บทที่ 4

การออกแบบฮาร์ดแวร์ค้นหาขอบภาพโดยแนวทางวิธีของ Canny

ในบทนี้จะเกี่ยวกับการออกแบบวงจรฮาร์ดแวร์ดิจิตอลจากขั้นตอนวิธีของ Canny โดยจะเริ่มจากแนวความคิดในการแบ่งส่วนประกอบของฮาร์ดแวร์ออกเป็นส่วนๆตามหน้าที่ในแต่ละขั้นตอนของ Canny

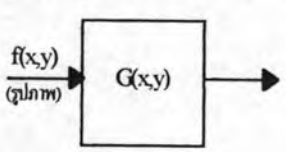
หลังจากนั้นจึงทำการออกแบบตามหน้าที่ที่ได้แบ่งไว้ แต่ในทุกหน้าที่ต้องมีการแก้ไขการคำนวณ เช่น ไม่มีการคำนวณค่าทศนิยมและการปัดเศษทศนิยม เป็นต้น เพื่อให้การออกแบบเป็นฮาร์ดแวร์นั้นสามารถกระทำได้สะดวกขึ้น แต่ขั้นตอนที่ 5 นั้นไม่เหมาะสมที่จะออกแบบเป็นฮาร์ดแวร์ เพราะต้องอาศัยการเวียนเกิด ดังนั้นขั้นตอนที่ 5 จะใช้เป็นซอฟต์แวร์แทน

การออกแบบจะใช้พอร์ตขนานของคอมพิวเตอร์หนึ่งพอร์ตเป็นตัวรับส่งข้อมูลกับฮาร์ดแวร์ โดยการติดต่อจะเป็นแบบอะซิงโครนัส(asynchronous)

หลักการออกแบบฮาร์ดแวร์

การออกแบบฮาร์ดแวร์จะแยกออกเป็นวงจรย่อยตามขั้นตอนวิธีของ Canny ดังนี้

1) ขั้นตอนกรองเอาความถี่ต่ำหรือการเกลี่ยภาพ



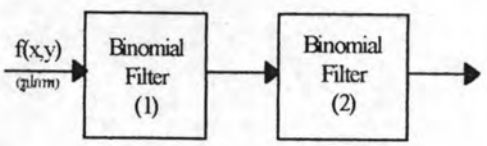
การหาค่าจุดภาพหลังจากผ่านการคอนโวลูชันกับตัวกรองความถี่แบบ G(x,y) (คือฟังก์ชันเกาส์เซียน) เราจะใช้ตัวกรองความถี่แบบไบโนเมียลแทน เพื่อลดความซับซ้อนของการคำนวณ, ทำให้เกิดความรวดเร็วในการคำนวณ และสามารถนำมาออกแบบเป็นฮาร์ดแวร์ได้ง่าย โดยผลที่ได้ยังคงมีค่าใกล้เคียงกับการใช้ตัวกรองความถี่แบบ G(x,y) ซึ่งตัวกรองความถี่แบบไบโนเมียลที่ใช้ได้นำมาจากของ ยุทธพงษ์ รังสรรค์เสรี ซึ่งเป็นหน้ากากเพื่อใช้คอนโวลูชันขนาด 3 x 3 ที่ให้ผลตอบสนองได้ใกล้เคียงกับตัวกรองความถี่แบบ G(x,y) ซึ่งมีรูปแบบดังนี้

$$f(x,y,n)=\frac{1}{16^n}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}^{*n}$$

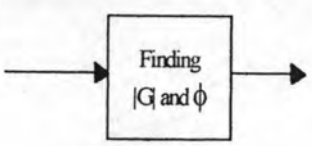
โดยให้สัญลักษณ์ '*n' แทนการทำคอนโวลูชันของตัวหน้ากากกับรูปภาพ n ครั้ง โดยที่ n เป็นพารามิเตอร์ที่ปรับค่าได้ เทียบกับ σ ในฟังก์ชันเกาส์เซียนได้ดังนี้

$$\sigma^2 = 0.5n$$

นั่นคือถ้าต้องการ σ เท่ากับ 1 ซึ่งเป็นค่าที่เราจะใช้เป็นค่าของตัวกรองความถี่ ก็ต้องใช้ n เท่ากับ 2 หรือคือต้องทำคอนโวลูชันของตัวมาสก์กับรูปภาพ 2 ครั้ง ดังนั้นจึงแยกส่วนฮาร์ดแวร์นี้ออกเป็น 2 วงจรที่มีวงจรที่เหมือนกันต่ออนุกรมกันดังแสดงข้างล่าง



2) ขั้นตอนหาค่าขนาดและทิศทางของกราเดียนต์



ขั้นตอนนี้จะทำการคำนวณหา Gy และ Gx โดยใช้หน้ากากแบบ

$$\begin{bmatrix}-1 & 0 & 1\end{bmatrix} \text{ และ } \begin{bmatrix}-1\\ 0\\ 1\end{bmatrix} \text{ ตามลำดับ}$$

แล้วมาคำนวณหาขนาดของกราเดียนต์ได้จาก

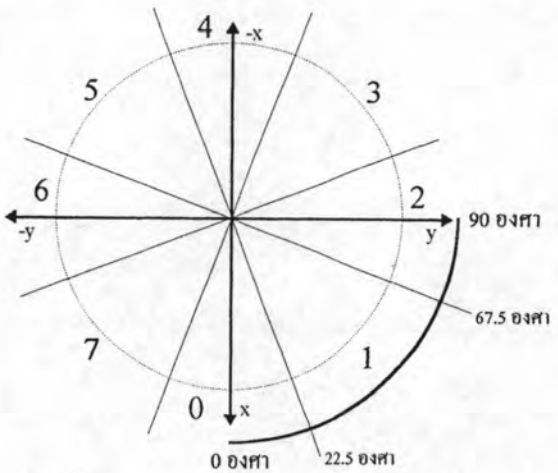
$$|G| = \sqrt{G_x^2 + G_y^2}$$

และทิศทางของกราเดียนต์ได้จาก

$$\phi = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

ซึ่งเราจะอิงการคำนวณตามรูปที่ 2.3(ก)

และทิศทางของกราเดียนต์ที่ได้เป็นมุมจะต้องถูกเปลี่ยนให้เป็นค่า 0 ถึง 7 โดยแบ่งมุม 360 องศาออกเป็น 8 ส่วน ส่วนละ 45 องศา ดังรูปที่ 4.1



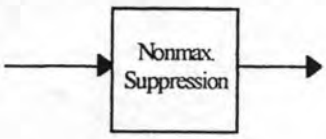
รูปที่ 4.1 แสดงการแบ่งค่ามุม ϕ ออกเป็น 8 ค่า

เริ่มจาก ทิศทางที่ 0 คือ มุมระหว่าง (-22.5, 22.5) องศา
ทิศทางที่ 1 คือ มุมระหว่าง [22.5, 67.5] องศา
ทิศทางที่ 2 คือ มุมระหว่าง (67.5, 112.5) องศา
ทิศทางที่ 3 คือ มุมระหว่าง[112.5, 157.5] องศา
ทิศทางที่ 4 คือ มุมระหว่าง(157.5, 202.5) องศา
ทิศทางที่ 5 คือ มุมระหว่าง[202.5, 247.5] องศา
ทิศทางที่ 6 คือ มุมระหว่าง(247.5, 292.5) องศา
ทิศทางที่ 7 คือ มุมระหว่าง[292.5, 337.5] องศา

ค่ามุมที่อยู่ระหว่างทิศทาง 2 ทิศทาง เช่น ค่ามุมที่ 22.5 จะอยู่ระหว่างทิศทางที่ 0 กับทิศทางที่ 1 จะกำหนดให้เป็นของทิศทางที่ 1 ดังแสดงด้วยเครื่องหมายวงเล็บสี่เหลี่ยม ที่กำหนดเป็นแบบนี้ไม่มีเหตุผลพิเศษ เพียงแต่ว่าต้องเลือกกำหนดแบบใดแบบหนึ่งออกมาเท่านั้น แต่จะทำให้มีผลต่อการหาจุดภาพที่มีขนาดสูงสุดเฉพาะถิ่น เพราะทิศทางจะเปลี่ยนไปตามวิธีที่เราเลือกใช้ แต่มีผลเพียงเล็กน้อยเท่านั้น เพราะโอกาสเกิดค่ามุมเช่นนี้มีโอกาสเกิดน้อยมากถึงไม่เกิดขึ้นเลย

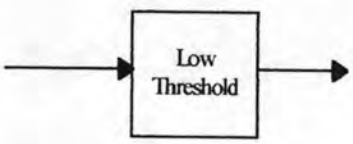
เหตุผลที่ต้องทำให้ทิศทางเป็นไปได้แค่ 0 ถึง 7 ตามรูป ก็เพื่อให้ง่ายแก่การเปรียบเทียบ และทำให้การออกแบบฮาร์ดแวร์เป็นไปได้ง่ายและทำงานได้รวดเร็ว เพราะค่าทั้ง 8 แทนตำแหน่งของจุดภาพทั้ง 8 ที่อยู่รอบจุดภาพตรงกลางที่เราต้องการหาว่าเป็นจุดภาพที่มีขนาดสูงสุดเฉพาะถิ่นหรือไม่ จึงทำให้ไม่ต้องทำการอินเตอร์โพเลชัน แบบที่ Canny ทำเพื่อหาค่าของกราเดียนต์ที่อยู่ก่อนหน้าและที่อยู่ถัดไปตามทิศทางของกราเดียนต์ที่อยู่ระหว่าง 2 จุดภาพ

3) ขั้นตอนหาจุดภาพที่มีขนาดสูงสุดเฉพาะถิ่น (Nonmax Suppression)



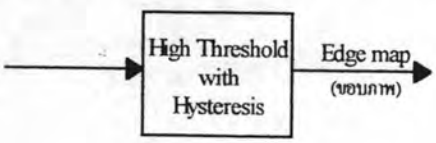
หลังจากที่ได้ค่ากราเดียนต์(|G|) และทิศทางของกราเดียนต์(ϕ) ของแต่ละจุดภาพจากขั้นตอนที่สองแล้ว ก็นำมาหาจุดภาพที่มีขนาดสูงสุดเฉพาะถิ่น โดยการนำค่า |G| ของจุดภาพขอแทนเป็น G เฉยๆ ไปเปรียบเทียบกับค่า |G| ของจุดภาพที่อยู่ถัดไปตามทิศทางของ ϕ ขอเรียกเป็น G_1 และค่า |G| ของจุดภาพที่อยู่ก่อนหน้าตามทิศทางของ ϕ ขอเรียกเป็น G_2 ซึ่งค่า G จะต้องเป็น $G > G_1$ และ $G \geq G_2$ ถึงจะแสดงว่าจุดภาพนี้เป็นตำแหน่งของขอบภาพก็ให้มีค่า G คงเดิม แต่ถ้าไม่ใช่ แสดงว่าจุดภาพนี้ไม่ใช่ตำแหน่งของขอบภาพ ก็ให้แก้ค่า G ของจุดภาพให้เป็นศูนย์

4) ขั้นตอนตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำ



จุดประสงค์ของการตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำก็เพื่อตัดค่าจุดภาพสูงสุดเฉพาะถิ่นที่มีค่าต่ำ ซึ่งมักเกิดจากสัญญาณรบกวนหรือขอบที่ไม่ชัดเจนออกไป เช่น เงาของวัตถุ หรือร่องรอยของแรเงา(shading)ของผิววัตถุ โดยการนำค่ากราเดียนต์ของจุดภาพที่เป็นจุดภาพที่มีขนาดสูงสุดเฉพาะถิ่นมาเปรียบเทียบกับค่าขีดเริ่มเปลี่ยนค่าต่ำที่เราเลือกไว้ ถ้าค่าของจุดภาพมีค่ามากกว่าหรือเท่ากับค่าขีดเริ่มเปลี่ยน ก็ให้คงค่าจุดภาพค่านี้ไว้ แต่ถ้าไม่ก็ให้เปลี่ยนค่าจุดภาพนี้เป็นค่าศูนย์

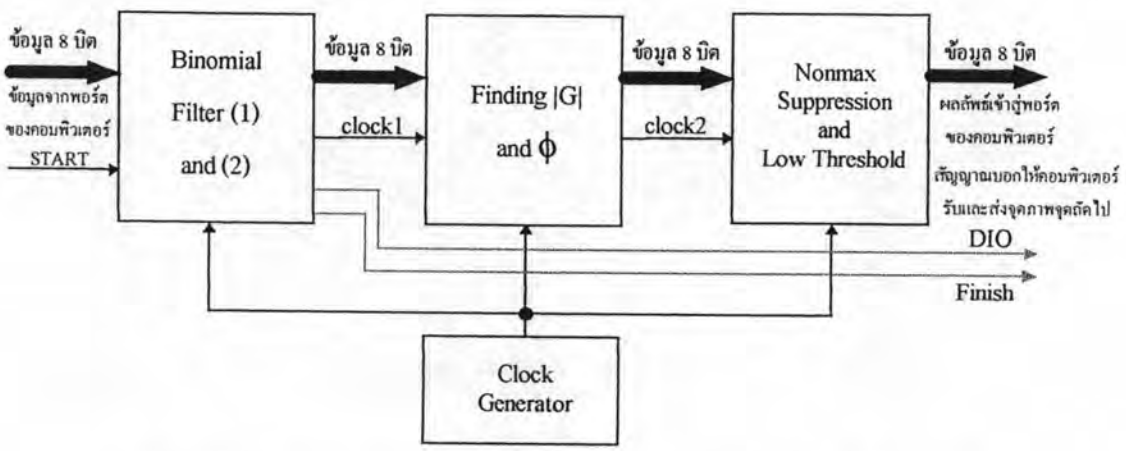
5) ขั้นตอนตรวจสอบขีดเริ่มเปลี่ยนค่าสูงแบบฮิสเตอร์เรซิส



จุดประสงค์ของการตรวจสอบขีดเริ่มเปลี่ยนค่าสูงแบบฮิสเตอร์เรซิสก็เพื่อให้ได้ลายเส้นขอบที่ต่อเนื่อง ซึ่งมีหลักการดังที่ได้อธิบายไว้ในบทที่ 3 ในหัวข้อ *"การหาขอบภาพของ Canny"* ซึ่งวิธีการที่สมบูรณ์สำหรับการตรวจสอบขีดเริ่มเปลี่ยนแบบฮิสเตอร์เรซิส จะต้องอาศัย

การคำนวณโดยทฤษฎีกราฟ ดังนั้นการออกแบบเป็นฮาร์ดแวร์จะต้องใช้หน่วยความจำและวงจรลอจิกเป็นจำนวนมากถึงจะทำได้ เพราะเป็นลักษณะการทำงานแบบค้นหา ดังนั้นจึงไม่ออกแบบขั้นตอนนี้เป็นฮาร์ดแวร์ โดยให้ขั้นตอนนี้ถูกรวมอยู่ในซอฟต์แวร์แทน

ดังนั้นจาก 4 ขั้นตอนแรกของ Canny (ไม่รวมขั้นตอนที่ 5) จึงนำมาออกแบบเป็นฮาร์ดแวร์ได้เป็น 3 วงจรต่อเรียงกันดังรูปที่ 4.2 ข้างล่าง โดยขั้นตอนที่ 3 และ 4 ของ Canny จะอยู่รวมกันเป็นวงจรที่ 3



รูปที่ 4.2 แสดงแผนภาพบล็อก(block diagram)ของวงจรฮาร์ดแวร์ Canny

ทั้ง 3 วงจรจะต่ออนุกรมเรียงกันไป เพราะเอาต์พุตของวงจรที่ 1 จะเป็นอินพุตของวงจรที่ 2 และเอาต์พุตของวงจรที่ 2 จะเป็นอินพุตของวงจรที่ 3 สัญญาณ START จะเป็นสัญญาณจากคอมพิวเตอร์ ซึ่งถ้าเป็น '0' จะใช้เพื่อทำการตั้งใหม่(reset)ของฮาร์ดแวร์ เพื่อให้ฮาร์ดแวร์อยู่ในสภาพที่พร้อมจะทำงาน และเป็น '1' จะใช้เพื่อบอกให้ฮาร์ดแวร์เริ่มทำงาน และจะต้องยังคงเป็น '1' ไปเรื่อยๆอย่างน้อยจนกว่าการทำงานของฮาร์ดแวร์จะเสร็จสิ้น เพราะถ้าเป็น '0' จะเท่ากับเป็นการหยุดการทำงานและเป็นการตั้งใหม่ฮาร์ดแวร์ทันที

เนื่องจากวงจรถูกออกแบบให้ทำงานแบบไพป์ไลน์ วงจรแต่ละวงจรจะเริ่มทำงานไม่พร้อมกัน โดยวงจรที่ 1 จะเริ่มทำงานก่อน แล้วตามด้วยวงจรที่ 2 และวงจรที่ 3 ตามลำดับ ทั้งนี้เพราะวงจรที่ 2 จะต้องรอเอาต์พุตจากวงจรที่ 1 และวงจรที่ 3 จะต้องรอเอาต์พุตจากวงจรที่ 2 ดังนั้นจึงต้องมีสัญญาณ clock1 ซึ่งเป็นสัญญาณนาฬิกาที่วงจรที่ 1 ส่งให้วงจรที่ 2 เพื่อบอกให้วงจรที่ 2 เริ่มทำงานตามหลังวงจรที่ 1 และสัญญาณ clock2 ก็จะเป็นสัญญาณนาฬิกาที่วงจรที่ 2 ส่งให้วงจรที่ 3 เพื่อบอกให้วงจรที่ 3 เริ่มทำงานตามหลังวงจรที่ 2 ได้อย่างถูกต้อง ซึ่งในส่วนของสัญญาณเหล่านี้จะอธิบายไว้ใน *"วิธีหาค่าการล้าหลัง"*

ข้อมูลอินพุตของวงจรที่ 1 จะได้จากพอร์ตของคอมพิวเตอร์ โดยวงจรที่ 1 จะส่งสัญญาณ DIO (Data In and Data Out) เป็น '1' เพื่อบอกให้คอมพิวเตอร์อ่านข้อมูลหนึ่งจุดภาพ(1 ไบต์)จากเอาต์พุตของวงจรที่ 3 เข้าสู่หน่วยความจำของคอมพิวเตอร์และส่งข้อมูลหนึ่งจุดภาพจากหน่วยความจำของคอมพิวเตอร์เป็นข้อมูลอินพุตให้กับวงจรที่ 1

เมื่อวงจรทำการหาขอบภาพเสร็จสิ้นก็จะส่งสัญญาณ Finish เป็น '1' เพื่อบอกคอมพิวเตอร์ว่าวงจรได้เสร็จสิ้นการทำงานแล้ว คอมพิวเตอร์ก็จะทำงานในส่วนของขั้นตอนที่ห้า(High Threshold) ซึ่งเป็นซอฟต์แวร์ต่อไป

ขั้นตอนการออกแบบฮาร์ดแวร์

เนื่องจากการออกแบบจะใช้พอร์ตขนานของคอมพิวเตอร์เป็นอินพุตและเอาต์พุต ดังนั้นจึงต้องระวังไม่ให้เกิดการแย่งกันระหว่างความต้องการข้อมูลอินพุตกับความต้องการข้อมูลเอาต์พุต และขนาดของรูปภาพที่ใช้คือ 256 x 256 จุดภาพ และเป็นระดับสีเทา ดังนั้น 1 จุดภาพจะใช้ 8 บิต หรือ 1 ไบต์พอดี

การออกแบบเริ่มจากกำหนดโครงสร้างของแต่ละส่วนที่สำคัญของวงจร ซึ่งจาก 3 วงจรแรกของ Canny จะพบว่าทุกวงจรต้องใช้หน้ากากขนาด 3 x 3 เพื่อการคำนวณ ดังนั้นจึงใช้รีจิสเตอร์ 9 ตัว เพื่อให้ทำการคำนวณได้รวดเร็ว และในการคำนวณจะทำการคำนวณเป็นแถวของข้อมูลจุดภาพจากซ้ายไปขวาของรูปภาพ ดังนั้นขนาดของหน่วยความจำที่ใช้จึงต้องมีขนาดไม่น้อยกว่า 3 แถวของรูปภาพ(3 x 256 = 768 จุดภาพ) ซึ่งเท่ากับ 768 ไบต์ จึงใช้หน่วยความจำขนาด 1 กิโลไบต์ และส่วนควบคุมขั้นตอนการทำงาน(sequencing control) จะใช้วงจรนับ(counter) เพื่อการควบคุม โดยแบ่งเป็นวงจรนับเพื่อควบคุมตำแหน่งแถว, วงจรนับเพื่อควบคุมตำแหน่งคอลัมน์, วงจรนับเพื่อควบคุมตำแหน่งแถวของข้อมูลในหน่วยความจำ(ดังจะอธิบายต่อไป) และวงจรนับเพื่อควบคุมขั้นตอนการทำงาน ซึ่งจะขออธิบายการออกแบบส่วนควบคุมขั้นตอนการทำงานของวงจรทั้ง 3 วงจรไปพร้อมกันดังนี้

การออกแบบส่วนควบคุมขั้นตอนการทำงาน

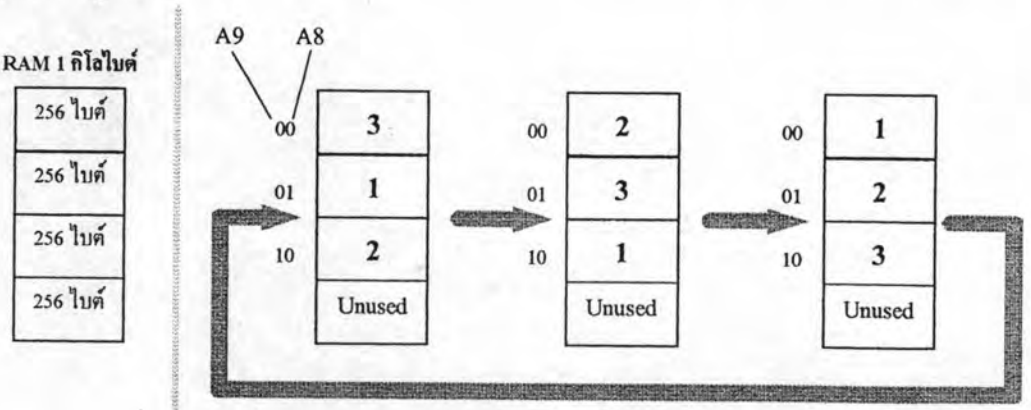
จากหลักการคำนวณจะพบว่าต้องใช้จำนวนแถว 256 แถว และจำนวนคอลัมน์ 256 คอลัมน์ เท่ากับมีการคำนวณทั้งหมด 256 x 256 ซึ่งเท่ากับ 65536 ครั้ง แต่การออกแบบเป็นฮาร์ดแวร์ออกมาจะต้องมีจำนวนครั้งของการคำนวณมากกว่านั้น ซึ่งเกิดจากเหตุผลดังนี้

การคำนวณต้องอาศัยค่าศูนย์เข้ามาช่วยดังที่เห็นค่าศูนย์ล้อมรอบจุดภาพของรูปภาพ ค่าศูนย์ที่นำเข้ามาจะช่วยในการคำนวณค่าของข้อมูลที่อยู่ที่ขอบ(frame) ดังนั้นทำให้จำนวนของแถวและคอลัมน์เพิ่มเป็น 258 แถวและ 258 คอลัมน์ดังในรูปที่ 4.3 (ค่าศูนย์ที่ต้องเพิ่มเข้ามาแสดงด้วย '0' และตำแหน่งของจุดภาพรูปภาพแสดงอยู่ในวงเล็บ (ดังเช่น (3,2) แสดงว่าเป็นจุดภาพของรูปภาพอยู่ในตำแหน่งแถวที่ 3 คอลัมน์ที่ 2))

'0'	'0'	'0'	'0'	--------	'0'	'0'
'0'	(1,1)	(1,2)	(1,3)	--------	(1,256)	'0'
'0'	(2,1)	(2,2)	(2,3)	--------	(2,256)	'0'
'0'	(3,1)	(3,2)	(3,3)	--------	(3,256)	'0'
⋮	⋮	⋮	⋮	⋱	⋮	⋮
'0'	(256,1)	(256,2)	(256,3)	--------	(256,256)	'0'
'0'	'0'	'0'	'0'	--------	'0'	'0'

รูปที่ 4.3 แสดงการเพิ่มค่าศูนย์เข้ามาช่วยในการคำนวณ

ดังนั้นวงจรนับสำหรับตำแหน่งแถวและคอลัมน์จึงต้องนับได้ไม่ต่ำกว่า 258 ค่า และเนื่องจากข้อมูลในหน่วยความจำเมื่อเริ่มจ่ายไฟให้ทำงาน จะเป็นข้อมูลแบบสุ่ม ดังนั้นในการโหลดข้อมูลเข้าสู่หน่วยความจำจึงต้องเริ่มต้นด้วยการป้อนค่าศูนย์เข้าสู่หน่วยความจำแถวหนึ่งก่อน แล้วจึงตามด้วยค่าของข้อมูลแถวที่หนึ่ง ตามด้วยข้อมูลแถวที่สองและสามต่อไปจนครบ 256 แถวของข้อมูล แล้วจึงตามด้วยค่าศูนย์อีกหนึ่งแถว แต่จะเห็นได้ว่า เราจะใช้หน่วยความจำในการเก็บข้อมูลแค่สามแถวเท่านั้น ดังนั้นจึงต้องมีการจัดลำดับแถวของข้อมูลในหน่วยความจำให้ถูกต้อง ดังแสดงด้วยรูปที่ 4.4

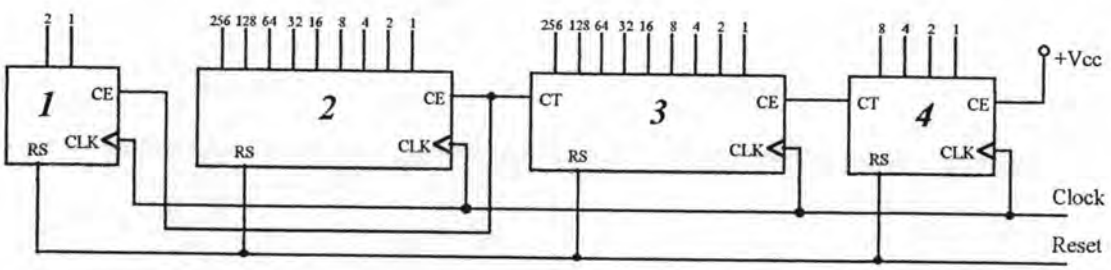


รูปที่ 4.4 แสดงการใช้หน่วยความจำในการเก็บข้อมูล

256 ไบต์สุดท้ายของหน่วยความจำจะไม่ใช้ จะใช้แค่ 768 ไบต์แรก ซึ่งถูกแบ่งออกเป็นแถวขนาด 256 ไบต์ได้ 3 แถว ดังนั้นการเข้าถึงแถวแต่ละแถวก็โดยอาศัยการอ้างผ่านขาแอสเดรส A9 และ A8 ของหน่วยความจำ ซึ่งลำดับของการอ้างถึงจะเป็นลำดับดังรูปข้างบนด้านขวา โดยหมายเลข 1, 2 และ 3 จะหมายถึงข้อมูลที่อยู่ในหน่วยความจำ ซึ่งหมายเลข 3 แทนแถวของข้อมูลที่จะถูกเขียนลงไปในหน่วยความจำ, หมายเลข 2 แทนแถวของข้อมูลที่ถูกเขียนลงไปก่อนแถวของหมายเลข 3 และหมายเลข 1 แทนแถวของข้อมูลที่ถูกเขียนลงไปก่อนแถวของหมายเลข 2 ดังนั้นจึงเห็นได้ว่าหลังจากเขียนข้อมูลครบทั้งแถว(ครบ 256 คอลัมน์) พอเริ่มเขียนข้อมูลแถวใหม่ที่อยู่ของข้อมูลหมายเลข 1 จะต้องกลายเป็นของหมายเลข 3, ที่อยู่ของข้อมูลหมายเลข 2 จะกลายเป็นของหมายเลข 1 และที่อยู่ของข้อมูลหมายเลข 3 จะกลายเป็นของหมายเลข 2 และที่กำหนดให้หมายเลข 3 เป็นหมายเลขแทนข้อมูลที่ถูกเขียนเพราะว่า ถือเอาหลักที่ว่าแถว 3 เป็นแถวของข้อมูลที่จะเข้ามาใหม่

และเนื่องจากในแต่ละวงจรหลักที่จะทำการออกแบบจะถูกออกแบบให้ทำงานเป็นลักษณะแบบไพป์ไลน์ ดังนั้นวงจรนับ 16 (modulo 16 counter) จึงเพียงพอแก่ความต้องการที่จะใช้เป็นวงจรนับเพื่อควบคุมขั้นตอนการทำงาน

ดังนั้นจึงสามารถออกแบบส่วนควบคุมขั้นตอนการทำงานได้ดังรูปที่ 4.5



<u>วงจรนับตัวที่ 1</u> คือวงจรนับเพื่อควบคุมตำแหน่งแถวของข้อมูลในหน่วยความจำ

<u>วงจรนับตัวที่ 2</u> คือวงจรนับเพื่อควบคุมตำแหน่งแถว

<u>วงจรนับตัวที่ 3</u> คือวงจรนับเพื่อควบคุมตำแหน่งคอลัมน์

<u>วงจรนับตัวที่ 4</u> คือวงจรนับเพื่อควบคุมขั้นตอนการทำงาน

<u>รูปที่ 4.5</u> แสดงส่วนควบคุมขั้นตอนการทำงานของวงจร

วงจรนับเพื่อควบคุมตำแหน่งแถวของข้อมูลในหน่วยความจำ จะใช้วงจรนับ 3 (modulo 3 counter) เพราะใช้หน่วยความจำเพื่อเก็บข้อมูล 3 แถว

วงจรนับเพื่อควบคุมตำแหน่งแถว จะใช้วงจรนับ 512 (modulo 512 counter) ซึ่งจะถูกหยุดการทำงานด้วยสัญญาณ '1' เข้าที่ขา RS (Reset) เมื่อการทำงานของวงจรเสร็จสิ้น

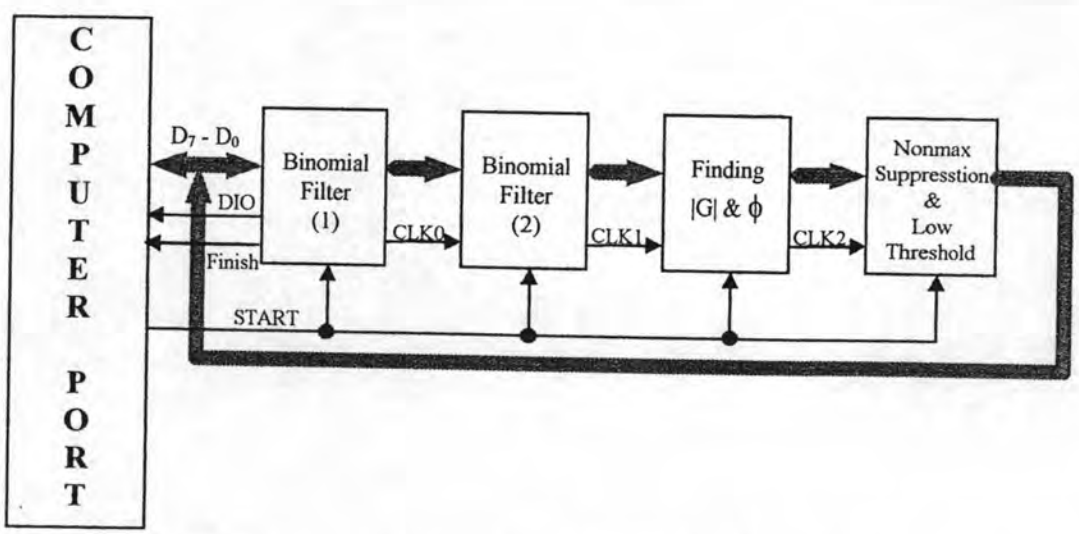
วงจรนับเพื่อควบคุมตำแหน่งคอลัมน์ จะใช้วงจรนับ 257 (modulo 257 counter) ทั้งนี้ที่ไม่ใช้วงจรนับ 258 เพราะว่าเราจะลบล้าง(clear)รีจิสเตอร์ทั้ง 9 ตัวให้เป็น '0' ทุกครั้งที่ทำการขึ้นแถวใหม่ก่อนการคำนวณ ดังนั้นค่า '0' ที่จะต้องป้อนให้กับรีจิสเตอร์ก่อนที่จะป้อนค่าของคอลัมน์ที่หนึ่งและสองต่อไปจึงไม่ต้องมี โดยเราสามารถป้อนค่าของคอลัมน์ที่หนึ่งได้เลย

วงจรนับเพื่อควบคุมขั้นตอนการทำงาน จะใช้วงจรนับ 16 (modulo 16 counter)

หลังจากนั้นก็มาออกแบบขั้นตอนการทำงาน โดยจัดลำดับลงในตารางเพื่อให้สะดวกต่อการออกแบบของทั้ง 3 วงจรพร้อมกัน ที่ต้องวางขั้นตอนการทำงานของ 3 วงจรพร้อมกัน ก็เพื่อให้การออกแบบวงจรทั้งสามวงจรมีความต่อเนื่องกัน เพราะว่าวงจรทั้งสามจะต้องเริ่มทำงานไม่พร้อมกัน โดยวงจรที่ 1 เริ่มก่อน ตามด้วยวงจรที่ 2 และวงจรที่ 3 ทั้งนี้เพราะว่าเราต้องการเอาเอาต์พุตของวงจรที่ 1 ไปเป็นอินพุตให้กับวงจรที่ 2 และเอาต์พุตของวงจรที่ 2 ไปเป็นอินพุตของวงจรที่ 3 ต่อไป และการออกแบบจะกำหนดให้มีการติดต่อกับคอมพิวเตอร์โดยอาศัยพอร์ตขนานเพียงพอร์ตเดียว ดังนั้นจึงต้องระวังเรื่องข้อมูลที่จะเข้าและออกจากพอร์ต เพื่อไม่ให้มีการแย่งกันใช้พอร์ต

วงจรที่ 1 จะต้องมี 2 ชุด กำหนดชื่อเป็น Binomial Filter (1) และ (2) ดังนั้นจึงเพิ่มวงจรขึ้นมาหนึ่งวงจร(คือวงจรที่ 1 มี 2 วงจร) รวมเป็นทั้งหมด 4 วงจร แต่จะขอเรียกวงจรที่เพิ่มขึ้นมานี้เป็นวงจรที่ 0

การออกแบบกำหนดให้มีสัญญาณที่ต้องใช้ในการติดต่อกับคอมพิวเตอร์ดังแสดงในรูปที่ 4.6



รูปที่ 4.6 แสดงการเชื่อมต่อระหว่างคอมพิวเตอร์กับฮาร์ดแวร์ที่ออกแบบ

สัญญาณ START จากคอมพิวเตอร์เริ่มแรกจะต้องเป็น '0' เพื่อให้ทุกวงจรอยู่ในสภาวะการตั้งใหม่ เพื่อเตรียมพร้อมในการคำนวณหาค่าของขอบภาพอันใหม่ หลังจากนั้นสัญญาณ START จะต้องเป็น '1' และจะต้องค้างค่า '1' นี้ไว้จนกว่าคอมพิวเตอร์จะได้รับข้อมูลของขอบภาพข้อมูลสุดท้าย จากนั้นจึงลบล้างสัญญาณ START หรือจะไม่ลบล้างก็ได้ เพราะวงจรทั้ง 3 ถูกออกแบบให้หยุดตัวเองหลังทำงานเสร็จ โดยไม่ต้องรอสัญญาณ START เป็น '0' คือแต่ละวงจรจะหยุดและทำการตั้งใหม่ตัวเองหลังจากที่วงจรทำงานเสร็จสิ้น

สัญญาณ DIO จะเป็นสัญญาณบอกให้คอมพิวเตอร์ทำการอ่านข้อมูลจุดภาพ(8 บิต)จากเอาต์พุตของวงจรสุดท้ายคือวงจร Nonmax Suppresstion and Low Threshold ซึ่งเป็นค่าจุดภาพของขอบภาพที่ได้จากการทำงานของฮาร์ดแวร์ ไปเก็บไว้ในหน่วยความจำของคอมพิวเตอร์ แล้วคอมพิวเตอร์ก็จะส่งข้อมูลจุดภาพถัดไปของรูปภาพเข้าสู่อินพุตของวงจรแรกคือวงจร Binomial Filter (1) ส่วนซอฟต์แวร์ที่ใช้กับคอมพิวเตอร์จะอธิบายภายหลัง

สัญญาณ Finish สัญญาณนี้ ตอนที่วงจรอยู่ในสภาวะตั้งใหม่ หรือกำลังทำงานจะมีค่าเป็น '0' และคงอยู่ค่า '0' ไปจนฮาร์ดแวร์ทำงานเสร็จเรียบร้อย สัญญาณนี้ถึงจะกลายเป็น '1' เพื่อบอกให้คอมพิวเตอร์ทราบว่าฮาร์ดแวร์ได้ทำการคำนวณหาค่าขอบภาพเสร็จเรียบร้อยแล้ว

ดังนั้นจึงสามารถแสดงสัญญาณหลักที่ถูกควบคุมโดย*ส่วนควบคุมขั้นตอนการทำงานของวงจร*ของวงจรทั้ง 4 ได้ดังตารางที่ 4.1 ในหน้า 38

ส่วนวิธีหาค่าการล้าหลัง(Lagging) หรือคือค่าที่วงจรถัดมาเริ่มช้ากว่าวงจรก่อนหน้าสามารถหาได้ดังนี้

วิธีหาค่าการล้าหลัง

	Row0							Row1							
No.	1	2	3	4	~	256	257	258	259	260	261	~			
Column	0	1	2	3	~	255	256	257	258	259	260	~			

จากตัวเลขข้างบนในบรรทัดที่สองจะเป็นตัวเลขแสดงตำแหน่งคอลัมน์ที่วงจรนับควบคุมตำแหน่งคอลัมน์เป็นตัวนับซึ่งจะนับ 0 ถึง 256 แล้ววนไป 0 ใหม่ ซึ่งเท่ากับขึ้น Row ใหม่ด้วย ดังที่เขียนไว้ข้างบนว่า Row0 และ Row1 ส่วนตัวเลขในบรรทัดแรกเป็นการนับจำนวนของคอลัมน์

ตัวเลขข้างบนจะใช้เพื่อช่วยในการหาค่าที่แต่ละวงจรควรจะเริ่มช้ากว่ากันเป็นจำนวนกี่สัญญาณนาฬิกา โดยแต่ละวงจรจะต้องโหลดค่า '0' เข้าสู่วงจรเมื่อวงจรเริ่มทำงาน ซึ่งวงจรทุกวงจรจะเริ่มทำงานที่ Row0 เสมอ และวงจรตัวที่อยู่หลังจะต้องเริ่มช้ากว่าวงจรตัวที่อยู่ก่อนหน้า

มันอย่างน้อยไม่ต่ำกว่า 1 แถวหรือเท่ากับ 257 คอลัมน์ เพราะต้องรอให้วงจรก่อนหน้าทำงานจนมีเอาต์พุตที่ถูกต้องออกมาเพื่อเป็นอินพุตให้กับวงจรที่อยู่ถัดมา ซึ่งค่าเอาต์พุตที่ถูกต้องที่จะได้จากวงจรใดก็ต่อเมื่อวงจรนั้นทำงานเป็นจำนวน 1 แถวกับอีก 1 คอลัมน์ (หรือเท่ากับ 257 + 1 คอลัมน์ ซึ่งขณะที่วงจรยังทำงานอยู่ในค่าคอลัมน์นี้วงจรจะโหลดค่าศูนย์เข้ามาเท่านั้น หลังจากผ่านจำนวนของคอลัมน์นี้วงจรก็จะเริ่มโหลดค่าของข้อมูลอินพุตที่จะทำการคำนวณเข้ามา) บวกกับค่าของสัญญาณนาฬิกาที่ตัวของวงจรก่อนหน้าต้องทำงานจนได้เอาต์พุตออกมาเป็นอินพุตให้กับวงจร ซึ่งค่านี้ขึ้นกับแต่ละวงจร และสามารถดูได้จากตารางที่ 4.1 ตารางแสดงขั้นตอนการทำงานที่ออกแบบโดยดูว่าค่าเอาต์พุต DataOut ของวงจรก่อนหน้า และค่าอินพุต DataIn ของวงจรถัดมาอยู่ที่สัญญาณนาฬิกาที่เท่าไร เพราะค่า DataOut ของวงจรก่อนหน้าจะไปเป็นค่า DataIn ของวงจรถัดไป โดยให้การโหลดค่า DataIn อยู่ตามหลังค่า DataOut อยู่อย่างน้อยอยู่ 1 สัญญาณนาฬิกา(แต่ไม่เกิน 15 สัญญาณนาฬิกา เพราะค่า DataOut จะเปลี่ยนเป็นค่าใหม่ไป) แล้วให้บวก 16 เพิ่มเข้าไป ที่ต้องบวก 16 เพิ่มเข้าไปก็เพราะว่าวงจรต้องทำงาน 1 รอบ(เท่ากับ 16 สัญญาณนาฬิกา) เพื่อโหลดข้อมูลเข้าให้กับรีจิสเตอร์ทั้ง 3 แถว และทำงานจนได้ค่า DataOut ออกมา จึงสามารถสรุปค่านี้จากตารางที่ 4.1 ได้ดังนี้

วงจรที่ 0	วงจรที่ 1	วงจรที่ 2
16 + 5	16 + 5	16 + 0

ดังนั้นจึงนำค่านี้ไปทำการคำนวณหาค่าการล้าหลังได้ดังนี้

<u>วงจรที่ 0</u> เริ่มทันทีเพราะเป็นวงจรแรก คือ ไม่มีวงจรอยู่ก่อนหน้า

<u>วงจรที่ 1</u> เริ่มหลังวงจรที่ 0 เป็นจำนวนอย่างน้อยที่สุดที่ 259 คอลัมน์ + 5 สัญญาณนาฬิกา แต่ที่ออกแบบจะให้วงจรนี้เริ่มหลังวงจรที่ 0 ที่คอลัมน์ที่ 259 คอลัมน์ + 6 สัญญาณนาฬิกา

<u>วงจรที่ 2</u> เริ่มหลังวงจรที่ 1 เป็นจำนวนอย่างน้อยที่สุดที่ 259 คอลัมน์ + 5 สัญญาณนาฬิกา แต่ที่ออกแบบจะให้วงจรนี้เริ่มหลังวงจรที่ 1 ที่คอลัมน์ที่ 259 คอลัมน์ + 6 สัญญาณนาฬิกา

<u>วงจรที่ 3</u> เริ่มหลังวงจรที่ 2 เป็นจำนวนอย่างน้อยที่สุดที่ 259 คอลัมน์ + 0 สัญญาณนาฬิกา แต่ที่ออกแบบจะให้วงจรนี้เริ่มหลังวงจรที่ 2 ที่คอลัมน์ที่ 259 คอลัมน์ + 7 สัญญาณนาฬิกา

เหตุผลหลักที่เปลี่ยนแปลงค่าการล้าหลัง ก็เพื่อหลีกเลี่ยงการชนกันระหว่างความต้องการข้อมูลเข้ากับข้อมูลออกจากพอร์ตของคอมพิวเตอร์

จากค่าการล้าหลังที่ได้ ทำให้สามารถจัดเรียงตารางที่ 4.1 ได้ใหม่ โดยเรียงขั้นตอนการทำงานของทั้ง 4 วงจรตามหมายเลขสัญญาณนาฬิกาของวงจรที่ 0 เช่น ถ้าวงจรที่ 0 ทำงานอยู่ที่สัญญาณนาฬิกาหมายเลขที่ 7 วงจรที่ 1, 2 และ 3 จะทำงานอยู่ที่จังหวะสัญญาณนาฬิกาใด จึงได้ออกมาเป็นตารางที่ 4.2 ในหน้าที่ 39

จากนั้นจึงนำค่าจากในตารางที่ 4.1 และ 4.2 มาออกแบบเป็นฮาร์ดแวร์ต่อไป โดยสัญลักษณ์และเครื่องหมายที่ใช้ในตารางอธิบายได้ดังข้างล่าง

อธิบายสัญลักษณ์และเครื่องหมายที่ใช้ในตาราง

เลข 0 ถึง 15 ในคอลัมน์ซ้ายมือสุด เป็นหมายเลขของสัญญาณนาฬิกาควบคุม ที่ใช้วงจรนับเพื่อควบคุมตำแหน่งแถวของข้อมูลในหน่วยความจำซึ่งเป็นวงจรนับ 16

ลบล้างรีจิสเตอร์หน้ากากทุกครั้งที่ขึ้นแถวใหม่ เป็นการลบล้างรีจิสเตอร์หน้ากากเมื่อมีการขึ้นแถวใหม่ของข้อมูล เพื่อล้างข้อมูลเก่าที่อยู่ในรีจิสเตอร์หน้ากาก

A.R$_n$Cx เป็นการที่วงจรส่งค่าแอดเดรสแถวที่ n คอลัมน์ที่ x ให้หน่วยความจำ (หมายเลขคอลัมน์ขึ้นอยู่กับรอบการทำงานซึ่งถูกควบคุมโดยวงจรนับเพื่อควบคุมตำแหน่งคอลัมน์)

L.R$_n$Cx เป็นการสั่งให้รีจิสเตอร์หน้ากากแถวที่ n อ่านข้อมูลจากหน่วยความจำหรือจากรีจิสเตอร์อินพุตเข้าไปเก็บไว้ ซึ่งแทนด้วย Pn โดย n คือหมายเลขแสดงตำแหน่งของรีจิสเตอร์หน้ากาก

WE เป็นสัญญาณควบคุมการอ่าน/เขียนของหน่วยความจำภายนอก เป็น '0' หมายถึงอยู่ในสภาวะเขียน และเป็น '1' คืออยู่ในสภาวะอ่าน

DataIn คือรีจิสเตอร์อินพุตของวงจรทำการเก็บค่าอินพุตที่ส่งมาจากพอร์ตของคอมพิวเตอร์ เพื่อเป็นอินพุตให้กับส่วนต่อไปของวงจร

DataOut คือรีจิสเตอร์เอาต์พุตของวงจรทำการเก็บค่าผลลัพธ์ที่วงจรคำนวณออกมา เพื่อส่งเป็นเอาต์พุตให้กับวงจร

DIO เป็นสัญญาณเอาต์พุตจากวงจร Binomial Filter (1) ใช้สั่งให้คอมพิวเตอร์อ่านข้อมูลและเขียนข้อมูลผ่านพอร์ต โดยไวงาน(active)ที่สัญญาณ '1'

Mn หรือ MGc คือการสั่งให้วงจรมัลติเพลกซ์ทำการเลือกรีจิสเตอร์หน้ากากตัวที่ n หรือตัวที่เป็นค่าของ Gc (Gx หรือ Gy) เข้าไปสู่ส่วนต่อไปของวงจร เพื่อทำการคำนวณต่อไป

Pout คือรีจิสเตอร์เก็บค่าผลบวก(มีอยู่ในวงจร Binomial Filter (1) และ (2)) ทำการเก็บผลบวกซึ่งต้องเก็บผลบวกของรีจิสเตอร์หน้ากากทั้ง 9 ตัว เพื่อให้ได้ผลลัพธ์ออกมา

Gx และ **Gy** คือรีจิสเตอร์เก็บค่ากราเดียนต์ในแนวแกน x (Gx) และ y (Gy) ตามลำดับ ให้ทำการเก็บค่า Gx และ Gy ที่วงจร Finding |G| and ϕ คำนวณไว้

L.ϕ อยู่ในวงจร Nonmax Suppression and Low Threshold เป็นการสั่งให้รีจิสเตอร์ที่ใช้เก็บค่า ϕ อ่านข้อมูลค่า ϕ จากหน่วยความจำเข้าไปเก็บไว้

No Output อยู่ในวงจร Nonmax Suppression and Low Threshold เป็นการสั่งให้บัฟเฟอร์สามสถานะกันไม่ให้เอาต์พุตจากรีจิสเตอร์เอาต์พุตผ่านออกไปสู่พอร์ตของคอมพิวเตอร์ เพื่อเป็นกันไม่ให้ไปปนกับข้อมูลที่ออกมาจากพอร์ตของคอมพิวเตอร์

	Binomial Filter (1) และ (2)	**Finding |G| & ∅**	**Nonmax Suppression**
0	we=1; A.R_3Cx; M7;	we=1; A.R_3Cx; MGx=1;	we=1; A.R_1Cx;
1	we=1; M8; Pout; **DataIn**;	we=1; MGx=1; **DataIn**;	we=1; **DataOut**; ลบล้างรีจิสเตอร์หน้ากากทุกครั้งที่ขึ้นแถวใหม่
2	we=0; M9; Pout;	we=0; MGx=1;	we=1;
3	we=0; Pout;	we=0; MGx=1; Gx;	we=1; L.R_1Cx > P3 > P2 > P1;
4	we=0; **DataOut**; ลบล้างรีจิสเตอร์หน้ากากทุกครั้งที่ขึ้นแถวใหม่	we=0; ลบล้างรีจิสเตอร์หน้ากากทุกครั้งที่ขึ้นแถวใหม่	we=1; A.R_2Cx;
5	we=0;	we=0;	we=1;
6	we=1; Reset Pout;	we=1;	we=1;
7	we=1; L.R_3Cx > P9 > P8 > P7;	we=1; L.R_3Cx > P9 > P8; **DataOut**;	we=1; L.R_2Cx > P6 > P5 > P4; L.ϕ;
8	we=1; A.R_2Cx; M1; DIO=1 (B.F.1)*;	we=1; A.R_2Cx;	we=1; A.R_3Cx; **DataIn**; No Output=1;
9	we=1; M2; Pout; DIO=1 (B.F.1);	we=1;	we=1; No Output=1;
10	we=1; M3; Pout; DIO=1 (B.F.1);	we=1;	we=0; No Output=1;
11	we=1; L.R_2Cx > P6 > P5 > P4; Pout; DIO=1 (B.F.1);	we=1; L.R_2Cx > P6 > P5 > P4;	we=0; No Output=1;
12	we=1; A.R_1Cx; M4;	we=1; A.R_1Cx; MGy=1;	we=0; No Output=1;
13	we=1; M5; Pout;	we=1; MGy=1;	we=0; No Output=1;
14	we=1; M6; Pout;	we=1; MGy=1;	we=1; No Output=1;
15	we=1; L.R_1Cx > P3 > P2 > P1; Pout;	we=1; L.R_1Cx > P3 > P2; MGy=1; Gy	we=1; L.R_3Cx > P9 > P8 > P7; No Output=1;

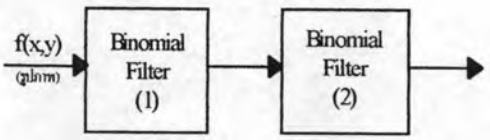
* B.F.1 = เป็นสัญญาณของวงจร Binomial Filter (1) เท่านั้น

ตารางที่ 4.1 แสดงสัญญาณควบคุมขั้นตอนการทำงานที่เกิดขึ้นในแต่ละรอบของวงจรที่ 0, 1, 2 และ 3 แยกอิสระกัน

Binomial Filter (1)		Binomial Filter (2)		Finding |G| & ∅		Nonmax Suppression	
0	we=1; A.R_2Cx; M7;	11	we=1; L.R_2Cx > P6 > P5 > P4; Pout;	6	we=1;	0	
1	we=1; M8; Pout; **DataIn**;	12	we=1; A.R_1Cx; M4;	7	we=1; L.R_1Cx > P9 > P8; **DataOut**;	1	we=1; **DataOut**; ลบล้างรีจิสเตอร์หน้ากากทุกครั้งที่ขึ้นแถวใหม่
2	we=0; M9; Pout;	13	we=1; M5; Pout;	8	we=1; A.R_2Cx;	2	we=1;
3	we=0; Pout;	14	we=1; M6; Pout;	9	we=1;	3	we=1; L.R_1Cx > P3 > P2 > P1;
4	we=0; **DataOut**; ลบล้างรีจิสเตอร์หน้ากากทุกครั้งที่ขึ้นแถวใหม่	15	we=1; L.R_1Cx > P3 > P2 > P1; Pout;	10	we=1;	4	we=1; A.R_2Cx;
5	we=0;	0	we=1; A.R_2Cx; M7;	11	we=1; L.R_2Cx > P6 > P5 > P4;	5	we=1;
6	we=1; Reset Pout;	1	we=1; M8; Pout; **DataIn**;	12	we=1; A.R_1Cx; MGy=1;	6	we=1;
7	we=1; L.R_1Cx > P9 > P8 > P7;	2	we=0; M9; Pout;	13	we=1; MGy=1;	7	we=1; L.R_2Cx > P6 > P5 > P4; L.∅;
8	we=1; A.R_2Cx; M1; DIO=1;	3	we=0; Pout;	14	we=1; MGy=1;	8	we=1; A.R_2Cx; **DataIn**; No Output=1;
9	we=1; M2; Pout; DIO=1;	4	we=0; **DataOut**; ลบล้างรีจิสเตอร์หน้ากากทุกครั้งที่ขึ้นแถวใหม่	15	we=1; L.R_1Cx > P3 > P2; MGy=1; Gy	9	we=1; No Output=1;
10	we=1; M3; Pout; DIO=1;	5	we=0;	0	we=1; A.R_1Cx; MGx=1;	10	we=0; No Output=1;
11	we=1; L.R_2Cx > P6 > P5 > P4; Pout; DIO=1;	6	we=1; Reset Pout;	1	we=1; MGx=1; **DataIn**;	11	we=0; No Output=1;
12	we=1; A.R_1Cx; M4;	7	we=1; L.R_1Cx > P9 > P8 > P7;	2	we=0; MGx=1;	12	we=0; No Output=1;
13	we=1; M5; Pout;	8	we=1; A.R_2Cx; M1;	3	we=0; MGx=1; Gx;	13	we=0; No Output=1;
14	we=1; M6; Pout;	9	we=1; M2; Pout;	4	we=0; ลบล้างรีจิสเตอร์หน้ากากทุกครั้งที่ขึ้นแถวใหม่	14	we=1; No Output=1;
15	we=1; L.R_1Cx > P3 > P2 > P1; Pout;	10	we=1; M3; Pout;	5	we=0;	15	we=1; L.R_1Cx > P9 > P8 > P7; No OutPut=1

ตารางที่ 4.2 แสดงสัญญาณควบคุมขั้นตอนการทำงานที่เกิดขึ้นในแต่ละรอบของวงจรที่ 0, 1, 2 และ 3 ที่เกิดขึ้นในจังหวะเดียวกัน

การออกแบบฮาร์ดแวร์วงจรที่ 1 Binomial Filter (1) และ (2)



ตัวกรองความถี่แบบไบโนเมียลที่ให้ผลตอบสนองได้ใกล้เคียงกับตัวกรองความถี่ที่ใช้ฟังก์ชันเกาส์เซียน มีรูปแบบดังนี้

$$f(x,y,n)=\frac{1}{16^n}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}^{*n}$$

ใช้ n เท่ากับ 2 คือต้องทำคอนโวลูชันของตัวหน้ากากกับรูปภาพ 2 ครั้ง ดังนั้นจึงต้องใช้วงจรที่เหมือนกันแยกออกมาเป็น Binomial Filter (1) และ Binomial Filter (2)

ซึ่งการทำคอนโวลูชันในที่นี้คือการนำค่าของจุดภาพรูปภาพคูณกับสัมประสิทธิ์ของตัวหน้ากาก แล้วจับมาบวกกันแล้วหารด้วย 16 เช่นสมมติข้อมูลขนาด 5 x 5 ดังรูปข้างล่างด้านซ้าย เมื่อทำการคอนโวลูชันกับตัวหน้ากาก ก็จะได้ผลลัพธ์ดังรูปข้างล่างด้านขวา

55	35	30	30	32
60	40	50	36	44
135	130	125	120	130
140	146	140	170	140
150	153	158	155	139

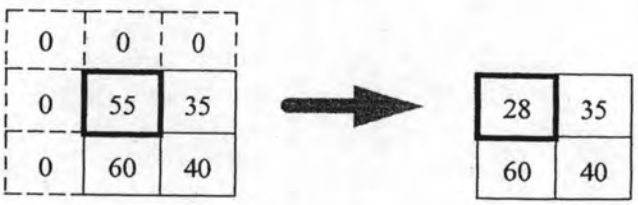
ข้อมูลก่อนผ่านการคอนโวลูชัน

28	31	26	25	19
54	65	61	59	45
86	112	110	111	83
106	142	144	146	107
83	112	115	114	82

ข้อมูลหลังผ่านการคอนโวลูชัน

รูปที่ 4.7 แสดงข้อมูลก่อนและหลังผ่านการคอนโวลูชัน

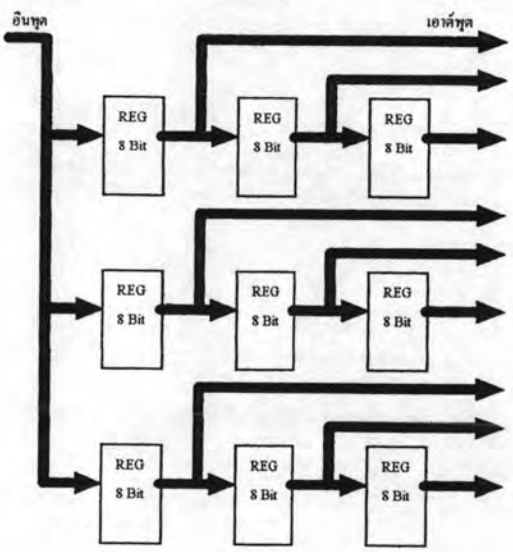
ซึ่งค่าที่อยู่ที่ขอบจะต้องใช้ค่าศูนย์เข้ามาช่วยในการคำนวณ ดังเช่นค่าที่ขอบบนด้านซ้ายของตัวอย่างข้อมูลข้างบนจะต้องทำการคำนวณดังนี้



$$((0x1) + (0x2) + (0x1)$$
$$(0x2) + (55x4) + (35x2) +$$
$$\underline{(0x1) + (60x2) + (40x1))/16} +$$
$$\underline{450/16}$$
$$\underline{= 28.125 = 28}$$

ค่าเศษจะถูกปัดทิ้งทั้งหมดเพื่อให้ง่ายต่อการออกแบบฮาร์ดแวร์ ทำให้ไม่ต้องมีส่วนของวงจรเพื่อทำการปัดเศษ ดังนั้นผลลัพธ์จึงได้ดังค่าตัวอย่างข้างบน

จากวิธีที่ต้องมีการนำค่าของจุดภาพรูปภาพคูณกับสัมประสิทธิ์ของตัวหน้ากาก แล้วจับมาบวกกันแล้วหารด้วย 16 ทำให้เราต้องการรีจิสเตอร์ขนาด 8 บิตทั้งหมด 9 ตัว เพื่อนำมาเก็บค่าของข้อมูลที่จะนำเข้ามาทำการคำนวณ โดยถูกจัดเรียงดังรูปที่ 4.8



<u>รูปที่ 4.8</u> แสดงการจัดเรียงรีจิสเตอร์ที่ใช้ในการเก็บค่าข้อมูลเพื่อการคำนวณ 9 ตัว

จากที่กล่าวไว้แล้วว่า ตัวหน้ากากจะมีค่าสัมประสิทธิ์เป็น 1, 2, 1, 2, 4, 2, 1, 2 และ 1 นั่นคือมีแค่สามค่าคือ 1, 2 และ 4 ดังนั้นการคูณข้อมูลด้วยค่าเหล่านี้ ก็เหมือนกับการเลื่อนบิตข้อมูลไปทางขวานั่นเอง ดังนั้นวงจรนี้ต้องมีตัวเลื่อนบิตข้อมูล และจะต้องใช้วงจรมัลติเพลกซ์เพื่อเป็นตัวเลือกข้อมูลจากรีจิสเตอร์ทีละตัวส่งผ่านตัวเลื่อนบิต เพื่อผ่านไปยังวงจรบวกจนครบ 9 ข้อมูล จึงต้องมีรีจิสเตอร์อีกตัวเป็นตัวเก็บค่าผลบวก และขนาดของรีจิสเตอร์ตัวนี้จะต้องสามารถเก็บค่าของผลบวกสูงสุดที่เป็นไปได้ได้คือค่า 255 x (1+2+1+2+4+2+1+2+1) = 255 x 16 = 4080 หรือเท่ากับ $FF0_{16}$ หรือเท่ากับ $1111\ 1111\ 0000_2$ ดังนั้นจึงต้องมีขนาด 12 บิต

มีข้อสังเกตอยู่ข้อหนึ่งคือ ในทางทฤษฎีเราจะเคลื่อนตัวหน้ากากผ่านข้อมูลรูปภาพเพื่อทำการคอนโวลูชัน โดยเคลื่อนตัวหน้ากากจากคอลัมน์ซ้ายไปขวา และแถวบนไปแถวล่าง แต่ถ้าเราให้ตัวหน้ากากอยู่กับที่และให้ข้อมูลรูปภาพเคลื่อนผ่านตัวหน้ากาก ก็จะพบว่าเหมือนกับมีการเลื่อนข้อมูลจากขวาไปซ้าย(left shift) (ให้ลองนึกว่ามองผ่านช่องหน้าต่างด้านซ้ายของรถยนต์ขณะรถวิ่งไปข้างหน้า หน้าต่างก็คือหน้ากาก ส่วนทิวทัศน์ที่เห็นก็คือข้อมูลรูปภาพ) แต่การออกแบบฮาร์ดแวร์โดยการเรียงรีจิสเตอร์ 9 ตัวไว้แทนหน้ากากดังรูปที่ 4.8 ก็จะเหมือนกับเป็นการยึดตัวหน้ากากให้อยู่กับที่และเมื่อทำการเลื่อนข้อมูลจากซ้ายไปขวาเพื่อส่งต่อไปยังวงจรบวก ทำให้ถ้าต้องการคงทิศทางของหน้ากากเสมือนว่าเลื่อนจากซ้ายไปขวาผ่านข้อมูลรูปภาพ ทำให้เหมือนผลิกตัวหน้ากากกลับซ้ายไปขวา และโดยหลักการข้อมูลแถวใหม่ที่เข้ามาจะต้องเข้าที่แถวล่างสุด ส่วนข้อมูลเก่าก็จะเลื่อนขึ้นแถวบน แต่ฮาร์ดแวร์ที่ออกแบบจะให้ข้อมูลที่เข้ามาใหม่เข้าที่แถวบนสุดและข้อมูลเก่าก็จะเลื่อนลงแถวล่าง ดังนั้นก็จึงเหมือนกับผลิกตัวหน้ากากกลับบนล่าง ดังแสดงในรูปที่ 4.9 นั่นคือ แถวที่สามอยู่แถวบน, แถวที่สองอยู่แถวกลาง และแถวที่หนึ่งอยู่แถวล่าง ซึ่งจะให้ผลไม่แตกต่างกันในกรณีของค่าสัมประสิทธิ์ของไบโนเมียล ของวงจร Binomial (1) และ Binomial (2) เพราะเป็นค่าที่มีความสมมาตรทั้งในแกน x และแกน y แต่มีผลที่แตกต่างในวงจร Finding |G| and ϕ และวงจร Nonmax Suppression and Low Threshold

(1,1)	(1,2)	(1,3)
(2,1)	(2,2)	(2,3)
(3,1)	(3,2)	(3,3)

หน้ากากทางทฤษฎี

(3,3)	(3,2)	(3,1)
(2,3)	(2,2)	(2,1)
(1,3)	(1,2)	(1,1)

หน้ากากในฮาร์ดแวร์

<u>รูปที่ 4.9</u> แสดงตำแหน่งของตัวหน้ากาก

ดังนั้นวงจร Binomial Filter (1) จึงมีส่วนประกอบดังแผนภาพบล็อกในรูปที่ 4.10 หน้า 44 จากในรูปแผนภาพบล็อกจะเห็นว่าวงจรเริ่มจากมีรีจิสเตอร์ขนาด 8 บิต(ในรูปอยู่ด้านซ้ายบน)ในชื่อ DATA_IN เพื่อรับข้อมูลจากพอร์ตขนานของคอมพิวเตอร์เข้ามาเก็บไว้ในรีจิสเตอร์นี้ เพื่อส่งต่อให้เป็นอินพุตให้กับหน่วยความจำและรีจิสเตอร์ที่ทำหน้าที่เป็นหน้ากากต่อไป มีหน่วยความจำ(ในรูปชื่อ MEMORY) เพื่อใช้เก็บข้อมูลไว้ 3 แถว เพื่อใช้ในการคำนวณ มีรีจิสเตอร์ขนาด 8 บิต 9 ตัวต่อเรียงกันเป็นเมตริกซ์ขนาด 3 x 3 เพื่อใช้เก็บค่าจุดภาพ 9 ค่า(ที่ได้จากหน่วยความจำและ DATA_IN) เพื่อจะส่งให้วงจรมัลติเพลกซ์เป็นตัวเลือกเอาทีละค่าส่งไปให้วงจรเลือกบิต(ในรูปชื่อ Shifter) เพื่อเป็นการคูณค่าของข้อมูลนั้นกับค่าสัมประสิทธิ์ของหน้ากาก แล้วส่งต่อให้วงจรบวก(ในรูปชื่อ ADDER) เพื่อทำการบวกค่าทั้ง 9 ที่ผ่าน Shifter แล้วเก็บผลลัพธ์ไว้ที่รีจิสเตอร์ขนาด 12 บิต(ในรูป

ชื่อ P_OUT) ซึ่งเมื่อได้ครบทั้ง 9 ค่าแล้ว ก็จะเอาผลลัพธ์ที่ได้แค่ 8 บิตบน ส่งไปให้กับรีจิสเตอร์ขนาด 8 บิตตัวต่อไป(ในรูปชื่อ DATA_OUT) ซึ่งก็จะเป็นอินพุตให้กับวงจร Binomial Filter (2) ต่อไป ซึ่งที่กล่าวมาทั้งหมดของวงจร Binomial Filter (1) จะมีขั้นตอนการทำงานตามตารางที่ 4.1

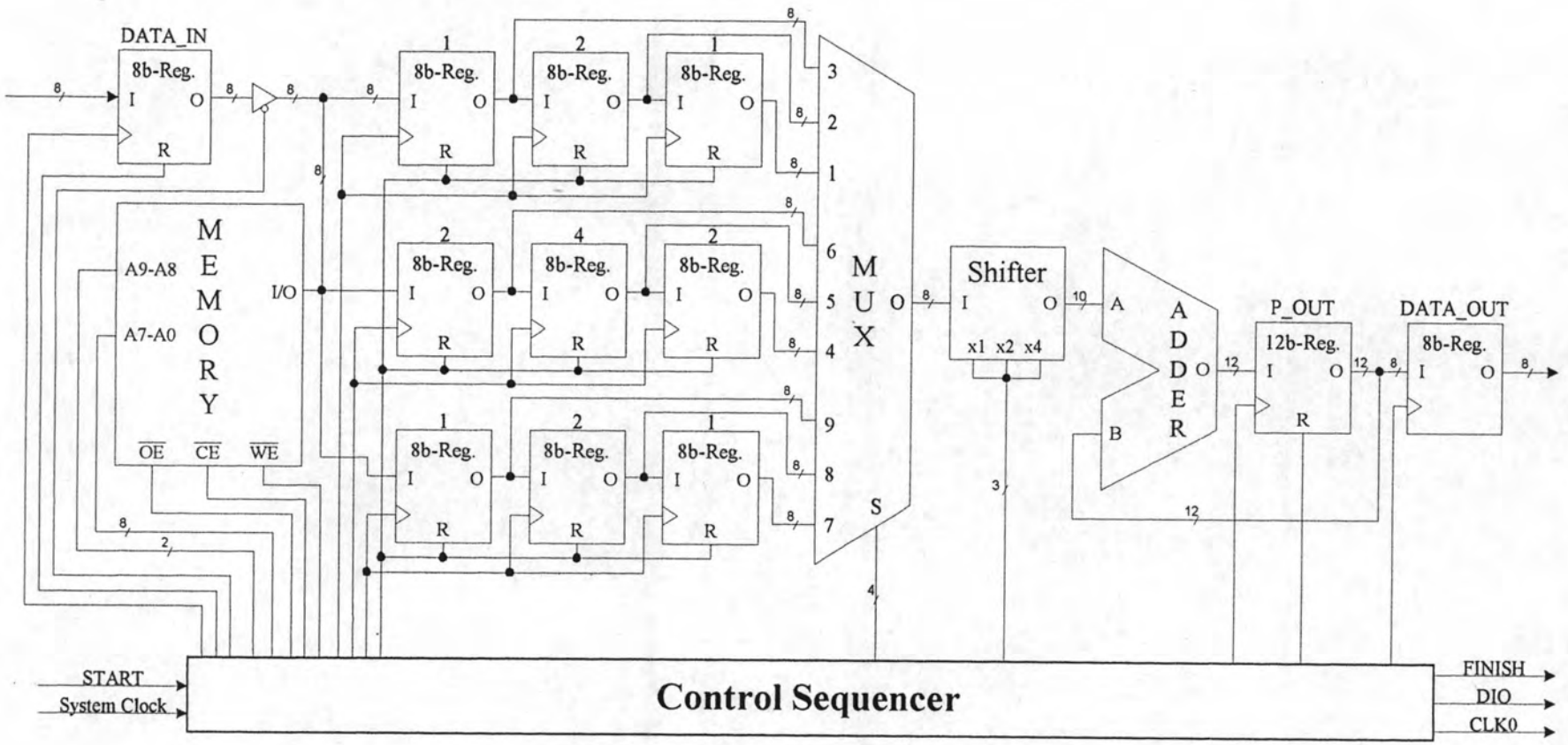
ในรูปวงจร Binomial Filter (1) ในส่วนของวงจรควบคุม(ในรูปชื่อ Control Sequencer) จะเป็นส่วนควบคุมลำดับขั้นการทำงานให้แก่ส่วนต่างๆของวงจร ซึ่งในส่วนนี้จะต้องการสัญญาณนาฬิกา(System Clock) เพื่อเป็นสัญญาณควบคุมความเร็วในการทำงานและต้องการสัญญาณ START จากคอมพิวเตอร์เพื่อให้วงจรเริ่มการทำงาน และจะให้สัญญาณเอาต์พุตออกมา 3 สัญญาณ คือ

1. *สัญญาณ CLK0* เพื่อส่งเป็นสัญญาณบอกให้วงจร Binomial Filter (2) เริ่มทำงานต่อจากวงจร Binomial Filter (1) ได้อย่างถูกต้อง
2. *สัญญาณ DIO (Data In and Data Out)* เป็นสัญญาณที่บอกให้คอมพิวเตอร์อ่านข้อมูลหนึ่งจุดภาพจากเอาต์พุตของวงจร Nonmax Suppression & Low Threshold เข้าไปเก็บไว้ในหน่วยความจำของคอมพิวเตอร์และส่งข้อมูลหนึ่งจุดภาพ(หนึ่งไบต์) จากหน่วยความจำของคอมพิวเตอร์เป็นข้อมูลอินพุตให้กับวงจร Binomial Filter (1)
3. *สัญญาณ FINISH* จะบอกให้คอมพิวเตอร์ทราบว่าฮาร์ดแวร์ได้ทำงานเสร็จสิ้นแล้ว

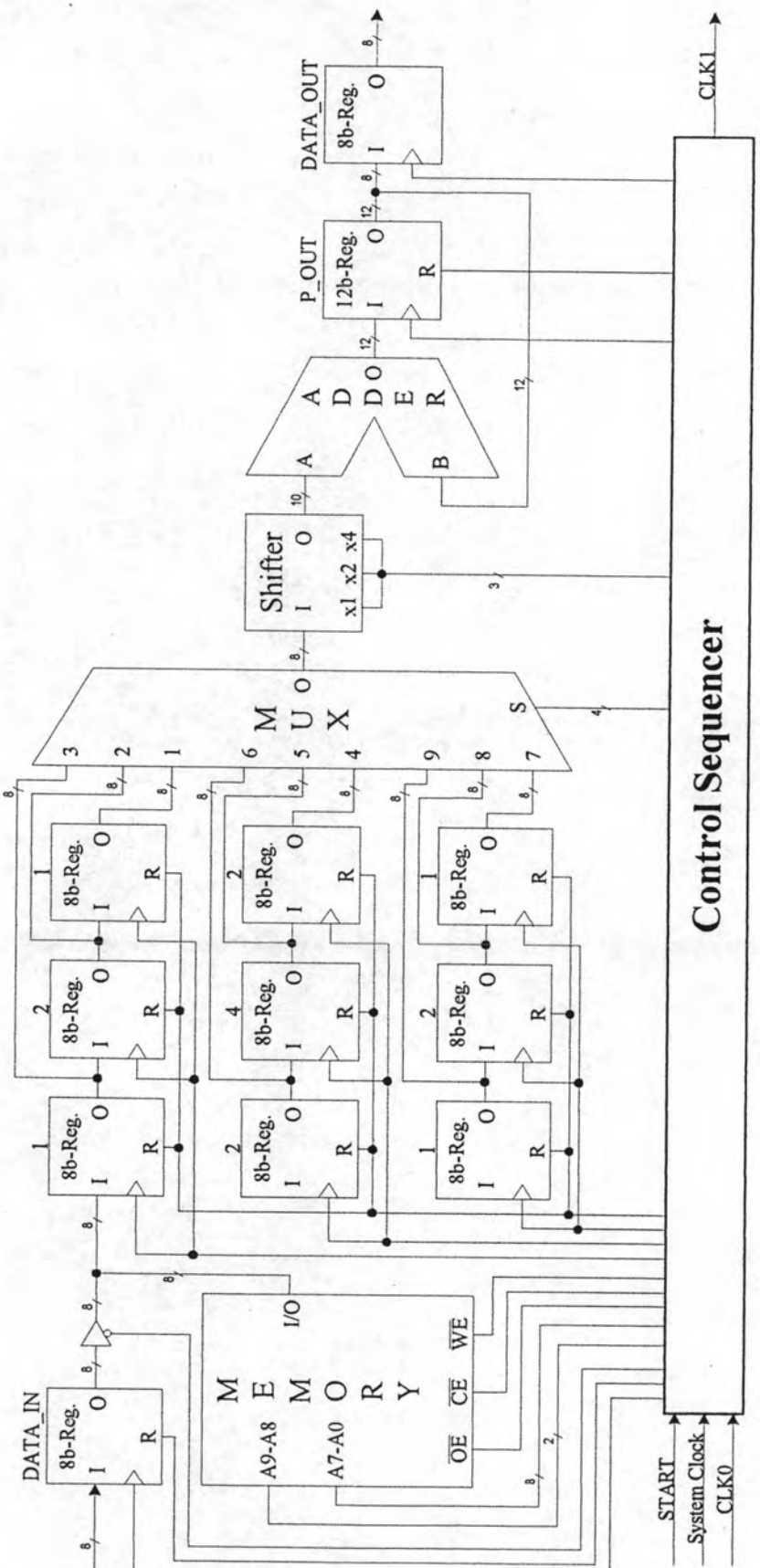
ส่วนวงจร Binomial Filter (2) ก็จะมีส่วนประกอบเหมือนกับวงจร Binomial Filter (1) ดังแผนภาพบล็อกของวงจร* ในรูปที่ 4.11 หน้า 45 จะต่างตรงที่ส่วนของวงจร Control Sequencer ของวงจร Binomial Filter (2) ต้องการสัญญาณอินพุตเพิ่มมาอีกหนึ่งสัญญาณ คือสัญญาณ CLK0 เพื่อให้วงจร Binomial Filter (2) สามารถรับรู้ได้ว่าจะต้องเริ่มทำงานหลังวงจร Binomial Filter (1) เมื่อใด และมีสัญญาณเอาต์พุตเพียงสัญญาณเดียว คือสัญญาณ CLK1 เพื่อเป็นสัญญาณบอกให้วงจร Finding |G| and ϕ เริ่มทำงานต่อจากวงจร Binomial Filter (2) ได้อย่างถูกต้อง ซึ่งตามตารางที่ 4.1 ในส่วนของ Binomial Filter (1) และ Binomial Filter (2) จะมีการทำงานที่เหมือนกัน ยกเว้น Binomial Filter (1) จะมีสัญญาณส่งออกขาเอาต์พุต DIO เพื่อควบคุมจังหวะการอินพุต/เอาต์พุตของพอร์ตคอมพิวเตอร์ และสัญญาณส่งออกขาเอาต์พุต FINISH เพื่อบอกคอมพิวเตอร์ว่าวงจรฮาร์ดแวร์ทำงานเสร็จสิ้นแล้ว

---

* ในการออกแบบวงจร ทั้งหมดเป็นการออกแบบระดับแบบแผน(schematic) แต่ที่แสดงเป็นระดับแผนภาพบล็อก ดังนั้นสำหรับผู้ที่สนใจสามารถติดต่อขอดูวงจรจริงได้จากอาจารย์ที่ปรึกษาวิทยานิพนธ์ฉบับนี้

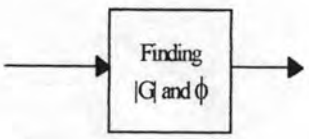


รูปที่ 4.10 แผนภาพบล็อกของวงจร Binomial Filter (1)



รูปที่ 4.11 แผนภาพบล็อกของวงจร Binomial Filter (2)

การออกแบบฮาร์ดแวร์วงจรที่ 2 Finding |G| and ϕ



ขั้นตอนนี้จะทำการคำนวณหา Gy และ Gx โดยใช้หน้ากากแบบ

$$\begin{bmatrix}-1 & 0 & 1\end{bmatrix} \text{ และ } \begin{bmatrix}-1\\0\\1\end{bmatrix} \text{ ตามลำดับ}$$

แล้วมาคำนวณหาขนาดของกราเดียนต์ได้จาก

$$|G| = \sqrt{G_x^2 + G_y^2}$$

และทิศทางของกราเดียนต์ได้จาก

$$\phi = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

และทิศทางของกราเดียนต์ที่ได้เป็นมุมจะต้องถูกเปลี่ยนให้เป็นค่า 0 ถึง 7 โดยแบ่งมุม 360 องศา ออกเป็น 8 ส่วน ส่วนละ 45 องศา

แต่การคำนวณจะต้องใช้วงจรที่ซับซ้อนและกินเวลาในการทำงาน ดังนั้นจึงใช้ตารางเปิดดู (look up table)แทน โดยอาศัยรอมขนาด 64 กิโลไบต์ 2 ตัว สำหรับเก็บค่า |G| ซึ่งมีขนาด 9 บิต(เพราะค่า |G| สูงสุดจะเกิดเมื่อค่าสัมบูรณ์(absolute)ของ Gy และ Gx มีค่าเท่ากับ 255 ซึ่งจะได้ค่า |G| ประมาณ 360.62 ปัดเศษได้เป็น 361 จึงมีขนาดสูงสุด 9 บิต) และ ϕ ซึ่งมีขนาด 3 บิต โดยมีค่า 0 ถึง 7 แต่ค่า ϕ นี้จะเก็บใน รอม แค่ 2 บิต ซึ่งได้มาจากการใช้ค่าสัมบูรณ์ของ Gy และ Gx ในการคำนวณ ก็จะได้ค่า ϕ ขนาด 2 บิต แล้วค่อยนำค่า ϕ 2 บิตนี้ร่วมกับบิตเครื่องหมาย(signbit)ของ Gy และ Gx มาเปลี่ยนให้เป็นค่า ϕ ขนาด 3 บิต

79	80	78	81	76
43	78	81	82	79
46	50	76	80	79
50	49	51	77	78
49	51	48	49	79

ข้อมูลก่อนการคอนโวลูชัน

80	-1	1	-2	-81
78	38	4	-2	-82
50	30	30	3	-80
49	1	28	27	-77
51	-1	-2	31	-49

ข้อมูล Gy

-43	-78	-81	-82	-79
-33	-30	-2	-1	3
7	-29	-30	-5	-1
3	1	-28	-31	0
-50	-49	-51	-77	-78

ข้อมูล Gx

รูปที่ 4.12 แสดงการหาค่า Gy และ Gx

สมมติข้อมูลขนาด 5 x 5 ดังรูปที่ 4.12 รูปด้านซ้าย นำมาคอนโวลูชันกับหน้ากากข้างบน เพื่อหาค่า Gy และ Gx จะได้ออกมาเป็นรูปตรงกลางสำหรับค่า Gy และรูปด้านขวาสำหรับค่า Gx ซึ่งค่าที่อยู่ที่ขอบจะต้องใช้ค่าศูนย์เข้ามาช่วยในการคำนวณ เช่นเดียวกับการออกแบบวงจร Binomial Filter (1)

จากการคำนวณจะเห็นได้ว่าค่าที่ได้มีค่าติดลบรวมอยู่ด้วย นั้นก็คือต้องใช้จำนวนบิต 9 บิต เพื่อแทนค่าข้อมูลที่มีค่าตั้งแต่ -255 ถึง 255 ซึ่งบิตตัวที่ 9 จะเป็นบิตเครื่องหมาย(เป็น '1' เท่ากับลบ และเป็น '0' เท่ากับบวก) จะไม่ใช้การแทนค่าลบด้วยการคอมพลีเมนต์(complement) เพื่อให้สะดวกในการออกแบบ

การคำนวณจะคำนวณหาค่า Gy ก่อนแล้วหาค่า Gx และใช้วงจรเปรียบเทียบเพื่อเปรียบเทียบว่าค่าบวกหรือค่าลบมีค่ามากกว่ากัน ยกตัวอย่างเช่น

79	80	78

คอนโวลูชันกับ

-1	0	1

จะได้ค่า Gy เท่ากับ '-1'

เมื่อใช้วงจรเปรียบเทียบก็จะได้ว่าค่าบวกน้อยกว่าค่าลบ (ค่าบวกก็คือ '78' และค่าลบคือ '79') และวงจรเปรียบเทียบก็จะกำหนดให้บิตเครื่องหมายมีค่าตามค่าที่มากกว่าซึ่งในที่นี้คือค่าลบ แล้วนำทั้งสองค่าไปเข้าวงจรลบ โดยค่าที่มากกว่าจะเป็นตัวตั้งและค่าที่น้อยกว่าเป็นตัวลบ ก็จะได้ผลลบออกมาเป็น '1' และมีบิตเครื่องหมายเท่ากับ '1' คือแสดงว่าเป็นค่าลบ แล้วนำผลลัพธ์นี้ไปเก็บไว้ในรีจิสเตอร์เก็บค่า Gy (เป็นรีจิสเตอร์ขนาด 9 บิต) แล้ววงจรก็ทำการคำนวณหาค่า Gx ต่อไป

เมื่อได้ทั้งค่า Gy และ Gx แล้ว วงจรก็จะนำค่าทั้งสองไปเป็นแอดเดรสอินพุตให้กับรอม เพื่อเป็นการอ่านค่าที่เก็บไว้ในตารางเปิดดู โดยรอมจะเก็บค่า |G| ขนาด 9 บิตไว้ และเก็บค่า ϕ ขนาด 2 บิตไว้ คือจะเก็บค่า 0 ถึง 2 (เฉพาะค่า ϕ ที่ได้จากค่า Gx และ Gy เป็นบวก) ที่ทำอย่างนี้เพราะจะกำหนดไว้ตายตัวว่าค่า Gy จะเข้าไปต่อกับแอดเดรส 8 บิตบนของรอม และค่า Gx จะต่อกับแอดเดรส 8 บิตล่างที่เหลือ โดยบิตเครื่องหมายของค่า Gx และ Gy จะรวมกับค่า ϕ 2 บิตที่ได้จากรอม ผ่านเข้าวงจรลอจิกเพื่อให้ได้ออกมาเป็นค่า ϕ 3 บิต ตามค่าบวกหรือลบของ Gx และ Gy ทั้งสอง

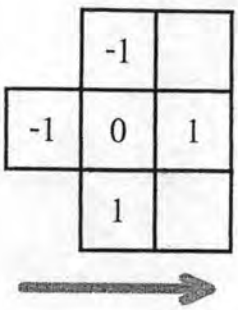
ก็จะได้ขนาดของกราเดียนต์(ค่า |G|) และทิศทางของกราเดียนต์(ค่า ϕ) เป็นเอาต์พุตให้กับวงจรถัดไป(วงจรที่ 3 วงจร Nonmax Suppression and Low Threshold)ต่อไป

ดังนั้นวงจร Finding |G| and ϕ จึงมีส่วนประกอบดังแผนภาพบล็อกในรูปที่ 4.14 หน้า 50 จากในรูปแผนภาพบล็อกจะเห็นว่าวงจรเริ่มจากมีรีจิสเตอร์ขนาด 8 บิต(ในรูปอยู่ด้านซ้ายบน)ในชื่อ DATA_IN เพื่อรับข้อมูลจากวงจร Binomial Filter (2) เข้ามาเก็บไว้ในรีจิสเตอร์นี้ เพื่อส่งต่อเป็นอินพุตให้กับหน่วยความจำและรีจิสเตอร์ที่ทำหน้าที่เป็นหน้ากากสำหรับหาค่า Gx และ Gy ต่อไป

มีหน่วยความจำ(ในรูปชื่อ MEMORY) เพื่อใช้เก็บข้อมูลไว้ 3 แถว เพื่อใช้ในการคำนวณ มีรีจิสเตอร์ขนาด 8 บิต 7 ตัวต่อเรียงกันเป็นเมตริกซ์ดังรูป เพื่อใช้เก็บค่าจุดภาพ 7 ค่า(ที่ได้จากหน่วยความจำและ DATA_IN) เพื่อจะส่งให้วงจรมัลติเพลกซ์ 2 ตัว(ในรูปชื่อ MUX) โดยตัวบนเป็นตัวเลือกค่าบวก(รีจิสเตอร์แถวบนที่มีเลขหนึ่งอยู่ข้างบน)และและตัวล่างเลือกค่าลบ(รีจิสเตอร์แถวล่างที่มีเลขลบหนึ่งอยู่ข้างบน)ของหน้ากาก Gx แล้วตัวบนค่อยเลือกค่าบวก(รีจิสเตอร์แถวกลางที่มีเลขหนึ่งอยู่ข้างบน)และตัวล่างเลือกค่าลบ(รีจิสเตอร์แถวกลางที่มีเลขลบหนึ่งอยู่ข้างบน)ของหน้ากาก Gy เพื่อส่งไปให้วงจรเปรียบเทียบ(ในรูปชื่อ Compar.) ทำการเปรียบเทียบว่าค่าบวกกับค่าลบค่าใดมากกว่ากัน เพื่อที่จะได้ส่งค่าที่มากกว่าไปเป็นตัวตั้ง และค่าที่น้อยกว่าไปเป็นตัวลบให้กับวงจรลบ(ในรูปชื่อ Sub) ทำให้ไม่ต้องใช้วงจรลบแบบคอมพลีเมนต์เพื่อประหยัดการออกแบบ โดยค่าเอาต์พุตของวงจรเปรียบเทียบนี้จะทำการควบคุมบัฟเฟอร์สามสถานะ 4 ตัว เพื่อเป็นเส้นทางในการส่งค่าบวกและค่าลบไปเป็นอินพุตของวงจรลบได้อย่างถูกต้อง และขาเอาต์พุต A<B ของวงจรเปรียบเทียบนี้ จะเป็นค่าของบิตเครื่องหมายด้วย

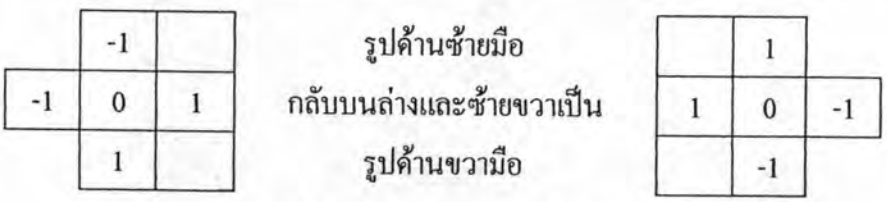
ค่าเอาต์พุตของวงจรลบจะเป็นค่าสัมบูรณ์ของ Gx และ Gy ตามลำดับ ซึ่งเมื่อรวมกับบิตเครื่องหมายจากขา A<B ของวงจรเปรียบเทียบ ก็จะกลายเป็น 9 บิต ซึ่งก็จะถูกนำไว้เก็บไว้ในรีจิสเตอร์ขนาด 9 บิตที่ชื่อ Gx และ Gy ตามลำดับ โดยเอาต์พุตของรีจิสเตอร์ Gx และ Gy เฉพาะ 8 บิตล่างคือค่าสัมบูรณ์ ยกเว้นบิตเครื่องหมาย รวมกันเป็นจำนวน 16 บิต จะไปเป็นขาแอดเดรสให้กับรอมที่ใช้เป็นตารางเปิดดูค่า |G| และค่า ϕ (ในรูปชื่อ 64KB ROM) ซึ่งรอมก็จะให้ค่า |G| ขนาด 9 บิตออกมาเก็บไว้ในรีจิสเตอร์ชื่อ |G|_OUT และให้ค่า ϕ ขนาด 2 บิต(เพราะคิดค่า ϕ เฉพาะในหนึ่งควอดแรนต์) ออกมารวมกับบิตเครื่องหมายจากรีจิสเตอร์ Gx และ Gy รวมเป็น 4 บิตเข้าเป็นอินพุตให้กับวงจรเปลี่ยนรหัส(ในรูปชื่อ Encoder) เพื่อเปลี่ยนค่า ϕ ขนาด 2 บิต เป็นค่า ϕ ขนาด 3 บิต(อาศัยเครื่องหมายบวกลบเพื่อเปลี่ยนค่า ϕ ในหนึ่งควอดแรนต์ให้กลายเป็น 4 ควอดแรนต์) แล้วส่งไปเก็บไว้ในรีจิสเตอร์ชื่อ ϕ_OUT ก็จะได้ค่า |G| และค่า ϕ จากเอาต์พุตของรีจิสเตอร์ |G|_OUT และรีจิสเตอร์ ϕ_OUT ไปเป็นอินพุตให้กับวงจร Nonmax Suppression and Low Threshold ต่อไป ซึ่งที่กล่าวมาทั้งหมดของวงจร Finding |G| and ϕ จะมีขั้นตอนการทำงานตามตารางที่ 4.1

มีข้อสังเกตเกี่ยวกับตำแหน่งของหน้ากาก Gx และ Gy และจำนวนรีจิสเตอร์ที่ใช้ดังนี้



ค่าสัมประสิทธิ์ของหน้ากากเพื่อใช้หาค่า Gx และ Gy จะรวมกันได้ดังรูปด้านซ้ายมือ ซึ่งแสดงให้เห็นว่าต้องใช้รีจิสเตอร์ 5 ตัว แต่ต้องใช้รีจิสเตอร์อีกสองตัวเพื่อการเก็บข้อมูลคือตำแหน่งที่หน้ากากเลื่อนไปในรูปแสดงด้วยกรอบสี่เหลี่ยมว่างเปล่า (หน้ากากเลื่อนจากซ้ายไปขวา)

แต่จากการออกแบบ ข้อมูลจะถูกป้อนเข้าสู่รีจิสเตอร์จึงเปรียบเสมือนข้อมูลเลื่อนเข้าหาหน้ากาก ดังนั้นการจัดเรียงจึงกลับทิศทางกันคือกลับซ้ายไปขวา และการป้อนข้อมูลเข้ารีจิสเตอร์ยังได้กำหนดให้รีจิสเตอร์ที่อยู่แถวบนสุดเป็นแถวที่ 3 ตามมาด้วยแถวที่ 2 อยู่ตรงกลาง และล่างสุดคือแถวที่ 1 นั้นก็คือมีการกลับบนล่าง ดังนั้นการจัดเรียงรีจิสเตอร์จึงต้องมีการกลับตามรูปที่ 4.13



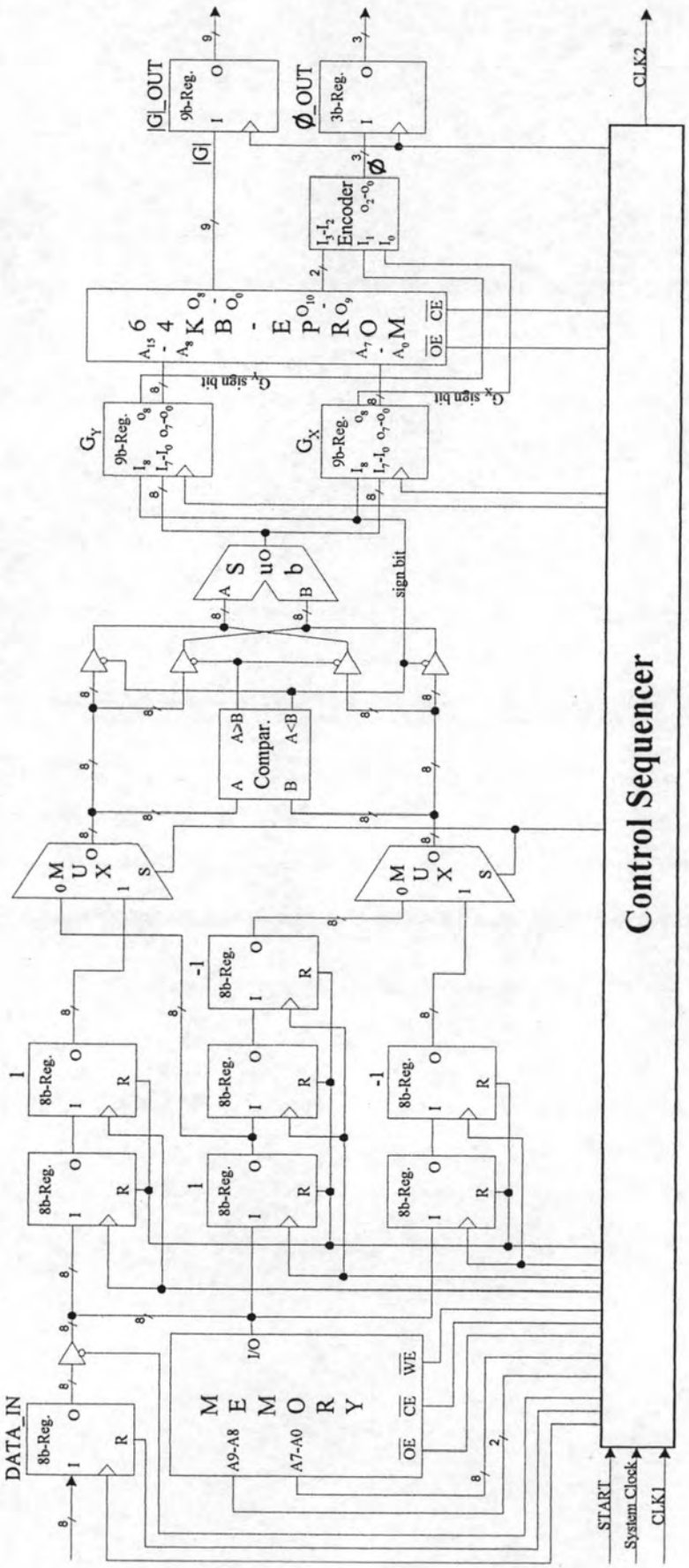
รูปที่ 4.13 แสดงตำแหน่งของตัวหน้ากาก

ในรูปวงจร Finding |G| and ϕ ในส่วนของวงจรควบคุม(ในรูปชื่อ Control Sequencer) จะเป็นส่วนควบคุมลำดับขั้นการทำงานให้แก่ส่วนต่างๆของวงจร ซึ่งในส่วนนี้จะต้องการสัญญาณอินพุต 3 สัญญาณคือ

1. *สัญญาณนาฬิกา(System Clock)* เพื่อเป็นสัญญาณนาฬิกาควบคุมวงจรในการทำงาน
2. *สัญญาณ START* จากคอมพิวเตอร์ เพื่อให้วงจรเริ่มการทำงาน
3. *สัญญาณ CLK1* เป็นสัญญาณจากวงจร Binomial Filter (2) เพื่อให้วงจร Finding |G| and ϕ สามารถเริ่มทำงานต่อจากวงจร Binomial Filter (2) ได้อย่างถูกต้อง

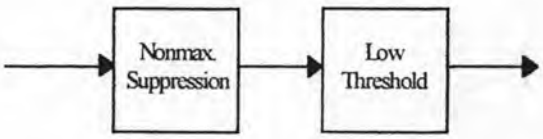
และจะให้สัญญาณเอาต์พุตออกมา 1 สัญญาณคือ

1. *สัญญาณ CLK2* เพื่อส่งเป็นสัญญาณบอกให้วงจร Nonmax Suppression and Low Threshold เริ่มทำงานต่อจากวงจร Finding |G| and ϕ ได้อย่างถูกต้อง



รูปที่ 4.14 แผนภาพบล็อกของวงจร Finding |G| and ϕ

การออกแบบฮาร์ดแวร์วงจรที่ 3 Nonmax Suppression and Low Threshold



หลังจากได้ขนาดของกราเดียนต์(|G|) และทิศทางของกราเดียนต์(ϕ) ก็ทำการหาจุดภาพที่มีขนาดสูงสุดเฉพาะถิ่น เรียกว่า Nonmax Suppression ซึ่งทิศทางของกราเดียนต์(ϕ) มีค่า 0 ถึง 7 เพื่อแทนจุดภาพ 8 ตัวที่อยู่รอบจุดภาพที่ต้องการหาค่าสูงสุดเฉพาะถิ่น(จุดภาพ G) ดังรูปที่ 4.15 ทำให้ใช้ 3 บิตในการแทนค่า ϕ โดยค่าๆหนึ่งของ ϕ แทนค่าละ 45 องศา ดังรูปข้างบน ยกตัวอย่างเช่น ถ้า ϕ มีค่าเป็น '0' ก็จะแทนค่ามุมตั้งแต่ -22.5 ถึง +22.5

5	4	3
6	G	2
7	0	1

รูปที่ 4.15 แสดงจุดภาพที่อยู่รอบจุดภาพที่ต้องการหาค่าสูงสุดเฉพาะถิ่น

เมื่อได้ค่า |G| (ซึ่งมีค่าขนาด 9 บิต)และ ϕ (ซึ่งมีค่าขนาด 3 บิต)มาจากวงจรที่ 2 (วงจร Finding |G| and ϕ) วงจรที่ 3 นี้ก็จะนำค่า |G| และ ϕ ไปเก็บไว้ในรีจิสเตอร์ขนาด 9 บิต 9 ตัวที่เรียงกันเป็นรูปเมตริกซ์ขนาด 3 x 3 โดยค่าที่จะถูกตรวจสอบว่าเป็นค่าสูงสุดเฉพาะถิ่นหรือไม่ จะเป็นค่าที่อยู่ตรงกลางของเมตริกซ์นี้ และ 8 รีจิสเตอร์ที่เหลือจะแทนจุดภาพ 8 จุดภาพที่อยู่รอบรีจิสเตอร์ตรงกลาง โดยอาศัยค่า ϕ เป็นตัวกำหนดทิศทางว่าค่าตรงกลางจะต้องถูกเปรียบเทียบกับค่าใด เช่นถ้าค่า ϕ มีค่าเป็น 2 ค่าตรงกลางก็จะต้องถูกเปรียบเทียบกับค่าที่อยู่ตรงด้านขวาและด้านซ้ายของมัน หรือถ้านับเป็นทิศก็คือทิศตะวันออกและตะวันตก หรือคือมีทิศทางการเปรียบเทียบตามแกน y จากตะวันตกไปตะวันออก ซึ่งก็คือทิศทางของขอบภาพ หรือถ้าค่า ϕ มีค่าเป็น 5 ก็จะต้องเปรียบเทียบกับทิศตะวันตกเฉียงเหนือและทิศตะวันออกเฉียงใต้ คือมีทิศทางของขอบภาพจากตะวันออกเฉียงใต้ไปตะวันตกเฉียงเหนือ ซึ่งค่าที่เก็บอยู่ในรีจิสเตอร์ตรงกลางจะต้องมีค่ามากกว่าค่าที่นำมาเปรียบเทียบทั้งสองค่า โดยจะแทนค่าเป็น G_1 และ G_2 โดยที่ G_2 แทนค่าขนาดของกราเดียนต์ที่มาก่อนหน้าค่า |G| และค่า G_1 แทนค่าขนาดของ กราเดียนต์ที่มาตามหลังค่า |G| ตามทิศทางของ ϕ นั่นคือ $|G| > G_1$ และ $|G| > G_2$ ถึงจะหมายความว่าค่า |G| ค่านี้คือค่าของขอบภาพหรือคือค่าจุดภาพสูงสุดเฉพาะถิ่นก็ให้คงค่านี้ไว้ไปเข้าวงจรในส่วนที่ทำ Low Threshold ต่อไป ไม่เช่นนั้นค่า |G| นี้ก็จะไม่ใช่ค่าจุดภาพสูงสุดเฉพาะถิ่น ก็จะกำหนดให้เป็นค่าศูนย์ไป แต่เนื่องจากการคำนวณใช้ค่าประมาณ

เพราะมีการปัดเป็นจำนวนเต็มและแบ่งทิศทางแค่ 8 ทิศทาง ทำให้ไม่สามารถใช้วิธีที่ว่าค่า |G| ต้องมากกว่าค่า G_1 และ G_2 ได้ เพราะจะทำให้ค่าขอบภาพอาจเกิดการขาดหายได้เนื่องจากเป็นไปได้ว่าค่า |G| อาจจะไปเท่ากับค่าใดค่าหนึ่งในค่า G_1 และ G_2 นี้ ดังนั้นจึงกำหนดใหม่เป็นว่าให้ค่า |G| นั้นมีค่าเป็น $|G| > G_1$ และ $|G| \geq G_2$ จึงจะหมายความว่าค่า |G| นี้คือค่าจุดภาพสูงสุดเฉพาะถิ่น

ส่วนประกอบหลักๆของวงจรที่ 3 นี้ก็จะคล้ายกับวงจรทั้งหมดที่กล่าวมาแล้ว แต่ที่สำคัญคือมีวงจรเปรียบเทียบขนาด 9 บิต 2 ตัว ตัวหนึ่งใช้เปรียบเทียบค่า |G| กับค่า G_1 และอีกตัวใช้เปรียบเทียบค่า |G| กับค่า G_2 และมีรีจิสเตอร์ขนาด 3 บิต 2 ตัวต่อเป็นแบบชิฟรีจิสเตอร์ เพื่อใช้เก็บค่า ϕ ซึ่งจะใช้เป็นตัวควบคุมให้วงจรมัลติเพลกซ์ (เลือก 1 ใน 8 ค่า ขนาด 9 บิต) 2 ตัว ทำการเลือกค่าอินพุตที่มาจากรีจิสเตอร์ 8 ตัวเป็นค่า G_1 และ G_2

หลังจากผ่านวงจรที่ทำ non-max suppression แล้ว ต่อมาก็วงจรส่วนที่ใช้ตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำ เพื่อตัดค่าจุดภาพสูงสุดเฉพาะถิ่นที่มีค่าต่ำ ซึ่งมักเกิดจากสัญญานรบกวนออกไป โดยการกำหนดค่าขีดเริ่มเปลี่ยนไว้ค่าหนึ่ง ซึ่ง Canny ไม่ได้บอกว่าควรจะกำหนดไว้ตายตัวที่เท่าไร บอกแต่เพียงว่าควรจะให้ขีดเริ่มเปลี่ยนค่าสูงมีค่าเป็น 2 หรือ 3 เท่าของขีดเริ่มเปลี่ยนค่าต่ำ ทั้งนี้ขึ้นอยู่กับความเหมาะสมของเส้นขอบภาพที่ได้ ว่าควรจะกำหนดค่าขีดเริ่มเปลี่ยนทั้งค่าสูงและค่าต่ำเป็นเท่าไร หลักการตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำก็คือ ถ้าค่าของจุดภาพมีค่ามากกว่า หรือเท่ากับค่าขีดเริ่มเปลี่ยนค่าต่ำ ก็ให้จุดภาพนั้นมีค่าคงเดิม ถ้าไม่ใช่ก็ให้แก้ค่าของจุดภาพนั้นให้เป็นศูนย์

วงจรที่ใช้ตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำนั้น ก็คือวงจรเปรียบเทียบ แต่ที่พิเศษคือมีอินพุตมาจากดิพสวิตช์(Dip Switch)ขนาด 8 สวิตช์ เพื่อให้ผู้ใช้สามารถเปลี่ยนแปลงค่าขีดเริ่มเปลี่ยนค่าต่ำได้ และที่ใช้ 8 สวิตช์แทนที่จะเป็น 9 สวิตช์(เท่ากับจำนวนบิตของค่า |G|) ก็เพราะว่าโอกาสที่ค่า |G| นั้นจะมีค่าออกมาเท่ากับ 256 หรือมากกว่า คือได้บิตสุดท้ายมีค่าเป็น '1' นั้นมีโอกาสน้อยมาก เพราะเป็นค่าที่สูงมาก เนื่องจากค่าจุดภาพได้ผ่านวงจรที่ 1 (วงจร Binomial Filter (1) และ (2)) เพื่อเกลี่ยภาพทำให้ค่าจุดภาพมีค่าต่ำลง และถ้าค่าที่ได้มีค่าสูงขนาดนี้ก็หมายความว่าค่าจุดภาพค่านี้ต้องเป็นค่าจุดภาพแทนตำแหน่งของขอบภาพอย่างแน่นอน จึงได้ส่วนของวงจรเพื่อตรวจสอบบิตสุดท้ายของค่า |G| นี้ว่า ถ้ามีค่าเป็น '1' ก็ให้เอาต์พุตของวงจรที่ 3 นี้มีค่าเป็น 255 คือ '$1111\ 1111_b$' แต่ถ้ามีค่าเป็น '0' ก็ให้บิตที่เหลืออีก 8 บิตต้องเปรียบเทียบกับการตั้งค่าดิพสวิตช์ที่เป็นอินพุตของวงจรเปรียบเทียบว่าค่าของบิต 8 บิตที่เหลือของค่า |G| นั้นมากกว่าหรือเท่ากับค่าของดิพสวิตช์หรือไม่ ถ้าใช่ก็ให้ออกค่า |G| 8 บิตที่เหลือนี้เป็นค่าเอาต์พุต แต่ถ้าไม่ใช่ก็ให้เอาต์พุตมีค่าเป็น '0'

ดังนั้นวงจร Nonmax Suppression and Low Threshold จึงมีส่วนประกอบดังแผนภาพบล็อกในรูปที่ 4.17 หน้า 56 จากในรูปแผนภาพบล็อกจะเห็นว่าวงจรเริ่มจากมีรีจิสเตอร์ขนาด 9 บิตและขนาด 3 บิต(ในรูปอยู่ด้านซ้ายบน) ในชื่อ |G|_IN และ ϕ_IN ตามลำดับ เพื่อรับข้อมูลจากวงจร

Finding |G| and ϕ เข้ามาเก็บไว้ในรีจิสเตอร์นี้ เพื่อส่งต่อให้เป็นอินพุตให้กับหน่วยความจำและรีจิสเตอร์ที่ทำหน้าที่เก็บค่า |G| ที่ต้องการหาว่าเป็นค่าสูงสุดเฉพาะถิ่นและค่า |G| อีก 8 ค่ารอบค่า |G| นี้ เพื่อแทนทิศทางที่เป็นไปได้ทั้ง 8 ทิศทางของค่า ϕ ซึ่งในรูปวงจรจะเห็นรีจิสเตอร์ขนาด 9 บิต 9 ตัวต่อกันเป็นเมตริกซ์ขนาด 3 x 3 เพื่อเก็บค่า |G| ทั้ง 9 ค่า โดยมีหมายเลขกำกับเพื่อแสดงตำแหน่งค่า |G| ที่อยู่รอบๆจากเลข 0 ถึง 7 และมีรีจิสเตอร์ขนาด 3 บิต เพื่อเก็บค่าทิศทางของค่า |G| ทำให้รู้ว่าจะต้องทำการเปรียบเทียบค่า |G| นี้กับหมายเลขใดซึ่งก็คือเปรียบเทียบกับค่า G_1 และ G_2 ดังที่ได้อธิบายไว้ตอนต้น ซึ่งค่า ϕ นี้ก็จะเป็นตัวกำหนดให้วงจรมัลติเพลกซ์(ในรูปชื่อ MUX)ตัวบนเลือกค่า G_1 และวงจรมัลติเพลกซ์ตัวล่างเลือกค่า G_2 แล้วนำมาเปรียบเทียบกับค่า |G| โดยวงจรเปรียบเทียบ (ในรูปชื่อ Compar.)ตัวบนทำการเปรียบเทียบค่า |G| กับค่า G_1 และวงจรเปรียบเทียบตัวล่างทำการเปรียบเทียบค่า |G| กับค่า G_2 ซึ่งถ้าค่า |G| มีค่าเป็น $|G| > G_1$ และ $|G| \geq G_2$ แสดงว่าค่า |G| นี้เป็นค่าสูงสุดเฉพาะถิ่น ทำให้บัฟเฟอร์สามสถานะที่อยู่ถัดจากวงจรเปรียบเทียบ ยอมให้ค่า |G| นี้ผ่านไปสู่วงจรเปรียบเทียบตัวสุดท้าย เพื่อทำการเปรียบเทียบค่า |G| นี้กับค่าขีดเริ่มเปลี่ยนค่าต่ำ(กำหนดได้จากการเปิดปิดสวิตช์) ซึ่งถ้าค่า |G| มีค่ามากกว่าหรือเท่ากับค่าขีดเริ่มเปลี่ยน ก็จะทำให้ค่า |G| นี้สามารถผ่านไปสู่รีจิสเตอร์ขนาด 8 บิตตัวสุดท้าย(ในรูปชื่อ EDGE_PIXEL) เป็นค่าขอบภาพส่งให้พอร์ตของคอมพิวเตอร์ แต่ถ้าไม่ก็จะทำให้เอาต์พุตของวงจรเปรียบเทียบนี้ทำการตั้งใหม่ให้กับรีจิสเตอร์ EDGE_PIXEL ซึ่งก็คือการตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำ และเนื่องจากค่า |G| สามารถมีค่าได้มากสุดถึง '361' หรือคือขนาด 9 บิต แต่จุดภาพขอบภาพจะมีค่าได้สูงสุดเพียง '255' ดังนั้นจึงต้องทำให้ค่า |G| ที่มากกว่า '255' กลายเป็นค่า '255' โดยการดูบิตที่ 9 ของค่า |G| ถ้ามีค่าเป็น '1' ก็จะทำให้เอาต์พุตของรีจิสเตอร์ EDGE_PIXEL มีค่าเป็น '11111111_b' ซึ่งที่กล่าวมาทั้งหมดของวงจร Nonmax Suppression and Low Threshold จะมีขั้นตอนการทำงานตามตารางที่ 4.1

รีจิสเตอร์ที่ใช้เก็บค่า |G| 9 ค่า จะมีทิศทางกลับซ้ายขวาและบนล่าง ดังแสดงเปรียบเทียบกันในรูปที่ 4.16 เพราะว่าการออกแบบได้กำหนดให้ตัวหน้ากากอยู่กับที่และป้อนข้อมูลใหม่เข้าแถวบน ดังที่ได้อธิบายไว้ใน *"การออกแบบวงจร Binomial Filter (1) และ (2)"*

5	4	3
6	G	2
7	0	1

รูปด้านซ้ายมือ
กลับบนล่างและซ้ายขวาเป็น
รูปด้านขวามือ

1	0	7
2	G	6
3	4	5

<u>รูปที่ 4.16</u> แสดงตำแหน่งของตัวหน้ากาก

และในรูปวงจร Nonmax Suppression and Low Threshold ในส่วนของวงจรควบคุม(ในรูปชื่อ Control Sequencer) จะเป็นส่วนควบคุมลำดับขั้นการทำงานให้แก่ส่วนต่างๆของวงจร ซึ่งในส่วนนี้จะต้องการสัญญาณอินพุต 3 สัญญาณคือ

1. *สัญญาณนาฬิกา(System Clock)* เพื่อเป็นสัญญาณนาฬิกาควบคุมวงจรในการทำงาน
2. *สัญญาณ START* จากคอมพิวเตอร์ เพื่อให้วงจรเริ่มการทำงาน
3. *สัญญาณ CLK2* เป็นสัญญาณจากวงจร Finding |G| and ϕ เพื่อให้วงจร Nonmax Suppression and Low Threshold สามารถเริ่มทำงานต่อจากวงจร Finding |G| and ϕ ได้อย่างถูกต้อง

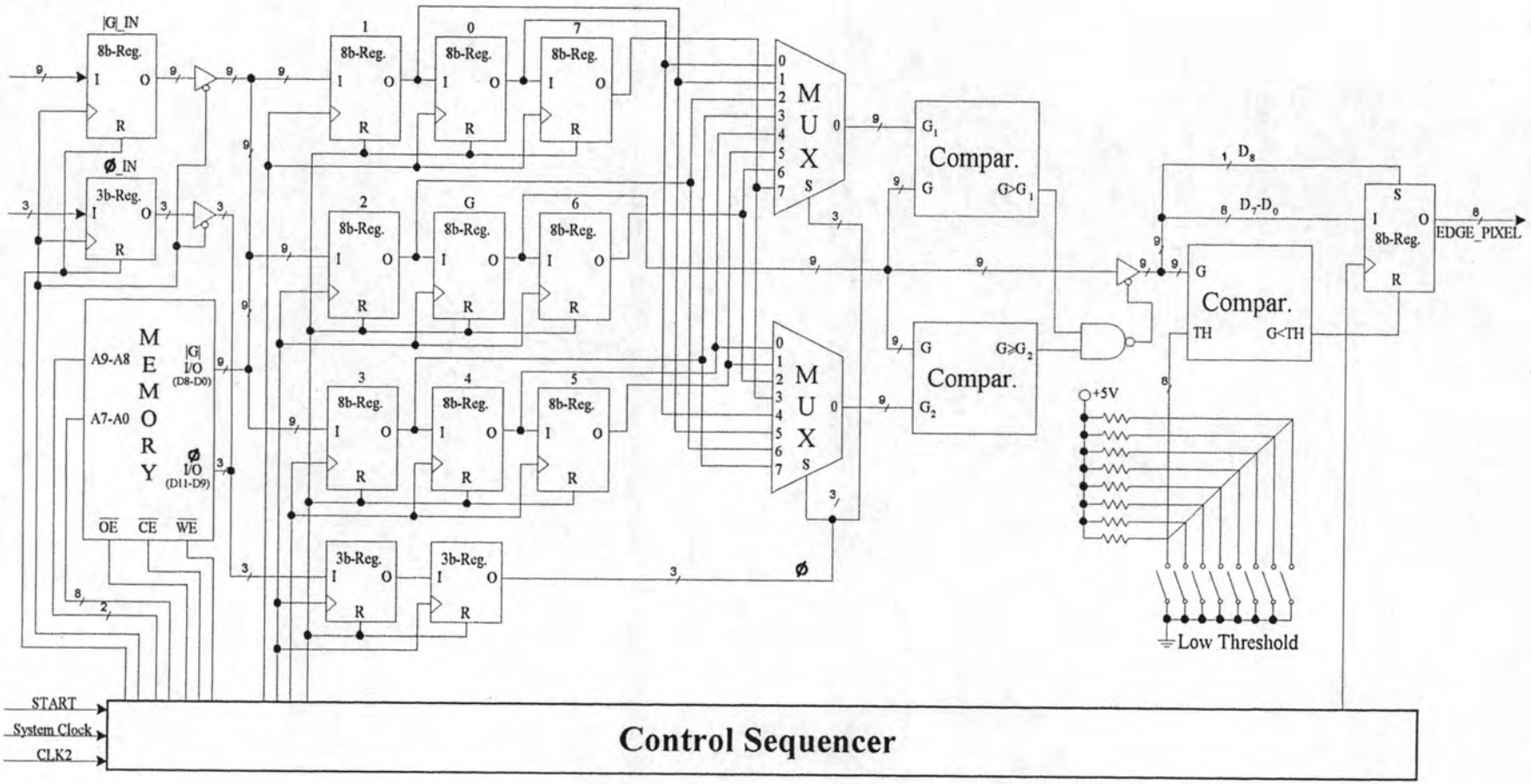
สรุปท้ายบท

การออกแบบฮาร์ดแวร์ตามขั้นตอนวิธีของ Canny โดยไม่ดัดแปลงเลยนั้น จะทำให้ใช้วงจรในการออกแบบฮาร์ดแวร์เป็นจำนวนมาก ซึ่งคงจะไม่สามารถออกแบบได้ในการวิจัยนี้ ดังนั้นจึงได้มีการดัดแปลงขั้นตอนให้มีความซับซ้อนน้อยลง ทำให้ความถูกต้องในการหาขอบภาพลดลง แต่ทำให้สามารถออกแบบฮาร์ดแวร์ได้

และขั้นตอนสุดท้ายคือการตรวจสอบขีดเริ่มเปลี่ยนแบบฮิสเตอร์เรซิสนั้น ต้องใช้การเวียนเกิดในการทำงาน ดังนั้นจึงไม่ออกแบบเป็นฮาร์ดแวร์ จะใช้เป็นซอฟต์แวร์แทน โดยอาศัยโปรแกรมที่ 3 ซึ่งอธิบายไว้ในบทที่ 4 ที่ใช้ตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำและค่าสูงแบบฮิสเตอร์เรซิส ดังนั้น 4 ขั้นตอนที่เหลือจึงนำมาออกแบบเป็นฮาร์ดแวร์ได้ 3 วงจรคือ

1. *วงจรที่ 1 วงจร Binomial Filter* ใช้ฟังก์ชันไบโนเมียลแทนการใช้ฟังก์ชันเกาส์เซียนในการคอนโวลูชันกับรูปภาพ เพื่อการเกลี่ยรูปภาพ และเพื่อให้ได้ค่า σ เท่ากับ 1 ซึ่งเป็นค่าที่ Canny ใช้ จึงต้องแบ่งวงจรนี้ออกเป็น 2 วงจร และใช้ชื่อเป็น *วงจรที่ 0 วงจร Binomial Filter (1)* และ *วงจรที่ 1 วงจร Binomial Filter (2)*
2. *วงจรที่ 2 วงจร Finding |G| and ϕ* ใช้หาค่ากราเดียนต์ |G| และทิศทางของกราเดียนต์ ϕ โดยใช้หน้ากากขนาด 3 x 1 คือ -1, 0, 1 มาหาค่า Gx และ Gy แล้วคำนวณหาค่า |G| และค่า ϕ และเปลี่ยนค่า ϕ ให้เหลือเพียง 8 ค่า คือแทน 8 ทิศทาง เพื่อลดขนาดในการออกแบบฮาร์ดแวร์
3. *วงจรที่ 3 วงจร Nonmax Suppression and Low Threshold* ใช้ค่า |G| และค่า ϕ ที่ได้จากวงจรที่ 2 มาทำการหาค่าสูงสุดเฉพาะถิ่น

การออกแบบจะใช้พอร์ตขนานของคอมพิวเตอร์เป็นตัวรับส่งข้อมูลกับฮาร์ดแวร์ และวงจรทั้ง 4 คือวงจรที่ 0 ถึง 3 จะออกแบบให้ทำงานแบบไพป์ไลน์ คือทำงานต่อเนื่องกันไป โดยวงจรที่ 0 ได้รับอินพุตมาจากพอร์ตขนานและวงจรที่ 3 ส่งเอาต์พุตให้พอร์ตขนาน นั่นคือวงจรที่ 0 รับข้อมูลจุดภาพหนึ่งข้อมูลมาจากคอมพิวเตอร์ ทำการคำนวณเพื่อเกลี่ยข้อมูล แล้วส่งเอาต์พุตให้วงจรที่ 1 ทำการเกลี่ยข้อมูลอีกที แล้วส่งเอาต์พุตให้วงจรที่ 2 ทำการคำนวณหาค่า |G| และค่า ϕ แล้วส่งเป็นเอาต์พุตให้วงจรที่ 3 หาค่าสูงสุดเฉพาะถิ่นและตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำ ก็จะได้ข้อมูลจุดภาพของขอบภาพที่ผ่านการตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำส่งไปให้พอร์ตของคอมพิวเตอร์ โดยสัญญาณที่ใช้ติดต่อกับคอมพิวเตอร์จะอยู่ในวงจรที่ 0



รูปที่ 4.17 แผนภาพบล็อกของวงจร Nonmax Suppression and Low Threshold