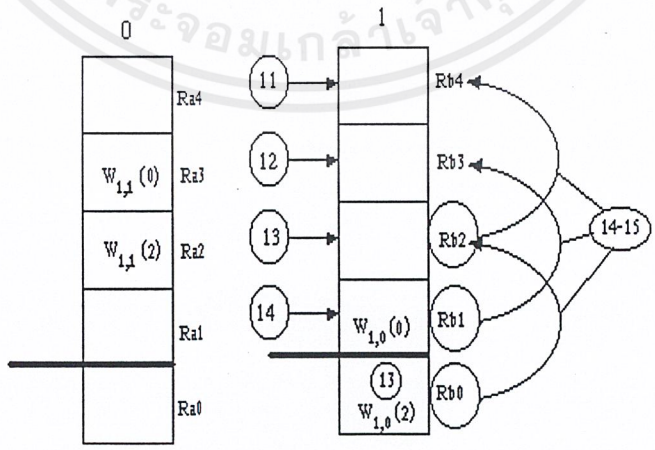


รูปที่ 3.25 การทำงานของวงจรเลื่อนข้อมูลในการแปลงเวฟเลตระดับที่ 2 Stage 3 ถึง Stage 6



รูปที่ 3.26 การทำงานของวงจรเลื่อนข้อมูลในการแปลงเวฟเลตระดับที่ 2 Stage 7 ถึง Stage 10

เมื่อเข้าสู่ Stage 3 จะมีการเปลี่ยนมาดึงเอาค่าที่เก็บไว้ใน Rb ออกไปโดย Stage 3 จนถึง Stage 4 จะมีเพียงการอ่านค่าจาก Rb4 และ Rb3 ออกไปทำการคูณกับค่าของเวฟเลตแม่ จนกระทั่งถึง Stage 5 จะมีการรับค่า $W_{1,0}(0)$ เข้ามาเก็บที่ Rb0 พร้อมทั้งส่งค่าที่เก็บใน Rb2 ออกไปคูณกับเวฟเลตแม่ด้วย และในช่วง Stage 5 ก่อนถึง Stage 6 จะต้องทำการเลื่อนข้อมูลที่เก็บใน Rb0 ซึ่ง ก็คือ $W_{1,0}(0)$ เข้าไปเก็บไว้ใน Rb1 แล้วใน Stage 6 จะทำการชี้ข้อมูล $W_{1,0}(0)$ ที่เก็บใน Rb1 ออกไปคูณกับค่าของเว75ฟเลตแม่ ดังรูปที่ 3.25 สำหรับใน Stage 7 และ Stage 8 จะกลับมาทำการอ่านค่าที่เก็บใน Ra4 และ Ra3 ตามลำดับ เพื่อส่งไปคูณกับเวฟเลตแม่ทางด้านต่ำ จนกระทั่งถึง Stage 9 จะมีการอ่านค่าจากระดับที่ 1 เข้ามาอีกโดยจะทำการอ่านข้อมูลเข้ามาอีกคือ $W_{1,1}(2)$ เก็บไว้ใน Ra0 แล้ว Stage 10 ก็ทำการอ่านค่าที่เก็บใน Ra1

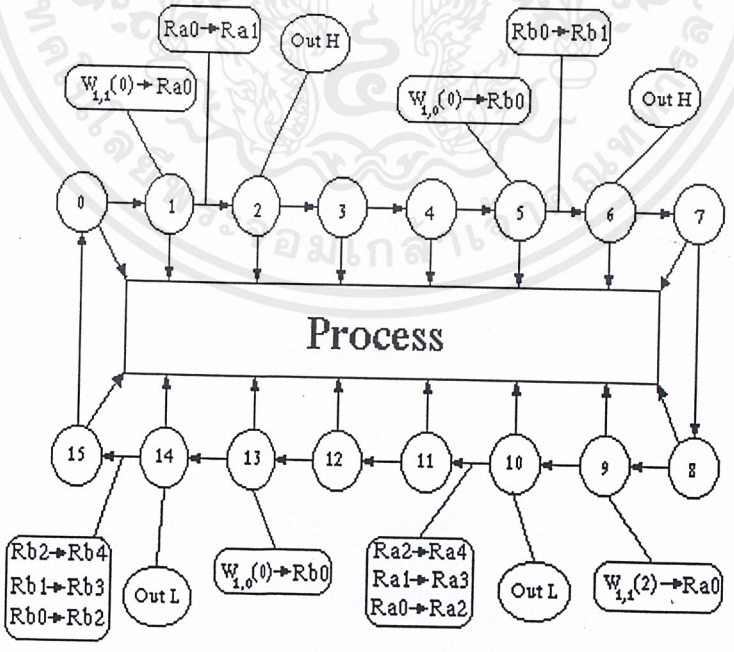


รูปที่ 3.27 การทำงานของวงจรเลื่อนข้อมูลในการแปลงเวฟเลตระดับที่ 2 Stage 11 ถึง Stage 14

ซึ่งเวลานี้เก็บค่าของ $W_{1,1}(0)$ ออกไปทำการคูณกับเวฟเลตแม่ทางด้านต่ำ และในช่วงระหว่าง Stage 10 ก่อนถึง Stage 11 จะต้องเลื่อนข้อมูลโดย Ra2 จะถูกเลื่อนไปเก็บไว้ใน Ra4 Ra1 ซึ่งเก็บค่า $W_{1,1}(0)$ เลื่อนข้อมูลไปเก็บไว้ใน Ra3 และ Ra0 ซึ่งเก็บค่า $W_{1,1}(2)$ เลื่อนไปเก็บไว้ใน Ra2 ดังรูปที่ 3.26

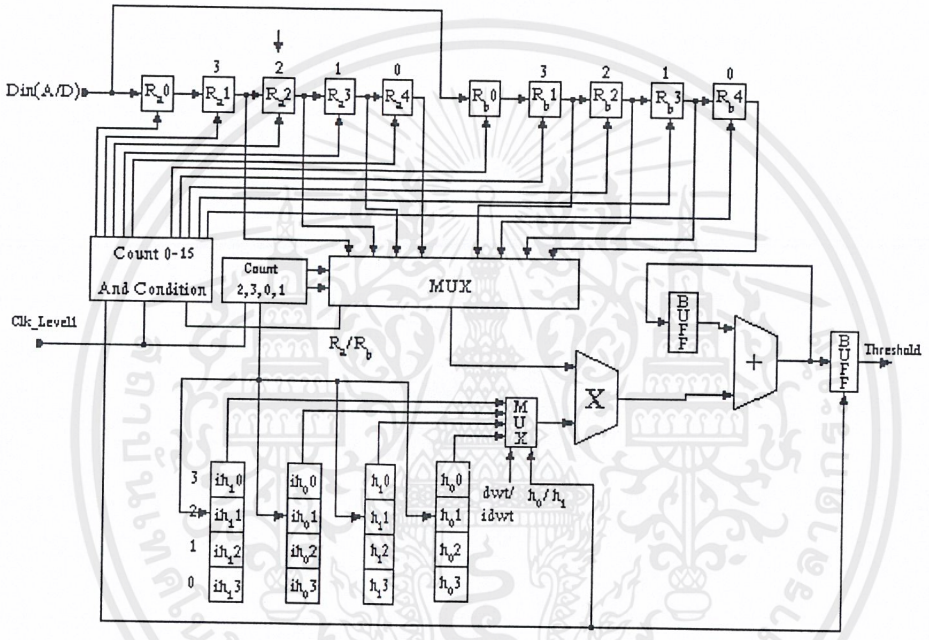
สำหรับ Stage 11 จนกระทั่งถึง Stage 12 จะทำการอ่านค่าจาก Rb4 และ Rb3 ออกไปคูณ กับเวฟเลตแม่ทางด้านต่ำตามลำดับ เมื่อเข้าสู่ Stage 13 จะต้องมีการอ่านค่าจากระดับที่ 1 อีกครั้ง โดยจะทำการอ่านค่า $W_{1,0}(2)$ เข้ามาเก็บไว้ใน Rb0 เมื่อถึง Stage 14 ก็อ่านค่าที่เก็บใน Rb1 นำไปกระทำการคูณกับค่าของเวฟเลตแม่ทางด้านต่ำปกติ พร้อมทั้งควบคุมให้ Buffer ส่งค่าออก (Out) ไปยังระดับที่ 3 ถัดจาก Stage 14 ก่อนถึง Stage 15 จะต้องมีการเลื่อนข้อมูล Rb2 ไปเก็บใน Rb4 Rb1 ซึ่งจัดเก็บ $W_{1,0}(0)$ ไปไว้ใน Rb3 และ Rb0 ซึ่งเก็บ $W_{1,0}(2)$ ไปเก็บใน Rb2 ดังรูปที่ 3.27 เมื่อเข้าสู่ Stage 15 จะมีลักษณะการทำงานวนกลับไปStage 0 อีกครั้ง ดังรูปที่ 3.27

ในการทำงานดังกล่าวระหว่างระดับที่ 1 และระดับที่ 2 จะต้องมีการทำงานบนฐานเวลาเดียวกัน ซึ่งจะต้องใช้สัญญาณนาฬิกาตัวเดียวกันกับสัญญาณนาฬิกาหลักที่ใช้ควบคุมการทำงานของระดับที่ 1 ในการแปลงเวฟเลตในระดับที่ 2 นี้จะมีลักษณะการทำงานเป็นลักษณะวนรอบดังแสดงใน รูปที่ 3.28



รูปที่ 3.28 แผนผังวงจรเลื่อนข้อมูลของการแปลงเวฟเลตระดับที่ 2

จากการทำงานในรูปที่ 3.28 นั้นสามารถที่จะออกแบบเป็นวงจรที่ใช้สำหรับแปลงเวฟเลตระดับที่ 2 ได้ ดังรูปที่ 3.29 ซึ่งการทำงานทั้งหมดจะอยู่ภายในฐานเวลาเดียวกับการแปลงเวฟเลตในระดับที่ 1 โดยในส่วนของการเลื่อนข้อมูลนั้นจะมีช่วงเวลาเพิ่มมากขึ้น เนื่องจากต้องจัดการกับวงจรเลื่อนข้อมูล ที่เพิ่มขึ้นอีก 1 ชุด ซึ่งจะมีการทำงานสลับกัน โดยส่วนของข้อมูลที่เข้ามาจะมีลักษณะสลับกันเก็บลงใน วงจรเลื่อนข้อมูล Ra และ Rb ซึ่งจะมีส่วนควบคุมเก๋าหนดว่าจะให้วงจรเลื่อนข้อมูลตัวในซึ่งเป็นตัวพักข้อมูล



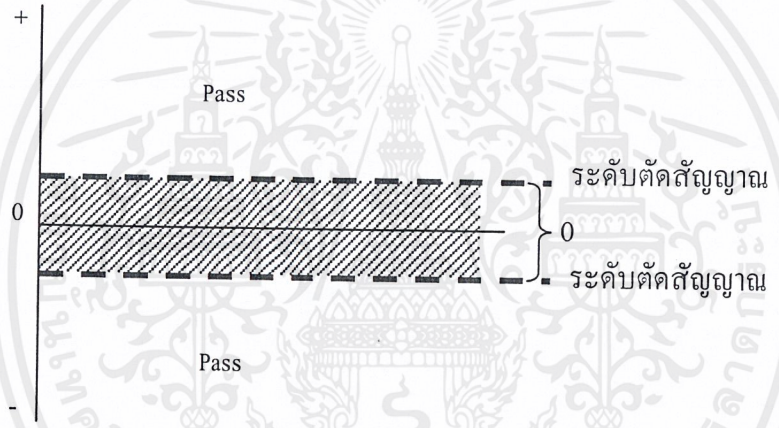
รูปที่ 3.29 วงจรแปลงเวฟเลตระดับที่ 2

ส่วนของการควบคุมการเลื่อนข้อมูลนั้นก็จะขึ้นอยู่กับว่าวงจรนับซึ่งถ้านับถึงเงื่อนไขตามที่กำหนดก็จะส่งให้ รีจิสเตอร์ ตัวใดที่จะต้องเลื่อนข้อมูล และตัวใดจะต้องหยุดข้อมูลไว้ ซึ่งในการอ่านข้อมูลนั้น จะต้องใช้มัลติเพล็กซ์เป็นตัวเลือกข้อมูลออกมาทำการคูณกับค่าของเวฟเลตแม่ โดยในแต่ละรอบของการเลือกค่าออกมาคูณนั้นจะต้องมีการนำผลลัพธ์ที่ได้จากการคูณนั้นไปทำการบวก แล้วนำไปเก็บไว้ใน Buffer เพื่อรอทำการบวกในกับค่าที่ชี้ออกมาคูณกับเวฟเลตแม่ในรอบถัดๆ ไป จนครบ 4 ค่าก็จะทำให้ได้ 1 ค่าของการแปลงเวฟเลต เมื่อทำการคูณ และบวกครบ 4 รอบจะได้ค่าที่ผ่านการแปลงเวฟเลต 1 ค่า ก็จะมีการส่งค่ามาบอกให้ Buffer ส่งค่าที่ได้จากการคำนวณไปยังส่วนถัดไป คือ ส่วนของวงจรตัดระดับสัญญาณ (Threshold) ต่อไป

3.4 การตัดระดับสัญญาณ (Threshold)

หลังจากการแปลงเวฟเลตข้อมูลที่ได้นั้นจะมีช่วงที่อยู่ทั้งความถี่สูง และความถี่ต่ำ ซึ่งสัญญาณทางด้านความถี่ต่ำนั้นจะคงรูปของสัญญาณรูปเดิมมากที่สุด ส่วนสัญญาณทางด้านความถี่สูงนั้นโดยส่วนมากจะเป็นสัญญาณรบกวนที่ปนมากับสัญญาณความถี่ต่ำ ซึ่งในการตัดระดับของสัญญาณนั้นมีเงื่อนไขดังต่อไปนี้

$$\text{Threshold (out)} = \begin{cases} \text{Pass} & ; \text{ระดับการตัดสัญญาณ} < |\text{ข้อมูลที่เข้ามา}| \\ 0 & ; \text{ระดับการตัดสัญญาณ} > |\text{ข้อมูลที่เข้ามา}| \end{cases}$$



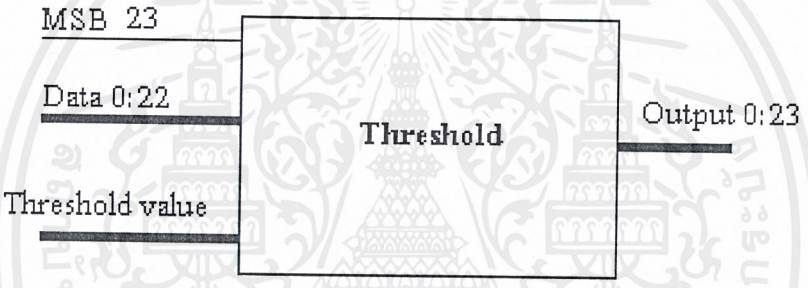
รูปที่ 3.30 การตัดระดับสัญญาณ

จากรูปที่ 3.30 ข้อมูลจะอยู่ในช่วงของส่วนที่แรงเงาจะถูกตัดให้มีค่าเป็นศูนย์ แล้วส่งไปยังส่วนของการจัดระดับสัญญาณ (Quantization) ซึ่งข้อมูลที่ถูกตัดให้เป็นศูนย์นั้นจะเป็นส่วนย่อยของสัญญาณความถี่สูงที่ส่งมาพร้อมกับข้อมูลเสียงที่ไม่จำเป็นในการนำไปจัดเก็บ ข้อมูลเหล่านี้จะสามารถที่ตัดให้เป็นศูนย์ได้ และต่อเมื่อทำการแปลงกลับเราสามารถที่จะสร้างกลับขึ้นมาได้โดยขบวนการของการทำเพิ่มอัตราการสุ่ม (Up Sampling) โดยในส่วนของการเพิ่มอัตราการสุ่มอยู่ในส่วนของการแปลงกลับ เวฟเลต (Inverse Wavelet Transform) ดังนั้นเมื่อผ่านขบวนการตัดระดับสัญญาณ ผลลัพธ์ที่ได้จะมีค่าเป็นศูนย์ซึ่งนำเอาผลลัพธ์ส่งต่อไหวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนเพื่อลดขนาดบิตของข้อมูลต่อไป

ในการกำหนดระดับการตัดสัญญาณนั้นสามารถที่จะกำหนดโดยการประมาณค่า ในที่นี้จะทำให้การจัดระดับแบบคงที่ (Fix Point Threshold) ซึ่งในที่นี้จะตัดข้อมูลที่ ร้อยละ 5 จากระดับของ

ข้อมูลที่สามารถเกิดขึ้นได้ โดยข้อมูลที่เกิดขึ้นได้นั้นจะอยู่ที่ 24 บิต หรือประมาณ 16,777,215 ค่า ซึ่งค่า ร้อยละ 5 ของข้อมูลที่ได้ คือ 838860.75 ≈ 838860 คือ เมื่อสัญญาณที่ผ่านการแปลงเวฟเลตเข้ามามีระดับสัญญาณอยู่ในช่วง 838860 ถึง -838860 ก็จะถูกตัดให้มีค่าเป็นศูนย์

ค่าที่ตัดทิ้งไปจะมีผลน้อยมากกับสัญญาณเมื่อทำการแปลงกลับข้อมูล โดยสัมประสิทธิ์ทางด้านต่ำจะเป็นตัวบ่งบอกลักษณะของสัญญาณ ส่วนที่ถูกตัดทิ้งไปจะมีคุณสมบัติคล้ายกับสัญญาณรบกวนของสัญญาณที่ติดมาด้วย เพียงแต่จะถูกตัดทิ้งไปได้ในขณะที่มีการจัดสัญญาณ และในการแปลงข้อมูลกลับมาจึงแทบจะไม่มีผลกระทบที่จะทำให้เกิดการเปลี่ยนแปลงรูปสัญญาณไปมาก จากขั้นตอนนี้จะทำให้เข้ารูปแบบที่จะนำไปเข้ารหัส คือ ความซ้ำซ้อนของข้อมูลที่เกิดขึ้นเป็น 0 จำนวนมากทำให้การตัดระดับของข้อมูลที่ผ่านการแปลงเวฟเลต



รูปที่ 3.31 วงจรตัดระดับสัญญาณ เมื่อแปลงเป็นแมโคร

โดยในการออกแบบวงจรนั้นจะใช้วงจรเปรียบเทียบ ดังรูป 3.31 เงื่อนไขที่ได้จากการเปรียบเทียบ คือ ถ้าข้อมูลที่เข้ามามีค่ามากกว่าระดับของจุดตัดสัญญาณจะให้ค่าที่ได้จากการเปรียบเทียบมีค่าระดับลอจิกเป็น 1 แต่ถ้าข้อมูลที่เข้ามามีค่าน้อยกว่าผลที่ได้จากการเปรียบเทียบมีระดับลอจิกเป็น 0 ซึ่งจะนำไป AND กับข้อมูลออกไปเป็นศูนย์

3.5 การจัดระดับข้อมูล (Quantization)

การจัดระดับข้อมูลเป็นขั้นตอนในการกำหนดข้อมูลที่ได้ออกมาจากการตัดระดับข้อมูล (Threshold) โดยข้อมูลที่ได้จากการแปลงเวฟเลตนั้นจะได้ข้อมูลขนาด 32 บิต ซึ่งเป็นข้อมูลที่มีขนาดใหญ่ ซึ่งวงจรเข้ารหัสที่ออกแบบจะเป็นวงจรที่ทำงานกับข้อมูลขนาด 8 บิตเท่านั้น จึงจำเป็นที่จะต้องทำการจัดระดับข้อมูลให้เหลือขนาดเพียง 8 บิต เพื่อที่จะทำการเข้ารหัส โดยข้อมูลที่ออกมาจากการแปลงเวฟเลตนั้น ถ้านำค่าในแต่ละทิศทางมารวมกันทั้งหมดแล้วคูณด้วยค่าสูงสุดของข้อมูล

ที่เข้ามา (Input) จาก A/D จะมีขนาดเพียง 8 บิต จึงทำให้เอาต์พุตของข้อมูลอยู่ที่ 23 บิต เท่านั้นส่วนบิตที่มีนัยสำคัญสูงสุดที่ผ่านการแปลงเวฟเลตจะถูกนำมาเป็นบิตเครื่องหมาย ซึ่งจำเป็นต้องใช้ในกระบวนการจัดระดับสัญญาณด้วยในการออกแบบวงจรจัดระดับข้อมูล จะต้องคำนึงถึงค่าสูงสุดที่เกิดขึ้นได้ ซึ่งในที่นี้จะมี ค่าสูงสุดได้ 24 บิต โดยบิตสุดท้าย 1 บิต จะเป็นบิตเครื่องหมายที่นำมาจากบิตที่มีนัยสำคัญมากที่สุดของข้อมูล 32 บิต ซึ่งจะให้ค่าได้ 2^{24} หรือประมาณ 16,777,215 ค่า โดยจะทำการจัดระดับให้เหลือเพียง 256 ระดับ จึงจะต้องนำค่าจริงที่สามารถเกิดขึ้นได้ของ 24 บิต มาทำการพิจารณาเพื่อให้ง่าย จึงยกตัวอย่างในการออกแบบ โดยให้ทำการจัดระดับของข้อมูล 16 บิตให้เหลือ 4 บิต

จากข้อมูลขนาด 16 บิต สามารถที่จะมีค่าทั้งหมดได้ 65,535 ข้อมูล โดยข้อมูลจะมีค่าทั้งด้านลบ และด้านบวก จึงทำให้ทางด้านบวกมีค่าจาก 0 ถึง 32,767 ค่า และด้านลบจาก 0 ถึง –32,768 ค่า ในการจัดระดับนั้นจำเป็นที่จะต้องหาระยะห่างของข้อมูลดังนี้

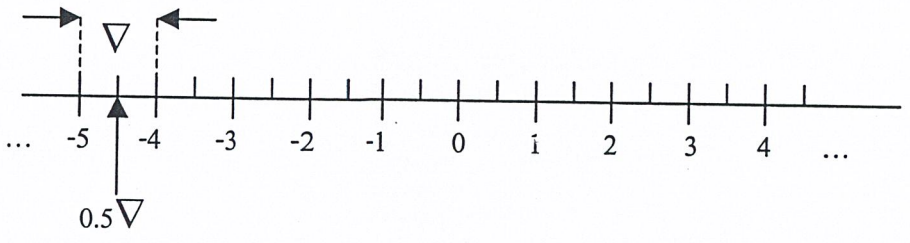
$$\nabla = 65,535 / 15 \quad \text{ข้อมูล / 1ระดับ}$$
$$= 4,369 \quad \text{ข้อมูล / 1 ระดับ}$$

โดยจุดตัดระดับของแต่ละระดับนั้นจะมีค่าเท่ากับ

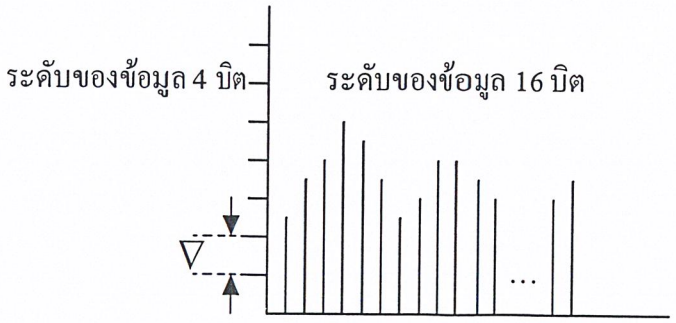
$$0.5\nabla = 4,369 / 2$$
$$= 2,184.5 \approx 2185$$

โดย

∇ = ระยะห่างของข้อมูลในแต่ละระดับทางด้าน 16 บิต



รูปที่ 3.32 การหาจุดตัดระดับของข้อมูล



รูปที่ 3.33 ลักษณะการจัดระดับของข้อมูลจาก 16 บิต ไปเป็น 4 บิต

จากรูปที่ 3.33 จะเห็นว่ามีการแบ่งข้อมูลระหว่าง 4 บิต และ 16 บิต ออกจากกัน โดยค่าทางด้าน 4 บิตนั้นจะมีค่าเรียงลำดับจาก 0000 ไปจนถึง 1111 ซึ่งจะมีค่าได้ทั้งหมด 16 ค่า โดยเริ่มต้นจากค่า 0 ส่วนทางด้านข้อมูลขนาด 16 บิตนั้นจะได้จากการนำเอาค่าของ ∇ คูณกับค่าของระดับทางด้าน 4 บิตสำหรับค่าของจุดตัดระดับจะหาได้จาก

$$\text{จุดตัดระดับ (n)} = \nabla \text{ X ระดับทางด้าน 4 บิต} + 2185 \quad (3.3)$$

$$\text{n} = \text{ระดับทางด้าน 4 บิต (0 – 15)} \quad (3.4)$$

สามารถแสดงได้ดังตาราง 3.3

จากตารางที่ 3.3 จุดที่ a จะเป็นจุดที่ใช้ในการเปรียบเทียบทางด้านบวกจุดแรก ถ้าข้อมูลเข้ามามีค่ามากกว่าก็จะทำการเพิ่มค่าจากตำแหน่ง a ขึ้นไปอีก โดยใช้ตำแหน่ง a บวกกับ 2∇ ซึ่งก็จะทำให้ได้ค่าตรงกับค่าในตำแหน่ง b ที่จะใช้ในการเปรียบเทียบครั้งต่อไป โดยถ้าข้อมูลที่เข้ามามีค่ามากกว่าค่าในตำแหน่ง b ก็จะเพิ่มค่าในตำแหน่ง b ขึ้นไปอีก ∇ ซึ่งก็จะทำให้ได้ค่าตรงกับค่าในตำแหน่ง d แล้วใช้ค่าในตำแหน่ง d ในการเปรียบเทียบต่อถ้ามีค่ามากกว่าก็จะสามารถชี้ตำแหน่งของการตัดระดับ โดยในการกระทำแต่ละครั้งจะต้องทำการจัดเก็บค่าทางด้าน 4 บิตไปด้วยเพื่อเป็นระดับทางด้าน 4 บิต

ในกรณีเดียวกันถ้าข้อมูลเข้ามามีค่าน้อยกว่าค่าในตำแหน่ง a ก็จะทำการลบค่าในตำแหน่ง a ลงสองเท่าของ ∇ ซึ่งก็จะทำให้ได้ค่าตรงกับค่าในตำแหน่ง c แล้วในการเปรียบเทียบครั้งต่อไปก็จะใช้ค่าของตำแหน่ง c เป็นตัวเปรียบเทียบต่อไป หรือกรณีที่ข้อมูลที่เข้ามามีค่าน้อยกว่าค่าในตำแหน่ง c ก็จะทำการลบค่าในตำแหน่ง c ลงไป ∇ ก็จะได้ตำแหน่งในการเปรียบเทียบทีตำแหน่ง g โดยเปรียบเทียบกับค่าทางด้าน 16 บิต ถ้าข้อมูลมีค่าน้อยกว่าก็จะสามารถชี้ตำแหน่งทางดาน 4 บิตได้ ซึ่ง

ในการทำแต่ละครั้งจะมีการจัดการกับข้อมูลทางด้าน 4 บิตตามไปด้วย เช่นเดียวกันถ้าข้อมูลเข้ามาเป็นทางด้านลบก็จะมีลักษณะการทำงานเหมือนกัน

ตารางที่ 3.3 การจัดระดับของข้อมูล 16 บิต เป็น 4 บิต

ข้อมูล 4 บิต					ข้อมูล 16 บิต													
7	0	1	1	1	30583	0	1	1	1	0	1	1	1	0	1	1	1	0	1
d					28399	0	1	1	0	1	1	1	0	1	1	1	0	1	1
6	0	1	1	0	26214	0	1	1	0	0	1	1	0	0	1	1	0	0	1
b					24030	0	1	0	1	1	1	0	1	1	1	0	1	1	1
5	0	1	0	1	21845	0	1	0	1	0	1	0	1	0	1	0	1	0	1
e					19661	0	1	0	0	1	1	0	0	1	1	0	0	1	1
4	0	1	0	0	17476	0	1	0	0	0	1	0	0	0	1	0	0	0	1
a					15292	0	0	1	1	1	0	1	1	1	0	1	1	1	1
3	0	0	1	1	13107	0	0	1	1	0	0	1	1	0	0	1	1	0	0
f					10923	0	0	1	0	1	0	1	0	1	0	1	0	1	0
2	0	0	1	0	8738	0	0	1	0	0	0	1	0	0	0	1	0	0	0
c					6554	0	0	0	1	1	0	0	1	1	0	0	1	1	0
1	0	0	0	1	4369	0	0	0	1	0	0	0	1	0	0	0	1	0	0
g					2185	0	0	0	0	1	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	1	1	1	1	-4369	1	1	1	0	1	1	1	0	1	1	1	0	1	1
D					-6554	1	1	1	0	0	1	1	0	0	1	1	0	0	1
-2	1	1	1	0	-8738	1	1	0	1	1	1	0	1	1	1	0	1	1	1
B					-10923	1	1	0	1	0	1	0	1	0	1	0	1	0	1
-3	1	1	0	1	-13107	1	1	0	0	1	1	0	0	1	1	0	0	1	1
E					-15292	1	1	0	0	0	1	0	0	0	1	0	0	0	1
-4	1	1	0	0	-17476	1	0	1	1	1	0	1	1	1	0	1	1	1	1
A					-19661	1	0	1	1	0	0	1	1	0	0	1	1	0	0
-5	1	0	1	1	-21845	1	0	1	0	1	0	1	0	1	0	1	0	1	0
F					-24030	1	0	1	0	0	0	1	0	0	0	1	0	0	0
-6	1	0	1	0	-26214	1	0	0	1	1	0	0	1	1	0	0	1	1	0

จุดตัดระดับข้อมูล

ในการทำงานแต่ละครั้งจะมีการจัดเก็บค่าทางด้าน 4 บิต โดยค่าของข้อมูลที่เข้ามามีค่ามากกว่าค่าในตำแหน่งที่ใช้เปรียบเทียบ ก็จะทำการจัดเก็บโดยในการเปรียบครั้งแรกกับตำแหน่ง a ทางด้านบวก หรือ A ทางด้านลบจะถูกเซตค่าไว้ 3 บิตหลังเป็น 000 ก่อนแล้วทำการเปรียบเทียบ ถ้าข้อมูลที่เข้ามามีค่ามากกว่า จะนำค่าที่เซตไว้ไป OR กับ 100 แต่ถ้าข้อมูลที่เข้ามามีค่าน้อยกว่าก็จะผ่านค่าที่เซตไว้โดยไม่มีการเปลี่ยนแปลงใดๆ แล้วไปทำการเปรียบเทียบในระดับถัดไป ถ้าข้อมูลมีค่ามากกว่าก็จะนำค่าที่ได้มาในระดับก่อนหน้านี้มากระทำทางลอจิกออร์ กับ 010 แต่ถ้าข้อมูลที่เข้ามามีค่าน้อยกว่าก็จะผ่านค่าที่ได้จากระดับก่อนหน้านี้ไปอีก แล้วทำการเปรียบเทียบในระดับถัดไป ถ้าข้อมูลที่เข้ามามีค่ามากกว่าก็จะนำค่าที่ได้จากกระดับก่อนหน้านี้มากระทำทางลอจิกออร์ กับ 001

แต่ถ้าข้อมูลที่เข้ามามีค่าน้อยกว่าก็จะทำการจัดระดับใหม่ เมื่อเสร็จก็นำค่าที่ได้จากกระบวนการดังกล่าวออกไปยังส่วนของการเข้ารหัสโดย จะนำไปต่อกับบิตเครื่องหมายก็จะได้ 4 บิตดังตัวอย่างต่อไปนี้

$$Din = 20596_{10}$$
$$= 0101000001011001_2$$

นำค่า Din มาเปรียบเทียบกับค่าทางด้าน 16 บิต ของตำแหน่ง a

$$Din = 0101000001011001_2$$
$$a = 0011101110111100_2$$

เมื่อเปรียบเทียบค่าของ Din จะมากกว่าค่าในตำแหน่ง a ก็จะทำการจัดเก็บค่าทางด้าน 4 บิตเอาไว้ คือ 100 แล้วทำการเพิ่มค่าของ a เข้าไป 2 ∇ ก็จะได้ดังนี้

$$Cmp1 = a + 2\nabla$$
$$= 15292_{10} + 2_{10} \times 4369_{10}$$
$$= 24030_{10}$$
$$= b$$

สังเกตว่าค่าที่ได้ในระดับถัดไปจะเป็นค่าของข้อมูลทางด้าน 16 บิตในตำแหน่ง b เมื่อได้ค่าดังกล่าวแล้วก็จะนำข้อมูลที่เข้ามาเปรียบเทียบกับค่าในตำแหน่ง b

$$Din = 0101000001011001_2$$
$$Cmp1 = 0101110111011110_2$$

ซึ่งเมื่อเปรียบเทียบกันแล้วค่าของ Din จะน้อยกว่าค่าในตำแหน่ง b ก็จะนำค่าที่เก็บไว้ทางด้าน 4 บิต ไปเก็บไว้ซึ่งก็จะเก็บค่า 100 ส่วนทางด้าน 16 บิต ก็ทำการลดค่าค่าของ b ลงไป ∇ ซึ่งก็จะได้ดังนี้

$$Cmp2 = b - \nabla$$
$$= 24{,}030_{10} - 4{,}369_{10}$$
$$= 19{,}661_{10}$$
$$= d$$

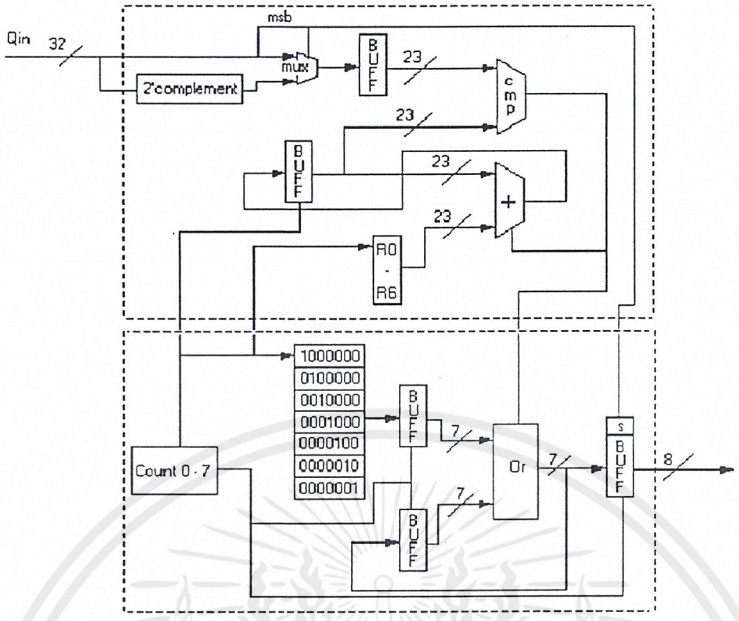
สังเกตว่าค่าที่ได้ในระดับถัดไปจะเป็นค่าของข้อมูลทางด้าน 16 บิตในตำแหน่ง d เมื่อได้ค่าดังกล่าวแล้วก็จะนำข้อมูลที่เข้ามาเปรียบเทียบกับค่าในตำแหน่ง d

$$Din = 0101111001011001_2$$
$$Cmp2 = 0100110011001101_2$$

ซึ่งเมื่อเปรียบเทียบกันแล้วค่าของ Din จะมากกว่าค่าในตำแหน่ง d ก็จะนำเอาค่าที่เก็บไว้ทางด้าน 4 บิต คือ 100 มา OR กับ 001 ซึ่งผลที่ได้ก็ คือ 101 และเอาต์พุตของการจัดระดับสัญญาณเมื่อนำบิตเครื่องหมายของข้อมูลที่เข้ามา ซึ่งในที่นี้ค่าของข้อมูลที่เข้ามามีค่าเป็นบวก บิตเครื่องหมายของข้อมูลมีระดับลอจิกเป็น 0 นำมาต่อกับ 3 บิตที่ผ่านการจัดระดับข้อมูล ค่าที่ได้ออกมาก็ คือ 0101 ดังนั้น

$$Dout = 0101_2$$

ซึ่งจากขบวนการดังกล่าวสามารถที่จะนำมาทำการออกแบบวงจรตามลักษณะการทำงานให้เป็นวงจรจัดระดับสัญญาณ (Quantizer) ให้มีการจัดระดับสัญญาณจาก 24 บิต ให้ลงมาเป็น 8 บิต ได้ดังในรูปที่ 3.34 (ในการทำงานของวงจรแปลงเวฟเลตนั้นจะให้ค่าได้สูงสุดเพียง 24 บิต จึงนำมาจัดระดับเพียง 24 บิต) เมื่อทำการจัดระดับสัญญาณจะต้องส่งข้อมูลและสัญญาณนาฬิกาไปพร้อมกับข้อมูล ซึ่งสัญญาณนาฬิกาที่ส่งไปพร้อมกับข้อมูลที่จัดระดับจะมีลักษณะ คือ ข้อมูล 1 ข้อมูลต่อสัญญาณนาฬิกา 1 ลูก โดยสัญญาณนาฬิกาที่ส่งไปยังวงจรเข้ารหัสฮัฟฟ์แมนจะต้องมีความถี่ที่สัมพันธ์กับวงจรเข้ารหัสฮัฟฟ์แมนที่ได้ออกแบบไว้ สำหรับผังการทำงานของวงจรจัดระดับสัญญาณ (Quantizer) จะแสดง ดังรูปที่ 3.34



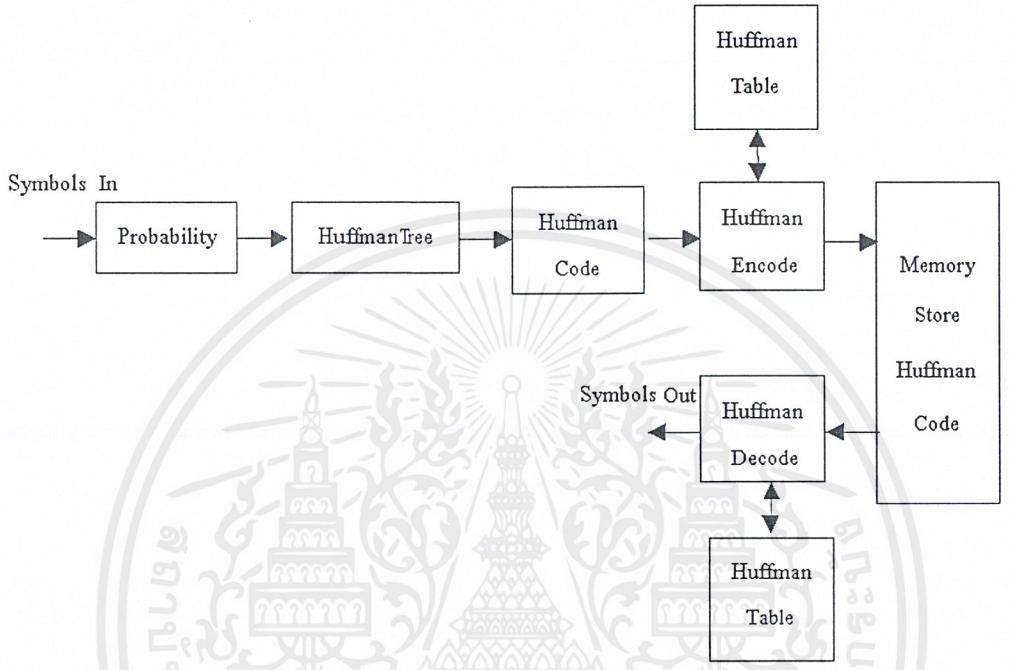
รูปที่ 3.34 ผังการทำงานของวงจรจัดระดับสัญญาณ (Quantizer)

จากรูปที่ 3.34 ในการทำงานของวงจรจัดระดับสัญญาณนั้น สังเกตจะมีการแบ่งวงจรส่วนที่เป็นวงจรเปรียบเทียบ ซึ่งจะเป็นชุดของวงจรที่มีการทำงานที่ 23 บิต โดยในบิตที่มีนัยสำคัญมากที่สุดจะเป็นบิตเครื่องหมาย จะถูกแยกออกไปควบคุมการเรียกใช้งานค่าจากตารางภายในวงจรนี้จะมีหน้าที่ในการเปรียบเทียบข้อมูลที่ได้มาจากการแปลงเวฟเลต เพื่อที่จะนำสัญญาณไปควบคุมอีกส่วนหนึ่ง คือ ส่วนที่เป็นวงจรที่ทำงานที่ 7 บิต โดยบิตสุดท้ายจะได้จากบิตเครื่องหมายที่ถูกส่งมาจากส่วนของวงจรเปรียบเทียบขนาด 23 บิต โดยวงจร 7 บิต นี้จะเป็นวงจรที่ทำหน้าที่รับสัญญาณที่จะส่งมาบอกว่าค่าที่เข้ามามีค่ามาก หรือน้อย โดยในการทำงานของวงจรนี้ใน 1 ข้อมูลจะมีการเปรียบเทียบเกิดขึ้นทั้งหมด 7 ครั้ง จึงทำให้ได้ข้อมูลที่เป็น 7 บิต เมื่อกระทำกระบวนการต่างๆ เสร็จสิ้นก็จะมีการนำเอาบิตเครื่องหมายที่ถูกส่งมาจากวงจรชุด 23 บิต มาเพิ่มใน SHF ซึ่งจะให้ข้อมูลออกไปเป็น 8 บิต

3.6 ความคิดเบื้องต้นในการสร้างวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน

วงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนสามารถออกแบบสร้างได้โดยใช้ ฮาร์ดแวร์หรือซอฟต์แวร์เพียงอย่างเดียวข้อมูลที่เข้ามาในวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนต้องมีความความน่าจะเป็นของข้อมูลค่อนข้างแน่นอนสำหรับการแทนรหัสให้กับข้อมูลที่มีความน่าจะเป็นสูงเป็นมี

ความยาวของรหัสสั้น ข้อมูลที่มีความน่าจะเป็นน้อยอาจจะมีความยาวของรหัสยาวกว่าข้อมูลที่เข้ามาซึ่งไปตามหลักการของการเข้ารหัส และถอดรหัสฮัฟฟ์แมนโดยสามารถอธิบายการทำงานได้ดังรูปที่ 3.35

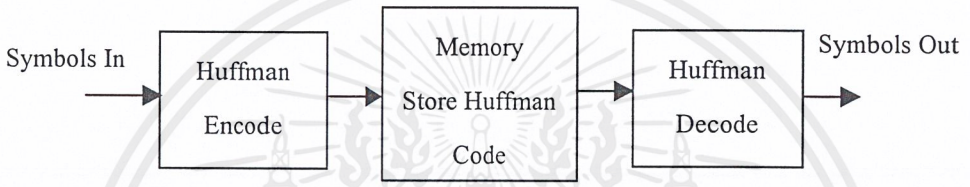


รูปที่ 3.35 แผนผังการทำงานของวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน

จากรูปที่ 3.35 สามารถอธิบายการทำงานได้ดังนี้ คือ ข้อมูลที่เข้ามาในวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน จะถูกหาความน่าเป็นของข้อมูลแต่ละข้อมูล จากนั้นจะจัดโครงสร้างข้อมูลเป็นแบบฮัฟฟ์แมนทรีซึ่งเป็นโครงสร้างข้อมูลตามความน่าจะเป็นของข้อมูล กระบวนการต่อมา คือ การแทนรหัสฮัฟฟ์แมนซึ่งการแทนแทนรหัสฮัฟฟ์แมนจะแทนรหัสตามโครงสร้างข้อมูลแบบฮัฟฟ์แมนทรี ในขั้นตอนนี้จะได้รหัสฮัฟฟ์แมนที่แทนข้อมูลแต่ละตัวซึ่งความยาวของรหัสฮัฟฟ์แมนจะสั้น หรือยาวขึ้นอยู่กับโครงสร้างข้อมูลแบบฮัฟฟ์แมนทรี หลังจากนั้นสร้างวงจรเข้ารหัสฮัฟฟ์แมนเพื่อที่จะนำรหัสฮัฟฟ์แมนไปจัดเก็บไว้ในหน่วยความจำซึ่งการจัดเก็บรหัสฮัฟฟ์แมนไว้ในหน่วยความจำจะสามารถลดบิตข้อมูลเมื่อเทียบกับการจัดเก็บข้อมูลโดยตรงในหน่วยความจำ และกระบวนการสุดท้าย คือ การถอดรหัสฮัฟฟ์แมนที่จัดเก็บไว้ในหน่วยความจำไปเป็นข้อมูลตามลำดับ

3.7 การออกแบบวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนโดยใช้ฮาร์ดแวร์เพียงอย่างเดียว

วงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนโดยใช้ฮาร์ดแวร์เพียงอย่างเดียวสามารถแบ่งเป็นสองส่วน คือ วงจรเข้ารหัสฮัฟฟ์แมนเป็นส่วนที่นำข้อมูลอินพุตภายนอกมาแทนรหัสฮัฟฟ์แมนเพื่อที่จะนำรหัสฮัฟฟ์แมนไปจัดเก็บไว้ในหน่วยความจำ และวงจรถอดรหัสฮัฟฟ์แมนเป็นส่วนที่ถอดรหัสที่อยู่ในหน่วยความจำออกมาเป็นข้อมูลที่เหมือนกันกับข้อมูลที่เข้ามาทางด้านอินพุตของวงจรเข้ารหัสและถอดรหัสฮัฟฟ์แมนสามารถเขียนเป็นแผนผังการทำงานได้ดังรูป 3.36



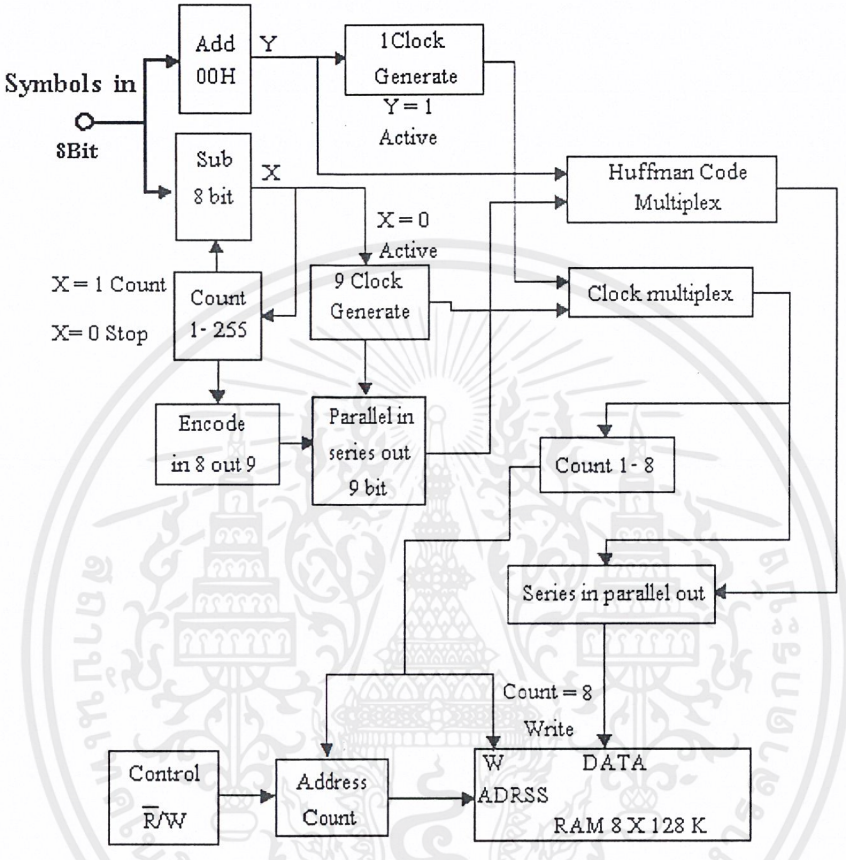
รูปที่ 3.36 วงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนโดยใช้ฮาร์ดแวร์เพียงอย่างเดียว

การออกแบบวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนออกแบบด้วยโปรแกรม Xilinx Foundation Series เวอร์ชัน 3.1 ใช้ Schematic Editor ในการออกแบบวงจรส่วนย่อยจากนั้นแปลงวงจรส่วยย่อยเป็นแมโครเพราะในหนึ่งหน้าต่างของ Schematics Editor ไม่สามารถที่จะออกแบบวงจรแต่ส่วนย่อยทั้งหมดรวมไว้ในหนึ่งหน้าต่างของ Schematics Editor ได้ การออกแบบวงจรจาก Schematics Editor จะได้เป็นวงจรดิจิตอลแบบอันดับ (Combination) และวงจรดิจิตอลแบบลำดับ (Sequential) เป็นส่วนย่อยรวมกันเป็นระบบวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน การออกแบบวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนโดยใช้ฮาร์ดแวร์เพียงอย่างเดียวจะแยกออกเป็นสองส่วนใหญ่ๆ คือ วงจรเข้ารหัสฮัฟฟ์แมน และวงจรถอดรหัสฮัฟฟ์แมน

3.8 วงจรเข้ารหัสฮัฟฟ์แมน

การออกแบบวงจรเข้ารหัสฮัฟฟ์แมนโดยใช้ฮาร์ดแวร์เพียงอย่างเดียวในกระบวนการก่อนหน้านี้จะต้องจัดโครงสร้างข้อมูลให้อยู่ในรูปของฮัฟฟ์แมนทรี ตามความน่าจะเป็นของข้อมูลของแต่ละข้อมูลจากนั้นแทนรหัสฮัฟฟ์แมนให้กับข้อมูลโดยที่รหัสฮัฟฟ์แมนที่แทนข้อมูลจะมีความยาวของรหัสมากหรือน้อยจะขึ้นอยู่กับความน่าจะเป็นของแต่ละข้อมูล ในกรณีที่ได้ทำการออกแบบสร้าง คือ การเก็บสัมประสิทธิ์ เสียงซึ่งเป็นข้อมูลขนาด 8 บิตความน่าจะเป็นของ ข้อมูล 0000 0000

หรือ S0 มากกว่าร้อยละ 50 ของข้อมูล ดังนั้นจึงจัดให้ความน่าจะเป็นของข้อมูลอื่นเท่ากันเมื่อจัดโครงสร้างความน่าจะเป็นของข้อมูลให้อยู่ในรูปของฮัฟฟ์แมนทรี แล้วทำการแทนรหัสฮัฟฟ์แมนให้กับข้อมูลดังกล่าว จะได้ รหัสฮัฟฟ์แมนแทนข้อมูลต่างๆ



รูปที่ 3.37 แผนผังการทำงานของวงจรเข้ารหัสฮัฟฟ์แมน

3.8.1 การออกแบบวงจรเข้ารหัสฮัฟฟ์แมน

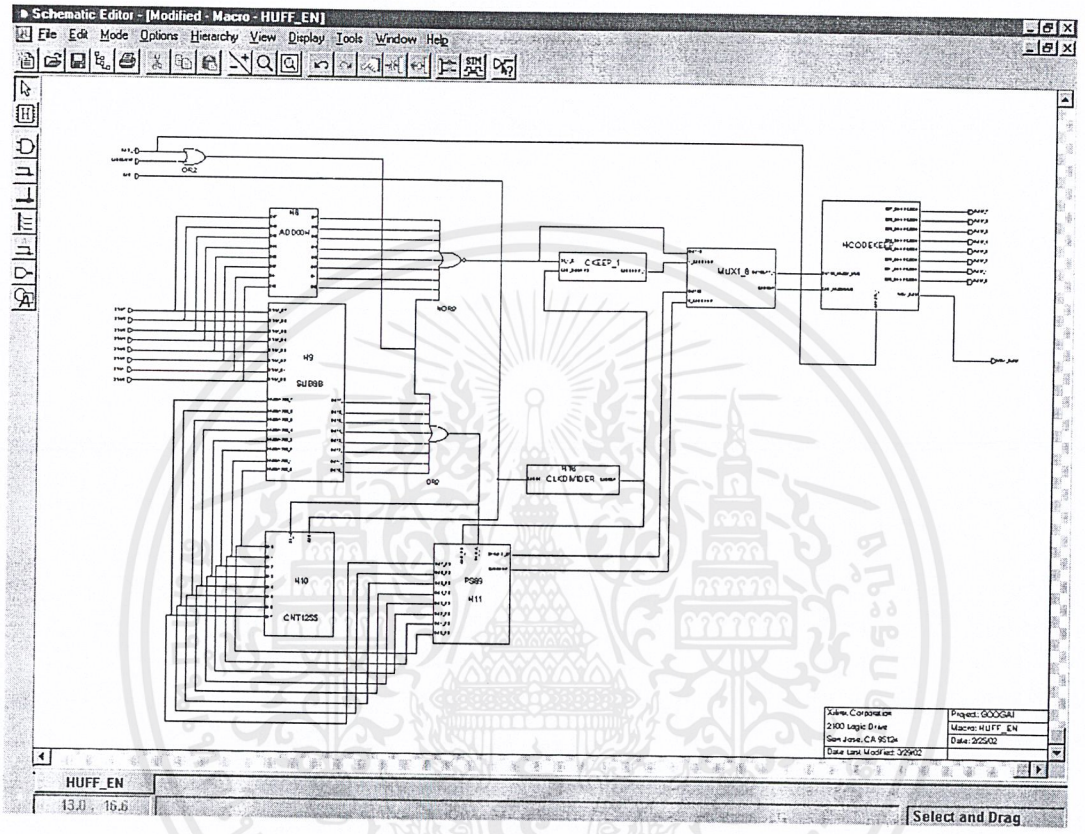
การออกแบบวงจรเข้ารหัสฮัฟฟ์แมนจะอาศัยคุณสมบัติวงจรลอจิกเกตแบบต่างๆ วงจรดิจิตอลแบบอันดับ และวงจรดิจิตอลแบบลำดับซึ่งจะนำคุณสมบัติของวงจรดังกล่าวมารวมกันเป็นวงจรเข้ารหัสฮัฟฟ์แมน มีหน้าที่เป็นวงจรจัดการนำรหัสฮัฟฟ์แมนไปจัดเก็บในหน่วยความจำขนาด 8 บิต 128 กิโลไบต์ วงจรเข้ารหัสฮัฟฟ์แมนสามารถเขียนแผนผังการทำงานได้ ดังรูปที่ 3.37

จากแผนผังการทำงานของวงจรเข้ารหัสฮัฟฟ์แมน สามารถอธิบายหลักการทำงานได้ดังนี้ เมื่อมีข้อมูลเข้ามา วงจรจะทำการแยกข้อมูลที่ต้องแทนรหัสฮัฟฟ์แมนออกเป็นสองส่วน คือ S0 ซึ่งจะแทนรหัสฮัฟฟ์แมนความยาวของรหัส 1 บิต และ S1 ถึง S255 จะแทนรหัสฮัฟฟ์แมนความยาว

ของรหัส 9 บิต โดยมีวงจรตัวนับ 1 ถึง 255 กับวงจรลบเลขขนาด 8 บิต เป็นวงจรแยกเอาข้อมูลที่ต้องแทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต โดยที่ข้อมูลที่ต้องแทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต จะลบเลขกับวงจรลบเลขขนาด 8 บิต จนผลลัพธ์ที่ได้มีค่าเป็น 0 ทำให้วงจรตัวนับ 1 ถึง 255 หยุดนับแล้วให้นำตำแหน่งของวงจรตัวนับ 1 ถึง 255 ชี้อยู่จะตรงกับข้อมูลที่เข้ามา ไปเข้ารหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต จากวงจรลบเลขจะพบว่าค่าของ X ซึ่งเป็นผลลัพธ์ การลบเลขระหว่างข้อมูล ที่เข้ามากับวงจรตัวนับ 1 ถึง 255 เป็นผลให้ค่า X = 0 ซึ่งในช่วงเวลาเดียวกันนั้นจะทำ ค่าของ Y = 0 จะส่งผลให้วงจรเข้ารหัสฮัฟฟ์แมนที่มีความยาวของรหัส 1 บิตไม่ทำงาน พร้อมกันนั้นวงจรสร้างสัญญาณนาฬิกา 1 ลูก จะไม่ทำงานเช่นเดียวกัน จากนั้นส่งค่าของรหัสฮัฟฟ์แมนที่แทนข้อมูลแต่ละตัวไปพร้อมกับสัญญาณนาฬิกา 9 ลูก เข้าสู่ส่วนมัลติเพล็กซ์ข้อมูล และมัลติเพล็กซ์สัญญาณนาฬิกาเพื่อทำการจัดเก็บรหัสฮัฟฟ์แมนลงไปในหน่วยความจำที่ได้กำหนดไว้ ในส่วนของ S0 จะเป็นข้อมูลที่แทนรหัสฮัฟฟ์แมนที่มีความยาวของรหัส 1 บิต ซึ่งจะมีวงจรบวกเลขขนาด 8 บิต บวกกับรหัสข้อมูล 0000 0000 จะทำให้ Y = 1 ซึ่งในขณะเดียวกันนั้น ค่า X = 1 จะทำให้วงจรส่วนเข้ารหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตไม่ทำงาน แต่วงจรสร้างสัญญาณนาฬิกา 1 ลูกทำงานพร้อมกับส่งรหัสฮัฟฟ์แมนที่มีความยาว 1 บิต กับสัญญาณนาฬิกา 1 ลูกไปพร้อมกับรหัสฮัฟฟ์แมน เข้าสู่ส่วนมัลติเพล็กซ์ข้อมูล และมัลติเพล็กซ์สัญญาณนาฬิกา เพื่อทำการจัดเก็บรหัสฮัฟฟ์แมนลงไปในหน่วยความจำ สำหรับการจัดเก็บข้อมูลลงในหน่วยความจำที่มีขนาด 8 บิต 128 กิโลไบต์ โดยมีวงจรเลื่อนข้อมูลแบบเข้าอนุกรมออกแบบขนานขนาด 8 บิต และใช้วงจรตัวนับมอดูลัส 1 ถึง 8 เป็นวงจรควบคุมการทำงานของการเขียนข้อมูลลงหน่วยความจำโดยวงจรเลื่อนข้อมูลแบบเข้าอนุกรมออกขนานก็จะทำการเก็บข้อมูล โดยวงจรตัวนับมอดูลัส 1 ถึง 8 นับไปถึง 8 หมายถึงส่วนของรหัสฮัฟฟ์แมนในวงจรเข้าอนุกรมออกขนานครบ 8 บิต จะเก็บข้อมูลลงไปในหน่วยความจำ พร้อมวงจรตัวนับมอดูลัส 1 ถึง 8 ส่งสัญญาณ เพิ่มตำแหน่งของหน่วยความจำเข้าไปอีก 1 ตำแหน่งหลังจากเขียนข้อมูลลงหน่วยความจำวงจรเข้ารหัสฮัฟฟ์แมนจะทำงานลักษณะอย่างนี้จนจัดเก็บรหัสในหน่วยความจำเต็มทุกตำแหน่ง วงจรควบคุมจะควบคุมให้วงจรในหน่วยของความจำหยุดจัดเก็บรหัสฮัฟฟ์แมน และพร้อมที่จะทำการถอดรหัสฮัฟฟ์แมนในกระบวนการของวงจรถอดรหัสฮัฟฟ์แมนจากแผนผังการทำงานจะเป็นการทำงานโดยรวมของวงจรเข้ารหัสฮัฟฟ์แมนในส่วนของการออกแบบวงจรจะออกแบบให้การทำงานเข้าจังหวะกับภาคควอนไตธ์ ซึ่งเป็นวงจรก่อนหน้าที่จะนำข้อมูลมาเข้ารหัสฮัฟฟ์แมน ซึ่งสัญญาณนาฬิกาที่ภาคควอนไตธ์ที่ส่งมาจากได้ออกแบบเอาไว้มีความถี่ 16 กิโลเฮิรตซ์ หรือเป็น 2 เท่าของสัญญาณนาฬิกาที่อ่านข้อมูลเข้ามาแปลงเวฟเลต

3.8.2 การออกแบบวงจรเข้ารหัสฮัฟฟ์แมนบน Schematics Editor

วงจรเข้ารหัสฮัฟฟ์แมนออกแบบวงจรรวมออกแบบวงจรเป็นส่วนย่อยแล้วสร้างเป็นแมโครนำมาสร้างเป็นวงจรรวมบน Schematic Editor ดังรูป 3.38

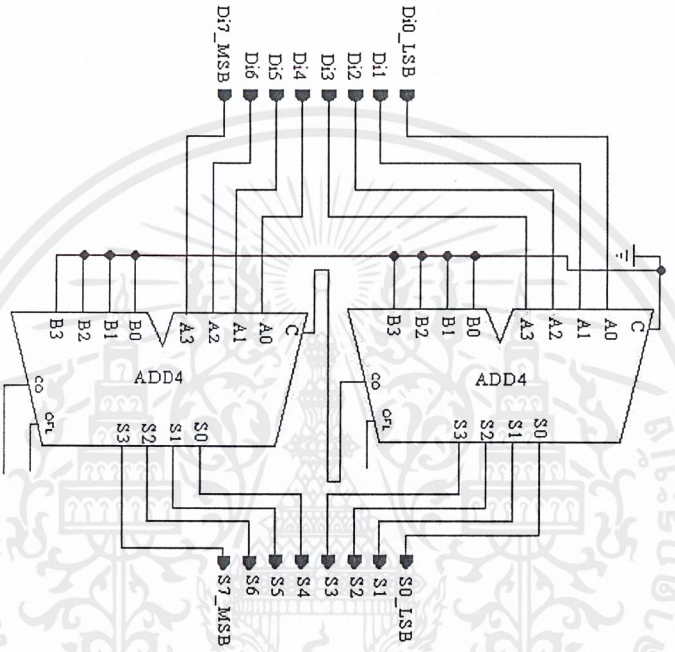


รูปที่ 3.38 วงจรเข้ารหัสฮัฟฟ์แมนบน Schematic Editor

จากวงจรจะพบว่าในส่วนของ อินพุตที่เข้ามา คือ D0 ถึง D7 จะเป็นข้อมูลขนาด 8 บิตจากภาคควอนไตร์สัมประสิทธิ์สัญญาณเสียงสัญญาณนาฬิกาที่ใช้ในการประมวลผล ของวงจรเข้ารหัสฮัฟฟ์แมน (สัญญาณ Clk) และสัญญาณนาฬิกาจากภาคควอนไตร์ (สัญญาณ Clk Quantize) เพื่อเป็นตัวบ่งชี้การเปลี่ยนแปลงของข้อมูล D0 ถึง D7 ในช่วงเวลาต่างๆ ที่นำมาเข้ารหัสฮัฟฟ์แมน และสัญญาณรีเซต เพื่อเป็นการเริ่มต้นในการเก็บรหัสฮัฟฟ์แมนสัญญาณนาฬิกา (สัญญาณ Clk) ที่ใช้ในการประมวลผลของวงจรเข้ารหัสฮัฟฟ์แมนจะมีความถี่มากกว่าสัญญาณนาฬิกาจากภาคควอนไตร์ เพราะวงจรตัวนับ 1 ถึง 255 จะต้องทำการนับอย่างรวดเร็ว เพื่อทำการลบกับข้อมูลที่แทนด้วย รหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต คือ S1 ถึง S255 การประมวลผลของวงจรเข้าหัสฮัฟฟ์แมน

กระทำในช่วงขอบขาลงของสัญญาณนาฬิกาจากภาคควอนไตธ์ ในช่วงขอบขาขึ้นของสัญญาณนาฬิกาของภาคควอนไตธ์ วงจรเข้ารหัส ฮัฟฟ์แมนพร้อมที่รับข้อมูลชุดใหม่เข้ามาแทนรหัสฮัฟฟ์แมนต่อไป สำหรับวงจรที่แยกออกมาเป็นส่วนๆ ที่ได้จะเป็นในลักษณะของการสร้างแมโครแล้วลิงค์เข้ามาสู่วงจรเข้ารหัสฮัฟฟ์แมน โดยแต่ละส่วนมีการออกแบบวงจร และการทำงานดังนี้

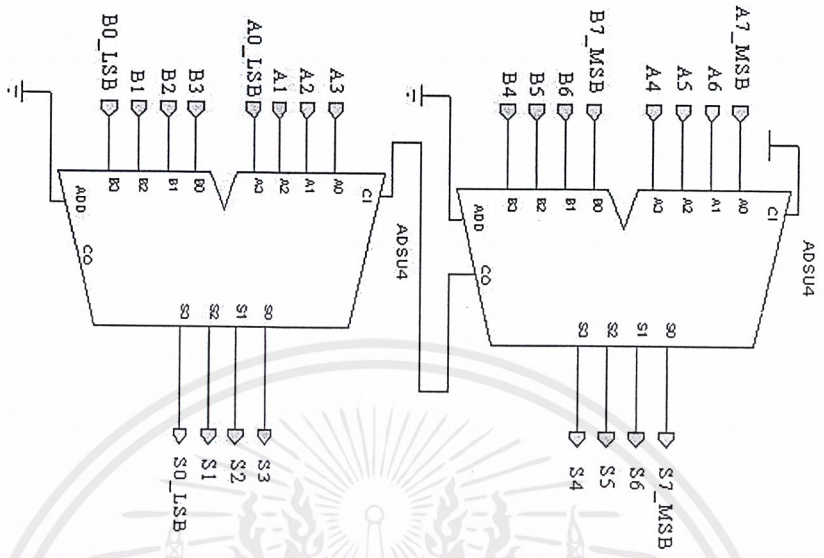
3.8.3 วงจรบวกเลขขนาด 8 บิต



รูปที่ 3.39 วงจรบวกเลขขนาด 8 บิต

วงจร บวกเลขขนาด 8 บิต เป็นการออกแบบวงจรโดยใช้วงจรบวกเลขขนาด 4 บิตที่มีอยู่ใน Library Virtexe มาต่อร่วมกันสองตัวเป็นวงจรบวกเลขขนาด 8 บิต การออกแบบวงจรในส่วนนี้เพื่อที่จะแยกเอา S0ซึ่งมีค่าไบนารีโค้ดเป็น 0000 0000 มาบวกเลขกับ 0000 0000 จะเป็นผลให้ผลลัพธ์ออกมาเป็น 0000 0000 ไปควบคุมวงจรเข้ารหัสฮัฟฟ์แมนที่มีความยาวของรหัส 1 บิต ให้กับรหัส S0 ที่เข้ามา ส่วนข้อมูลอื่นที่ไม่ใช่ S0 คือตั้งแต่ S1 ถึง S255 นั้นเมื่อมากระทำการบวกเลขกับ 0000 0000 จะได้ ผลลัพธ์ที่มีค่ามากกว่า 0 และส่งผลให้ค่าที่ได้ออกมาเป็น 0000 0000 ทำให้วงจรเข้ารหัสฮัฟฟ์แมนที่มีความยาวของรหัส 1 บิต ไม่ทำงาน และค่าที่ได้จะนำไปกระทำทางลอจิกนอร์เกต 9 อินพุต ในผลบวกแต่ละบิต ซึ่งอีกอินพุตหนึ่งจะมาจาก สัญญาณนาฬิกาของภาคควอนไตธ์ เพื่อที่จะให้วงจรฮัฟฟ์แมน เอ็นโค้ดทำงานเฉพาะช่วงขอบขาลงเท่านั้น ส่วนขอบขาขึ้นจะเป็นช่วงที่วงจรฮัฟฟ์แมนเอ็นโค้ด พร้อมที่จะรับข้อมูลชุดใหม่เข้ามา

3.8.4 วงจรลบเลขขนาด 8 บิต



รูปที่ 3.40 วงจรลบเลขขนาด 8 บิต

วงจรลบเลขเป็นการออกแบบวงจรใช้วงจรบวกเลข และลบเลขขนาด 4 บิตที่มีอยู่ใน Library Vertexe ซึ่งถ้านำมาต่อกันสองส่วน ขาบวก (ADD) ของวงจร เป็น 0 จะทำให้วงจรเป็นวงจรลบเลขกลับกันถ้าขาบวก (ADD) ของวงจร เป็น 1 จะทำให้เป็นวงจรบวกเลข และวงจรลบเลขเป็นวงจรที่มีส่วนสำคัญในการแยกเอาข้อมูลที่ต้องเข้ารหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต คือ S1 ถึง S255 โดยวงจร จะทำการลบเลขระหว่างข้อมูลที่เข้ามากับวงจรตัวนับ 1 ถึง 255 แล้วผลลบขนาด 8 บิตไปกระทำทางลอจิกออร์กับสัญญาณนาฬิกาของภาคควอนไตร์ เพื่อที่จะให้ได้ผลลัพท์ X ออกมาเป็นระดับลอจิก1 หรือ 0 เพื่อเป็นอินพุตให้กับวงจรส่วนย่อยต่างๆ ของฮัฟฟ์แมนเอ็นโค้ดที่มีความยาวขนาด 9 บิตให้ทำงานในกรณีที่ สัญญาณนาฬิกาของภาคควอนไตร์เป็นขอบขาลง พร้อมกับข้อมูลที่เข้ามาจะถูกแทนด้วยรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตส่วนขอบขาขึ้นจะเป็นช่วงที่วงจรฮัฟฟ์แมนเอ็นโค้ดพร้อมที่จะรับ ข้อมูลชุดใหม่เข้ามา

3.8.5 วงจรตัวนับ 1 ถึง 255

วงจรนับ 1 ถึง 255 ในวงจรเข้ารหัสฮัฟฟ์แมนออกเป็นฟังก์ชันการทำงานดังรูปที่ 3.41 และ 3.42

Input : $X(t) \in \{0, 1\}$

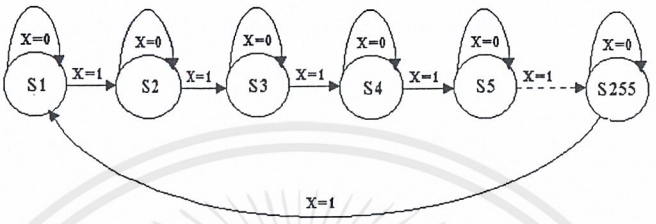
Output : $Z(t) = \{1, 2, 3, 4, 5, 6, 7, 8, \ldots, 255\}, Z(t) \in (0, 1)$

State : S(t) ∈ {S1, S2, S3, S4, S5, S6, S7, S8, …, S255}

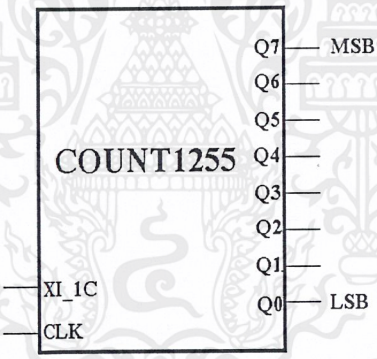
Initial State : S(0) = S1

Function : Counts Modulus 1 to 255 i.e., (1, 2, 3, 4, 5, …, 255, 1, 2, 3, 4, 5, …, 255)

State DiagRam : ดังรูปที่ 3.41



รูปที่ 3.41 แผนผังสถานะของวงจรตัวนับ 1 ถึง 255



รูปที่ 3.42 แมโครตัวนับ 1 ถึง 255

วงจรตัวนับ 1 ถึง 255 เป็นวงจรตัวนับ ขนาด 8 บิต แบบมีอินพุต คือ XI_1C โดยวงจรจะมีหลักการทำงาน คือ ถ้า XI_1C มีระดับลอจิกเป็น 1 วงจรจะนับไปข้างหน้าทีละ 1 สถานะ ถ้า XI_1C มีระดับลอจิกเป็น 0 ทำให้วงจรจะหยุดนับ และค้างสถานะอยู่ในสถานะนั้น โดยเริ่มต้น คือ S1 ส่วนการทำงานของวงจรตัวนับ 1 ถึง 255 ในฮัฟฟ์แมนเอ็นโค้ดจะเป็นตัวชี้ข้อมูลที่เข้ามาไปแทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต คือ S1 ถึง S255 มีลักษณะการทำงานเมื่อมีข้อมูลเข้ามาพร้อมกับสัญญาณนาฬิกาของภาคควอนไตซ์ในช่วงขอบขาลง วงจรตัวนับ 1 ถึง 255 จะทำการนับอย่างรวดเร็วเพื่อนำค่าที่ได้จากวงจรตัวนับไปลบกับข้อมูลที่เข้ามา เมื่อค่าของวงจรตัวนับตรงกับข้อ

มูลที่เข้ามา จะทำให้ผลลัพธ์ที่ได้เป็น 0000 0000 และเมื่อนำผลลัพธ์ทั้งหมดทุกบิต ไปกระทำทางลอจิกออร์กับกับสัญญาณนาฬิกาของภาคควอนไตธ์ ซึ่งเป็นผลทำให้การกระทำทางลอจิกออร์ ทำให้มีระดับลอจิกเป็น 0 แล้วนำค่าไปเป็นอินพุต XI_1C ทำให้วงจรตัวนับ 1 ถึง 255 วงจรจะหยุดนับค่าของวงจรตัวนับค้างสถานะอยู่เท่ากับข้อมูลที่เข้ามา และผลจากการกระทำทางลอจิกทำให้ระดับลอจิกเป็น 0 อีกส่วนหนึ่งจะไปเป็น อินพุตของ วงจรย่อยอีกวงจรพร้อมกัน คือ วงจรสร้างสัญญาณนาฬิกา 9 ลูก วงจรแทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต

3.8.6 วงจรสร้างสัญญาณนาฬิกา 9 ลูก และวงจรแทนรหัสฮัฟฟ์แมนความยาว 9 บิต

วงจรสร้างสัญญาณนาฬิกา 9 ลูกในวงจรเข้ารหัสฮัฟฟ์แมนออกแบบตามฟังก์ชันการทำงาน สามารถอธิบายการทำงานดังตารางที่ 3.4 และรูปที่ 3.43

Input : $X(t) \in \{0, 1\}$

Output : $Z(t) \in \{0, 1\}$

State : $S(t) \in \{A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T\}$

Initial State : $S(0) = A$

Function : ดังตารางที่ 3.4

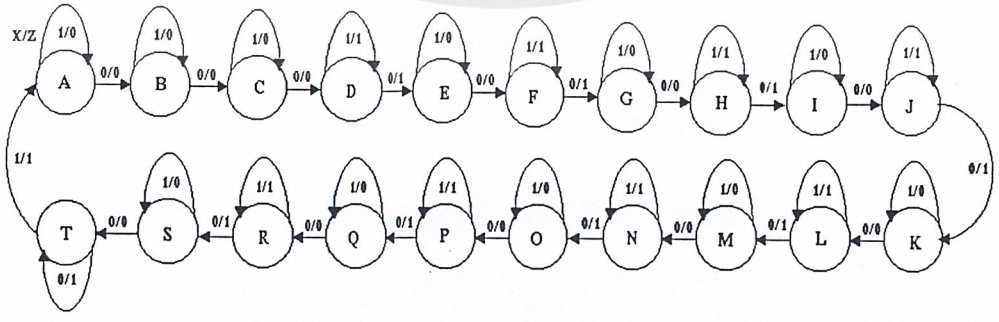
ตารางที่ 3.4 ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 9 ลูก

Present State		Next State X,Z	
State Name	Binary Code	X = 0	X = 1
A	00000	B, 0	A, 0
B	00001	C, 0	B, 0
C	00010	D, 0	C, 0
D	00011	E, 1	D, 1
E	00100	F, 0	E, 0
F	00101	G, 1	F, 1
G	00110	H, 0	G, 0
H	00111	I, 1	H, 1

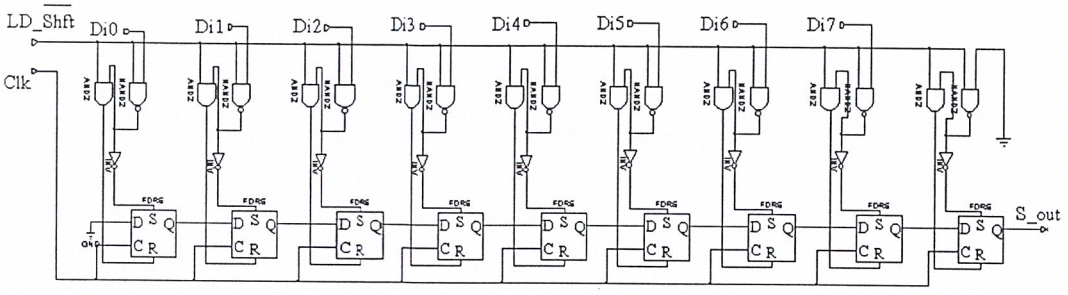
ตารางที่ 3.4 (ต่อ) ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 9 ลูก

Present State		Next State X, Z	
State Name	Binary Code	X = 0	X = 1
I	01000	J, 0	I, 0
J	01001	K, 1	J, 1
K	01010	L, 0	J, 0
L	01011	M, 1	L, 1
M	01100	N, 0	M, 0
N	01101	O, 1	N, 1
O	01110	P, 0	O, 0
P	01111	Q, 1	P, 1
Q	10000	R, 0	Q, 0
R	10001	S, 1	R, 1
S	10010	T, 0	S, 0
T	10011	T, 1	A, 1

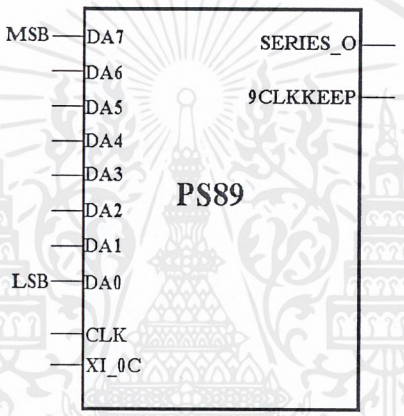
State Diagram : ดังรูปที่ 3.43



รูปที่ 3.43 แผนผังสถานะของวงจรสร้างสัญญาณนาฬิกา 9 ลูก



รูปที่ 3.44 วงจรแทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต



รูปที่ 3.45 วงจรสร้างสัญญาณนาฬิกา 9 ลูกวงจรแทนรหัสฮัฟฟ์แมน ความยาวของรหัส 9 บิต เมื่อแปลงเป็นแมโคร

จากรูปที่ 3.45 เป็นวงจรที่ทำหน้าที่สร้างสัญญาณนาฬิกา 9 ลูก และวงจรแทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต และซึ่งได้สร้างบน Schematic Editor สร้างเป็นแมโครเดียวกัน โดยมีอินพุตที่เป็นข้อมูล (D0~D7) มาจาก เอาต์พุตของวงจรตัวนับ 1 ถึง 255 วงจรสร้างสัญญาณนาฬิกา 9 ลูกเป็นการออกแบบวงจรดิจิตอลแบบลำดับ แบบมีอินพุต คือ XI_0C ซึ่งค่าของ XI_0C มาจากการกระทำทางลอจิกออร์ของค่าจากการลบเลขระหว่างวงจรตัวนับ 1 ถึง 255 กับข้อมูลที่เข้ามาในวงจรเข้ารหัสฮัฟฟ์แมนกับสัญญาณนาฬิกาจากภาคควอนไตธ์ จะนำข้อมูลไปเข้ารหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตเมื่อ อินพุตวงจร ตัวนับ 1 ถึง 255 คือ XI_1C มีระดับลอจิกเป็น 0 ซึ่งหมายถึงมีข้อมูลที่ต้องแทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตในช่วงเวลาสัญญาณนาฬิกาจากภาคควอนไตธ์ช่วงขอบขาลง วงจร สร้างสัญญาณนาฬิกา 9 ลูกจะสร้างสัญญาณนาฬิกาแล้วส่งสัญญาณนาฬิกากับรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต ไปเก็บไว้ในหน่วยความจำแล้ววงจรจะ

หยุดสร้างสัญญาณนาฬิกาจนกว่ามีข้อมูลที่จะต้องเข้ารหัสฮัฟฟ์แมน ความยาวของรหัส 9 บิต ใหม่จากภาคควอนไตร์ข้อมูลถัดไปเมื่อ พิจารณา XI_0C มีระดับลอจิก เป็น 0 ซึ่งสถานะนี้ คือ มีข้อมูลเข้ามาในวงจรเข้ารหัสฮัฟฟ์แมน จากแผนผังสถานะของวงจรสร้างสัญญาณนาฬิกา 9 ลูกส่วนของวงจรจะสร้างสัญญาณนาฬิกาเป็นจำนวน 10 ลูกโดยสัญญาณนาฬิกาลูกแรกจะนำไปใช้แทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตวงจรสร้างสัญญาณนาฬิกา 9 ลูกจะสร้างสัญญาณนาฬิกาแล้วส่งสัญญาณนาฬิกากับรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต นำไปเก็บไว้ในหน่วยความจำจากนั้นวงจรจะหยุดสร้ างสัญญาณนาฬิกาจนกว่ามีข้อมูลที่จะต้องเข้ารหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตใหม่จากภาคควอนไตร์ข้อมูลถัดไปพิจารณาแผนผังสถานะของวงจรสร้างสัญญาณนาฬิกา 9 ลูกเมื่อ XI_0C มีระดับลอจิก เป็น 0 ซึ่งในสถานะนี้ คือ มีข้อมูลเข้ามาในวงจรเข้ารหัสฮัฟฟ์แมน ส่วนของวงจรจะสร้างสัญญาณนาฬิกาเป็นจำนวน 10 ลูก โดยสัญญาณนาฬิกาลูกแรกจะทำหน้าที่แทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต โดยการเพิ่ม 0 ข้างหน้าบิตที่มีนัยสำคัญสูงสุด ของข้อมูลที่วงจรตัวนับ 1 ถึง 255 ได้ทำการสร้างข้อมูลแทนข้อมูลที่ได้เข้ามาในวงจรเข้ารหัสฮัฟฟ์แมนใหม่จากนั้นสัญญาณนาฬิกาลูกแรกจะทำหน้าที่อีกอย่าง คือโหลดข้อมูล ที่ทำการแทนรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตจากรหัสฮัฟฟ์แมนแบบขนานให้เป็นรหัสฮัฟฟ์แมนแบบอนุกรม เพราะการออกแบบวงจรจะใช้ดีฟลิป-ฟลอป ซึ่งอาศัยการเซต และรีเซตแบบซิงโครนัสจึงต้องมีสัญญาณนาฬิกามากระตุ้นวงจรให้โหลดข้อมูลพอสัญญาณนาฬิกาลูกที่ 2 ถึงลูกที่ 10 จะเป็นช่วงเวลาการเลื่อนข้อมูลโดยในช่วงนี้จะทำการส่งรหัสฮัฟฟ์แมนแบบอนุกรมโดยเริ่มจากบิตที่มีนัยสำคัญสูงสุดไปจนบิตที่มีนัยสำคัญต่ำสุด พร้อมกันนั้นจะส่งสัญญาณนาฬิกา 9 ลูกที่เหลือพร้อมกับรหัสฮัฟฟ์แมนที่ได้ถูกส่งออกไปโดยสัญญาณนาฬิกา 1 ลูกพร้อมกับรหัสฮัฟฟ์แมน 1 บิตแล้วจะนำรหัสฮัฟฟ์แมนไปเก็บไว้ในหน่วยความจำดังนั้นสัญญาณนาฬิกาลูกแรกที่ได้สร้างขึ้นมาจึงไม่ต้องส่งไปพร้อมกับรหัสฮัฟฟ์แมน

3.8.7 วงจรสร้างสัญญาณนาฬิกา 1 ลูก

วงจรสร้างสัญญาณนาฬิกา 1 ลูกในวงจรเข้ารหัสฮัฟฟ์แมนออกแบบตามฟังก์ชันการทำงาน ดังตารางที่ 3.5, รูปที่ 3.46 และรูปที่ 3.47

Input : $X(t) \in \{0, 1\}$

Output : $Z(t) \in \{0, 1\}$

State : $S(t) \in \{A, B, C\}$

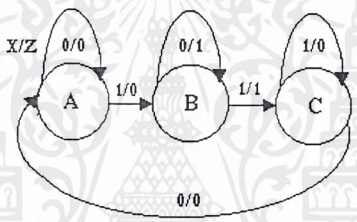
Initial State : $S(0) = A$

Function : ดังตารางที่ 3.5

ตารางที่ 3.5 ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 1 ลูก

Present State		Next State X, Z	
State Name	Binary code	X = 0	X = 1
A	00	B, 0	A, 0
B	01	B, 1	C, 1
C	10	A, 0	C, 0

State Diagram : ดังรูปที่ 3.46



รูปที่ 3.46 แผนผังสถานะของวงจรสร้างสัญญาณนาฬิกา 1 ลูก

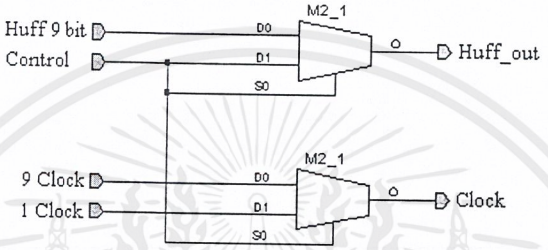
รูปที่ 3.47 วงจรสร้างสัญญาณนาฬิกา 1 ลูก เมื่อแปลงเป็นแมโคร

วงจรสร้างสัญญาณนาฬิกา 1 ลูก เป็นวงจรดิจิตอลแบบลำดับ ซึ่งได้ออกแบบสำหรับส่งสัญญาณนาฬิกาไปพร้อมกับกับรหัสฮัฟฟ์แมนความยาวของรหัส 1 บิตซึ่งแทนข้อมูล S0 หรือไบนารีโค้ด คือ 0000 0000 หลักการออกแบบเมืออินพุต คือ XI_1C มีระดับลอจิกเป็น 1 สถานะ วงจรจะเลื่อนสถานะไปข้างหน้าจนถึงสถานะ C แล้วแล้วค้างสถานะการทำงานที่สถานะ C รอจนกว่าอินพุต XI_1C จะมีระดับลอจิกเป็น 0 ซึ่งในเวลานั้นเป็นช่วงขอบขาขึ้นของสัญญาณนาฬิกาจากภาคควอนไตซ์จะทำให้สถานะของวงจรไปที่ สถานะ A แล้วค้างสถานะจนกว่า XI_1C จะมีระดับ

ลอจิกเป็น 1 ซึ่งจะมีข้อมูล S0 เข้ามาแทนรหัสฮัฟฟ์แมน 1 บิต วงจรจะสร้างสัญญาณนาฬิกา 1 ลูกจะทำงานลักษณะนี้ สัญญาณนาฬิกาจำนวน 1 ลูก ที่สร้างขึ้นมาจะถูกส่งไปพร้อมกับรหัสฮัฟฟ์แมนความยาวของรหัส 1 บิตที่ได้แทนข้อมูล S0

3.8.8 วงจรมัลติเพล็กซ์รหัสฮัฟฟ์แมน และมัลติเพล็กซ์สัญญาณนาฬิกา

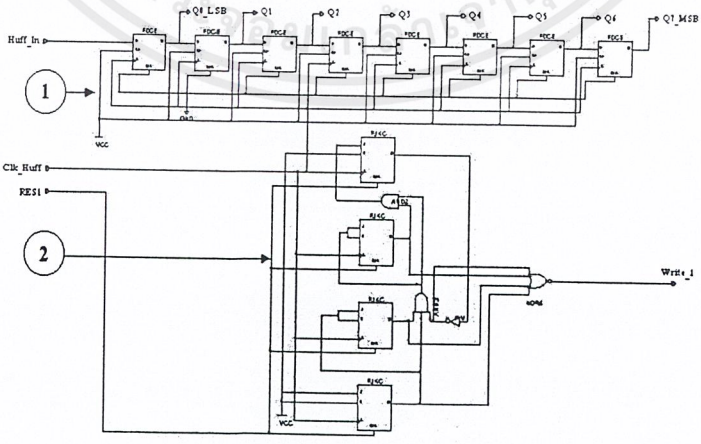
วงจรมัลติเพล็กซ์รหัสฮัฟฟ์แมน และมัลติเพล็กซ์สัญญาณนาฬิกาเป็นวงจรมัลติเพล็กซ์แบบเข้า 2 ออก 1 สามารถเลือกใช้ได้จาก โดยการออกแบบสร้างจะเป็นดังรูป 3.48



รูปที่ 3.48 วงจรมัลติเพล็กซ์รหัสฮัฟฟ์แมน และมัลติเพล็กซ์สัญญาณนาฬิกา

วงจรมัลติเพล็กซ์รหัสฮัฟฟ์แมน และมัลติเพล็กซ์สัญญาณนาฬิกาใช้สำหรับเลือกส่งรหัสฮัฟฟ์แมนที่ทำการแทน S0 ด้วยรหัสฮัฟฟ์แมนความยาวของรหัส 1 บิต หรือส่งรหัสฮัฟฟ์แมนที่ทำการแทน S1 ถึง S255 ด้วยรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต และเลือกส่งสัญญาณนาฬิกาจำนวน 1 ลูก หรือ 9 ลูกตามรหัสฮัฟฟ์แมน

3.8.9 วงจรควบคุมการจัดเก็บรหัสฮัฟฟ์แมนในหน่วยความจำ



รูปที่ 3.49 วงจรควบคุมการจัดเก็บรหัสฮัฟฟ์แมนลงในหน่วยความจำ

จากรูปที่ 3.49 แบ่งวงจรเป็นสองส่วน คือ ส่วนที่ 1 คือ วงจรอนุกรมออกขนานซึ่งใช้หลักการของวงจรเลื่อนข้อมูลวงจรในวงจรส่วนนี้จะรับข้อมูลมาจากส่วนของวงจรมัลติเพล็กซ์รหัสฮัฟฟ์แมน และสัญญาณนาฬิกาที่ใช้เลื่อนข้อมูลไปข้างหน้า 1 ตำแหน่งเมื่อมีสัญญาณนาฬิกา 1 ลูก ส่วนที่ 2 เป็นวงจรตัวนับมอดูลัส 1 ถึง 8 สำหรับนับบิตของส่วนรหัสฮัฟฟ์แมนที่เข้ามาเข้ามาครบ 8 บิต วงจรจะส่งสัญญาณควบคุมหน่วยความจำให้เก็บรหัสฮัฟฟ์แมนลงไปในหน่วยความจำโดยที่วงจรทั้งสองส่วนจะใช้สัญญาณนาฬิการ่วมกัน วงจรมีหลักการทำงานได้ดังต่อไปนี้เริ่มต้นเมื่อได้ทำการ รีเซตระบบให้วงจรเข้ารหัสฮัฟ่ฟ์แมนพร้อมที่จะทำงาน จะทำให้วงจรตัวนับมอดูลัส 1 ถึง 8 มีสถานะอยู่ที่ 0 จะพบว่า รหัสฮัฟฟ์แมนที่แทนข้อมูลที่ได้เข้ามา มีเพียง 2 ความยาวของรหัส คือ รหัสฮัฟฟ์แมนความยาวของรหัส 1 บิตที่แทนข้อมูล S0 และรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต ซึ่งแทนข้อมูล S1 ถึง S255 สมมุติข้อมูลแรกที่เขามาเป็น S0 รหัสฮัฟฟ์แมนที่แทน S0 คือ 1 วงจรสร้างสัญญาณนาฬิกา 1 ลูก และวงจรเข้ารหัสฮัฟฟ์แมน 1 บิต จะทำการส่งข้อมูลมาพร้อมกับสัญญาณนาฬิกา 1 ลูก ทำให้ วงจรตัวนับมอดูลัส 1 ถึง 8 นับไป จาก 0 ไปเป็น 1 พร้อมกับวงจรเข้าอนุกรมออกขนานจะเลื่อนข้อมูลไป 1 ช่วงเวลานี้ Q0_LSB จะเก็บค่า 1 ซึ่งเป็นรหัสฮัฟฟ์แมนของ S0 เอาไว้ ต่อมาสมมุติ ข้อมูลที่ 2 ที่เข้ามา คือ S1 รหัสฮัฟฟ์แมนแทน S1 คือ 0 0000 0001 จะพบว่าเป็นรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต วงจรเข้ารหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต และวงจรสร้างสัญญาณนาฬิกาขนาด 9 ลูก จะทำการส่งรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตโดยเริ่มจากบิตที่มีนัยสำคัญสูงสุด ของโค้ดรหัสฮัฟฟ์แมนพร้อมกับส่งสัญญาณนาฬิกา 9 ลูกผ่านวงจรมัลติเพล็กซ์ เข้ามาในวงจรควบคุมการจัดเก็บรหัสฮัฟฟ์แมนลงในหน่วยความจำ สัญญาณนาฬิกาลูกที่ 1 จะทำให้บิตสูงของรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตเลื่อนมาที่ Q0 LSB ส่วนของรหัสฮัฟฟ์แมนของ S0 คือ 1 ซึ่งค้างสถานะอยู่ที่ Q0_LSB ก่อนหน้านี้จะถูกเลื่อนไปอีก คือ อยู่ที่ Q1 และในเวลาเดียวกัน วงจรตัวนับมอดูลัส 1 ถึง 8 จะนับไปข้างหน้าอีก 1 ตามสัญญาณนาฬิกา คือ นับ 2 สำหรับสัญญาณนาฬิกาลูกที่ 2, 3, 4, 5, 6 จะทำให้ข้อมูลจะเลื่อนไปลักษณะเดียวกันตามเป็นคุณสมบัติของวงจรเลื่อนข้อมูลพร้อมวงจรตัวนับมอดูลัส1 ถึง 8 นับเป็น 3, 4, 5, 6, 7 ตามลำดับสัญญาณนาฬิกาที่ส่งเข้ามา เมื่อสัญญาณนาฬิกาลูกที่ 7 จะพบว่า วงจรตัวนับมอดูลัส 1 ถึง 8 จะนับเป็น 8 เป็นตัวบ่งชี้ว่าส่วนของรหัส ฮัฟฟ์แมนที่เข้ามาครบ 8 บิต จะส่งสัญญาณเขียนข้อมูลลงไปในหน่วยความจำขนาด 8 บิต 128 กิโลไบต์ เมื่อเขียนข้อมูลลงในหน่วยความจำเสร็จ 1 ตำแหน่งให้เพิ่มตำแหน่งของหน่วยความจำไป 1 ตำแหน่ง ส่วนสัญญาณนาฬิกาที่เหลืออีก 2 ลูกจะทำให้ส่วนของรหัสฮัฟฟ์แมนที่แทน S1 บิตรองต่ำสุด ไปเก็บอยู่ที่ Q1 และ ส่วนของรหัสฮัฟฟ์แมนที่แทน S1 บิตที่มีนัยสำคัญต่ำสุด เก็บอยู่ที่ Q0_LSB และในเวลาเดียวกัน วงจรตัวนับมอดูลัส 1 ถึง 8 จะนับ จะเริ่มนับ 1 ใหม่ที่สัญญาณนาฬิกาลูกที่ 8 และนับ 2 ที่สัญญาณนาฬิกาลูกที่ 9 จะค้างสถานะ

อยู่ที่ 2 รอจนกว่ารหัสฮัฟฟ์แมนชุดใหม่ที่เข้ามาใหม่พร้อมสัญญาณนาฬิกาที่ส่งมาจากวงจรสร้างสัญญาณนาฬิกา โดยมีวงจรตัวนับมอดูลัส 1 ถึง 8 ควบคุมการเขียนส่วนของรหัสฮัฟฟ์แมนลงในหน่วยความจำเมื่อมีส่วนของรหัสฮัฟฟ์แมนครบ 8 บิต เมื่อวงจรตัวนับมอดูลัสนับครบ 8 ในรอบใหม่ แล้วควบคุมให้เพิ่มตำแหน่งของหน่วยความจำไป 1 ตำแหน่ง วงจรจะทำงานลักษณะนี้จนข้อมูลที่เก็บในหน่วยความจำเต็มตำแหน่ง 1FFFFH หรือ ครบ ตำแหน่งของหน่วยความจำขนาด 8 บิต 128 กิโลไบต์

3.9 วงจรถอดรหัสฮัฟฟ์แมน

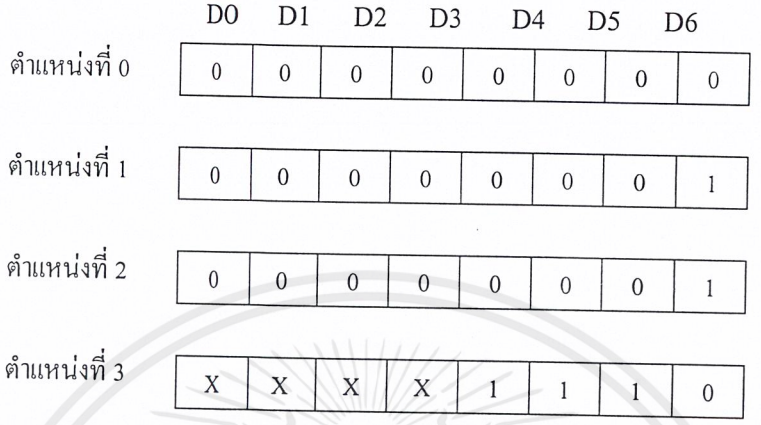
รหัสฮัฟฟ์แมนที่จัดเก็บไว้ในหน่วยความจำโดยรหัสฮัฟฟ์แมนจะอนุกรมกันอยู่ในรูปแบบของบิตที่มีนัยสำคัญต่ำสุดของข้อมูลที่ตำแหน่ง ตำแหน่งที่ 0 ของหน่วยความจำ ต่ออนุกรมกับบิตที่มีนัยสำคัญสูงสุด ของตำแหน่งที่ 1 และบิตที่มีนัยสำคัญต่ำสุดของข้อมูลที่ตำแหน่งของตำแหน่งที่ 1 ต่ออนุกรมกับบิตที่มีนัยสำคัญสูงสุด ของตำแหน่งที่ 2 ไปเรื่อย จนตำแหน่งสุดท้ายของหน่วยความจำ สมมุติ A, B, C, D เป็นรหัสฮัฟฟ์แมนที่แทนข้อมูลต่างๆ โดย A, B และ C เป็นรหัสฮัฟฟ์แมนที่มีความยาว 9 บิต ส่วน D เป็นรหัสฮัฟฟ์แมนที่มีความยาวของรหัส 1 บิต ลักษณะรหัสฮัฟฟ์แมนจะเรียงตัวกันอยู่ดังรูปในหน่วยความจำขนาด 8 บิต จะเป็นไปดังต่อไปนี้

	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
ตำแหน่งที่ 0	A1	A2	A3	A4	A5	A6	A7	A8
ตำแหน่งที่ 1	B2	B3	B4	B5	B6	B7	B8	A0
ตำแหน่งที่ 2	C3	C4	C5	C6	C7	C8	B0	B1
ตำแหน่งที่ 3	X	X	X	X	D	C0	C1	C2

รูปที่ 3.50 ตัวอย่างรหัสฮัฟฟ์แมนที่แทนข้อมูลในหน่วยความจำขนาด 8 บิต

จากรูปที่ 3.50 ถ้าสมมุติให้ข้อมูล A, B, C และ เป็นรหัสฮัฟฟ์แมนโดยที่ A แทน S1 ซึ่งรหัสฮัฟฟ์แมน คือ 0 0000 0001 B แทน S2 ซึ่งรหัสฮัฟฟ์แมน คือ 0 0000 0010 C แทน S3 ซึ่ง

รหัสฮัฟฟ์แมน คือ 0 0000 0011 และ D แทน S0 รหัสฮัฟฟ์แมน คือ 1 เมื่อทำการแทนบิตของรหัสฮัฟฟ์แมนแทน A, B, C และ D การจัดเก็บรหัสฮัฟฟ์แมนในหน่วยความจำจะเป็นดังต่อไปนี้



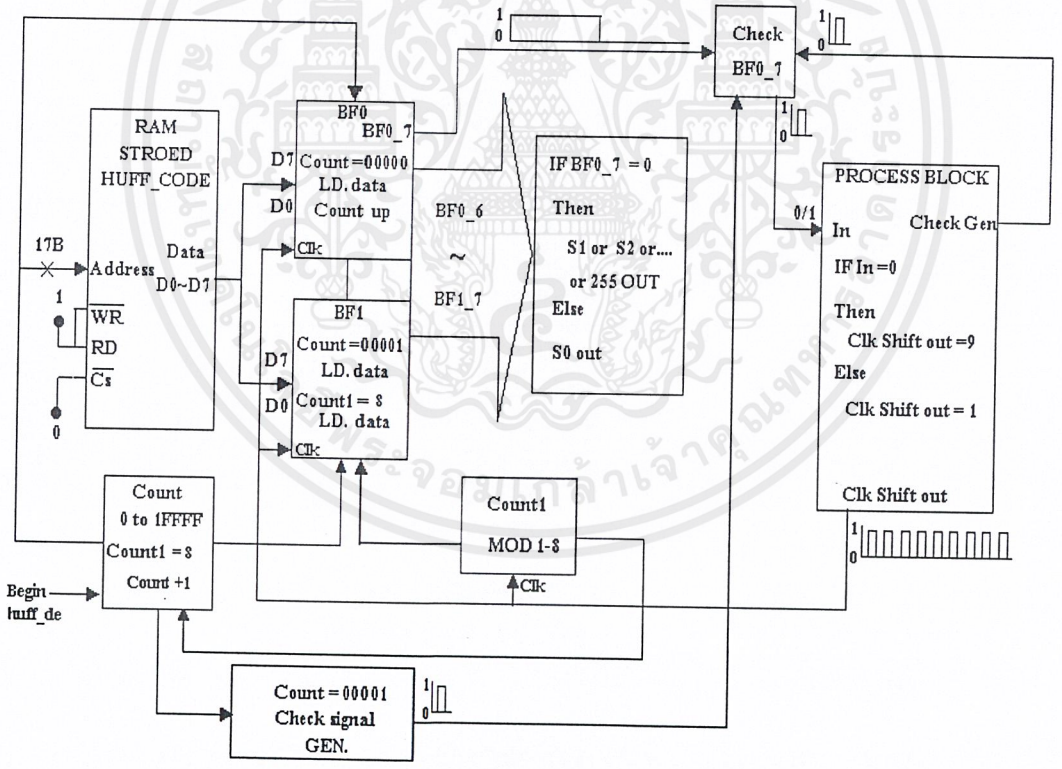
รูปที่ 3.51 ตัวอย่างรหัสฮัฟฟ์แมนที่จัดเก็บในหน่วยความจำขนาด 8 บิต

ตามหลักการถอดรหัสฮัฟฟ์แมนที่ได้กล่าวไปแล้วข้างต้นในบทที่ 2 กรณีที่ความยาวของโค้ดของรหัสฮัฟฟ์แมนที่แทนข้อมูลที่เข้ามาในวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนมีความยาวหลายระดับตามโครงสร้างข้อมูลแบบฮัฟฟ์แมนทรี หลักการถอดรหัสฮัฟฟ์แมนจะใช้หลักการตรวจสอบบิต ที่เป็น 0 จากบิตที่มีนัยสำคัญสูงสุดของรหัสฮัฟฟ์แมนเมื่อได้จำนวนบิตที่เป็น 0 ที่อยู่หน้าโค้ดแล้วจะทำการถอดรหัสฮัฟฟ์แมนออกมาเป็นข้อมูลตามที่เข้ามาในวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน การถอดรหัสฮัฟฟ์แมนที่ได้ออกแบบนี้จะพบว่ามีความยาวของรหัสบิตเพียง 2 ระดับ คือ ความยาวของรหัส 1 บิต และ 9 บิต จะพบว่าข้อมูล S0 ไบนารีโค้ด คือ 0000 0000 แทนรหัสฮัฟฟ์แมนด้วย 1 ส่วน ข้อมูล S1 ถึง S255 ซึ่งมีความยาวของรหัส 9 บิตจะพบว่ารหัสฮัฟฟ์แมนจะแตกต่างกับไบนารีโค้ด คือ รหัสฮัฟฟ์แมนจะเป็นลักษณะของไบนารีโค้ดที่เติม 0 ถัดไปจากบิตที่มีนัยสำคัญสูงสุด เช่น S1 ไบนารีโค้ด คือ 0000 0001 รหัสฮัฟฟ์แมน คือ 0 0000 0001 S2 ไบนารีโค้ด คือ 0000 0010 รหัสฮัฟฟ์แมน คือ 0 0000 0010 S255 ไบนารีโค้ด คือ 1111 1111 รหัสฮัฟฟ์แมน คือ 0 1111 1111 เป็นต้น ดังนั้นการถอดรหัสจะตรวจสอบจะใช้หลักการดังกล่าวข้างต้น คือ การตรวจสอบบิตแรกซึ่งเป็นบิตที่มีนัยสำคัญสูงสุดของรหัสฮัฟฟ์แมนเป็นระดับลอจิกเป็น 1 หรือ 0 กรณีที่เป็นระดับลอจิก 1 จะหมายถึงรหัสฮัฟฟ์แมนแทน S0 จากนั้นทำการเลื่อนโค้ดไป 1 ตำแหน่งแล้วทำการตรวจสอบรหัสฮัฟฟ์แมนใหม่ถ้าเป็นระดับลอจิก 1 จะหมายถึงรหัสฮัฟฟ์แมนแทน S0 แล้วทำการเลื่อนข้อมูล 1 ตำแหน่ง แล้วทำการตรวจสอบรหัสฮัฟฟ์แมนใหม่ ถ้าระดับลอจิกเป็น 0 แสดงว่า ตั้งแต่บิตที่ 7 รองจากบิตที่มีนัยสำคัญสูงสุดของรหัสฮัฟฟ์แมนจนถึงบิต

ที่นัยสำคัญต่ำสุดเป็นตัวบ่งบอกว่ารหัสฮัฟฟ์แมนแทนข้อมูลใดๆ เช่น 0 0000 00001 เมื่อได้ทำการตรวจสอบบิตที่ 9 จะพบระดับลอจิก 0 แสดงว่าข้อมูลที่ได้ทำการแทนรหัสฮัฟฟ์แมน คือ 0000 0001 หรือ S1 เมื่อได้ข้อมูลแล้วเลื่อนรหัสฮัฟฟ์แมนไปอีก 9 ตำแหน่งหลังจากนั้นถอดรหัสฮัฟฟ์แมนเป็นข้อมูลใหม่ซึ่งการถอดรหัสฮัฟฟ์แมนจะกระทำไปกระบวนการถอดรหัสจนครบทุกตำแหน่งของหน่วยความจำที่ได้จัดเก็บรหัสฮัฟฟ์แมนไว้ การออกแบบวงจรถอดรหัสฮัฟฟ์แมนจะมีกลไกของวงจรย่อยหลายๆ ส่วนที่แตกต่างกันไปรวมกันเป็นวงจรถอดรหัสฮัฟฟ์แมน

3.9.1 การออกแบบวงจรถอดรหัสฮัฟฟ์แมน

การออกแบบวงจรถอดรหัสฮัฟฟ์แมนจะสร้างแต่ละส่วนย่อยของวงจรเป็นแมโครในลักษณะเดียวกันกับวงจรเข้ารหัสฮัฟฟ์แมน วงจรที่ได้ออกแบบในวงจรเข้ารหัสฮัฟฟ์แมนในบางส่วนสามารถนำมาใช้ออกแบบวงจรถอดรหัสฮัฟฟ์แมน การออกแบบจะใช้หลักการถอดรหัสที่ได้กล่าวมาแล้วข้างต้นซึ่งจากหลักการดังกล่าวสามารถเขียนเป็นแผนผังการทำงานได้ ดังรูปที่ 3.52



รูปที่ 3.52 แผนผังการทำงานของวงจรถอดรหัสฮัฟฟ์แมน

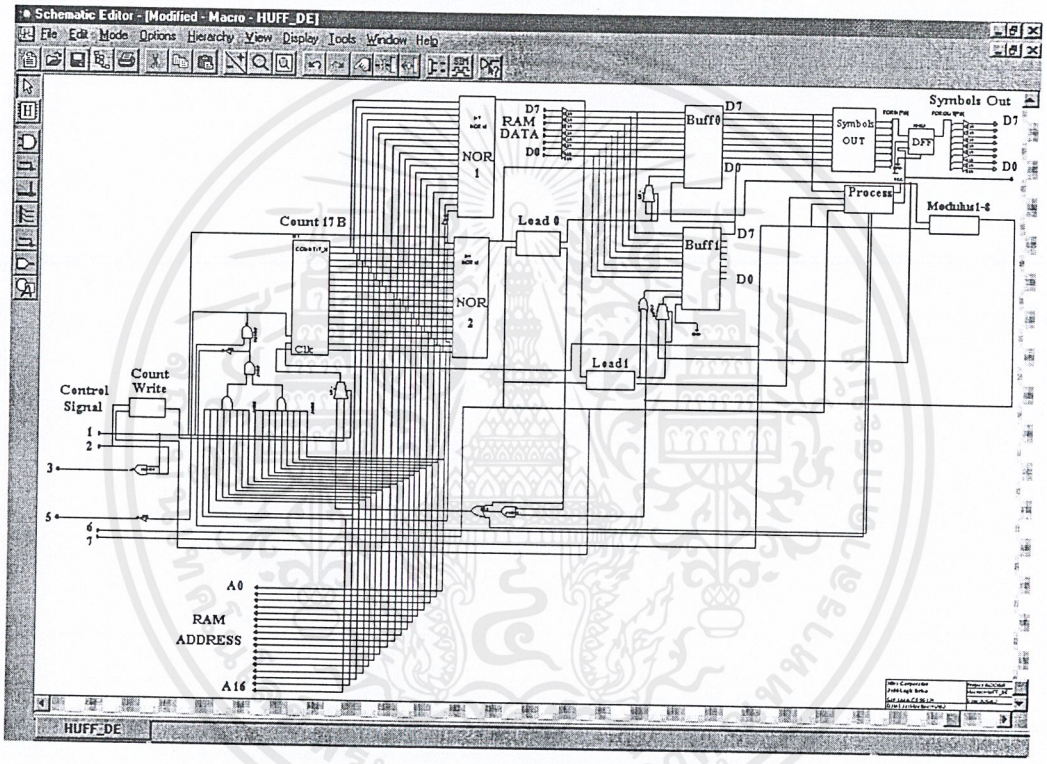
จากรูปที่ 3.52 แผนผังการทำงานของวงจรถอดรหัสฮัฟฟ์แมน หลักการทำงานมีดังนี้ คือ เมื่อวงจรเข้ารหัสฮัฟฟ์แมนได้จัดเก็บรหัสฮัฟฟ์แมนในหน่วยความจำจนครบทุกตำแหน่ง

ของหน่วยความจำเรียบร้อยแล้ว การถอดรหัสฮัฟฟ์แมนสำหรับคืนค่าข้อมูลที่เข้ามาในวงจรเข้ารหัสและถอดรหัสฮัฟฟ์แมนให้เป็นข้อมูลตามลำดับที่เข้ามาก่อนหลังได้ดังนี้การถอดรหัสฮัฟฟ์แมนตอนขั้นแรกสัญญาณที่ขาควบคุมหน่วยความจำจะเป็นดังนี้ สัญญาณอ่าน (RD) เป็นระดับลอจิก 1 สัญญาณเขียน (WR) มีระดับลอจิกเป็น1 และสัญญาณเลือกหน่วยความจำ Cs เป็นระดับลอจิก 0 เพื่อถอดรหัสฮัฟฟ์แมนวนซ้ำ การทำงานเมื่อมีสัญญาณนาฬิกากระตุ้นวงจรตัวนับ 0 ถึง 1FFFFH (Count) ซึ่งเป็นวงจรตัวนับชี้ตำแหน่งของหน่วยความจำชี้ที่ตำแหน่งแรกของหน่วยความจำ คือ 00000H จากตำแหน่งดังกล่าวจะโหลดข้อมูลจากตำแหน่ง 00000H ลงไปในบัฟเฟอร์ 0 (BF0) หลังจากโหลดที่ได้โหลดข้อมูลจากตำแหน่ง 00000H ลงไปใบบัฟเฟอร์ 0 ให้นับวงจรตัวนับ 0 ถึง 1FFFFH (Count) ชี้ตำแหน่งของหน่วยความจำเพิ่มไปอีก 1 ตำแหน่ง คือ ไปชี้อยู่ที่ตำแหน่ง 00001H แล้วโหลดข้อมูลลงจากตำแหน่ง 00001H ไปในบัฟเฟอร์ 1 (BF1) จากนั้นส่วนควบคุมย่อยจะสร้างสัญญาณถอดรหัสฮัฟฟ์แมนข้อมูลแรก โดยตรวจสอบบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 0 (BF0_7) จะได้ข้อมูลจากการถอดรหัสฮัฟฟ์แมนตัวแรก สัญญาณถอดรหัสฮัฟฟ์แมนข้อมูลแรกจะนำไปตรวจสอบระดับลอจิกของบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 0 (BF0_7) เป็น 0 หรือ 1 กรณีที่ระดับลอจิกเป็น 1 คือ รหัสฮัฟฟ์แมนที่แทนเป็นข้อมูล S0 ไบนารีโค้ด คือ 0000 0000 หลังจากที่ได้ ข้อมูลแรกส่วนประมวลผล (Process Block) ควบคุมบัฟเฟอร์ 0 และบัฟเฟอร์ 1 เลื่อนข้อมูลไป 1 ตำแหน่งแล้วสร้างสัญญาณถอดรหัสฮัฟฟ์แมนไปตรวจสอบระดับลอจิกบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 0 (BF0_7) ใหม่หลังจากที่เลื่อนรหัสฮัฟฟ์แมนที่ได้ทำการถอดรหัสไปแล้ว และในกรณีที่ระดับลอจิกเป็น 0 หมายถึง ข้อมูล คือ บิตรองนัยสำคัญสูงสุดของบัฟเฟอร์ 0 ไปจนถึงบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 1 (BF0_6 – BF1_7) เป็นข้อมูลที่ถอดรหัสฮัฟฟ์แมนข้อมูลนี้ ซึ่งบิตรองนัยสำคัญสูงสุดของบัฟเฟอร์ 0 (BF0_6) จะเป็นบิตที่มีนัยสำคัญสูงสุด ของข้อมูลนี้ และบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 1 (BF1_7) เป็นบิตที่มีนัยสำคัญต่ำสุดของข้อมูลนี้ หลังจากที่ได้ข้อมูลนี้แล้วส่วนประมวลผล (Process Block) ควบคุมให้บัฟเฟอร์ 0 (BF0) และบัฟเฟอร์ 1 (BF1) เลื่อนรหัสฮัฟฟ์แมนไป 9 ครั้ง แล้วสร้างสัญญาณถอดรหัสฮัฟฟ์แมนไปตรวจสอบระดับลอจิกบิตที่มีนัยสำคัญสูงสุด ของ บัฟเฟอร์0 (BF0_7) ใหม่ ในส่วนของวงจรมอดูลัส 1 ถึง 8 จะเป็นตัวตรวจสอบการเลื่อนรหัสฮัฟฟ์แมนที่บัฟเฟอร์ 0 และ บัฟเฟอร์ 1 ครบ 8 ครั้งแล้วให้วงจรตัวนับ 0 ถึง 1FFFFH เลื่อนชี้ตำแหน่งของหน่วยความจำเพิ่มไป 1 ตำแหน่งแล้วโหลดรหัสฮัฟฟ์แมนจากตำแหน่งดังกล่าวลงไปในบัฟเฟอร์ 1 (BF1) ใหม่เนื่องจากรหัสฮัฟฟ์แมนที่อยู่ในบัฟเฟอร์ 1 (BF1) ได้เลื่อนไปที่บัฟเฟอร์ 0 (BF0) เรียบร้อยแล้ว หลังจากที่โหลดส่วนของรหัสฮัฟฟ์แมนลงไปในบัฟเฟอร์ 1 ใหม่ให้ส่วนของรหัสฮัฟฟ์แมนชุดใหม่เลื่อนส่วนของรหัสฮัฟฟ์แมนไปพร้อมกับ รหัสฮัฟฟ์แมนที่มีอยู่ในบัฟเฟอร์ 0 และการทำงานของวงจรถอดรหัสฮัฟฟ์แมนจะทำงานในลักษณะนี้ไป

จน ตำแหน่งสุดท้ายของหน่วยความจำ คือ ตำแหน่ง ที่ 1FFFFH วงจรจะถอดรหัสซ้ำเริ่มจากตำแหน่ง 00000H ไปจนตำแหน่ง 1FFFFH อีกครั้ง

3.9.2 การออกแบบวงจรถอดรหัสฮัฟฟ์แมนบน Schematic Editor

วงจรถอดรหัสฮัฟฟ์แมนออกแบบโดยใช้ Schematics Editor วงจรในบางส่วนของวงจรจะใช้วงจรที่ได้ออกแบบไว้ในส่วนของวงจรเข้ารหัสฮัฟฟ์แมนซึ่งวงจรโดยรวมที่ได้ออกแบบบน Schematics Editor ดังรูปที่ 3.53



รูปที่ 3.53 ถอดรหัสฮัฟฟ์แมนบน Schematic Editor

สำหรับวงจรถอดรหัสฮัฟฟ์แมนตามรูปที่ 3.53 ได้ทำการออกแบบวงจรตามหลักการที่ได้อธิบายการงานตามแผนผังการทำงาน และได้รวมเอาวงจรส่วนควบคุมเอาไว้ด้วย ส่วนควบคุมนี้จะทำหน้าที่ให้วงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนทั้งหมดให้เข้ารหัสฮัฟฟ์แมนแทนข้อมูลไปเก็บไว้หน่วยความจำ หรือถอดรหัสฮัฟฟ์แมนที่ได้จัดเก็บไว้ในหน่วยความจำออกมาเป็นข้อมูลสู่เอาต์พุตของ วงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน โดยสัญญาณควบคุม (Control Signal) ทั้งหมดจะมีดังนี้

1) เป็นสัญญาณอินพุตควบคุมจากภายนอก โดยสัญญาณมีระดับลอจิกเป็น 1 จะเลือกให้เข้ารหัสฮัฟฟ์แมนทำงานจะทำการเข้ารหัส หรือนำรหัสฮัฟฟ์แมนที่ได้ทำการแทนข้อมูลไปเก็บไว้

ในหน่วยความจำ ระดับลอจิกเป็น 0 ควบคุมให้วงจรถอดรหัสฮัฟฟ์แมนถอดรหัสฮัฟฟ์แมนที่ได้จัดเก็บในหน่วยความจำไปเป็นข้อมูลตามลำดับของข้อมูลที่เข้ามา

2) เป็นสัญญาณอินพุตที่ใช้ควบคุมการเขียนข้อมูลลงไปในหน่วยความจำซึ่งมาจากวงจรเข้ารหัสฮัฟฟ์แมนโดยจะทำการเขียนข้อมูลลงไปในหน่วยความจำ เมื่อระดับสัญญาณจากวงจรเข้ารหัสฮัฟฟ์แมน เป็นลอจิก 1 และระดับสัญญาณในข้อ 1 เป็นลอจิก 0

3) เป็นสัญญาณเอาต์พุตที่นำไปต่อกับขา ($\overline{WR}$) ของหน่วยความจำหลักที่เก็บรหัสฮัฟฟ์แมน

4) เป็นสัญญาณเอาต์พุตเลือกหน่วยความจำ ($\overline{Cs}$) โดยจะเป็นวงจรควบคุมไม่ให้เกิดการเขียนทับข้อมูลตำแหน่งสุดท้าย

5) เป็นสัญญาณนาฬิกาที่ป้อนให้กับวงจรถอดรหัสฮัฟฟ์แมน

6) เป็นอินพุตสัญญาณรีเซตสำหรับให้วงจรถอดรหัสฮัฟฟ์แมนพร้อมทำงาน

นอกจากนี้ยังมีขาแอดเดรส (A0~A16) และข้อมูล (D0~D7) ที่จะต้องนำไปต่อกับแอดเดรสและข้อมูลของหน่วยความจำ การทำงานของวงจรเกิดจากการทำงานอย่างสัมพันธ์กันระหว่างวงจรส่วนย่อยที่ได้สร้างแมโคร แล้วลิงค์เข้ามาสู่หน้าต่าง Schematics Editor ของวงจรถอดรหัสฮัฟฟ์แมน ซึ่งแต่ละวงจรได้มีการออกแบบสำหรับให้วงจรทำงานเป็นส่วนย่อยแต่ละส่วนดังนี้

3.9.3 วงจรตัวนับ 00000H ถึง 1FFFFH

การออกแบบวงจรตัวนับ 00000H ถึง 1FFFFH เป็นวงจรตัวนับเป็นเลขฐานสิบที่ใช้เพื่อความสะดวกในการออกแบบ คือ นับ 0 ถึง 131071 สำหรับชี้ตำแหน่งในหน่วยความจำสำหรับการจัดเก็บรหัสฮัฟฟ์แมนในหน่อยความจำ หรือ การถอดรหัสฮัฟฟ์แมนซึ่งมีส่วนของรหัสฮัฟฟ์แมนอยู่ในแต่ละตำแหน่งในการออกแบบวงจรตัวนับจะทำการออกแบบวงจรตัวนับ 0 ถึง 255 แล้วนำมาต่อรวมกันเป็นวงจรตัวนับ 0 ถึง 131071 การออกแบบวงจรตัวนับ 0 ถึง 255 ออกแบบตามฟังก์ชันการทำงานดังรูปที่ 3.54 ถึง 3.56

Input : $X(t) \in \{0, 1\}$

Output : $Z(t) \in \{0, 1\}$

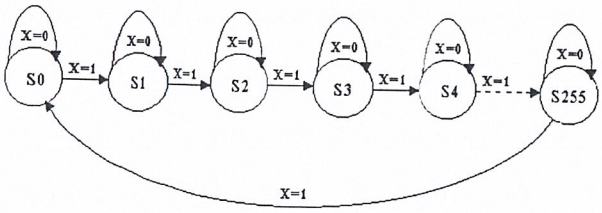
State : $S(t) \in \{S0, S1, S2, S3, S4, \ldots, S255\}$

Initial State : $S(0) = \{S0\}$

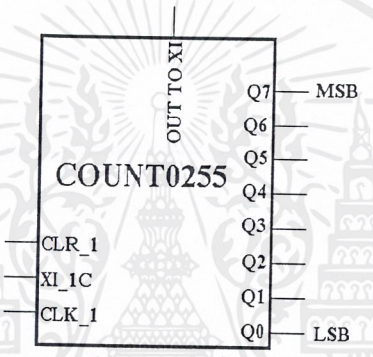
Function : Counts Modulus 0 to 255 i.e., (0, 1, 2, 3, 4, …, 255, 0, 0, 1, 2, 3, 4, …, 255, 0…)

If X = 1 Stop Counts

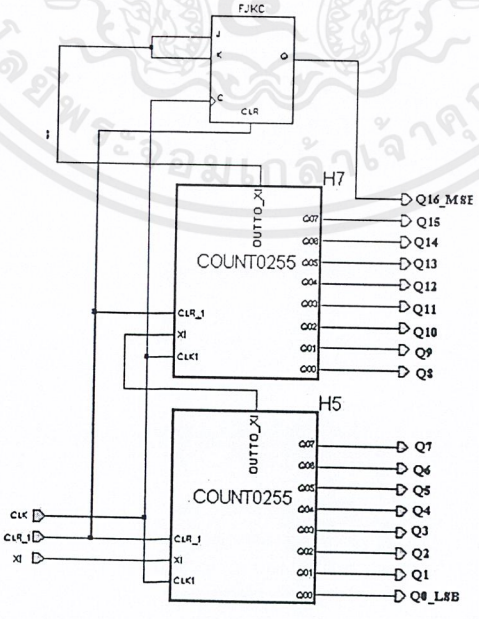
State Diagram : ดังรูปที่ 3.54



รูปที่ 3.54 แผนผังสถานะของวงจรตัวนับ 0 ถึง 255



รูปที่ 3.55 วงจรตัวนับ 0 ถึง 255 เมื่อแปลงเป็นแมโคร



รูปที่ 3.56 วงจรตัวนับ 00000H ถึง 1FFFFH ที่สร้างจากการวงจรตัวนับ 0 ถึง 255

3.9.4 วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0

การออกแบบวงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0 เป็นการออกแบบวงจรดิจิตอลแบบลำดับ ออกแบบวงจรตามฟังก์ชันการทำงานดังรูปที่ 3.57 และ 3.58

Input : $X(t) \in \{0, 1\}$

Output : $Z(t) = \{Z1, Z0\}, Z(t) \in \{0, 1\}$

State : $S(t) \in \{A, B, C, D\}$

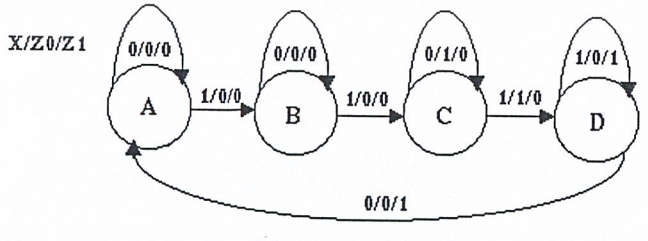
Initial State : $S(0) = \{A\}$

Function : ดังตารางที่ 3.6

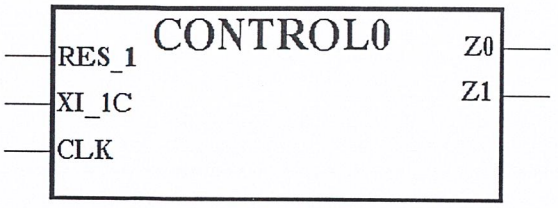
ตารางที่ 3.6 ฟังก์ชันการทำงานของวงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0

Present state		Next state X, Z0, Z1	
State Name	Binary Code	X = 0	X = 1
A	00	A, 0, 0	B, 0, 0
B	01	B, 0, 0	C, 0, 0
C	10	C, 1, 0	D, 1, 0
D	11	A, 0, 1	D, 0, 1

State Diagram : ดังรูปที่ 3.57



รูปที่ 3.57 แผนผังสถานะวงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0



รูปที่ 3.58 วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0

วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0 เป็นวงจรแบบลำดับที่มีอินพุต XI_1C มาจากการกระทำทางลอจิกนอร์ของตำแหน่งของหน่วยความจำที่ 00000H การออกแบบสร้างวงจรแบบลำดับนี้ขึ้นสำหรับนำ Z0 ซึ่งเป็นเอาต์พุตที่ 1 ของวงจรไปทำการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0 อีกต่อมาได้นำ Z1 ซึ่งเป็นเอาต์พุตที่ 2 ไปเป็นสัญญาณนาฬิกาของวงจรตัวนับ 00000H ถึง 1FFFFH ให้วงจรตัวนับไปข้างหน้าอีก 1 ตำแหน่ง เพื่อชี้ตำแหน่งหน่วยความจำ คือ 00001H

3.9.5 วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00001H ถึง 1FFFFH ลงในบัฟเฟอร์ 1

สำหรับวงจรนี้มีหน้าที่สำคัญอีกอย่าง คือ เป็นวงจรสร้างสัญญาณถอดรหัสฮัฟฟ์แมนเป็นข้อมูลตัวแรก การออกแบบวงจรตามฟังก์ชันการทำงานดังรูปที่ 3.59

Input : $X(t) \in \{0, 1\}$

Output : $Z(t) = \{Z0, Z1\}, Z(t) \in \{0, 1\}$

State : $S(t) \in \{A, B, C, D, E\}$

Initial State : $S(0) = \{A\}$

Function : ดังตารางที่ 3.7

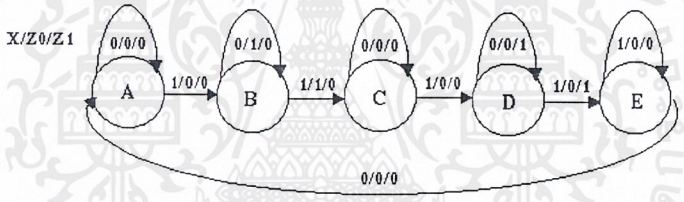
ตารางที่ 3.7 ฟังก์ชันการทำงานของวงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00001H ลงในบัฟเฟอร์ 1

Present State		Next State X, Z0, Z1	
State Name	Binary Code	X = 0	X = 1
A	000	A, 0, 0	B, 0, 0

ตารางที่ 3.7 (ต่อ) ฟังก์ชันการทำงานของวงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00001H ลงในบัฟเฟอร์ 1

Present State		Next State X, Z0, Z1	
State Name	Binary Code	X = 0	X = 1
B	001	B, 1, 0	C, 1, 0
C	010	C, 0, 0	D, 0, 0
D	011	D, 0, 1	D, 0, 1
E	100	A, 0, 0	E, 0, 0

State Diagram : ดังรูปที่ 3.59



รูปที่ 3.59 แผนผังสถานะวงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00001H ลงในบัฟเฟอร์ 1

รูปที่ 3.60 วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00001H ลงในบัฟเฟอร์ 1 เมื่อแปลงเป็นแมโคร

วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00001H ลงในบัฟเฟอร์ 1 เป็นวงจรแบบลำดับที่มีอินพุต XI_1C มาจากการกระทำทางลอจิกนอร์ของตำแหน่งของหน่วยความจำที่ 00001H หรือ

ตำแหน่งที่ 2 ของหน่วยความจำ ออกแบบวงจรสำหรับนำ Z0 ซึ่งเป็นเอาต์พุตที่ 1 ของวงจร ไปทำการโหลด ข้อมูลตำแหน่งที่ 00001H ลงในบัฟเฟอร์ 1 และช่วงเวลาอีกต่อมาได้นำ Z1 ซึ่งเป็นเอาต์พุตที่ 2 ไปเป็นสัญญาณตรวจสอบรหัสฮัฟฟ์แมนครั้งแรกเพื่อถอดรหัสฮัฟฟ์แมนได้ข้อมูลแรก ซึ่งในกระบวนการนี้จะได้ข้อมูลแรกสุดในกระบวนการถอดรหัสฮัฟฟ์แมน

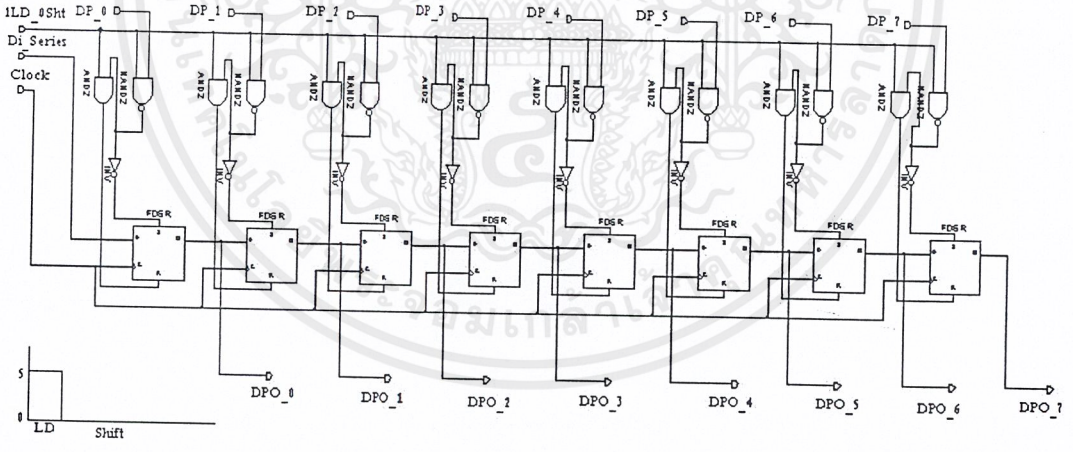
3.9.6 วงจรบัฟเฟอร์พักรหัสฮัฟฟ์แมนจากหน่วยความจำ

ในที่นี้เรียกสั้นๆ ว่าบัฟเฟอร์ บัฟเฟอร์เป็นวงจรที่มีหน้าที่พักรหัสฮัฟฟ์แมนสำหรับให้วงจรส่วนของการประมวลผลทำการถอดรหัสฮัฟฟ์แมนออกไปเป็นข้อมูลที่เข้ามาในวงจรเข้ารหัสและถอดรหัสฮัฟฟ์แมนบัฟเฟอร์นี้จะต้องมีคุณสมบัติดังนี้

1) เป็นวงจรเข้าขนานออกอนุกรมเข้าขนานออกขนานพร้อมจะโหลดรหัสฮัฟฟ์แมนจากหน่วยความจำ

2) เป็นวงจรเข้าอนุกรมออกขนาน และเข้าอนุกรมออกอนุกรมสำหรับนำเลื่อนรหัสจากบัฟเฟอร์ชุดถัดมาได้รวมทั้งมีประโยชน์ในการถอดรหัสฮัฟฟ์แมน

3) เป็นวงจรเลื่อนข้อมูลสำหรับใช้ในการเลื่อนรหัสฮัฟฟ์แมนที่ทำการถอดรหัสแล้ว ซึ่งจากคณสมบัติดังกล่าว สามารถออกแบบเป็นวงจรได้ ดังรปที่ 3.61



รูปที่ 3.61 วงจรบัฟเฟอร์พักรหัสฮัฟฟ์แมนจากหน่วยความจำ

บัฟเฟอร์พักรหัสต้องมีการโหลดข้อมูลลงจากหน่วยความจำเสมอเมื่อการเลื่อนรหัสฮัฟฟ์แมนที่ได้ทำการถอดรหัสครบ 8 ตำแหน่งการเลื่อนรหัสในบัฟเฟอร์ครบ 8 ตำแน่ง หมายถึงส่วนของรหัสฮัฟฟ์แมนในบัฟเฟอร์ 1 ไปที่บัฟเฟอร์ 0 หมดทุกส่วนของรหัสแล้วจำเป็นต้องโหลดข้อมูลที่ตำแหน่งหน่วยความจำตำแหน่งถัดไปลงในบัฟเฟอร์ 1 การโหลดลงในบัฟเฟอร์ 1 จำเป็นจะต้องเข้าจังหวะการเลื่อนข้อมูลไปพร้อมกับรหัสฮัฟฟ์แมนที่ได้ถอดรหัสเป็นข้อมูล การใช้จำนวนชุด

ของบัฟเฟอร์ขึ้นอยู่กับจำนวนรหัสฮัฟฟ์แมนที่มีความยาวของรหัสมากที่สุดบวกด้วยจำนวนของบัสข้อมูลของหน่วยความจำลบด้วย 1 แล้วนำค่าที่ได้ไปหารด้วยจำนวนของบัสข้อมูลของหน่วยความจำ และในกรณีที่มีเศษจากการหารให้เพิ่มจำนวนของบัฟเฟอร์ไปอีกหนึ่งชุด

บัฟเฟอร์ที่ได้กล่าวไปแล้ว คือ บัฟเฟอร์ 0 บิตที่มีนัยสำคัญสูงสุดของบัฟเฟอร์จะเป็นตัวสำคัญในการถอดรหัสฮัฟฟ์แมนในแต่ละครั้ง คือ จะเป็นตัวบ่งบอกถึงการนำข้อมูลออกสู่เอาต์พุตของวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนซึ่งในที่นี้เป็นการถอดรหัสฮัฟฟ์แมนความยาวของรหัส 1 บิต และ 9 บิตดังที่ได้กล่าวไปแล้ว หลักการ คือ การถอดรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิตจะนำเอาบิตรองนัยสำคัญสูงสุดของบัฟเฟอร์ 0 เป็นบิตที่มีนัยสำคัญสูงสุดของข้อมูล เรียงลำดับลงไปจนถึงบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 1 ส่วนการถอดรหัสฮัฟฟ์แมนขนาด 1 บิตจะนำเอาบิตที่มี นัยสำคัญสูงสุดของบัฟเฟอร์ 0 ไปถอดรหัสฮัฟฟ์แมนเป็น S0 บัฟเฟอร์เมื่อมีการถอดรหัสข้อมูลแต่ละครั้งจะมีสัญญาณนาฬิกาสำหรับเลื่อนรหัสฮัฟฟ์แมนที่ได้ถอดรหัสไปเรียบร้อยแล้วการเลื่อนข้อมูลจะเลื่อนข้อมูลตามความยาวของรหัสฮัฟฟ์แมนที่ได้ทำการถอดรหัสตัวนั้น คือ ถ้าเป็นรหัสฮัฟฟ์แมนความยาวของรหัส 9 บิต เมื่อทำการถอดรหัสแล้วจะมีการเลื่อนข้อมูลไป 9 ครั้ง กรณีที่เป็นรหัสฮัฟฟ์แมนความยาวของรหัส 1 บิต เมื่อได้ถอดรหัสแล้วจะทำการเลื่อนรหัสไป 1 ตำแหน่ง จะมีวงจรมอดูลัส 1 ถึง 8 ตรวจสอบว่าถ้ารหัสเลื่อนไป 8 ครั้งให้ทำการเลื่อนตำแหน่งของหน่วยความจำไปอีก 1

3.9.7 วงจรสร้างสัญญาณนาฬิกาเลื่อนรหัสฮัฟฟ์แมน

วงจรสร้างสัญญาณนาฬิกาเลื่อนรหัสฮัฟฟ์แมนดังแสดงในรูปที่ 3.62 เป็นวงจรที่มีหน้าที่สร้างสัญญาณนาฬิกาสำหรับเลื่อนรหัสฮัฟฟ์แมนที่ได้ถอดเป็นข้อมูลรหัสเรียบร้อยแล้ว โดยวงจรจะทำงานตามเงื่อนไขสัญญาณที่สร้างขึ้นสำหรับถอดรหัสโดยการตรวจสอบที่บิตที่มีนัยสำคัญสูงสุดของบัฟเฟอร์ 0 โดยการทำงานหลัก คือ ถ้าบิตที่มีนัยสำคัญสูงสุดของบัฟเฟอร์ 0 ระดับลอจิกเป็น 0 จะทำการสร้างสัญญาณนาฬิกา 9 ลูกแล้วส่งไปเลื่อนรหัสฮัฟฟ์แมนในบัฟเฟอร์ 0 และบัฟเฟอร์ 1 หลังจากเลื่อนรหัสฮัฟฟ์แมนจะสร้างสัญญาณถอดรหัสฮัฟฟ์แมนไปตรวจสอบที่บิตที่มีนัยสำคัญสูงสุดของ บัฟเฟอร์ 0 สำหรับเลื่อนรหัสฮัฟฟ์แมนถ้าตรวจสอบบิตที่มีนัยสำคัญสูงสุด บัฟเฟอร์ 0 เป็นระดับ ลอจิก 1 จะทำการสร้างสัญญาณนาฬิกา 1 ลูกแล้วส่งไปเลื่อนรหัสฮัฟฟ์แมนในบัฟเฟอร์ 0 และบัฟเฟอร์ 1 หลังจากที่ส่งสร้างสัญญาณนาฬิกาจะสร้างสัญญาณถอดรหัสัฟฟ์แมนไปตรวจสอบที่บิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 0 ขึ้นมาใหม่โดยวงจรสร้างสัญญาณนาฬิกา สัญญาณต่างๆ ของวงจรมีดังนี้ คือ

1) Initial Check in เป็นสัญญาณอินพุตจาก วงจรโหลดข้อมูลตำแหน่ง 00001H ถึง 1FFFFH ซึ่งทำหน้าที่สร้างสัญญาณถอด รหัสฮัฟฟ์แมนสัญญาณแรกดังที่ได้กล่าวไปแล้ว

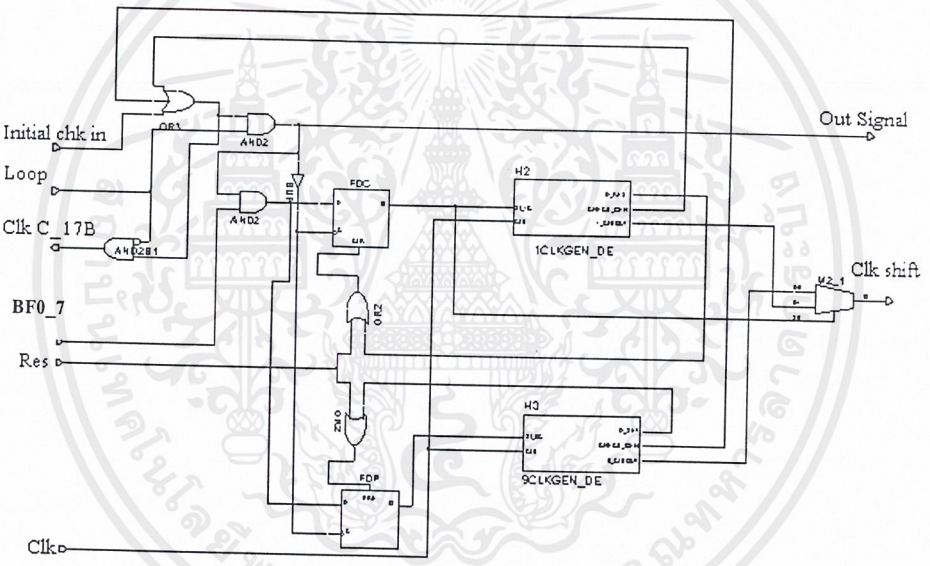
2) ลูปที่เป็นสัญญาณอินพุตจากหน่วยความจำที่ตำแหน่ง 1FFFFH โดยเมื่อระดับสัญญาณเป็น 1 วงจรจะทำการถอดรหัสถ้าสัญญาณมีระดับลอจิกเป็น 0 จะทำให้การถอดรหัสเริ่มถอดรหัสฮัฟฟ์แมนที่ตำแหน่ง 00000H ของหน่วยความจำใหม่

3) BF0_7 เป็นข้อมูลจาก บัฟเฟอร์ชุดที่ 0 บิตที่มีนัยสำคัญสูงสุด

4) RES เป็นสัญญาณอินพุตเพื่อการรีเซตวงจรจะทำการรีเซตเมื่อระดับสัญญาณเป็น 1

5) Out Signal เป็นสัญญาณทางด้านเอาต์พุต สำหรับใช้เป็นสัญญาณนาฬิกาที่ส่งพร้อมกับข้อมูลที่ ทำการถอดรหัสฮัฟฟ์แมนเรียบร้อยแล้ว

6) Clk Shift เป็นสัญญาณเอาต์พุตสำหรับใช้เป็นสัญญาณนาฬิกาสำหรับใช้เลื่อนข้อมูลในบัฟเฟอร์ 0 และบัฟเฟอร์ 1



รูปที่ 3.62 วงจรสร้างสัญญาณนาฬิกาเลื่อนรหัสฮัฟฟ์แมน

การทำงานโดยรวมของวงจร คือ ที่ตำแหน่งหน่วยความจำที่ 00000H ถึง 1FFFEH สัญญาณลูป จะมีระดับลอจิกเป็น 1 ดังนั้นการกระทำทางลอจิกแอนด์กับสัญญาณใดๆ ทำให้ผลการกระทำทางลอจิกเท่ากับสัญญาณส่วนนั้นเมื่อสัญญาณ Initial Check In จากวงจรโหลดข้อมูลตำแหน่ง 00001H ถึง 1FFFFH ลงในหน่วยความจำ ซึ่งมีหน้าที่สร้างสัญญาณถอดรหัสฮัฟฟ์แมนครั้งแรกซึ่งมีลักษณะเป็น สัญญาณนาฬิกา 1 ลูก มากระทำทางลอจิกออร์กับสัญญาณถอดรหัสฮัฟฟ์แมนที่สร้างจาก ส่วนวงจรย่อยวงจรสร้างสัญญาณนาฬิกา 1 ลูก (1 ClkGen_DE) และวงจร สร้างสัญญาณนาฬิกา 9 ลูก (9 ClkGen_DE) ซึ่งในเวลานี้เป็น 0 ทั้งสองสัญญาณระดับลอจิกเป็น 0 ทำให้การกระทำทางลอจิกเป็น 1 ในช่วงขอบขาขึ้นของสัญญาณ Initial Check in ไปกระทำทาง ลอจิกแอนด์

กับบิตที่มีนัยสำคัญสูงสุดของบัฟเฟอร์ 0 กรณีที่ข้อมูลจากบิตที่มีนัยสำคัญสูงสุดของบัฟเฟอร์ 0 มีระดับลอจิกเป็น 1 ซึ่งเป็นรหัสฮัฟฟ์แมนข้อมูลเป็น S0 การกระทำทางลอจิกระหว่าง กับบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 1 มีระดับลอจิกเป็น 1 ในช่วงขอบขาขึ้นของสัญญาณ Initial Check in เป็นผลให้เอาต์พุตของ ดี-ฟลิปฟลอป ตัวที่หนึ่ง และตัวที่สองมีระดับลอจิกเป็น 1 เนื่องจากมีสัญญาณนาฬิกาป้อนให้จากสัญญาณจากสัญญาณ Initial Check in ฟังก์ชันการออกแบบวงจรสร้างสัญญาณนาฬิกา 1 ลูก (1 ClkGEN_DE) ซึ่งเป็นวงจรแบบลำดับนั้นให้ทำการสร้างสัญญาณนาฬิกาเมื่ออินพุตของวงจรเป็น 1 และการออกแบบวงจรสร้างสัญญาณนาฬิกา 9 ลูก (9 ClkGEN_DE) เป็นวงจรแบบลำดับนั้นให้ทำการสร้างสัญญาณนาฬิกาเมื่ออินพุตมีระดับลอจิกเป็น 0 ดังนั้นเมื่ออินพุตมีระดับลอจิกเป็น 1 ทำให้วงจรสร้างสัญญาณนาฬิกา 1 ลูก (1 ClIGEN_DE) สร้างสัญญาณนาฬิกา 1 ลูกจะสร้างสัญญาณนาฬิกาเพื่อไปเลื่อนข้อมูลที่บัฟเฟอร์ 0 และ บัฟเฟอร์ 1 หลังจากนั้นจะสร้างสัญญาณรีเซต ดี–ฟลิปฟลอป ที่ต่อกับอินพุต ของวงจรให้เป็นระดับลอจิก 0 พร้อมที่จะเลื่อนข้อมูลชุดใหม่กรณีที่ทำการถอดรหัสฮัฟฟ์แมนครั้งต่อไปเป็น รหัสฮัฟฟ์แมนความยาวของรหัส 1 บิต หลังจากนั้นวงจรจะสร้างสัญญาณ ถอดรหัสฮัฟฟ์แมน 1 ลูกเพื่อทำการตรวจสอบข้อมูลในบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 0 ในลักษณะเดียวกับ สัญญาณ Initial Check in แล้วพร้อมที่จะทำงานใหม่เมื่ออินพุตเข้ามาเป็น 1 ในกรณีที่บิตที่มีนัยสำคัญสูงสุดของบัฟเฟอร์ 0 มีระดับลอจิกเป็น 0 เอาต์พุตของ ดี-ฟลิปฟลอปทั้งสองระดับลอจิกเป็น 0 จะทำให้วงจรสร้างสัญญาณนาฬิกา 9 ลูก (9 ClkGEN_DE) ทำงานวงจรจะทำการสร้างสัญญาณนาฬิกา 9 ลูก เพื่อเลื่อนรหัส ฮัฟฟ์แมนที่ได้ทำการถอดรหัสในบัฟเฟอร์ 0 และบัฟเฟอร์ 1 พร้อมกันหลังจากที่ได้ทำการสร้างสัญญาณนาฬิกา 9 ลูกแล้วสร้างสัญญาณสำหรับมาเซต ดี-ฟลิปฟลอปที่ต่อกับอินพุต วงจรให้เป็นระดับลอจิก 1 หลังจากนั้นจากนั้นวงจรจะสร้างสัญญาณ ถอดรหัสฮัฟฟ์แมน 1 ลูกเพื่อทำการเช็คข้อมูลในบิตที่มีนัยสำคัญสูงสุด ของบัฟเฟอร์ 0 แล้ววงจรแล้วพร้อมที่จะทำงานใหม่เมื่ออินพุตเข้ามาเป็นระดับลอจิก 0 วงจรจะทำงานลักษณะอย่างนี้ไปเรื่อยๆ จนเมื่อถึงตำแหน่งของหน่วยความจำเป็น 1FFFFH หรือตำแหน่งสุดท้ายจะเป็นผลให้ส่วนอินพุตลูป จะมีลอจิกเป็น 0 ทำให้ให้ผลการกระทำทาง ลอจิกกับสัญญาณใดๆ เป็น 0 ซึ่งจะทำให้ ดี-ฟลิปฟลอป ทั้งสองตัวไม่มีสัญญาณนาฬิกาที่เป็นช่วงขอบขาขึ้นเนื่องจากการกระทำทางลอจิกกับ อินพุตลูป จึงทำให้วงจรสร้างสัญญาณนาฬิกาทำงานเป็นช่วงสุดท้ายของการถอดรหัสฮัฟฟ์แมนไปเป็นข้อมูลซึ่งเมื่อวงจรทั้งสองที่กำลังทำงานสร้างสัญญาณถอดรหัสฮัฟฟ์แมนลูกสุดท้ายทำให้ไปเป็นอินพุตให้กับวงจรตัวนับ 00000H ถึง 1FFFFH ไปชี้ตำแหน่งหน่วยความจำที่ 00000H ใหม่แล้ววงจรถอดรหัสฮัฟฟ์แมนจะทำการถอดรหัสซ้ำอีก

3.9.8 การออกแบบวงจรสร้างสัญญาณนาฬิกา 9 ลูก

การออกแบบสัญญาณนาฬิกาสัญญาณนาฬิกา 9 ลูกจะแตกต่างจากส่วนของวงจรเข้ารหัสฮัฟฟ์แมน คือ จะต้องมีเอาต์พุต 3 เอาต์พุตโดยเอาต์พุตที่ 1 (Z0) สำหรับเป็นสัญญาณนาฬิกาสำหรับเลื่อนข้อมูลในบัฟเฟอร์ เอาต์พุตที่ 2 (Z1) สำหรับเซต ดี-ฟลิปฟลอปที่ต่อเป็นอินพุตของวงจรและเอาต์พุตที่ 3 (Z2) สำหรับสร้างสัญญาณ ไปถอดรหัสฮัฟฟ์แมนดังที่ได้กล่าวมาแล้วข้างต้น

วงจรสร้างสัญญาณนาฬิกา 9 ลูกในวงจรถอดรหัสฮัฟฟ์แมนออกแบบตามฟังก์ชันการทำงานดังรูปที่ 3.63 และ รูปที่ 3.64

Input : $X(t) \in \{0, 1\}$

Output : $Z(t) = \{Z0, Z1, Z2\}, Zo \in \{0, 1\}$

State : $S(t) \in \{A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U\}$

Initial State : $S(0) = A$

Function : ดังรูปที่ 3.63

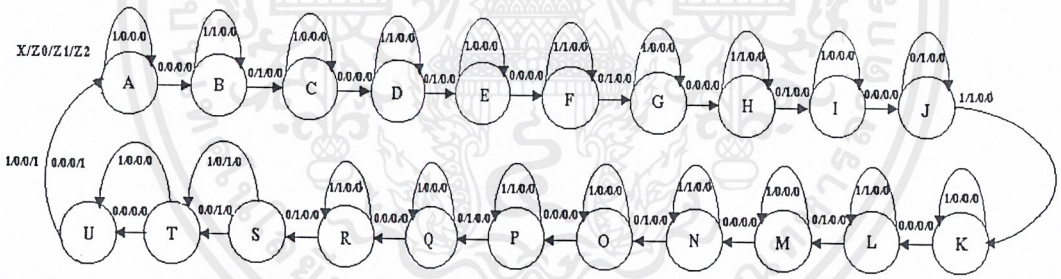
ตารางที่ 3.8 ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 9 ลูก

Present State		Next state X, Z0, Z1, Z2	
State Name	Binary code	X = 0	X = 1
A	00000	B, 0, 0, 0	A, 0, 0, 0
B	00001	C,1, 0, 0	B, 1, 0, 0
C	00010	D, 0, 0, 0	C, 0, 0, 0
D	00011	E, 1, 0, 0	D, 1, 0, 0
E	00100	F, 0, 0, 0	E, 0, 0, 0
F	00101	G, 1, 0, 0	F, 1, 0, 0
G	00110	H, 0, 0, 0	G, 0, 0, 0
H	00111	I, 1, 0, 0	H, 1, 0, 0
I	01000	J, 0, 0, 0	I, 0, 0, 0
J	01001	K, 1, 0, 0	J, 1, 0, 0
K	01010	L, 0, 0, 0	K, 0, 0, 0
L	01011	M, 1, 0, 0	L, 1, 0, 0
M	01100	N, 0, 0, 0	M, 0, 0 ,0

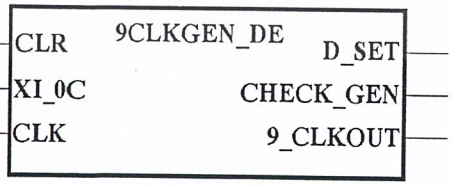
ตารางที่ 3.8 (ต่อ) ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 9 ลูก

Present State		Next state X, Z0, Z1, Z2	
State Name	Binary code	X = 0	X = 1
N	01101	O, 1, 0, 0	N, 1, 0, 0
O	01110	P, 0, 0, 0	O, 0, 0, 0
P	01111	Q, 1, 0, 0	P, 1, 0, 0
Q	10000	R, 0, 0, 0	Q, 0, 0, 0
R	10001	S, 1, 0, 0	R, 1, 0, 0
S	10010	T, 0, 1, 0	T, 0, 1, 0
T	10011	U, 0, 0, 0	U, 0, 0, 0
U	10100	A, 0, 0, 1	A, 0, 0, 1

State Diagram : ดังรูปที่ 3.63



รูปที่ 3.63 แผนผังสถานะวงจรสร้างสัญญาณนาฬิกา 9 ลูก



รูปที่ 3.64 วงจรสร้างสัญญาณนาฬิกา 9 ลูกเมื่อแปลงเป็นแมโคร

3.9.9 การออกแบบวงจรสร้างสัญญาณนาฬิกา 1 ลูก

การออกแบบสัญญาณนาฬิกาของวงจรฮัฟฟ์แมนดีโค้เดอร์จะแตกต่างจากส่วนของการเข้ารหัสตรงที่จะต้องมีเอาต์พุต 3 เอาต์พุตโดยเอาต์พุตที่ 1 (Z0) สำหรับสร้างสัญญาณนาฬิกาสำหรับเลื่อนข้อมูลในบัฟเฟอร์เอาต์พุตที่ 2 (Z1) สำหรับรีเซตดี-ฟลิปฟลอปที่ต่อเป็นอินพุตของวงจร และเอาต์พุตที่ 3 (Z2) สำหรับสร้างสัญญาณ สำหรับถอดรหัสฮัฟฟ์แมนดังที่ได้กล่าวมาแล้วข้างต้น เนื่องจากการถอดรหัสฮัฟฟ์แมนเป็นข้อมูลออกแบบให้ระยะเวลาของการถอรหัสฮัฟฟ์แมนที่ต้องเท่ากันดังนั้นการออกแบบต้องคำนวณ สถานะ การทำงานที่เท่ากันกับวงจรสร้างสัญญาณนาฬิกา 9 ลูกในวงจรถอดรหัสฮัฟฟ์แมนวงจรสร้างสัญญาณนาฬิกา 1 ลูกในวงจรถอดรหัสฮัฟฟ์แมนออกแบบตามฟังก์ชันการทำงานดังรูปที่ 3.56

Input : $X(t) \in \{0, 1\}$

Output : $Z(t) = \{Z0, Z1, Z2\}, Zo \in \{0, 1\}$

State : $S(t) \in \{A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U\}$

Initial State : $S(0) = A$

Function : ดังตารางที่ 3.9

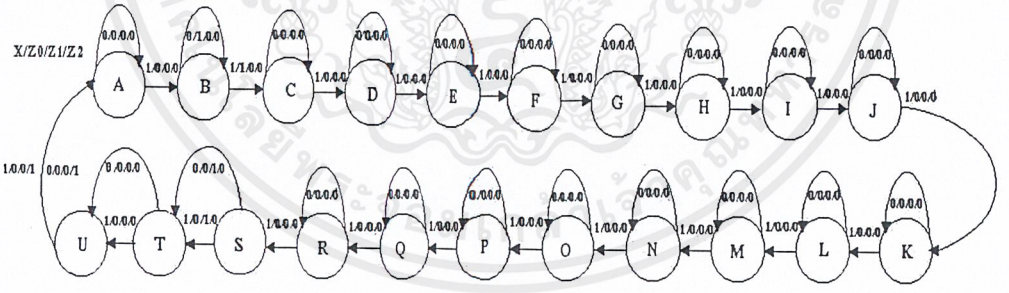
ตารางที่ 3.9 ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 1 ลูก

Present State		Next State X, Z0, Z1, Z2	
State Name	Binary Code	X = 0	X = 1
A	00000	A, 0, 0, 0	B, 0, 0, 0
B	00001	B, 1, 0, 0	C, 1, 0, 0
C	00010	C, 0, 0, 0	D, 0, 0, 0
D	00011	D, 0, 0, 0	E, 0, 0, 0
E	00100	E, 0, 0, 0	F, 0, 0, 0
F	00101	F, 0, 0, 0	G, 0, 0, 0
G	00110	G, 0, 0, 0	H, 0, 0, 0
H	00111	H, 0, 0, 0	I, 0, 0, 0
I	01000	I, 0, 0, 0	J, 0, 0, 0
J	01001	J, 0, 0, 0	K, 0, 0, 0

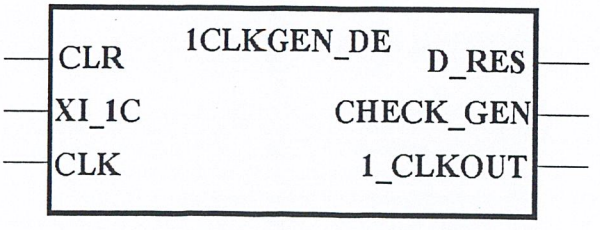
ตารางที่ 3.9 (ต่อ) ฟังก์ชันการทำงานของวงจรสร้างสัญญาณนาฬิกา 1 ลูก

Present State		Next State X, Z0, Z1, Z2	
State Name	Binary Code	X = 0	X = 1
K	01010	K, 0, 0, 0	L, 0, 0, 0
L	01011	L, 0, 0, 0	M, 0, 0, 0
M	01100	M, 0, 0, 0	N, 0, 0, 0
N	01101	N, 0, 0, 0	O, 0, 0, 0
O	01110	O, 0, 0, 0	P, 0, 0, 0
P	01111	P, 0, 0, 0	Q, 0, 0, 0
Q	10000	Q, 0, 0, 0	R, 0, 0, 0
R	10001	R, 0, 0, 0	S, 0, 0, 0
S	10010	T, 0, 1, 0	T, 0, 1, 0
T	10011	U, 0, 0, 0	U, 0, 0, 0
U	10100	A, 0, 0, 1	A, 0, 0, 1

State Diagram : ดังรูปที่ 3.65



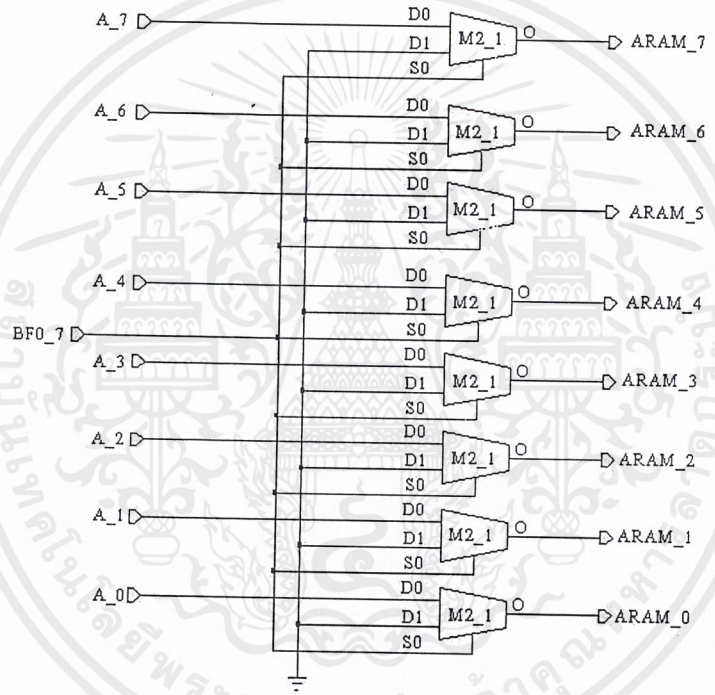
รูปที่ 3.65 แผนผังสถานะวงจรสร้างสัญญาณนาฬิกา 1 ลูก



รูปที่ 3.66 วงจรสร้างสัญญาณนาฬิกา 1 ลูกเมื่อแปลงเป็นแมโคร

3.9.10 วงจรชี้ข้อมูล

วงจรชี้ข้อมูลถือได้ว่าเป็นกระบวนการสุดท้ายของวงจรถอดรหัสฮัฟฟ์แมนสำหรับที่จะนำข้อมูลออกจากวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนไปสู่ภาคประมวลผลข้อมูลอื่นๆ สำหรับให้ข้อมูลที่ได้จากการถอดรหัสฮัฟฟ์แมนพร้อมกับสัญญาณที่ชี้ข้อมูลแต่ละตัวที่ได้ทำการถอดรหัสดังนั้นสัญญาณถอดรหัสฮัฟฟ์แมนที่สร้างขึ้นจะมีข้อมูลค้างอยู่ที่ ดี–ฟลิปฟลอปซึ่งขึ้นอยู่กับช่วงเวลาที่ทำการสร้างสัญญาณถอดรหัสฮัฟฟ์แมนในแต่ละสัญญาณ วงจรชี้ข้อมูลที่ได้ทำการออกแบบในวงจรถอดรหัสฮัฟฟ์แมนแสดงได้ดังรูปที่ 3.67



รูปที่ 3.67 วงจรชี้ข้อมูล

จากการออกแบบรหัสฮัฟฟ์แมนที่ได้ทำการเข้ารหัสจะมีอยู่ 2 ระดับ ดังนั้นการถอดรหัสโดยการสร้างสัญญาณตรวจสอบ บิตที่สูงที่สุดของบัฟเฟอร์ 0 โดยการนำเอาบิตที่สูงที่สุดของบัฟเฟอร์ 0 เป็นวงจรควบคุมการส่งข้อมูลมัลติเพล็กซ์เข้า 2 ออก 1 จำนวน 8 ตัวในวงจร ถ้าบิตที่สูงที่สุดของบัฟเฟอร์ 0 เป็น 1 จะชี้ข้อมูล S0 ซึ่งไบนารีโค้ด คือ 0000 0000 ออกไป ถ้าบิตที่สูงที่สุดของบัฟเฟอร์ 0 เป็น 0 จะทำการส่งข้อมูลซึ่งได้จากการถอดรหัสฮัฟฟ์แมน คือ บิตรองสูงสุดบัฟเฟอร์ 0 เป็นบิตที่มีนัยสำคัญสูงสุดของข้อมูล และบิตที่สูงสุดของบัฟเฟอร์ 1 เป็นบิตที่มี นัยสำคัญต่ำสุดของข้อมูล

3.10 การแปลงข้อมูลกลับ (Reconstruction)

จากข้อมูลที่ผ่านกระบวนการแปลงเวฟเล็ตเพื่อให้มีคุณสมบัติตรงตามที่ ฮัฟฟ์แมนต้องการ คือ การเข้ารหัสข้อมูลที่มีความซ้ำซ้อนสูงเพื่อทำการจัดเก็บลงในหน่วยความจำค่าที่ทำการจัดเก็บจะไม่สามารถที่จะนำออกสู่วงจรแปลงสัญญาณดิจิตอลเป็นแอนะลอกได้ เพราะค่าที่ทำการจัดเก็บนั้นจะมีการแยกองค์ประกอบของข้อมูลออกจากกัน ซึ่งจะไม่อยู่ในรูปของข้อมูลเสียงต้นแบบ จึงจำเป็นอย่างยิ่งที่จะต้องมีขั้นตอนในการแปลงกลับเวฟเลต (Inverse Wavelet Transfrom) ซึ่งขั้นตอนนี้เองจะเป็นกระบวนการนำข้อมูลที่เก็บไว้ในหน่วยความจำมาทำการจัดรูปแบบ จากข้อมูลที่มีการแยกส่วนกันอยู่ให้มีลักษณะคล้ายกับข้อมูลที่เข้ามา โดยในการแปลงกลับเวฟเลตนั้นจะต้องมีขบวนการที่คล้ายคลึงกับตอนเข้า ซึ่งจะมีขบวนการดังต่อไปนี้

3.10.1 การรีควอนไตซ์ (Requantization)

การทำการรีควอนไตซ์ (Requantization) คือ ขบวนการคืนค่าโดยการนำข้อมูลที่มีขนาด 8 บิต ที่ผ่านการถอดรหัสฮัฟฟ์แมน มาทำการคืนค่ากลับสู่ข้อมูลขนาด 24 บิต ซึ่งเป็นข้อมูลที่ถูกกระทำก่อนเข้าสู่ขั้นตอนการเข้ารหัสฮัฟฟ์แมน โดยข้อมูลทีมีขนาด 8 บิต นั้นจะถูกนำไปทำการชี้ ตารางของการจัดระดับสัญญาณ ซึ่งค่าเหล่านี้จะถูกจัดเก็บไว้ในหน่วยความจำโดยข้อมูลขนาด 8 บิต จะเป็นตำแหน่งของข้อมูลขนาด 8 บิต ซึ่งจะเป็นระดับของการควอนไตซ์โดยวงจรที่ทำการออกแบบจะเป็นวงจรถอดรหัสธรรมดา ทำให้ผลลัพธ์ที่ได้มีขนาด 8 บิต ก่อนที่จะไปทำการแปลงกลับเวฟเล็ต

3.10.2 การแปลงกลับเวฟเลต (Inverse Wavelet Transfrom)

ในการที่จะนำข้อมูลที่ได้จากการเข้ารหัสไปใช้งานนั้นจะต้องนำค่าที่ผ่านการถอดรหัสฮัฟฟ์แมน ไปทำการรีควอนไตซ์ (Requantization) และจากค่าที่ได้ก็จะถูกนำไปหาร (เพื่อจัดการกับค่าที่ถูกคูณเข้าไปในตอนต้น) จากขั้นตอนการรีควอนไตซ์ผลที่ได้ คือ ค่าที่มีช่วง 127 ถึง –128 ซึ่งก็คือ ข้อมูลที่มีขนาด 8 บิตนั่นเองจากข้อมูลที่ได้จะต้องนำไปทำการแปลงกลับเวฟเลต (Inverse Wavelet Transfrom) ซึ่งจากวงจรที่ออกแบบนั้นเป็นวงจรที่สามารถแปลงเวฟเลตไป และแปลงกลับได้ในวงจรเดียวกัน โดยในการแปลงกลับข้อมูลนั้นจะต้องทำการเปลี่ยนตัวกรอง โดยใช้สมการเดิมที่ใช้ในการแปลงเวฟเลต จะมีการเปลี่ยนเพียงเวฟเลตแม่ที่ใช้ในการแปลงกลับเท่านั้น ซึ่งกระบวนการในการแปลงกลับทั้งหมดนั้นจะมีการทำงานเหมือนกับการแปลงไปของเวฟเลตโดยทั้งสิ้น

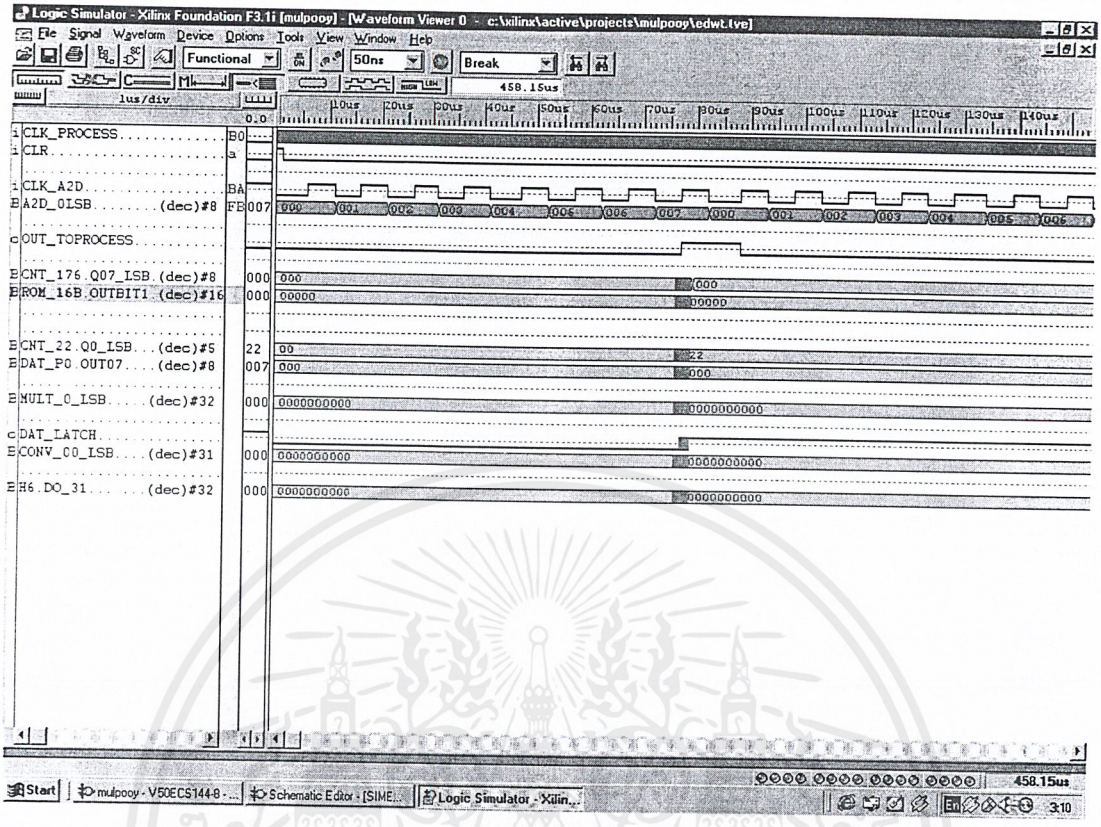
บทที่ 4

การทดลอง และผลการทดลอง

จากการทำงานของวงจรที่ถูกแบ่งออกเป็นส่วนซึ่งประกอบไปด้วย วงจรที่ออกแบบภายนอก และวงจรภายในอุปกรณ์ FPGAs ซึ่งวงจรภายนอกนั้นจะเป็นชุดของการรับข้อมูลที่เป็นเสียง โดยจะทำการแปลงสัญญาณจากสัญญาณแอนะลอกไปเป็นดิจิตอล และส่วนของการแปลงสัญญาณดิจิตอลไปเป็นแอนะลอก วงจรภายในเป็นวงจรที่ได้ออกแบบเพื่อทำการแปลงข้อมูล เพื่อจัดเก็บในรูปแบบของสัมประสิทธิ์ เวฟเลตเพคเกจการเข้า และถอดรหัสด้วยฮัฟฟ์แมน โดยการทดลองเป็นผลจำลองการทำงาน (Simulate) โดยการกำหนดข้อมูลให้กับวงจรในแบบต่างๆ แล้วตรวจสอบจากระดับของข้อมูลที่ได้จากอุปกรณ์ที่สร้างขึ้นแต่ละส่วน มีการทำงานเข้าจังหวะกับสัญญาณนาฬิกา และเงื่อนไขของการทำงานในส่วนต่างๆ มีการทำงานสอดคล้องกับวัตถุประสงค์ที่ได้ออกแบบวงจรในส่วนต่างๆ จาก บทที่ 3 โดยการทดลองจะแบ่งออกเป็น 4 ส่วนใหญ่ๆ ด้วยกัน คือ การแปลงเวฟเลตการเข้ารหัส และถอดรหัสฮัฟฟ์แมนการถอดรหัสฮัฟฟ์แมน และการแปลงกลับเวฟเลต ซึ่งทั้งวงจร 4 ส่วนนี้เป็นองค์ประกอบหลักที่สำคัญของเครื่องบันทึกเสียงพูดระบบดิจิตอลโดยส่วนของการควบคุมของแต่ละกระบวนการจะแยกกันโดยจะอาศัยเงื่อนไขในการควบคุมที่ถูกสร้างขึ้นจากแต่ละส่วนเพื่อทำการควบคุมการทำงานของแต่ละส่วนให้ทำงานได้อย่างมีประสิทธิภาพ

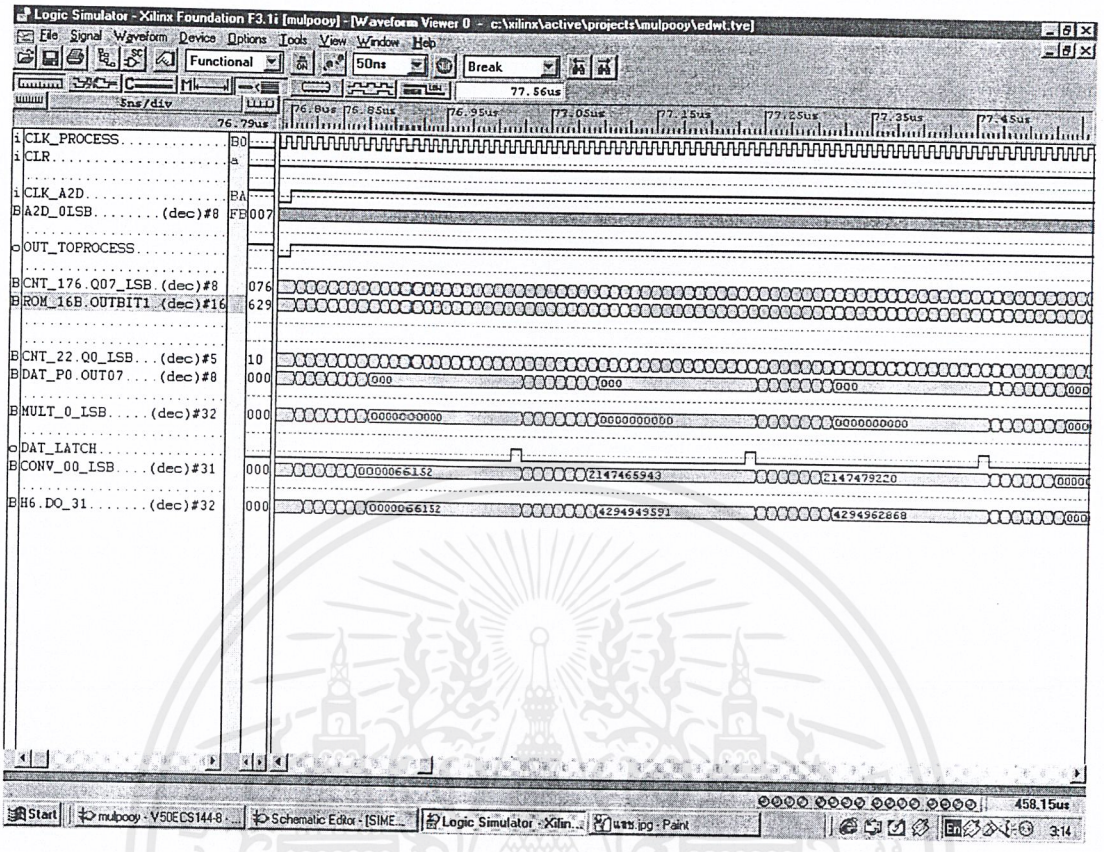
4.1 ผลการทดลองการแปลงเวฟเลต

ในการแปลงเวฟเลตนั้นจะอาศัยสัญญาณนาฬิกาควบคุมการทำงานจาก 2 ส่วน คือ สัญญาณนาฬิกาของ A/D (CLK_A2D) ซึ่งจะเป็นสัญญาณนาฬิกาที่มีช่วงของคาบเวลามากที่สุดในระบบ โดยจะเป็นสัญญาณที่ควบคุมการอ่านข้อมูลที่รับเข้ามา และอีกส่วนจะเป็นสัญญาณนาฬิกาที่ใช้ในการทำการแปลงเวฟเลต (CLK_PROCESS) ซึ่งจะมีช่วงของคาบเวลาน้อยกว่า เพื่อที่จะทำการแปลงเวฟเลตให้เสร็จสิ้นภายในสัญญาณนาฬิกาของ A/D (CLK_A2D) 1 ลูก



รูปที่ 4.1 การ Latch ข้อมูลพร้อมกับสัญญาณควบคุม

จากรูปที่ 4.1 สัญญาณนาฬิกาควบคุมจาก CLK_A2D จะใช้ในการควบคุมการอ่านข้อมูลเข้ามาโดยภายใน 1 ลูกของสัญญาณนาฬิกาจะต้องทำการนำค่าที่ถูก Latch เก็บไว้ในในวงจรเลื่อนข้อมูล แล้วทำการอ่านออกมาเพื่อทำการคูณกับค่าของตัวกรอง แล้วทำการบวกไปเรื่อยๆ จนครบ 22 ค่า สำหรับ 1 ทิศทาง ซึ่งจะทำทั้งหมด 8 ทิศทาง ดังนั้นจึงทำการคูณค่า และบวกค่าทั้งหมด 176 ครั้ง โดยรอบแรกของการทำงานจะต้องทำการ Clear หลายๆ ส่วนภายในวงจร เช่น หน่วยความจำวงจรเลื่อนข้อมูล วงจรนับ และวงจรส่วนอื่นๆ ซึ่งในรอบแรกจะทำการ Latch ค่าเข้ามา 8 ค่า คือ 0, 1, 2, 3, 4, 5, 6 และ 7 โดยจะทำการเก็บในวงจรเลื่อนข้อมูล และในการอ่านข้อมูลแต่ละครั้งก็จะกระทำในช่วงขอบขาขึ้นของสัญญาณนาฬิกาจาก CLK_A2D เมื่อทำการเก็บข้อมูลครบครบภายในสัญญาณนาฬิกา (CLK_A2D) 8 ลูก ซึ่งที่ขา OUT_TOPROCRESS วงจรตัวนับมอดูลัส 1 ถึง 8 ทำหน้านับจำนวนข้อมูลที่เข้ามาแปลงเวฟเลต เมื่อวงจรนับ ถึง 8 หมายถึงข้อมูลเข้ามาครบ 8 ข้อมูล จะทำการแปลงเวฟเลตกับข้อมูลที่รับเข้ามา

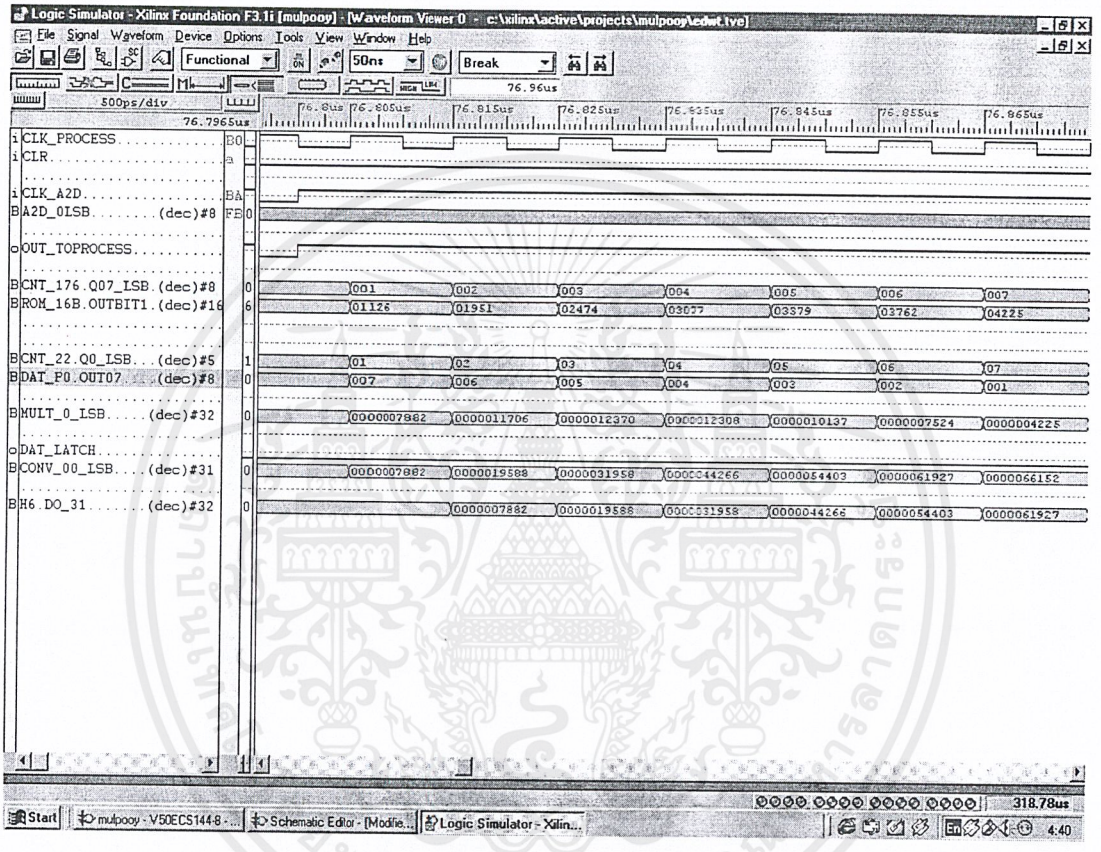


รูปที่ 4.2 การนำค่าที่ Latch ทำการคำนวณ

ในการคำนวณนั้นจะต้องทำให้เสร็จภายในสัญญาณนาฬิกาของ CLK_A2D เพียง 1 ลูก เมื่อทำงานครบวงจรนับแบบมีเงื่อนไขก็จะส่งสัญญาณออกมาควบคุมให้หยุดทำการชี้ค่าจาก ROM และวงจรเลื่อนข้อมูล

ในการเลื่อนข้อมูลภายในวงจรเลื่อนข้อมูลนั้นจะทำการเลื่อนข้อมูลที่อ่านเข้ามาครั้งละ 8 ข้อมูล ดังรูปที่ 4.2 โดยในการอ่านข้อมูลออกมาแต่ละครั้งจากวงจรเลื่อนข้อมูล (DAT_P0.OUT07) นั้นจะต้องทำการอ่านค่าของตัวกรองที่ถูกจัดเก็บไว้ใน ROM (ROM16B.OUTBIT) พร้อมทั้งต้องทำการคำนวณหาค่าของผลคูณไปด้วย ทำให้ผลคูณดังรูปที่ 4.2 ใน Bus สัญญาณที่ชื่อ MULT_0_LSB โดยจะแสดงเป็นเลขฐานสิบ เสร็จแล้วจะนำค่าที่ได้จากการคูณไปทำการบวกกับค่าของผลคูณของข้อมูลชุดถัดไป ซึ่งจะทำการบวกไปเรื่อยๆ จนครบ 22 ครั้ง ผลที่ได้ คือ จะเป็นค่าของการบวก หรือสัมประสิทธิ์ของการแปลงเวฟเลต โดยจะแสดงดังรูปที่ 4.3 ใน Bus สัญญาณที่ชื่อ CONV_00_LSB

เมื่อทำการบวกจนครบ 22 ครั้ง ค่าที่ได้จากการบวกแต่ละรอบนั้นจะแสดงดังรูปที่ 4.3 ใน BUS สัญญาณที่ชื่อ H6.D0 เสร็จแล้ววงจรตัวนับ 22 แบบมีเงื่อนไขก็จะทำการส่งระดับลอจิก 1 ออกไปเพื่อ ควบคุมการ Latch ข้อมูลในวงจรเลื่อนข้อมูลก่อนที่จะส่งข้อมูลออกไปสู่วงจรจัดระดับสัญญาณ



รูปที่ 4.3 ผลลัพธ์ของการคำนวณในแต่ละครั้ง

จากรูปที่ 4.3 จะเห็นว่าเมื่อมีสัญญาณนาฬิกาที่ส่งมาจากวงจรตัวนับแบบมีเงื่อนไขทางขา OUT_TOPROCRESS โดยจะมีค่าระดับลิจิกเป็น 1 ซึ่งจะทำให้วงจรนับทำการอ่านค่าที่ถูกเก็บไว้ใน ROM และทำการอ่านค่าจากวงจรเลื่อนข้อมูลมาทำการคูณ ค่าที่อ่านเข้ามาทำการคูณนั้นจะถูกอ่านเข้ามาดังรูปที่ 4.3 โดยจะอ่านเข้ามาในช่วงขอบขาขึ้นของสัญญาณนาฬิกาจาก CLK_A2D คือ 0, 2, 3, 4, 5, 6, และ 7 ใน ช่วง 8 ลูกของสัญญาณนาฬิกา CLK_A2D โดยในการอ่านค่าที่ถูกเก็บไว้ในวงจรเลื่อนข้อมูลนั้นจะถูกอ่านขึ้นมาเพื่อคำนวณซึ่งจะแสดงได้ใน BUS ที่ชื่อ DAT_P0.OUT พร้อมกับค่าของตัวกรองที่ถูกจัดเก็บไว้ใน ROM ที่อ่านออกมาไปทำการคูณ ซึ่งผลที่ได้จากการคูณ

แต่ละครั้งจะต้องนำไปบวกกันกับผลคูณครั้งที่ผ่านมาจนครบ 22 รอบ เมื่อทำการบวกครบ 22 รอบ จะมีวงจรตัวนับ 22 จะส่งเงื่อนไขออกมาเป็น 1 แล้วทำการส่งข้อมูลให้วงจรเลื่อนข้อมูลทำการเลื่อนข้อมูลที่มีค่าสัมประสิทธิ์ในวงจรเลื่อนข้อมูลเพื่อส่งออกไปยังวงจรจัดระดับสัญญาณ

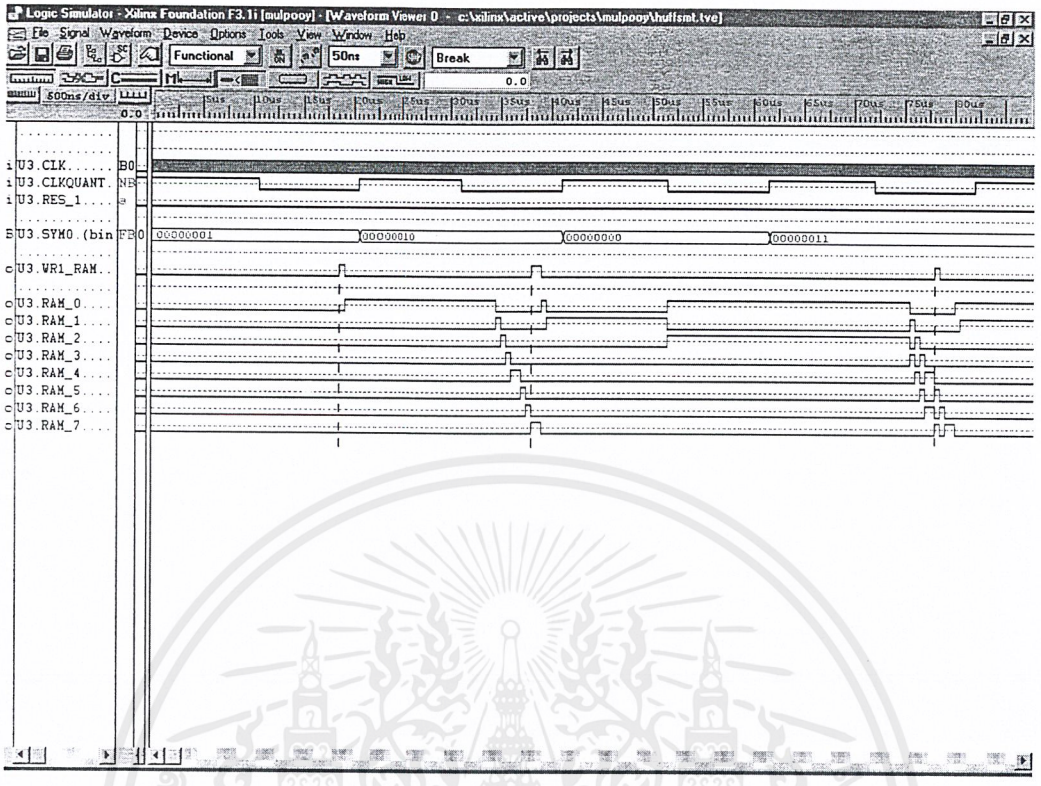
ในรูปที่ 4.3 จะสังเกตเห็นได้ว่าเมื่อมีสัญญาณนาฬิกา CLK_PROCRESS เข้ามาในช่วงขอบขาขึ้นนั้น วงจรเลื่อนข้อมูลจะถูก Latch เป็นตัวสุดท้ายเข้ามาทำการคูณ พร้อมทั้งอ่านค่าที่ถูกจัดเก็บไว้ภายใน ROM เข้ามาคูณด้วย จะเห็นได้ว่าค่าที่อ่านเข้ามาในสัญญาณนาฬิกา CLK_PROCRESS ลูกแรกจะอ่านค่าออกมาจาก ROM ทาง BUS โดยจะมีค่าเท่ากับ 1126 ส่วนข้อมูลที่ผ่าน A/D เข้ามานั้นจะอยู่ใน BUS ที่ชื่อว่า DATA_P0.OUT07 โดยจะมีค่า 007 ซึ่งจะเป็นค่าสุดท้ายที่ถูก Latch เข้ามาจาก A/D เมื่อคูณกันเสร็จจะมีค่าเท่ากับ 7882 แล้วนำมาทำการบวก ซึ่งในรอบแรกจะบวกกับศูนย์ โดยผลของการบวกนั้นจะถูกแสดงใน BUS ที่ชื่อ CONV เมื่อกระทำรอบที่สองนั้น ค่าที่อ่านจาก ROM ออกมาได้ คือ 1951 ส่วนค่าที่อ่านเข้ามาจากการ Latch ของ A/D นั้นจะมีค่า 0006 เมื่อคูณกันเสร็จจะมีค่าเท่ากับ 11706 โดยผลที่ได้จากการคูณนี้จะถูกนำมาบวกกับผลที่ได้ในรอบแรก ซึ่งจะทำให้มีค่าเท่ากับ 19588 ซึ่งจะกระทำเช่นนี้ไปจนครบ 22 รอบ แล้วผลที่ได้จะถูกนำไปจัดเก็บในวงจรเลื่อนข้อมูลในช่วงขาออก ของข้อมูล

4.2 การทดลอง และผลการทดลองวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน

ผลการทดลองของวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมนที่ได้นั้นเกิดจากจากการรวมวงจรย่อยที่ออกแบบเป็นแมโครในแต่ละส่วนย่อยมารวมเป็นวงจรใหญ่รวมสามารถนำมาสังเคราะห์และจำลองการทำงานได้สามารถแบ่งออกเป็น 2 ส่วน คือ วงจรเข้ารหัสฮัฟฟ์แมน และวงจรถอดรหัสฮัฟฟ์แมน

4.3 วงจรเข้ารหัสฮัฟฟ์แมน

จากขั้นตอนการออกแบบ วงจรส่วนย่อยของวงจรเข้ารหัสฮัฟฟ์แมนในหัวข้อ 3.10 ซึ่งได้ออกแบบเป็นวงจรซึ่งได้ออกแบบทั้งวงจรดิจิตอลแบบอันดับ และวงจรดิจิตอลแบบลำดับออกแบบให้วงจรทำงานตามเงื่อนไขของอินพุตเมื่อได้นำมารวมเป็นวงจรเข้ารหัสฮัฟฟ์แมนแล้วนำมาสังเคราะห์ และจำลองการทำงานได้ผลการทำงานดังรูปที่ 4.4



รูปที่ 4.4 ผลจำลองการทำงานของวงจรเข้ารหัสฮัฟฟ์แมน

การทดลองจะเป็นการจัดเก็บรหัสฮัฟฟ์แมนลงในหน่วยความจำสัญญาณต่างๆ ที่ได้จากวงจรเข้ารหัสฮัฟฟ์แมนซึ่งจำลองการทำงานโดยการทดลองป้อนสัญญาณนาฬิกาเข้าที่อินพุตโดยสัญญาณนาฬิกาแบ่งเป็น 2 ส่วน คือ สัญญาณนาฬิกาในการประมวลผลเข้ารหัสฮัฟฟ์แมน (CLK) และสัญญาณนาฬิกาจากภาคควอนไตร์ (CLKQUAT) ซึ่งเป็นสัญญาณบ่งบอกการเปลี่ยนแปลงข้อมูลในแต่ละข้อมูล ซึ่งเป็นตัวแปรสำคัญในการกำหนดระยะเวลาของการเข้ารหัสฮัฟฟ์แมนแต่ละข้อมูล สัญญาณรีเซต (RES_1) สำหรับให้วงจรพร้อมที่จะทำงานอินพุตอีกส่วนหนึ่ง คือ ข้อมูลที่จะเข้าแทนรหัสฮัฟฟ์แมน (Sym0-Sym7) ซึ่งเป็นข้อมูลขนาด8บิตในส่วนของเอาต์พุตจะมีสัญญาณเขียนข้อมูลลงไปในหน่วยความจำ (WR1_RAM) ซึ่งสัญญาณนี้จะนำไปเพิ่มตำแหน่งของหน่วยความจำไปอีก 1 ตำแหน่งหลังจากเขียนข้อมูลลงในหน่วยความจำ และส่วนของรหัสฮัฟฟ์แมนที่จะจัดเก็บในหน่วยความจำ (RAM_0~ RAM_7)

ผลการทดลองสามารถอธิบายได้ดังนี้จากรูปที่ 4.4 วงจรเข้ารหัสฮัฟฟ์แมนจะทำการเข้ารหัสฮัฟฟ์แมนช่วงขอบขาลงของสัญญาณนาฬิกาจากภาคควอนไตร์ (CLKQUAT) ผลการทดลองจะเห็นได้ข้อมูลที่เข้ามาในฮัฟฟ์แมนมี 4 ข้อมูลตาลำดับดังนี้ คือ 0000 0001 (S1), 0000 0010 (S2), 0000 0000 (S0), และ 0000 0011 (S3) รหัสฮัฟฟ์แมนในบทที่ 3 เมื่อได้ทำการแทนรหัสฮัฟฟ์แมน

ทั้งสามข้อมูลตามลำดับดังนี้ 0 0000 0001, 0 0000 0010, 1, 0 0000 0011 ดังนั้นรหัสฮัฟฟ์แมนที่แทนข้อมูลทั้งสามจะจัดเก็บในหน่วยความจำขนาด 8 บิตดังนี้

	D0	D1	D2	D3	D4	D5	D6	D7
ตำแหน่งที่ 0	0	0	0	0	0	0	0	0
ตำแหน่งที่ 1	0	0	0	0	0	0	0	1
ตำแหน่งที่ 2	0	0	0	0	0	1	0	1
ตำแหน่งที่ 3	X	X	X	X	1	1	0	0

รูปที่ 4.5 จำลองจัดเก็บรหัสฮัฟฟ์แมนที่แทนข้อมูล S1, S2, S0, S3 ตามลำดับ

จากรูปที่ 4.5 จะพบว่าแต่ละตำแหน่งของหน่วยความจำจะเก็บส่วนรหัสฮัฟฟ์แมนที่แทนข้อมูลแต่ละข้อมูลเอาไว้นอกจาก S0 ซึ่งเป็นรหัสฮัฟฟ์แมนความยาวโคดซึ่งจะเรียงตามบิตสุดท้ายของส่วนรหัสฮัฟฟ์แมนที่แทนข้อมูลอื่นๆ จากผลการทดลองการจัดเก็บรหัสฮัฟฟ์แมนจะขึ้นอยู่กับสัญญาณ เขียนข้อมูลในหน่วยความจำในที่นี้ คือ สัญญาณ WR1_RAM ซึ่งจะเป็นสัญญาณเขียนข้อมูลลงไปในหน่วยความจำจะให้สัญญาณช่วงขอบขาขึ้นไปเขียนข้อมูลลงไปในหน่วยความจำ จากผลจำลองการทำงานของวงจรเข้ารหัสฮัฟฟ์แมนเมื่อสังเกตจากเส้นประจะพบว่าสัญญาณเขียนข้อมูลลงในหน่วยความจำลูกที่ 1 จะเขียนข้อมูล 0000 0000 ลงไปในตำแหน่งที่ 0 ของหน่วยความจำ สัญญาณเขียนข้อมูลลงในหน่วยความจำลูกที่ 2 จะเขียนข้อมูล 1000 0000 ลงไปในตำแหน่งที่ 1 และสัญญาณเขียนข้อมูลลงในหน่วยความจำลูกที่ 3 จะเขียนข้อมูล 1010 0000 ลงไปในตำแหน่งที่ 2 จะพบว่ามีบางส่วนของรหัส ฮัฟฟ์แมนที่แทน ข้อมูล S3 ยังไม่ได้จัดเก็บ ลงในหน่วยความจำ จะเกิดสัญญาณเขียนข้อมูลลงไปในหน่วยความจำอีกครั้งเมื่อมีส่วนของรหัสฮัฟฟ์แมนที่แทนข้อมูลชุดใหม่ซึ่งเข้ามาในระบบฮัฟฟ์แมนโค้ดเดอร์มารวมกับรหัสฮัฟฟ์แมนที่เหลือแล้วได้ 8 บิต สำหรับการทดลองนี้ต้องมีส่วนของรหัสฮัฟฟ์แมนที่เข้ามาใหม่อีก 4 บิตเพราะส่วนของรหัสฮัฟฟ์แมนที่ยังไม่ได้จัดเก็บลงในหน่วยความจำอยู่ ของ S3 มี 4 บิต ซึ่งผลจำลองการทำงานของวงจรเข้ารหัสฮัฟฟ์แมนเป็นไปตามวัตถุประสงค์ของวงจรเข้ารหัสฮัฟฟ์แมนที่ได้ออกแบบสร้างเอาไว้ในบทที่ 3

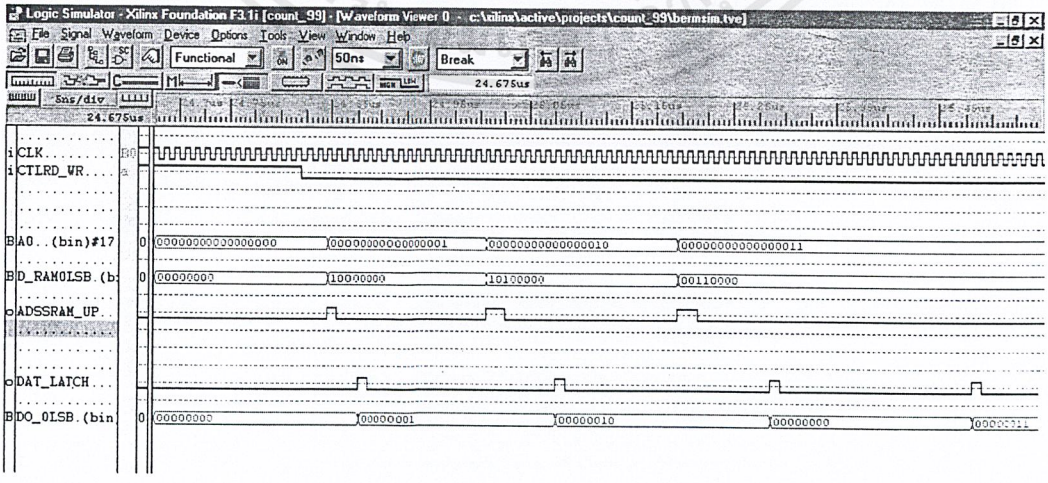
4.4 วงจรถอดรหัสฮัฟฟ์แมน

จากการออกแบบ วงจรส่วนย่อยของวงจรเข้ารหัส และถอดรหัสฮัฟฟ์แมน 3.12 เมื่อนำวงจรในส่วนย่อยมารวมกันเป็นวงจรถอดรหัสฮัฟฟ์แมนแล้วนำมาทดลองถอดรหัสฮัฟฟ์แมน การทดลองวงจรเข้ารหัสฮัฟฟ์แมน โดยใช้รหัสฮัฟฟ์แมนที่อยู่ในหน่วยความจำจำลองที่สร้างขึ้นขนาด 8 บิต 4 ตำแหน่ง โดยแต่ละตำแหน่งมีข้อมูล ดังรูปที่ 4.6

	D0	D1	D2	D3	D4	D5	D6	D7
ตำแหน่งที่ 0	0	0	0	0	0	0	0	0
ตำแหน่งที่ 1	0	0	0	0	0	0	0	1
ตำแหน่งที่ 2	0	0	0	0	0	1	0	1
ตำแหน่งที่ 3	0	0	0	0	1	1	0	0

รูปที่ 4.6 ตำแหน่ง และข้อมูลในหน่วยความจำเพื่อทดสอบการดีโค้ดรหัสฮัฟฟ์แมน

จากรูปจะเห็นได้ว่าสอดคล้องกับรูปที่ 4.5 ในการทดลองวงจรเข้ารหัสฮัฟฟ์แมน เพื่อทดสอบการดีโค้ดฮัฟฟ์แมนให้ได้ข้อมูลที่ออกมาที่ถูกต้องตามลำดับข้อมูลที่ได้ทำการแทนรหัสฮัฟฟ์แมนจะพบว่าเมื่อได้ทำการดีโค้ดรหัสฮัฟฟ์แมนจะได้ข้อมูลตามลำดับดังนี้ คือ ข้อมูลแรก 0000 0001 หรือ S1 ข้อมูลที่ 2 คือ 0000 0010 หรือ S2 ข้อมูลที่ 3 คือ 0000 0000 หรือ S0 และข้อมูลที่ 4 คือ 0000 0011 หรือ S3 สำหรับผลที่ได้จากการทดลองวงจรถอดรหัสฮัฟฟ์แมน ดังรูปที่ 4.7



รูปที่ 4.7 ผลจำลองการทำงานของวงจรถอดรหัสฮัฟฟ์แมน

การทดลองจะเป็นการนำรหัสฮัฟฟ์แมนหน่วยความจำที่ได้จำลองขึ้นสัญญาณต่างๆ ที่ได้จาก วงจรเข้ารหัสฮัฟฟ์แมนซึ่งจำลองการทำงานโดยการทดลองป้อนสัญญาณนาฬิกาเข้าที่อินพุตโดยเป็นสัญญาณนาฬิกาในการประมวลผลถอดรหัสฮัฟฟ์แมน (CLK) สัญญาณนาฬิกาอ่านข้อมูล (CTLRD_WR) ซึ่งเป็นสัญญาณจากกภายนอกโดยสัญญาณเป็น0จะมีการถอดรหัสฮัฟฟ์แมนในส่วนของเอาต์พุตจะส่วนของรหัสฮัฟฟ์แมนในหน่วยความจำ (D0) และตำแหน่งในหน่วยความจำ (A0) ซึ่งได้แสดงเป็นเลขฐานสองซึ่งจะเห็นได้ว่ากรณีที่เป็นรหัสฮัฟฟ์แมนที่มีความยาวรหัส 9 บิตจะใช้ตำแหน่งของหน่วยความจำในการจัดเก็บ 2 ตำแหน่งซึ่งรหัสฮัฟฟ์แมนที่แทนข้อมูลจะอนุกรมกันอยู่ ดังที่ได้กล่าวไปแล้วสัญญาณเพิ่มตำแหน่งหน่วยความจำ (ADDRAM_UP) สัญญาณชี้ข้อมูล (DAT_LATCH) ที่ได้ทำการถอดรหัสฮัฟฟ์แมนซึ่งมีความสำคัญสำหรับนำข้อมูลที่ได้ถอดรหัสฮัฟฟ์แมนส่งให้ภาคต่อไปประมวลผลข้อมูลต่อไป

ผลการทดลองของวงจรถอดรหัสฮัฟฟ์แมนสามารถอธิบายได้ดังนี้ คือ จากรูปที่ 4.7 จะพบว่า วงจรถอดรหัสฮัฟฟ์แมนจะทำงานเมื่อมีสัญญาณอ่านข้อมูล (CTLRD_WR) ซึ่งเป็นสัญญาณที่ได้จากภายนอกโดยสัญญาณเป็น 0 ซึ่งจะสังเกตได้จากสัญญาณทางเอาต์พุตต่างๆ จะเกิดมีการเปลี่ยนแปลงหลังจากที่สัญญาณอ่านข้อมูล (CTLRD_WR) เป็น 0 วงจรถอดรหัสฮัฟฟ์แมนจะถอดรหัสเป็นข้อมูลตัวแรกที่ตำแหน่งที่ 1 สอดคล้องกับหลักการทำงานของวงจรที่ออกแบบ คือ วงจรถอดรหัสฮัฟฟ์แมนจะนำส่วนของรหัสฮัฟฟ์แมนในตำแหน่งที่ 0 ลงในบัฟเฟอร์ 0 นำส่วนของรหัสฮัฟฟ์แมนในตำแหน่งที่ ลงในบัฟเฟอร์ 1 แล้วถอดรหัสเป็นข้อมูลตัวแรก สัญญาณเพิ่มตำแหน่งหน่วยความจำ (ADDRAM_UP) จะไม่เป็นจังหวะที่คงที่เห็นได้จากการถอดรหัสได้ข้อมูล 0000 0000 หรือ S0 แล้วจังหวะการเลื่อนตำแหน่งในหน่วยความจำจะเปลี่ยนไปสอดคล้องกับหลักการทำงานของวงจรเพราะในกรณีที่ถอดรหัส ฮัฟฟ์แมนความยาวรหัส 1 บิต จำเป็นจะต้องหน่วงเวลาให้เท่ากับการถอดรหัสฮัฟฟ์แมนความยาวรหัส 9 บิต ซึ่งเราจะเห็นได้ว่าข้อมูลแต่ละชุดที่ถอดรหัสออกมามีช่วงเวลาที่แน่นอนไม่ว่าจะเป็นการถอด รหัสฮัฟฟ์แมนความยาวรหัส 1 บิต หรือ 9 บิต ก็ตามซึ่งผลจำลองการทำงานของวงจรถอดรหัสฮัฟฟ์แมนเป็นไปตามวัตถุประสงค์ของวงจรถอดรหัสฮัฟฟ์แมนที่ได้ออกแบบสร้างเอาไว้

บทที่ 5
ปัญหา และแนวทางพัฒนา

5.1 บทสรุป

จากกระบวนการในการลดขนาดของข้อมูล (Compress) จะประกอบไปด้วยขั้นตอนที่ยุ่งยากซับซ้อนในหลายๆ อย่างเกิดขึ้นจากความไม่เหมาะสมของการเลือกวงจรใช้งาน ซึ่งในบางกรณีนั้นเป็นผลที่ทำให้ข้อมูลที่ทำการแปลงกลับออกมามีลักษณะของข้อมูลที่มีการผิดเพี้ยนไปมาก ซึ่งสิ่งเหล่านี้ ก็อาจจะเกิดขึ้นจากหลายสาเหตุ ทั้งในเรื่องของการจัดตำแหน่งของข้อมูลภายในหน่วยความจำ การจัดระดับข้อมูล (Quantization) ซึ่งจะส่งผลให้เกิดความผิดพลาดของการจัดระดับ (Quantize Error) การสร้างตารางการเข้ารหัส หรือแม้กระทั่งการตัดระดับข้อมูล (Threshold) เองก็ตาม ทุกอย่างล้วนแต่เป็นองค์ประกอบที่ทำให้ค่าที่ผ่านการแปลงกลับข้อมูล (Inverse Wavelet Transfrom) เกิดการผิดพลาด ทั้งยังมีในส่วนของวงจรภายนอก เช่น อัตราการสุ่มตัวอย่างข้อมูลขาเข้า (Sampling Rate) จำนวนข้อมูลในการในการสุ่มตัวอย่างแต่ละครั้ง แล้วยังในส่วนของการเข้ารหัส ซึ่งการเข้ารหัสเองก็มีหลายแนวคิดในการเข้ารหัส ประสิทธิภาพต่างๆ ในการลดขนาดข้อมูลเสียงก็ขึ้นอยู่กับสิ่งเหล่านี้ด้วย

ในการออกแบบวงจรต่างๆ นั้นจะกระทำในโปรแกรม Xilinx Foundation 3.1 โดยใช้การออกแบบด้วย Schematic เป็นการออกแบบวงจรในระดับเกต ซึ่งจะต้องมีหลายวงจรที่ต้องใช้งานตามแต่ ละกระบวนการ โดยจะต้องมีส่วนของการควบคุม (Control) ซึ่งจะใช้วงจรนับ (Counter) ทั้งแบบมีเงื่อนไข และไม่มีเงื่อนไข เพื่อที่จะควบคุมขั้นของการทำงานในแต่ละวงจร โดยในการทดลองขั้นตอนต่างๆ นั้นจำเป็นต้องกระทำร่วมทั้งในส่วนของการจำลองการทำงาน (Simulate) ภายในโปรแกรม และส่วนของการทำงานจริง คือ ต้องทำการ Down Load วงจรลงในอุปกรณ์ FPGAs เพื่อที่จะทดลองการทำงานจึงต้องมีการทดลองควบคู่กันไปเพื่อให้ได้ผลออกมาของวงจรที่มีการทำงานที่ถูกต้อง ทั้งภาย ในโปรแกรม Xilinx Foundation และการ Down Load วงจรลงในอุปกรณ์ FPGAs ด้วย

5.2 ปัญหา และแนวทางการแก้ไข

จากขั้นตอนต่างๆ ในการจัดทำโครงงานชิ้นนี้สามารถที่จะสรุปปัญหาที่เกิดขึ้นระหว่างทำโครงงานนี้ พร้อมกับแนวทางในการแก้ไขปัญหาดังนี้

1) ปัญหา มีสัญญาณรบกวนเกิดขึ้นกับไฟเลี้ยงของอุปกรณ์แต่ละตัวภายในวงจรทำให้การใช้งานของอุปกรณ์ แต่ละตัวถูกรบกวนเอาต์พุตที่ได้จึงมีความผิดพลาด

แนวทางการแก้ไข ต่อตัวเก็บประจุค่า 0.1 ไมโครฟารัดชนิดเซรามิกระหว่างขาไฟเลี้ยงของไอซีกับกราวน์ของไอซีทุกตัวโดยต่อให้ใกล้กับขาไฟเลี้ยงของตัวไอซีมากที่สุด โดยจะแยกกราวด์ระหว่าง วงจรภาคแอนะลอก และดิจิตอล และใช้หม้อแปลงที่ลดสัญญาณรบกวน เช่น หม้อแปลงเทอร์รอยด์จะช่วยลดสัญญาณรบกวนลดลงได้มาก

2) ปัญหา ไม่สามารถใช้สัญญาณนาฬิกาในการสุ่ม (Sampling Clock) ที่มีความถี่สูงกว่า 20 MHz ได้ เพราะไอซีหน่วยความจำไม่สามารถทำงานได้ที่ความถี่เกิน 20 MHz

แนวทางการแก้ไข ให้เปลี่ยนไปใช้หน่วยความจำที่มีความเร็วสูงกว่านี้ เช่น หน่วยความจำเบอร์ AM720-10

3) ปัญหา ไอซีแปลงสัญญาณแอนะลอกเป็นดิจิตอลรับความถี่ที่สัญญาณอินพุตได้สูงสุดไม่เกิน 4.43 MHz

แนวทางการแก้ไข ให้เปลี่ยนไปใช้ไอซีแปลงสัญญาณแอนะลอกเป็นดิจิตอลที่สามารถรับความถี่อิน พุตได้สูง กว่านี้ เช่นเบอร์ TDA8798, TDA8776 เป็นต้น

4) ปัญหา วงจรทริกเกอร์ทางฮาร์ดแวร์ไม่สามารถทำการทริกซ์ได้ที่ระดับแรงดันดีซี

แนวทางการแก้ไข เปลี่ยนการทริกซ์เกอร์จากฮาร์ดแวร์มาเป็นการทริกซ์เกอร์ทางซอฟต์แวร์

5) ปัญหา จากการวัดสัญญาณแบบ Real Time รูปสัญญาณที่ได้จะไม่นิ่ง

แนวทางการแก้ไข พัฒนาวงจรทริกซ์เกอร์ให้มีประสิทธิภาพมากขึ้น โดยการเพิ่มภาคขยายสัญญาณทริกซ์เกอร์ให้สูงขึ้น

6) ปัญหา ในการทำการ Implement วงจรลงในอุปกรณ์ FPGAs นั้นจะเกิดข้อผิดพลาดขึ้นในกรณีที่ไม่มีการกำหนด IBUF ให้กับ IPAD และ OBUF ให้กับ OPAD ภายใน Schematics เสมอ

แนวทางการแก้ไข ต้องมีการกำหนด IBUF ให้ IPAD และกำหนด OBUF ให้กับ OPAD เสมอ

7) ปัญหา ในการใช้งานสัญญาณนาฬิกาจากภายนอกที่จะทำการรส่งเข้ามาภายในวงจรที่อยู่ในอุปกรณ์ FPGA นั้น เกิดการผิดพลาด คือ วงจรภายในจะไม่รู้จักสัญญาณนาฬิกาที่ป้อนเข้ามา

แนวทางการแก้ไข จะต้องมีการกำหนด IBUFG ในส่วนขาเข้าของสัญญาณนาฬิกาภายใน Schematic เพื่อให้วงจรที่อยู่ภายในอุปกรณ์ FPGAs รู้จักสัญญาณนาฬิกาที่ป้อนให้

8) ปัญหา ในการ Implement วงจรนั้น บางครั้งจะเกิดข้อผิดพลาดโดยไม่รู้สาเหตุ โดยจะไม่มีการรายงานข้อผิดพลาดหรือสิ่งที่เกิดขึ้นจากการ Implement

แนวทางการแก้ไข ให้ปิดโปรแกรม Xilinx Foundation Series 3.1 แล้วเปิดโปรแกรม Xilinx Foundation Series 3.1 ขึ้นมาใหม่แต่ถ้ายังมีอาการเหมือนเดิมก็ให้ปิดเครื่องคอมพิวเตอร์แล้วเปิดเครื่องคอมพิวเตอร์เพื่อใช้งานใหม่

9) ปัญหา ข้อผิดพลาดที่เกิดจากการกำหนด PAD ซ้ำกัน และการกำหนด PAD ตรงกับขาที่ถูกสงวนไว้ของอุปกรณ์ FPGAs

แนวทางการแก้ไข ตรวจสอบขาที่สามารถนำไปใช้งานได้ของอุปกรณ์ FPGAs แต่ละตระกูลอย่างละเอียด

10) ปัญหา เกิดข้อผิดพลาดจากการใช้งานอุปกรณ์ OSC (อุปกรณ์กำเนิดสัญญาณนาฬิกา) มากกว่า 1 ตัวภายใน 1 หน้ากระดาษของ Schematic ของโปรแกรม Xilinx Foundation V2.1

แนวทางการแก้ไข จะต้องกำหนดการใช้งาน OSC ภายใน 1 หน้ากระดาษของโปรแกรม Xilinx Foundation 2.1 ได้เพียง 1 ตัว แต่ถ้าต้องการที่จะใช้งานสัญญาณนาฬิการที่นอกเหนือไปจากความถี่ ที่กำหนดให้ จะต้องนำสัญญาณนาฬิกาไปผ่านวงจรหารความถี่

11) ปัญหา ขบวนการอันซับซ้อนที่เกิดขึ้นจากการแปลงเวฟเล็ตแพคเก็จหลายระดับ ซึ่งจะทำให้ต้องสร้าง วงจรนับเป็นจำนวนมาก เงื่อนไขที่ต้องเพิ่มมากขึ้น วงจรคูณจำนวน 16 บิต หลายตัววงจรบวกขนาด 32 บิตหลายตัว รีจิสเตอร์ขนาด 32 บิต จำนวนมาก ฯลฯ

แนวทางการแก้ไข ทำการลดขั้นตอนโดยการนำสมการที่เกิดขึ้นในแต่ละระดับแทนค่าลงไปในแต่ละระดับจนครบทุกระดับ ซึ่งจะสามารถกำหนดค่าคงที่ให้กับค่าตัวกรอง โดยจะเหลือค่าที่ต้องเปลี่ยนแปลง คือ ค่าของข้อมูลที่เข้ามาเพียงอย่างเดียวค่าตัวกรองที่ได้ออกมาจะต้องทำการเก็บในรอม ทำให้ขั้นตอนที่ยุ่งยากลดลงไป

5.3 แนวทางในการพัฒนา

แนวทางในการพัฒนาฮาร์ดแวร์

1) ควรใช้ไอซีแปลงสัญญาณแอนะลอกเป็นดิจิตอลที่มีความสามารถในการแปลงข้อมูลที่เร็ว วัดสัญญาณอินพุตได้ที่ความถี่สูงๆ และมีจำนวนบิตมากกว่าไอซีเบอร์ TDA8703 เช่นเบอร์ TDA8776 โดยมีจำนวนบิต และอัตราการสุ่มสัญญาณที่สูงกว่า

2) ในการออกแบบควรพัฒนาให้มีขีดความสามารถทางด้านการวัดสัญญาณขนาดเล็กๆ เป็น มิลลิโวลต์โดยวิธีเพิ่มภาคขยายสัญญาณอินพุตให้มากขึ้น

3) การเขียนโปรแกรมสามารถเขียนได้หลายภาษาเช่น ภาษาวิชวลเบสิก, ภาษาวิชวลซี และภาษา เดลไฟล์ เป็นต้น ซึ่งขึ้นอยู่กับความถนัด ถ้าต้องการความเร็วที่สูงที่สุดให้เลือกใช้ภาษาวิชวลซี

แนวทางในการพัฒนาการบีบอัดข้อมูลเสียง

1) ในการเลือกใช้งานตัวกรอง ควรหาตัวกรองที่มีคุณสมบัติในการกรองความถี่ที่มีช่วงตั้งแต่ 300 ถึง 3500 Hz ให้เหมาะสม

2) ในการคำนวณเปลี่ยนส่วนของการคำนวณให้มีคุณสมบัติการทำงาน เป็นแบบทศนิยม (Floating Point) ในการกำหนดจุดตัดระดับข้อมูล ควรคำนวณหาจุดตัดระดับที่เหมาะสมกับ ข้อมูลเสียงที่ผ่านการแปลงกลับออกมาว่ามีมีคุณภาพเป็นที่ต้องการหรือไม่

3) ในบางกรณีที่มีความยุ่งยากทางในการออกแบบวงจรควรใช้ VHDL เขียนโปรแกรมแล้วสร้างเป็นแมโครสำหรับนำมาใช้ใน Schematics Editor

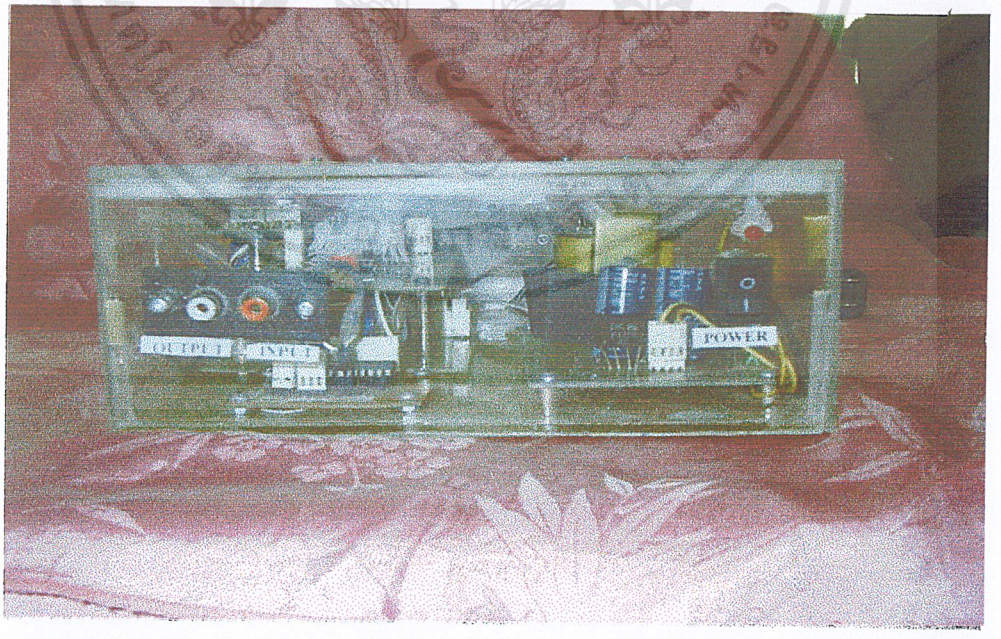
4) การเข้ารหัส และถอดรหัสฮัฟฟ์แมนควรทำการคำนวณหรือทดลองหาสถิติความน่าจะเป็นของข้อมูลหลาย ๆ ระดับ เพราะจะทำให้การลดบิตข้อมูลได้มีเปอร์เซนต์สูงขึ้น

5) การลดรูปสมการลอจิกเกตควรใช้โปรแกรมช่วยเพราะกรณีที่ออกแบบวงจรแบบลำดับที่มีหลาย สถานะจะช่วยประหยัดเวลาในการออกแบบ

6) วงจรแบบลำดับที่มีลักษณะการทำงานใกล้เคียงกันมาใช้งานร่วมกัน

ภาคผนวก ก
เครื่องบันทึกเสียงพูดระบบดิจิตอลต้นแบบ

รูปที่ ก.1 เครื่องบันทึกเสียงพูดระบบดิจิตอล



รูปที่ ก.2 เครื่องบันทึกเสียงพูดระบบดิจิตอล

ภาคผนวก ข
ตารางฮัฟฟ์แมน และวงจรพิมพ์

ตารางที่ ข.1 รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S0	0000 0000	1
S1	0000 0001	0 0000 0001
S2	0000 0010	0 0000 0010
S3	0000 0011	0 0000 0011
S4	0000 0100	0 0000 0100
S5	0000 0101	0 0000 0101
S6	0000 0110	0 0000 0110
S7	0000 0111	0 0000 0111
S8	0000 1000	0 0000 1000
S9	0000 1001	0 0000 1001
S10	0000 1010	0 0000 1010
S11	0000 1011	0 0000 1011
S12	0000 1100	0 0000 1100
S13	0000 1101	0 0000 1101
S14	0000 1110	0 0000 1110
S15	0000 1111	0 0000 1111
S16	0001 0000	0 0001 0000
S17	0001 0001	0 0001 0001
S18	0001 0010	0 0001 0010
S19	0001 0011	0 0001 0011
S20	0001 0100	0 0001 0100
S21	0001 0101	0 0001 0101

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S22	0001 0110	0 0001 0110
S23	0001 0111	0 0001 0111
S24	0001 1000	0 0001 1000
S25	0001 1001	0 0001 1001
S26	0001 1010	0 0001 1010
S27	0001 1011	0 0001 1011
S28	0001 1100	0 0001 1100
S29	0001 1101	0 0001 1101
S30	0001 1110	0 0001 1110
S31	0001 1111	0 0001 1111
S32	0010 0000	0 0010 0000
S33	0010 0001	0 0010 0001
S34	0010 0010	0 0010 0010
S35	0010 0011	0 0010 0011
S36	0010 0100	0 0010 0100
S37	0010 0101	0 0010 0101
S38	0010 0110	0 0010 0110
S39	0010 0111	0 0010 0111
S40	0010 1000	0 0010 1000
S41	0010 1001	0 0010 1001
S42	0010 1010	0 0010 1010
S43	0010 1011	0 0010 1011

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S44	0010 1100	0 0010 1100
S45	0010 1101	0 0010 1101
S46	0010 1110	0 0010 1110
S47	0010 1111	0 0010 1111
S48	0011 0000	0 0011 0000
S49	0011 0001	0 0011 0001
S50	0011 0010	0 0011 0010
S51	0011 0011	0 0011 0011
S52	0011 0100	0 0011 0100
S53	0011 0101	0 0011 0101
S54	0011 0110	0 0011 0110
S55	0011 0111	0 0011 0111
S56	0011 1000	0 0011 1000
S57	0011 1001	0 0011 1001
S58	0011 1010	0 0011 1010
S59	0011 1011	0 0011 1011
S60	0011 1100	0 0011 1100
S61	0011 1101	0 0011 1101
S62	0011 1110	0 0011 1110
S63	0011 1111	0 0011 1111
S64	0100 0000	0 0100 0000
S65	0100 0001	0 0100 0001

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S66	0100 0010	0 0100 0010
S67	0100 0011	0 0100 0011
S68	0100 0100	0 0100 0100
S69	0100 0101	0 0100 0101
S70	0100 0110	0 0100 0110
S71	0100 0111	0 0100 0111
S72	0100 1000	0 0100 1000
S73	0100 1001	0 0100 1001
S74	0100 1010	0 0100 1010
S75	0100 1011	0 0100 1011
S76	0100 1100	0 0100 1100
S77	0100 1101	0 0100 1101
S78	0100 1110	0 0100 1110
S79	0100 1111	0 0100 1111
S80	0101 0000	0 0101 0000
S81	0101 0001	0 0101 0001
S82	0101 0010	0 0101 0010
S83	0101 0011	0 0101 0011
S84	0101 0100	0 0101 0100
S85	0101 0101	0 0101 0101
S86	0101 0110	0 0101 0110
S87	0101 0111	0 0101 0111

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S88	0101 1000	0 0101 1000
S89	0101 1001	0 0101 1001
S90	0101 1010	0 0101 1010
S91	0101 1011	0 0101 1011
S92	0101 1100	0 0101 1100
S93	0101 1101	0 0101 1101
S94	0101 1110	0 0101 1110
S95	0101 1111	0 0101 1111
S96	0110 0000	0 0110 0000
S97	0110 0001	0 0110 0001
S98	0110 0010	0 0110 0010
S99	0110 0011	0 0110 0011
S100	0110 0100	0 0110 0100
S101	0110 0101	0 0110 0101
S102	0110 0110	0 0110 0110
S103	0110 0111	0 0110 0111
S104	0110 1000	0 0110 1000
S105	0110 1001	0 0110 1001
S106	0110 1010	0 0110 1010
S107	0110 1011	0 0110 1011
S108	0110 1100	0 0110 1100
S109	0110 1101	0 0110 1101

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S110	0110 1110	0 0110 1110
S111	0110 1111	0 0110 1111
S112	0111 0000	0 0111 0000
S113	0111 0001	0 0111 0001
S114	0111 0010	0 0111 0010
S115	0111 0011	0 0111 0011
S116	0111 0100	0 0111 0100
S117	0111 0101	0 0111 0101
S118	0111 0110	0 0111 0110
S119	0111 0111	0 0111 0111
S120	0111 1000	0 0111 1000
S121	0111 1001	0 0111 1001
S122	0111 1010	0 0111 1010
S123	0111 1011	0 0111 1011
S124	0111 1100	0 0111 1100
S125	0111 1101	0 0111 1101
S126	0111 1110	0 0111 1110
S127	0111 1111	0 0111 1111
S128	1000 0000	0 1000 0000
S129	1000 0001	0 1000 0001
S130	1000 0010	0 1000 0010
S131	1000 0011	0 1000 0011

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S132	1000 0100	0 1000 0100
S133	1000 0101	0 1000 0101
S134	1000 0110	0 1000 0110
S135	1000 0111	0 1000 0111
S136	1000 1000	0 1000 1000
S137	1000 1001	0 1000 1001
S138	1000 1010	0 1000 1010
S139	1000 1011	0 1000 1011
S140	1000 1100	0 1000 1100
S141	1000 1101	0 1000 1101
S142	1000 1110	0 1000 1110
S143	1000 1111	0 1000 1111
S144	1001 0000	0 1001 0000
S145	1001 0001	0 1001 0001
S146	1001 0010	0 1001 0010
S147	1001 0011	0 1001 0011
S148	1001 0100	0 1001 0100
S149	1001 0101	0 1001 0101
S150	1001 0110	0 1001 0110
S151	1001 0111	0 1001 0111
S152	1001 1000	0 1001 1000
S153	1001 1001	0 1001 1001

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S154	1001 1010	0 1001 1010
S155	1001 1011	0 1001 1011
S156	1001 1100	0 1001 1100
S157	1001 1101	0 1001 1101
S158	1001 1110	0 1001 1110
S159	1001 1111	0 1001 1111
S160	1010 0000	0 1010 0000
S161	1010 0001	0 1010 0001
S162	1010 0010	0 1010 0010
S163	1010 0011	0 1010 0011
S164	1010 0100	0 1010 0100
S165	1010 0101	0 1010 0101
S166	1010 0110	0 1010 0110
S167	1010 0111	0 1010 0111
S168	1010 1000	0 1010 1000
S169	1010 1001	0 1010 1001
S170	1010 1010	0 1010 1010
S171	1010 1011	0 1010 1011
S172	1010 1100	0 1010 1100
S173	1010 1101	0 1010 1101
S174	1010 1110	0 1010 1110
S175	1010 1111	0 1010 1111

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S176	1011 0000	0 1011 0000
S177	1011 0001	0 1011 0001
S178	1011 0010	0 1011 0010
S179	1011 0011	0 1011 0011
S180	1011 0100	0 1011 0100
S181	1011 0101	0 1011 0101
S182	1011 0110	0 1011 0110
S183	1011 0111	0 1011 0111
S184	1011 1000	0 1011 1000
S185	1011 1001	0 1011 1001
S186	1011 1010	0 1011 1010
S187	1011 1011	0 1011 1011
S188	1011 1100	0 1011 1100
S189	1011 1101	0 1011 1101
S190	1011 1110	0 1011 1110
S191	1011 1111	0 1011 1111
S192	1100 0000	0 1100 0000
S193	1101 0001	0 1101 0001
S194	1101 0010	0 1101 0010
S195	1101 0011	0 1101 0011
S196	1101 0100	0 1101 0100
S197	1101 0101	0 1101 0101

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S198	1100 0110	0 1100 0110
S199	1100 0111	0 1100 0111
S200	1100 1000	0 1100 1000
S201	1100 1001	0 1100 1001
S202	1100 1010	0 1100 1010
S203	1100 1011	0 1100 1011
S204	1100 1100	0 1100 1100
S205	1100 1101	0 1100 1101
S206	1100 1110	0 1100 1110
S207	1100 1111	0 1100 1111
S208	1101 0000	0 1101 0000
S209	1101 0001	0 1101 0001
S210	1101 0010	0 1101 0010
S211	1101 0011	0 1101 0011
S212	1101 0100	0 1101 0100
S213	1101 0101	0 1101 0101
S214	1101 0110	0 1101 0110
S215	1101 0111	0 1101 0111
S216	1101 1000	0 1101 1000
S217	1101 1001	0 1101 1001
S218	1101 1010	0 1101 1010
S219	1101 1011	0 1101 1011

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

Symbols	Binary Code	Huffman Code
S220	1101 1100	0 1101 1100
S221	1101 1101	0 1101 1101
S222	1101 1110	0 1101 1110
S223	1101 1111	0 1101 1111
S224	1110 0000	0 1110 0000
S225	1110 0001	0 1110 0001
S226	1110 0010	0 1110 0010
S227	1110 0011	0 1110 0011
S228	1110 0100	0 1110 0100
S229	1110 0101	0 1110 0101
S230	1110 0110	0 1110 0110
S231	1110 0111	0 1110 0111
S232	1110 1000	0 1110 1000
S233	1110 1001	0 1110 1001
S234	1110 1010	0 1110 1010
S235	1110 1011	0 1110 1011
S236	1110 1100	0 1110 1100
S237	1110 1101	0 1110 1101
S238	1110 1110	0 1110 1110
S239	1110 1111	0 1110 1111
S240	1111 0000	0 1111 0000
S241	1111 0001	0 1111 0001

ตารางที่ ข.1 (ต่อ) รหัสฮัฟฟ์แมน

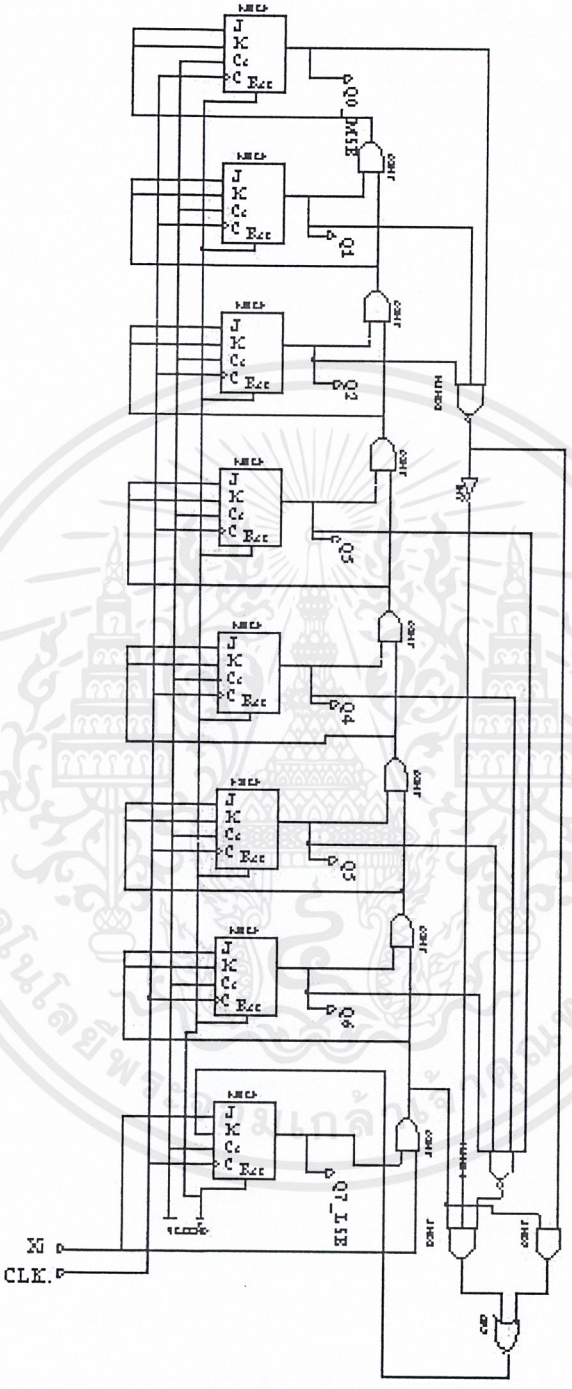
Symbols	Binary Code	Huffman Code
S242	1111 0010	0 1111 0010
S243	1111 0011	0 1111 0011
S244	1111 0100	0 1111 0100
S245	1111 0101	0 1111 0101
S246	1111 0110	0 1111 0110
S247	1111 0111	0 1111 0111
S248	1111 1000	0 1111 1000
S249	1111 1001	0 1111 1001
S250	1111 1010	0 1111 1010
S251	1111 1011	0 1111 1011
S252	1111 1100	0 1111 1100
S253	1111 1101	0 1111 1101
S254	1111 1110	0 1111 1110
S255	1111 1111	0 1111 1111

ตารางที่ ข.2 การลดข้อมูลเมื่อเข้ารหัสฮัฟฟ์แมน

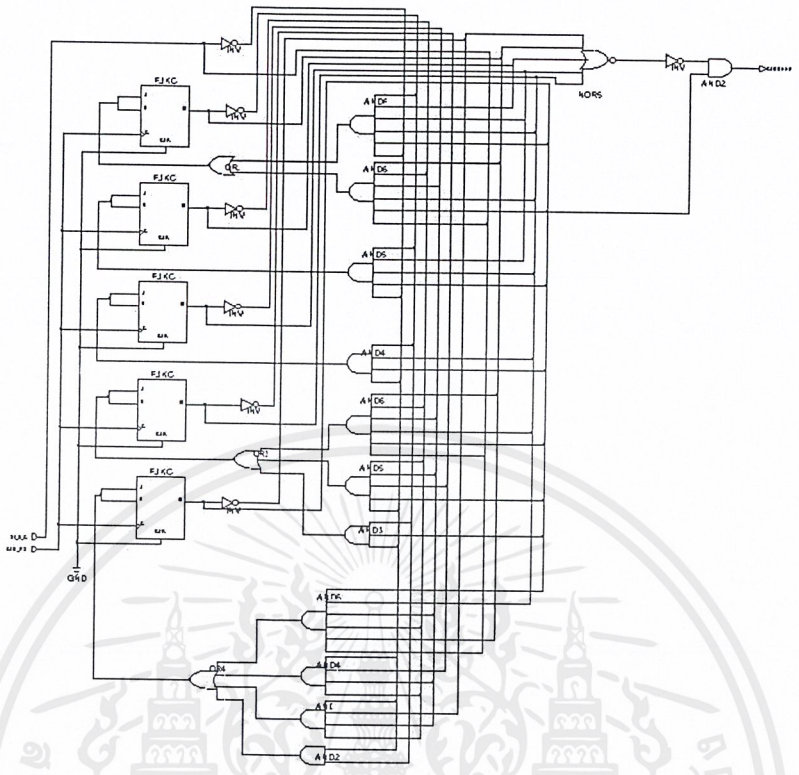
จำนวน S0 จาก 100 ข้อมูล	การเก็บข้อมูลโดยตรง (บิต)	การเก็บบิตข้อมูลเมื่อเข้ารหัส (บิต)	การลดบิตข้อมูล (บิต)
0	800	900	เพิ่มขึ้น 100
4	800	868	เพิ่มขึ้น 68
8	800	836	เพิ่มขึ้น 36

ตารางที่ ข.2 (ต่อ) การลดข้อมูลเมื่อเข้ารหัสฮัฟฟ์แมน

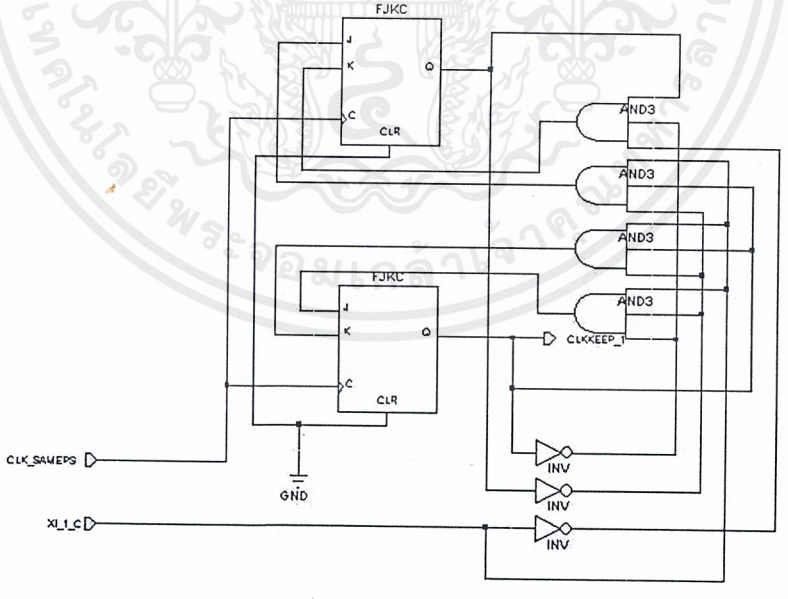
จำนวน S0 จาก 100 ข้อมูล	การเก็บข้อมูลโดยตรง (บิต)	การเก็บบิตข้อมูลเมื่อเข้ารหัส (บิต)	การลดบิตข้อมูล (บิต)
12	800	804	เพิ่มขึ้น 4
16	800	772	ลดลง 28
20	800	740	ลดลง 60
24	800	708	ลดลง 92
28	800	676	ลดลง 124
32	800	644	ลดลง 156
36	800	612	ลดลง 188
40	800	580	ลดลง 220
44	800	548	ลดลง 252
48	800	516	ลดลง 248
52	800	484	ลดลง 316
56	800	452	ลดลง 348
60	800	420	ลดลง 380
64	800	388	ลดลง 412
68	800	356	ลดลง 444
72	800	324	ลดลง 476
76	800	292	ลดลง 508
80	800	260	ลดลง 540
84	800	228	ลดลง 572
88	800	196	ลดลง 604
92	800	164	ลดลง 636
96	800	132	ลดลง 668
100	800	100	ลดลง 700



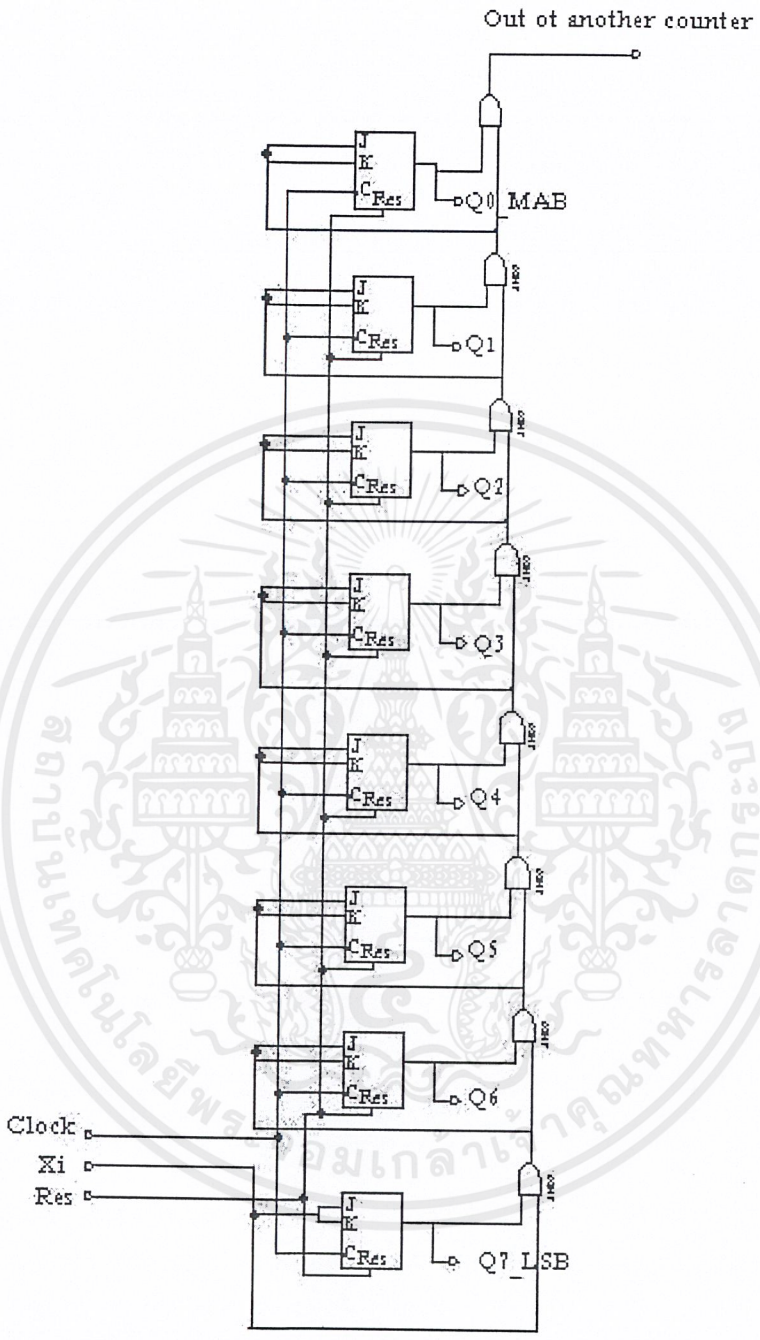
รูปที่ ข.1 วงจรตัวนับ 1 ถึง 255



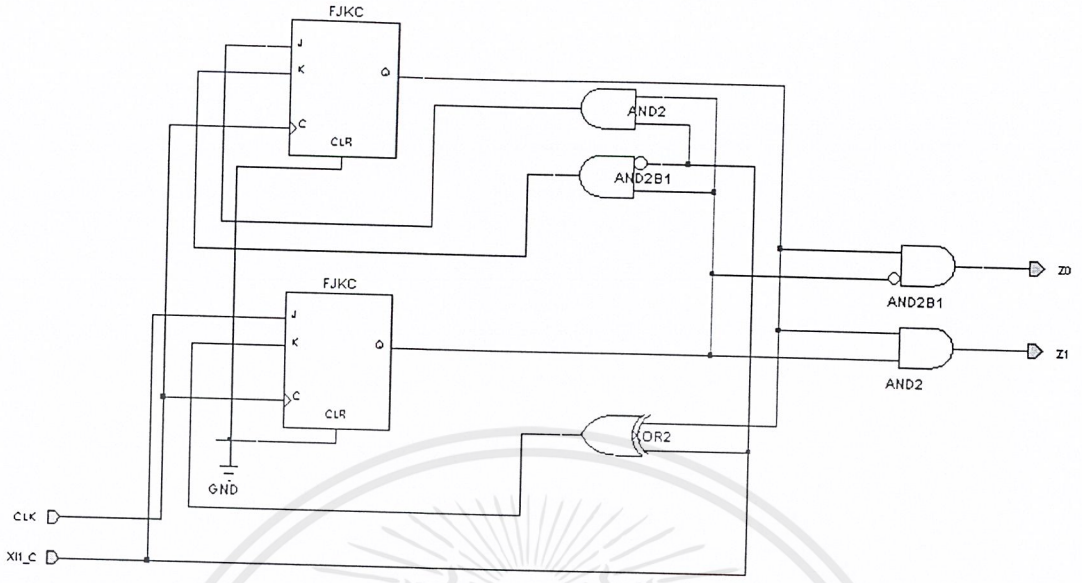
รูปที่ ข.2 วงจรกำเนิดสัญญาณนาฬิกา 9 ลูก



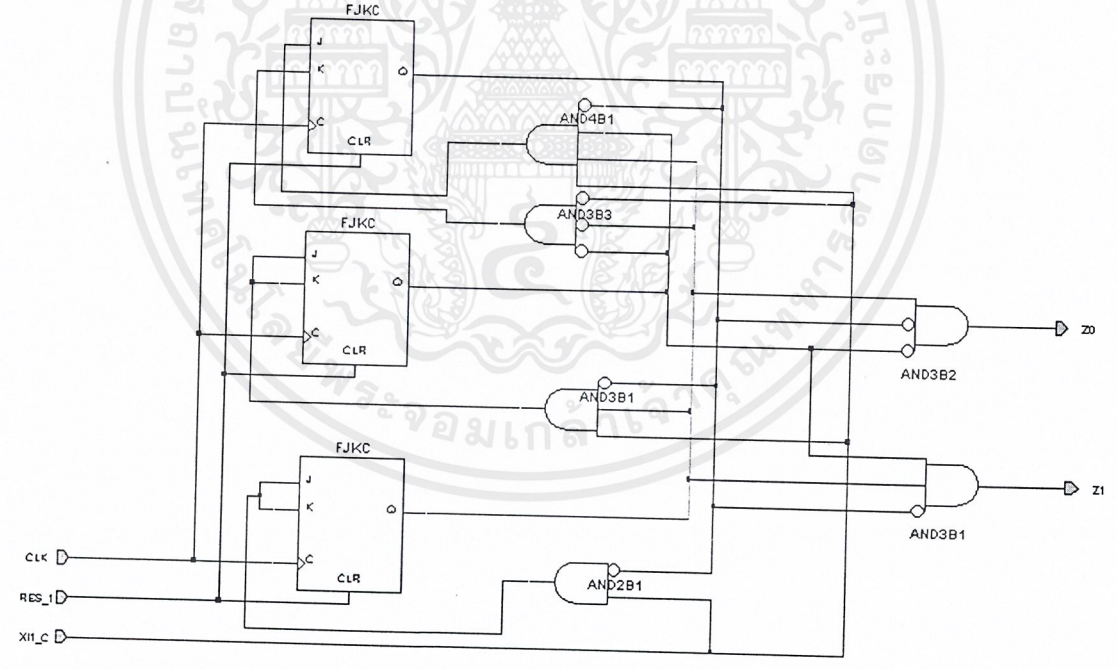
รูปที่ ข.3 วงจรกำเนิดสัญญาณนาฬิกา 1 ลูก



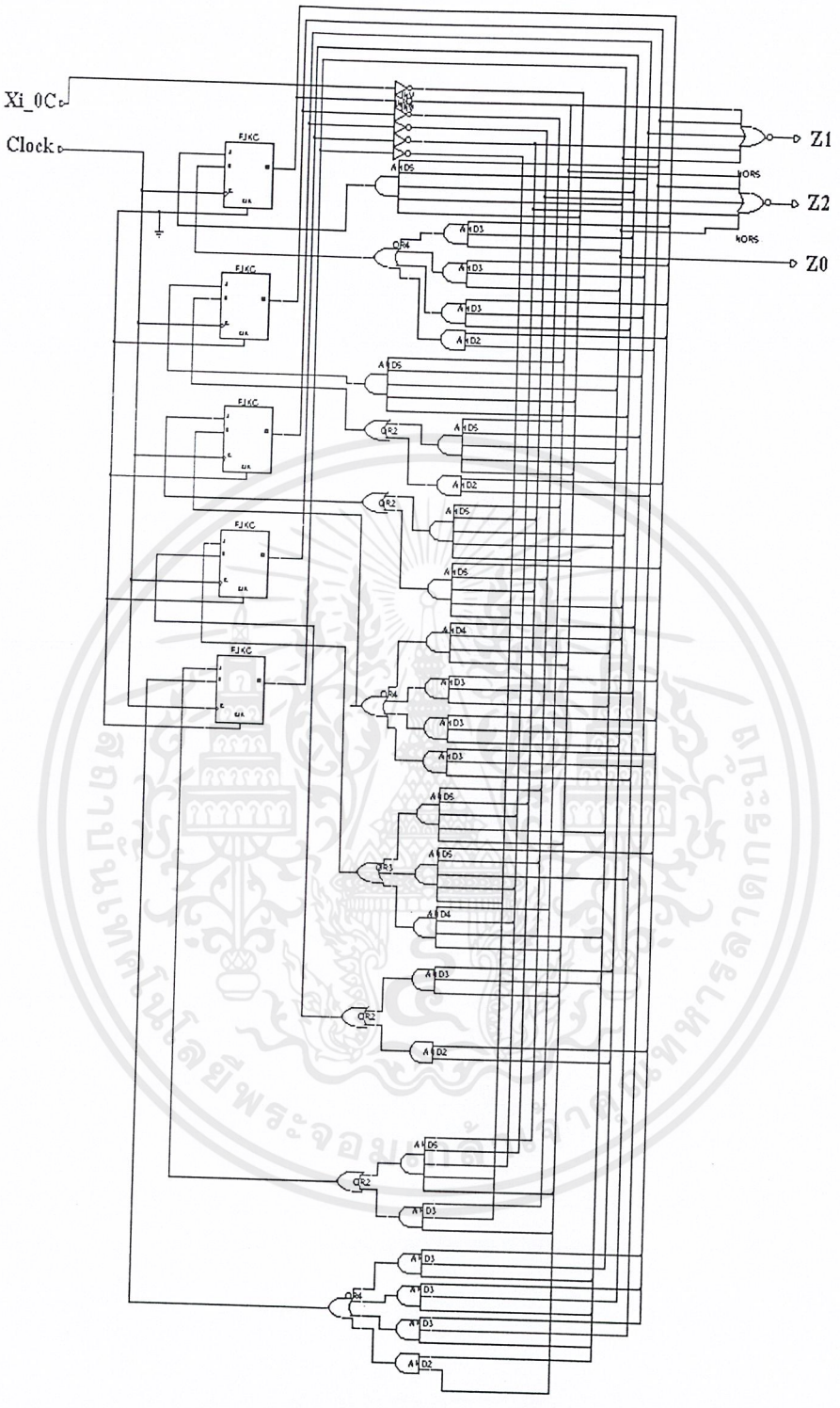
รูปที่ ข.4วงจรตัวนับ 0 ถึง 255



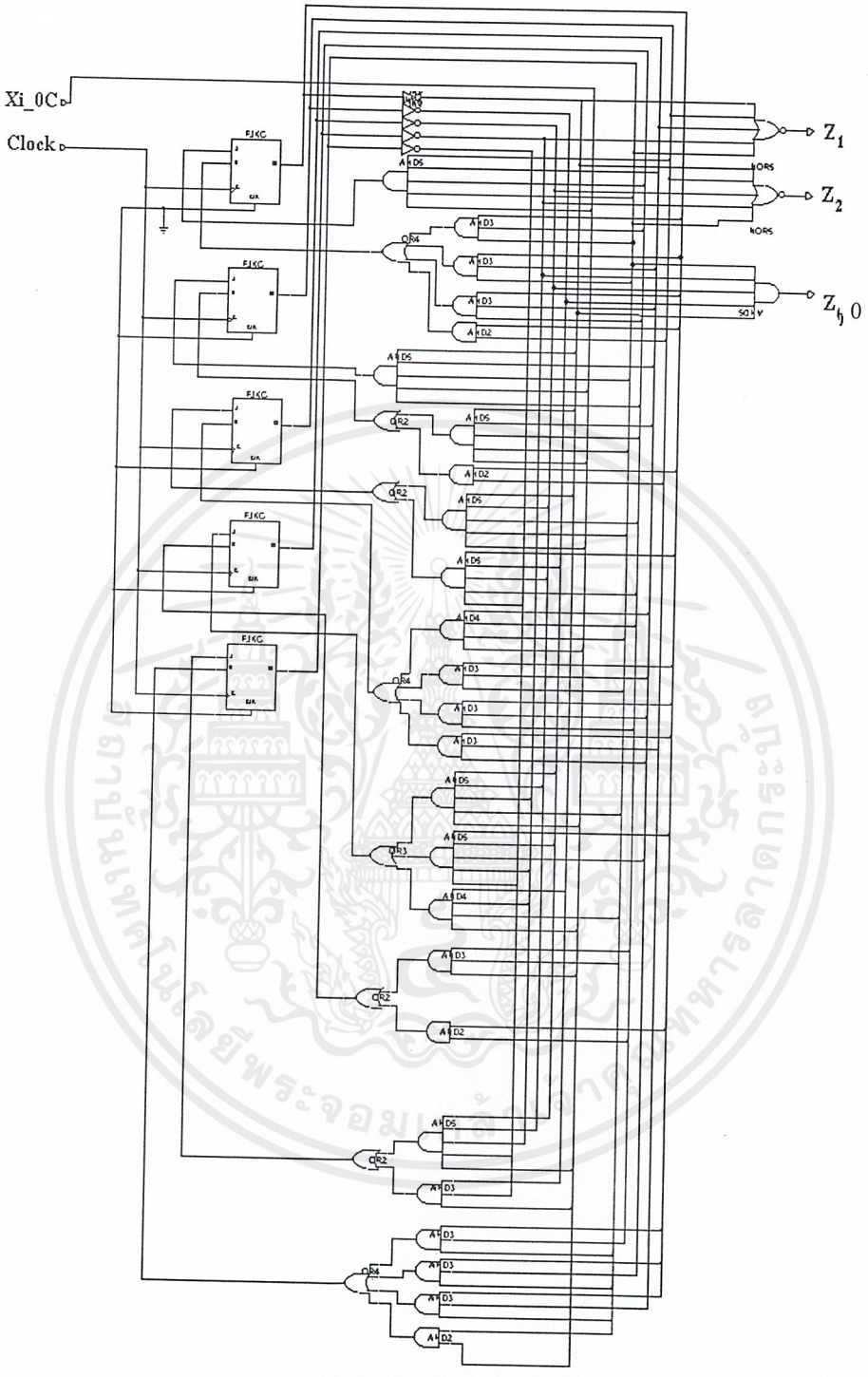
รูปที่ ข.5 วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00000H ลงในบัฟเฟอร์ 0



รูปที่ ข.6 วงจรควบคุมการโหลดข้อมูลตำแหน่งที่ 00001H ถึง 1FFFFH ลงในบัฟเฟอร์ 1



รูปที่ ข.7 วงจรสร้างสัญญาณนาฬิกา 9 ลูก



รูปที่ ข. 8 การออกแบบวงจรสร้างสัญญาณนาฬิกา 1 ลูก

บรรณานุกรม

ชินภัทร นันทจิวากรชัย, **การลดขนาดข้อมูลเสียงภาพโดยใช้เวฟเล็ตทรานส์ฟอร์ม**, วิทยานิพนธ์วิศวกรรมศาสตร์มหาบัณฑิต สาขาวิชาวิศวรรมไฟฟ้า บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2542.

Multirate Systems and Filter Banks, PTR Prentice Hall, Englewood Cliffs, New Jersey

Randy K. Young, **Wavelet Theory and Its Applications**, Kluwer Acadamic Publishers.

Speech and Audio Coding for Wireless and Network Applications, Kluwer Acadamic Publishers.

Steven W. Smith, **The Scientist and Engineer's Guide to Digital Signal Processing**, California Technical Publishing

Xlinx Inc, **Schematic-Base Programamble Logic Training Course**, 1996

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายเทพารักษ์ พุทธรักษ์
วันเดือนปีเกิด	17 สิงหาคม พ.ศ. 2522
สถานที่เกิด	จังหวัดเลย
ภูมิลำเนาเดิม	29 หมู่ 3 บ้านหัวนา ตำบลแสงภา อำเภอนาแห้ว จังหวัดเลย 42170
ที่อยู่ปัจจุบัน	94/1 หมู่ 7 ซอย 71 ถนนรามอินทรา แขวงบางชัน เขตคลองสามวา กรุงเทพฯ 10510
โทรศัพท์	0-2943-3237
ประวัติการศึกษา	
ประถมศึกษา	โรงเรียนบ้านแสงภา
มัธยมศึกษาตอนต้น	โรงเรียนนาแห้ววิทยา
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคเลย
ประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.)	วิทยาลัยเทคนิคเลย
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง
ผลงานที่ได้รับ	รางวัลชมเชยสิงประดิษฐ์คนรุ่นใหม่
ทุนการศึกษา	-
คติพจน์	เราไม่ถูกทุกอย่าง ไม่ผิดทุกอย่างทำไปแก้ไป

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายปิยะ ศรีปทุมภรณ์
วันเดือนปีเกิด	21 ตุลาคม พ.ศ. 2521
สถานที่เกิด	จังหวัดอุบลราชธานี
ภูมิลำเนาเดิม	9 ซอยชยางกูร 30 ถนนชยางกูร ตำบลในเมือง อำเภอเมือง จังหวัดอุบลราชธานี 34000
ที่อยู่ปัจจุบัน	9 ซอยชยางกูร 30 ถนนชยางกูร ตำบลในเมือง อำเภอเมือง จังหวัดอุบลราชธานี 34000
โทรศัพท์	0-4531-2941
ประวัติการศึกษา	
ประถมศึกษา	โรงเรียนอัสสัมชัญอุบลราชธานี
มัธยมศึกษาตอนต้น	โรงเรียนศรีปทุมพิทยาคาร
ประกาศนียบัตรวิชาชีพ (ปวช.)	โรงเรียนเทคโนอาชีวศึกษา
ประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.)	วิทยาลัยเทคนิคอุบลราชธานี
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง
ทุนการศึกษา	-
คติพจน์	อิสระทำให้เราสามารถที่จะจินตนาการถึงสิ่งต่าง ๆ ได้ ความหวังทำให้เรากล้าที่จะคิดถึงสิ่งที่อยู่ไกล

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายอนุวัฒน์ ธารเอี่ยม
วันเดือนปีเกิด	14 เมษายน พ.ศ. 2522
สถานที่เกิด	จังหวัดกาฬสินธ์
ภูมิลำเนาเดิม	163 หมู่ 14 ตำบลหลุบ อำเภอเมือง จังหวัดกาฬสินธุ์ 46000
ที่อยู่ปัจจุบัน	261/139 ยุคลรัตน์คอนโดมิเนียม ซอยทับยาว เขตลาดกระบัง กรุงเทพฯ 10520
โทรศัพท์	0-9105-6273
ประวัติการศึกษา	
ประถมศึกษา	โรงเรียนกาฬสินธุ์พิทยาสัย
มัธยมศึกษาตอนต้น	โรงเรียนเมืองกาฬสินธุ์
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยสารพัดช่างกาฬสินธุ์
ประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.)	วิทยาลัยเทคนิคกาฬสินธุ์
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง
ผลงานที่ได้รับ	รางวัลชมเชยสิงประดิษฐ์คนรุ่นใหม่
ทุนการศึกษา	-
คติพจน์	“Wisdom can never be communicated. It has to be experienced”

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายสุรินทร์ แท่นรัตน์
วันเดือนปีเกิด	4 กุมภาพันธ์ พ.ศ. 2523
สถานที่เกิด	จังหวัดพระนครศรีอยุธยา
ภูมิลำเนาเดิม	51 หมู่ 10 ตำบลปากกราน
	อำเภอพระนครศรีอยุธยา
	จังหวัดพระนครศรีอยุธยา 13000
ที่อยู่ปัจจุบัน	51 หมู่ 10 ตำบลปากกราน
	อำเภอพระนครศรีอยุธยา
	จังหวัด พระนครศรีอยุธยา 13000
โทรศัพท์	0-1780-3692
ประวัติการศึกษา	
ประถมศึกษา	โรงเรียนวัดปากกราน
มัธยมศึกษาตอนต้น	โรงเรียนอยุธยาวิทยาลัย
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคพระนครศรีอยุธยา
ประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.)	วิทยาลัยเทคนิคพระนครศรีอยุธยา
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์
	ภาควิชาครุศาสตร์วิศวกรรม
	คณะครุศาสตร์อุตสาหกรรม
	สถาบันเทคโนโลยีพระจอมเกล้า
	เจ้าคุณทหารลาดกระบัง
ผลงานที่ได้รับ	รางวัลชมเชยสิ่งประดิษฐ์คนรุ่นใหม่
ทุนการศึกษา	ทุนยกเว้นหน่วยกิต
คติพจน์	ความอึด คือ สิ่งพิชิตอุปสรรค