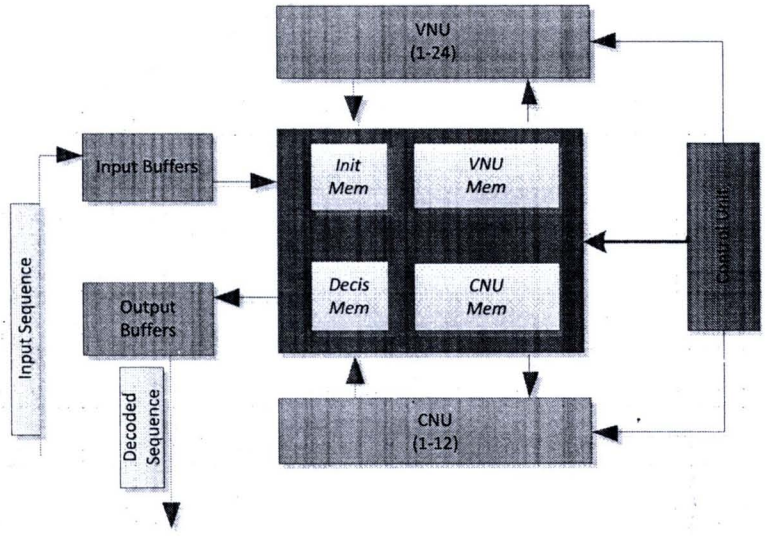
บทที่ 4

การออกแบบโครงสร้างภายในเอฟพีจีเอ

ในส่วนนี้ เราจะอธิบายถึงการออกแบบโครงสร้างภายในของการถอดรหัสแอลดีพีซี สำหรับการนำไปใช้งานบนแอฟพีจีเอการออกแบบสำหรับแอฟพีจีเอของเรานั้นจะเป็นการออกแบบในระดับ RTLหรือในระดับเกต เพื่อส่งผลให้การสังเคราะห์ของโปรแกรมมีการใช้งานที่เกตต่ำๆ โดยจะเริ่มออกแบบเป็นโมดูลย่อยแล้วจึงนำมารวมกันเป็นระบบ ขนาดของคำรหัสที่เรานำมาออกแบบมีขนาด 648 บิตอัตรารหัส ½ มีขนาดบิตที่ 6 บิต ส่วนประกอบหลักในการออกแบบเริ่มแรกคือหน่วยประมวล CNU และ VNU เมื่อทำการออกแบบจนได้โครงสร้างที่เหมาะสมแล้ว จากนั้นทำการออกแบบหน่วยความจำเพื่อใช้เก็บข้อมูลจากหน่วยประมวลผล ส่วนถัดมาคือหน่วยอินพุตบัฟเฟอร์และเอาต์พุตบัฟเฟอร์ เมื่อออกแบบโครงสร้างดังกล่าวแล้วจึงนำโครงสร้างทั้งหมดมารวมกันและทำการออกแบบหน่วยควบคุมเพื่อให้ระบบทำงานสอดคล้องกัน

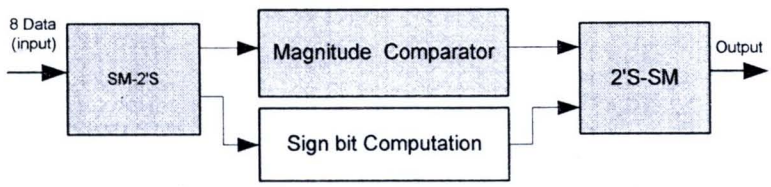


รูปที่4.1โครงสร้างระบบการถอดรหัสแอลดีพีซี

ในรูปที่ 4.1จะแสดงถึงโครงสร้างการถอดรหัสของแอลดีพีซีโดยสามารถใช้ได้ทั้งวิธีการผลรวมต่ำสุด (MS)และวิธีการผลรวมต่ำสุดดัดแปลง (MMS)โดยจะมีโครงสร้างหลักอยู่ 4 โครงสร้างคือ หน่วยประมวลผลบิตโหนด (VNU) หน่วยประมวลผลเช็คโหนด (CNU) หน่วยความจำ (memory unit) และ หน่วยควบคุม (Control unit) โดยหน่วยความจำจะทำการเก็บข้อมูลที่คำนวณได้จากหน่วยประมวลผล บิตโหนดกับเช็คโหนด ซึ่งขนาดของหน่วยความจำจะขึ้นอยู่กับขนาดของเมตริกซ์พาริตี้เช็คที่นำมาใช้ ส่วนอินพุตบัฟเฟอร์ (input buffer unit) กับเอาต์พุตบัฟเฟอร์ (output buffer unit) จะทำหน้าที่สลับตำแหน่ง (permutation) ของคำรหัสที่เข้ามาและสลับตำแหน่งคืน (de-permutation) หลังจากทำการถอดรหัสเสร็จสิ้น

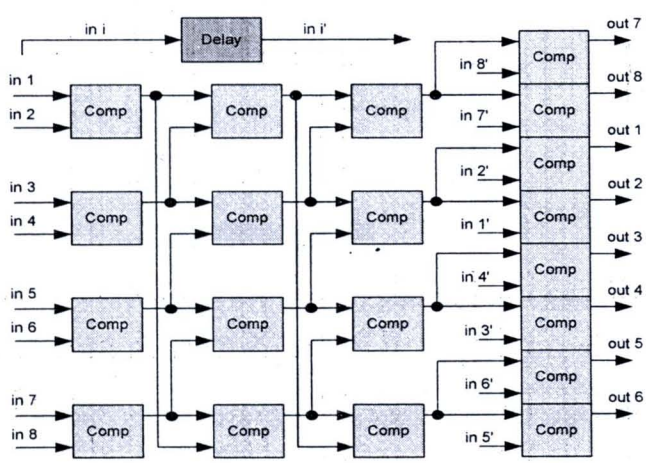
4.1 หน่วยประมวลผลเช็คโหนด (CNUs Unit)

หน่วย CNU จะทำหน้าที่ในการประมวลผลของแต่หละแถว ซึ่งเป็นไปตามสมการที่ 2.53โดยจะทำการหาค่าต่ำสุดจากทั้งหมด 24 ตำแหน่ง ในแถวนั้นๆ แต่อย่างไรก็ตามเนื่องจากในแถวจะประกอบไปด้วยค่าศูนย์กับค่าหนึ่ง ซึ่งในตำแหน่งที่มีค่าศูนย์จะไม่นำมาคิด จึงมีเพียง 7-8 ตำแหน่งต่อแถวเท่านั้นที่นำมาพิจารณา



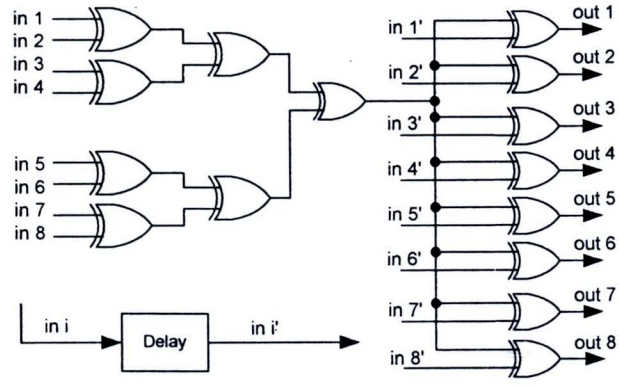
รูปที่4.2โครงสร้างของหน่วยประมวลผล CNU

ในแต่หละตำแหน่งจะมีขนาดของบิตอยู่ที่ 6 บิต ซึ่งรวมบิตเครื่องหมายด้วย โดยอินพุตที่เข้ามาจะเป็นระบบเลข 2's complement จากการแสดงในรูปที่ 4.2เมื่ออินพุตเข้ามาจะทำการเปลี่ยน 2's complement ให้แยกออกเป็นสองส่วนคือ ขนาดบิต (magnitude bits) กับ บิตเครื่องหมาย (sign bit) โดยขนาดบิตจะถูกนำไปเปรียบเทียบกันเพื่อหาค่าต่ำสุด รูปแบบของเส้นทางในการเปรียบเทียบสามารถดูได้จากรูปที่ 8 ซึ่งจำนวนของการเปรียบเทียบจะมีทั้งหมด 20 บล็อก แต่หละบล็อกจะเปรียบกัน 2 อินพุต และออก 1 เอาต์พุต สำหรับแถวที่มีค่าไม่ใช่ศูนย์อยู่ 7 ตำแหน่ง ในจำนวน 8 อินพุตจะมีหนึ่งอินพุตที่ถูกตั้งค่าให้เป็นค่าสูงสุดเพื่อละเลยในการเปรียบเทียบ



รูปที่4.3เส้นทางในการเปรียบเทียบเพื่อหาค่าต่ำสุด

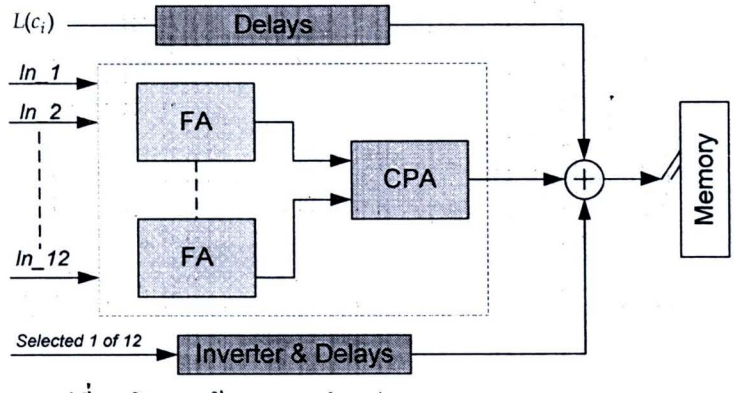
การคำนวณของบิตเครื่องหมายสามารถคำนวณได้โดยง่าย ซึ่งแสดงให้เห็นดังรูปที่ 4.4จะเป็นวงจรที่ประกอบไปด้วยเกต XORเท่านั้น



รูปที่4.4โครงสร้างเพื่อการคำนวณหาบิตเครื่องหมาย (sign bit)

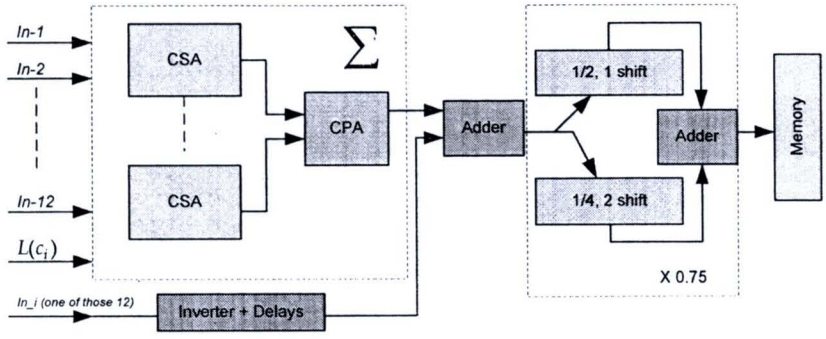
4.2 หน่วยประมวลผลบิตโหนด (VNUs Unit)

ในส่วนของโครงสร้างหน่วยประมวลผลบิตโหนดเราได้ออกแบบให้รองรับวิธีการถอดรหัสเป็นสองวิธีเพื่อที่จะสามารถใช้ได้ทั้งวิธีการผลรวมต่ำสุด (MS) และวิธีการผลรวมต่ำสุดดัดแปลง (MMS) และยังได้พัฒนาโครงสร้างของ VNUที่ใช้วิธีการ MMS เพื่อเป็นอีกทางเลือกหนึ่งในการใช้งาน



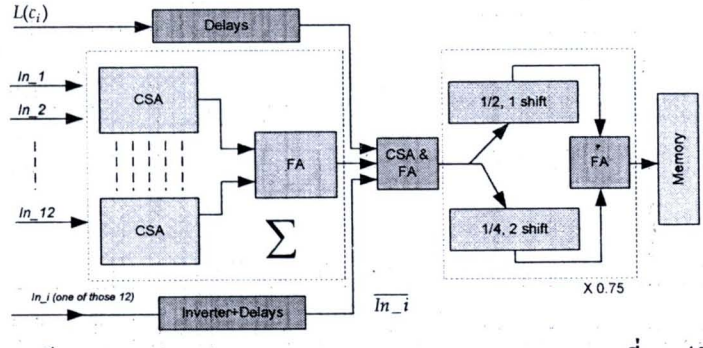
รูปที่4.5โครงสร้างของหน่วยประมวลผล VNUs min-sum

การถอดรหัสแอลดีพีซีด้วยวิธีการผลรวมต่ำสุดจะเป็นการคำนวณตามสมการที่ 2.54 โดยสามารถแสดงให้เห็นถึงโครงสร้างภายในได้ดังรูปที่ 4.5 ซึ่งการคำนวณจะเป็นการหาผลรวมของทั้ง 12อินพุต ที่อยู่ในแต่หละหลักมาคำนวณ เริ่มจากการนำเอาจำนวนอินพุตทั้งหมดนำมาบวกกันก่อน และในขณะเดียวกันอินพุตทั้งหมดจะถูกแยกไปผ่านอินเวอร์เตอร์เพื่อกลับเครื่องหมายจากนั้นนำไปบวกผลรวมที่ได้ก่อนหน้านี้และรวมกับค่า Lciซึ่งจะทำให้เราจะได้ผลลัพธ์ในตำแหน่งนั้นๆ



รูปที่4.6โครงสร้างของหน่วยประมวลผล VNUs modified min-sum

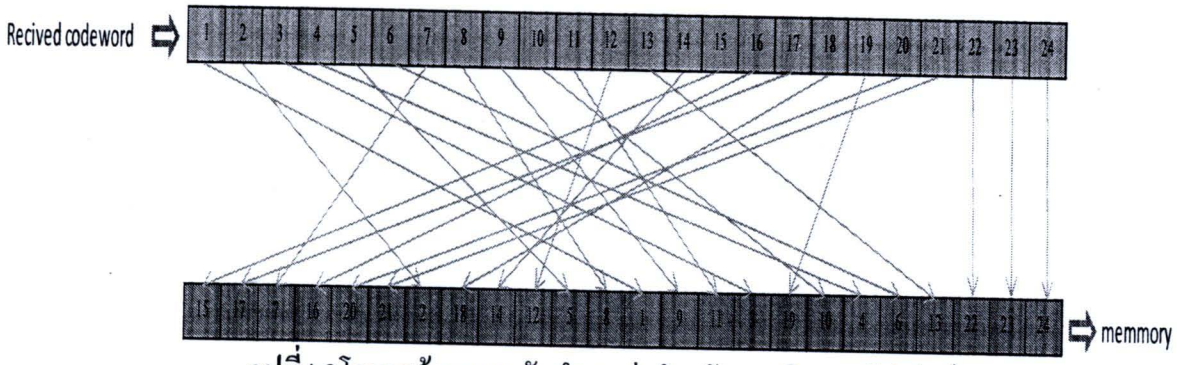
ในส่วนของวิธีการผลรวมต่ำสุดดัดแปลง (MMS) หน่วย VNU จะเป็นการนำสมการที่ 2.57 มาใช้งาน โดยสามารถแสดงให้เห็นถึงโครงสร้างภายในได้ดังรูปที่ 4.6 ซึ่งการหาผลรวมของทั้ง 12 ตำแหน่งในหลักมาคำนวณ โดยขั้นตอนแรกจะนำเอาจำนวนอินพุตทั้งหมดนำมาบวกกันทั้งหมดรวมทั้งค่า Lci อย่างไรก็ตามเมื่อเราพิจารณาหาค่าที่ตำแหน่งนั้นๆ จะนำค่าที่ตำแหน่งนั้นมาลบกับผลรวมในขั้นตอนแรกโดยการผ่านแอดเดอร์ซึ่งจะได้ผลลัพธ์ในตำแหน่งนั้นๆ วงจรในการรวมค่าทั้ง 12 อินพุตเราจะใช้ CSA 10 ยูนิต ตามด้วยยูนิตCPA 1 ยูนิต โดยผลลัพธ์ที่นำไปใช้งานจริงจะถูกคำนวณผ่าน scaling factor อีกหนึ่งขั้นตอนโดยมีค่าสเกลเท่ากับ 0.75 ด้วยวิธีการเลื่อนบิตและบวกกันโดยตรง



รูปที่4.7โครงสร้างของหน่วยประมวลผล VNUs modified min-sumที่ถูกปรับปรุงแล้ว

ความเร็วในการทำงานสูงสุดของหน่วย VNU ตามรูปที่ 4.6 จะมีความเร็วที่สูงกว่าหน่วย CNU ในทางทฤษฎีการทำงานของ CNU และ VNU จะทำงานในรูปแบบขนาน อย่างไรก็ตามสำหรับรูปแบบที่เราสร้างเป็นพิเศษเราจะต้องใช้หนึ่ง CNU และหนึ่ง VNU ในการทำงานแต่หละรอบ เราสามารถปรับความเร็วให้เหมาะสมกันโดยพิจารณาจากรูปแบบที่ 4 (pattern 4) ในรูปที่ 3.5 จะเห็นว่าการทำงานของ CNU ในบล็อกของแถวที่ 7 กับ 8 จะต้องทำงานพร้อมกันทั้งสองแถวในหนึ่ง CNU และหน่วย VNU จะทำงานที่หลักที่ 1-3 ขนานกันไปด้วย ดังนั้นการทำงานของ VNU ควรจะทำงานเร็วเป็น 1.5 เท่า ของหน่วย CNU เมื่อพิจารณาได้ดังนี้หน่วย VNU จึงถูกออกแบบใหม่เพื่อเพิ่มความเร็วให้ได้ตามความต้องดังแสดงในรูปที่ 4.7

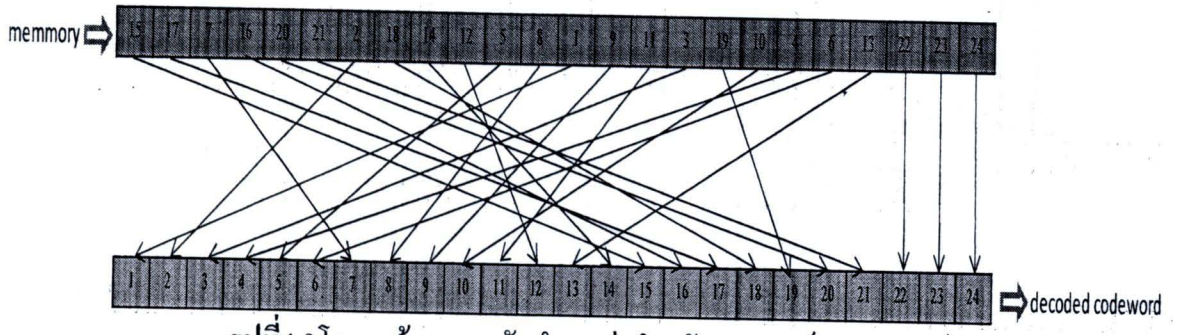
4.3 หน่วยอินพุตบัฟเฟอร์(Input Buffer Unit)



รูปที่4.8โครงสร้างการสลับตำแหน่งคำรหัสของอินพุทบัฟเฟอร์

ในส่วนของโครงสร้างอินพุตบัฟเฟอร์จะทำหน้าสลับตำแหน่งของคำรหัสที่รับเข้ามาเพื่อใช้ในการคำนวณให้เป็นตามดังเมตริกซ์ที่เราได้ออกแบบไว้ข้างต้น ดังแสดงให้ในรูปที่ 4.8เมื่อเรารับคำรหัสจากช่องสัญญาณแล้ว จากนั้นจะนำคำรหัสที่ได้มาเข้ากระบวนการเปลี่ยนตำแหน่ง เมื่อเปลี่ยนตำแหน่งเรียบร้อยแล้วจะนำคำรหัสที่ได้ไปเก็บลงในหน่วยความจำเพื่อทำการประมวลผลต่อไป

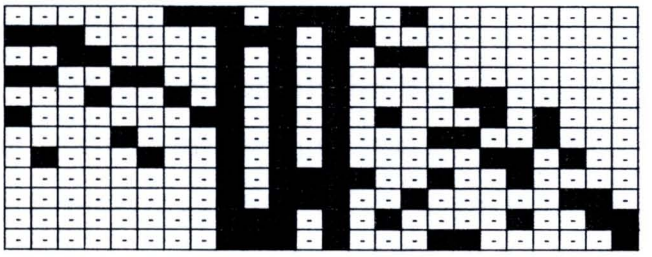
4.4 หน่วยเอาต์พุตบัฟเฟอร์(Output Buffer Unit)



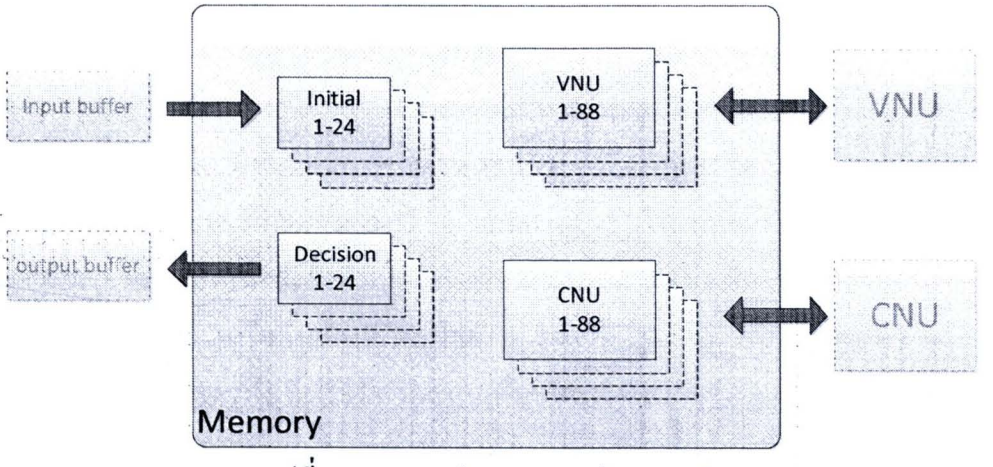
รูปที่4.9โครงสร้างการสลับตำแหน่งคำรหัสของเอาต์พุตบัฟเฟอร์

ในส่วนของโครงสร้างเอาต์พุตบัฟเฟอร์จะทำหน้าสลับตำแหน่งกลับตำแหน่งเดิมจากที่เราได้สลับในส่วนแรก เพื่อให้ได้คำรหัสที่ได้ถูกต้อง ดังแสดงให้ในรูปที่ 4.9เมื่อเราประมวลผลจนเป็นที่เรียบร้อยแล้วคำรหัสที่ได้นั้นยังมีตำแหน่งที่ไม่ถูกต้องเพราะว่าเราได้สลับตำแหน่งไปก่อนหน้านี้ ฉะนั้นเราจะนำคำรหัสจากหน่วยความจำมาสลับตำแหน่งคืนดังเดิมให้ได้ตามเมตริกซ์พื้นฐาน เมื่อเปลี่ยนตำแหน่งเรียบร้อยแล้วเราก็จะได้คำรหัสที่ถูกต้องตามมาตราฐาน

4.5 หน่วยความจำ (Memmory Unit)



รูปที่4.10 เมตริกซ์พาริตี้เช็คที่ใช้ในการถอดรหัส



รูปที่4.11แสดงรูปแบบของหน่วยความจำ

จากรูปที่ 4.11จะแสดงถึงจำนวนของหน่วยความจำเพื่อใช้เก็บข้อมูล โดยโครงสร้างภายในจะเป็น Look up table (LUT) เนื่องจากการประมวลผลที่เราได้ออกแบบไว้เป็นแบบกึ่งขนาน ทำให้การออกแบบโครงสร้างต้องแยกออกเป็นบล็อคตามเมตริกซ์พาริตี้เช็คดังรูปที่ 4.10 เพื่อให้อ่านและเขียนลงหน่วยความจำสามารถทำงานในรูปแบบขนานได้เช่นเดียวกัน โดยแต่หละบล็อกจะมีตำแหน่งเก็บข้อมูลอยู่ 27 ตำแหน่ง และมีความกว้างของบิตที่ 6 บิต ในส่วนของหน่วยความจำหลังจากผ่านอินพุตบัฟเฟอร์คือหน่วยความจำเก็บค่าเริ่มต้น (Lci) จะมีจำนวน 24 บล็อก ถัดมาเป็นหน่วยความจำเพื่อเก็บค่าจากการประมวลผล VNU และ CNU โดยแต่หละหน่วยจะมีจำนวนหน่วยเก็บข้อมูลอยู่ 88 บล็อก เมื่อทำการประมวลผลเสร็จเรียบร้อยหน่วยความจำสุดท้ายคือการเก็บข้อมูลจากการประมวลผลเพื่อตัดสินใจมีจำนวน 24 บล็อก

4.6 หน่วยควบคุม (Control Unit)

หน่วยควบคุมจะเป็นหน่วยที่ควบคุมการทำงานของระบบทั้งหมด เมื่อรับคำรหัสผ่านช่องสัญญาณมาแล้ว ข้อมูลที่เก็บในอินพุตบัฟเฟอร์จะถูกเรียกออกมาเพื่อทำการสลับตำแหน่งให้ถูกต้องตามเมตริกซ์พาริตี้เช็คที่ได้ออกแบบไว้ เมื่อทำการสลับตำแหน่งเรียบร้อยแล้วค่าที่ได้จะนำไปเป็นเก็บลงในหน่วยความจำค่าเริ่มต้น (Lci) ซึ่งอยู่ในขั้นตอนของการเซ็ทค่าตั้งต้น (Initial) จากนั้นหน่วยควมคุมจะเริ่มสั่งให้ระบบประมวลผลทำงานโดยการโหลดค่าLciลงในหน่วยความจำ VNUs เพื่อนำค่าเรานี้ไปทำตามสมการที่ 2.53 ซึ่งเป็นการประมวลของ CNUs เมื่อทำการประมวลเช็คโหนดเสร็จแล้วจะนำค่าไปเก็บลงในหน่วยความจำ CNUs จากนั้นหน่วยควมคุมจะสั่งทำการประมวลผลที่บิตโหนดตามสมการ 2.54 (ถ้าเป็นวิธีการผลรวมต่ำสุดดัดแปลงจะทำตามสมการ 2.57)เมื่อประมวลผล VNUs เรียบร้อยจะทำการเก็บข้อมูลลงหน่วยความจำ VNUs เช่นเดียวกัน ซึ่งการกระทำเช่นนี้จะเป็นการประมวลผลเสร็จหนึ่งรอบ เมื่อเริ่มทำการประมวลผลในรอบที่ถัดไป หน่วยควบคุมไม่จำเป็นต้องสั่งให้อ่านข้อมูลจากค่า Lciมาเก็บลงในหน่วยความจำ VNUs อีกเพราะเราได้ข้อมูลจากการประมวลในรอบที่ผ่านมาเรียบร้อยแล้ว จากนั้นหน่วยควบคุมจะสั่งให้ประมวลผลตามขั้นตอนการถอดรหัสจนกระทั่งถึงรอบสูงสุดที่ตั้งไว้ จากนั้นหน่วยควบคุมจะสั่งให้ทำการตัดสินใจแบบระเอียดสมการ 2.55 และการตัดสินใจแบบหยาบสมการ 2.56 และส่งผ่านข้อมูลไปยังหน่วยเอาต์พุตบัฟเฟอร์เพื่อทำการสลับตำแหน่งคืนตำแหน่งเดิม เมื่อทำตามขั้นตอนที่ผ่านมาทั้งหมดถือว่าการถอดรหัสนั้นเสร็จสิ้นเป็นที่เรียบร้อย

จากระบบการถอดรหัสที่เราได้ออกแบบเพื่อนำไปใช้บน FPGAจะเห็นได้ว่าเราได้ละเลยการคำนวณของขั้นตอนพาริตี้เช็ค $H\cdot(\hat{c}_1,\hat{c}_2,...,\hat{c}_n)^T=0$ เนื่องจากการทำขั้นตอนนี้จะมีความซับซ้อนเพิ่มมากยิ่งขึ้นเพราะต้องนำผลลัพท์ทั้งสองค่ามาคูณกันมีผลทำให้ขนาดของฮาร์ดแวร์ใหญ่มากขึ้นจากเดิมเป็นจำนวนมาก และอีกประการหนึ่งคือเมื่อทำการประมวลผลจนครบรอบสูงสุดที่กำหนดไว้การถอดรหัสถือว่าเสร็จสิ้นจึงไม่จำเป็นต้องทำขั้นตอนนี้ เพียงแต่ว่าเมื่อมีการถอดรหัสที่ถูกต้องเสร็จก่อนระบบยังคงทำงานต่อไปทำให้เวลาการถอดรหัสช้าลงแต่ระบบยังคงมีความเร็วเป็นไปตามมาตราที่เรานำมาใช้