บทที่ 4
การออกแบบระบบแผงแสดงภาพสีไดโอดเปล่งแสง โดยใช้ระบบเครือข่ายท้องถิ่น

วิทยานิพนธ์นี้ต้องการสร้างแผงแสดงภาพสีไดโอดเปล่งแสงที่สามารถแสดงภาพเคลื่อนไหวได้โดยใช้ระบบเครือข่ายท้องถิ่นในการส่งข้อมูล เนื่องจากในปัจจุบันหลอดไดโอดเปล่งแสงมีประสิทธิภาพมากขึ้น สามารถแสดงสีได้มากขึ้น และมีราคาถูกลง ทำให้แผงแสดงภาพสีไดโอดเปล่งแสงมีการใช้งานกันอย่างแพร่หลาย โดยรูปแบบในการส่งข้อมูลภาพเคลื่อนไหวนิยมใช้ใยแก้วนำแสงเป็นตัวกลางในการส่ง ซึ่งมีราคาแพง จึงมีความคิดจะใช้อุปกรณ์เครือข่ายท้องถิ่นแทน เนื่องจากปัจจุบันอุปกรณ์เครือข่ายท้องถิ่นมีราคาถูก , สามารถหาได้ทั่วไป และสามารถส่งข้อมูลได้ที่ความเร็วสูง

ระบบแผงแสดงภาพสีไดโอดเปล่งแสงที่ต้องการออกแบบ จะแยกโมดูลแต่ละโมดูลให้อิสระออกจากกัน เพื่อลดการเชื่อมต่อสายระหว่างโมดูล และออกแบบให้ระบบสามารถขยายขนาดได้ เพื่อรองรับการใช้งานในหลายรูปแบบ

4.1 แนวความคิดในการออกแบบระบบ

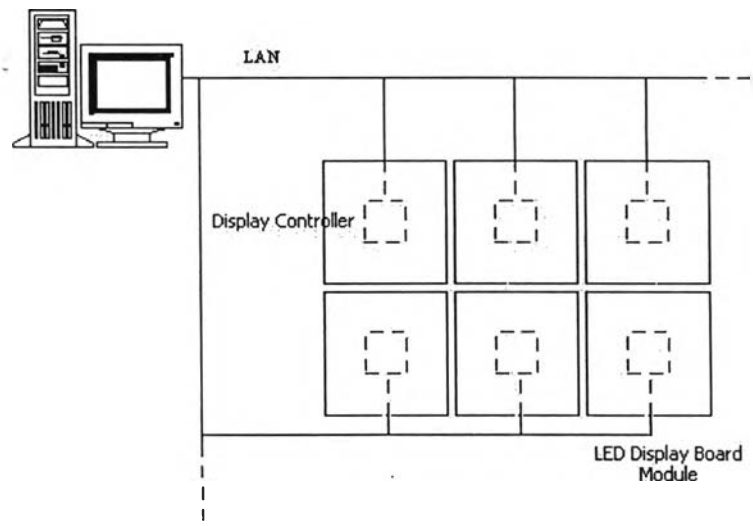
ระบบที่แผงแสดงภาพสีไดโอดเปล่งแสงโดยใช้ระบบเครือข่ายท้องถิ่นที่ออกแบบ มีแนวคิดในส่วนต่างๆ ดังนี้

- ใช้สัญญาณภาพจากจอแสดงผลคอมพิวเตอร์เป็นหลักในการแสดง เนื่องจากปัจจุบันคอมพิวเตอร์ส่วนบุคคลสามารถรับภาพจากอุปกรณ์ต่างๆ เช่น โทรทัศน์ , เครื่องเล่นวีดีโอ , กล้องวีดีโอ มาแสดงบนหน้าจอคอมพิวเตอร์ได้ ทำให้สามารถนำภาพจากแหล่งกำเนิดหลายแบบมาแสดงได้ โดยไม่ต้องออกแบบส่วนรับสัญญาณใหม่ตามรูปแบบสัญญาณที่ต้องการใช้แสดง
- ด้านแผงแสดงภาพไดโอดเปล่งแสงใช้ไมโครคอนโทรลเลอร์ที่มีระบบปฏิบัติการภายใน (Operating System) เป็นตัวรับข้อมูลผ่านระบบเครือข่ายท้องถิ่น สะดวกในการรับข้อมูลผ่านระบบเครือข่ายท้องถิ่น

- ใช้โปรโตคอล TCP/IP ในการส่งข้อมูลเนื่องจากมีความเชื่อถือได้สูง สามารถใช้งานได้โดยไม่เสียค่าลิขสิทธิ์ และใช้ระบบเครือข่ายแบบอีเธอร์เน็ต เนื่องจากมาตรฐานระบบเครือข่ายนี้ไม่มีลิขสิทธิ์ ทำให้ราคาอุปกรณ์ถูก , หาได้ง่าย และสามารถส่งข้อมูลได้เร็ว ระบบเครือข่ายนี้เป็นที่นิยมใช้งานในปัจจุบัน

- การใช้งานแต่ละแผงแสดงภาพแยกจากกันอิสระ แต่ละแผงจะมี IP ต่างกันไป การส่งข้อมูลจะส่งเฉพาะข้อมูลภาพที่ต้องการแสดงบนแผงนั้นๆเท่านั้น

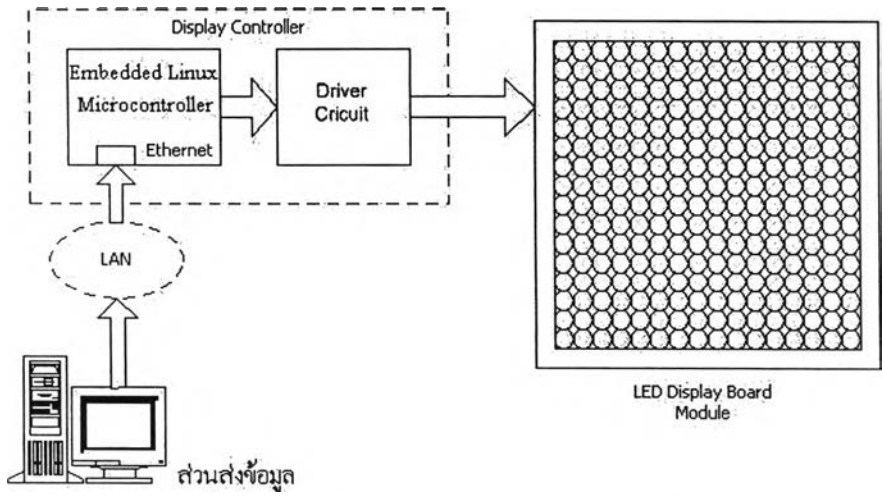
- ใช้กระแสคงตัวในการควบคุมระดับสีของหลอดไดโอดเปล่งแสง โดยควบคุมด้วยวิธีการ PWM



รูปที่ 4.1 ระบบแผงแสดงภาพแบบแยกอิสระ

4.2 องค์ประกอบของระบบ

ระบบแผงแสดงภาพที่ออกแบบจะแบ่งออกเป็นส่วนๆ ได้แก่ ส่วนส่งข้อมูล , ส่วนควบคุมการแสดงภาพ และ แผงแสดงภาพ ซึ่งแต่ละส่วนมีรายละเอียดดังนี้



รูปที่ 4.2 ระบบแผงแสดงภาพไดโอดเปล่งแสงโดยใช้ระบบเครือข่ายท้องถิ่น

1. ส่วนส่งข้อมูล

ส่วนนี้หมายถึงคอมพิวเตอร์ส่วนบุคคลและโปรแกรมภายใน ซึ่งทำหน้าที่จับภาพบนจอแสดงผลของคอมพิวเตอร์ แปลงภาพที่ได้อยู่ในรูปข้อมูลสี ก่อนจะส่งสัญญาณข้อมูลสีที่ได้ไปยังตัวควบคุมการแสดงภาพบนแผงแสดงภาพไดโอดเปล่งแสง ผ่านระบบเครือข่ายท้องถิ่น

2. ส่วนควบคุมการแสดงภาพ (Display Controller)

เป็นส่วนรับข้อมูลจากระบบเครือข่ายท้องถิ่น และประมวลผลเพื่อส่งสัญญาณไปขับหลอดไดโอดเปล่งแสงแต่ละสีบนแผงแสดงภาพ โดยแบ่งออกเป็น 2 ส่วนหลักๆคือ

- ส่วนรับข้อมูล หน้าที่หลัก คือ การรับสัญญาณจากเครือข่ายท้องถิ่น
- ส่วนวงจรขับ (Driver Circuit) หน้าที่หลัก คือ ขับหลอดไดโอดเปล่งแสง

3. แผงแสดงภาพไดโอดเปล่งแสง

แผงแสดงภาพไดโอดเปล่งแสงประกอบด้วยไดโอดเปล่งแสง 3 สี คือ สีแดง , สีเขียว และสีน้ำเงิน รับสัญญาณสีจากวงจรขับมาแสดงผล ลักษณะของแผงแสดงผลจะแตกต่างกันไปตามการใช้งาน

4.3 การออกแบบระบบ

จากแนวความคิด และองค์ประกอบในการออกแบบที่กล่าวมา จึงได้เลือกและออกแบบระบบในแต่ละส่วนขึ้น โดยในแต่ละส่วน มีรายละเอียดดังนี้

1. ส่วนส่งข้อมูล

โปรแกรมบนคอมพิวเตอร์ส่วนบุคคลเลือกใช้โปรแกรม Visual Basic ในการออกแบบ เนื่องจากใช้งานได้ง่าย การส่งข้อมูลในเครือข่ายท้องถิ่น เลือกใช้มาตรฐาน Fast Ethernet (100 Mbps) , โดยใช้ชุดโปรโตคอล TCP/IP เป็นตัวส่งผ่าน และใช้สายคู่บิดเกลียวแบบไม่มีชีลด์เป็นสายสัญญาณ เนื่องจากชุดโปรโตคอล TCP/IP และมาตรฐานอีเธอร์เน็ตเองไม่มีค่าลิขสิทธิ์ ทำให้อุปกรณ์มีราคาถูกและใช้งานกันแพร่หลาย ส่วนสายคู่บิดเกลียวแบบไม่มีชีลด์สามารถส่งได้ในระยะทางไกล (ประมาณ 100 เมตรเมื่อไม่มีการเชื่อมต่อสาย) หากเปรียบเทียบการใช้สายคู่บิดเกลียวแบบไม่มีชีลด์ส่งข้อมูล กับการใช้สายใยแก้วนำแสงส่งข้อมูล แม้ว่าการใช้สายคู่บิดเกลียวแบบไม่มีชีลด์จะมีระยะทางในการส่งใกล้กว่า แต่ก็มีข้อได้เปรียบในด้านราคา และความสะดวกในการเชื่อมต่ออุปกรณ์ เนื่องจากใยแก้วนำแสงต้องมีการแปลงสัญญาณแสงกลับเป็นสัญญาณไฟฟ้าอีกครั้งหนึ่ง

ในการส่งข้อมูลผ่านชุดโปรโตคอล TCP/IP เลือกใช้โปรโตคอล TCP ในการส่ง แม้ว่าการส่งข้อมูลภาพเคลื่อนไหวส่วนใหญ่ทางอีเธอร์เน็ตจะนิยมใช้งานโปรโตคอล UDP มากกว่าโปรโตคอล TCP เนื่องจากใช้แบนด์วิดท์น้อยกว่า เพราะไม่มีการตรวจสอบความผิดพลาดในการส่งข้อมูล อย่างไรก็ตามในการส่งข้อมูลในระบบที่แบ่งการแสดงภาพออกเป็นส่วนๆ โดยแต่ละส่วนอิสระจากกัน หากไม่สนใจการสูญหายของข้อมูลที่ส่งไป อาจทำให้ภาพที่แสดงผิดเพี้ยนไปได้ จึงได้เลือกใช้โปรโตคอล TCP ในการส่งข้อมูล เพราะสามารถรับประกันได้ว่าข้อมูลจะถึงปลายทางแน่นอน แม้ว่าจะใช้แบนด์วิดท์มากกว่า

ขั้นตอนการทำงานของส่วนส่งข้อมูล

ขั้นตอนการทำงานมีไดอะแกรมดังรูปที่ 4.4 โดยมีรายละเอียดการทำงานดังนี้

1. สร้างช่องทางที่ใช้เชื่อมต่อไปยังเซิฟเวอร์

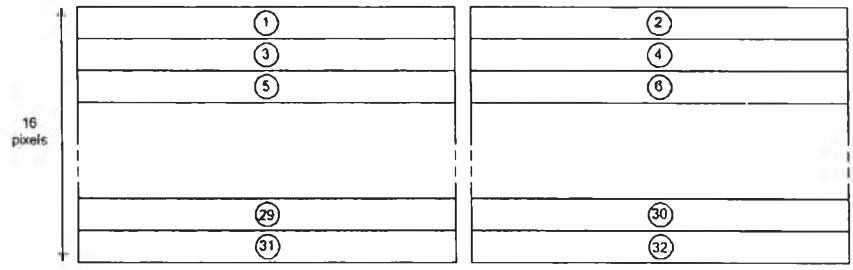
2. เชื่อมต่อกับเซิฟเวอร์ และรอการยืนยันการเชื่อมต่อ

3. การแสดงผล สามารถเลือกให้แสดงจาก desktop , โปรแกรมที่ใช้งานอยู่ในขณะนั้น (active window) หรือเปิดจากไฟล์ที่ต้องการแสดง โดยไฟล์ที่ต้องการแสดงสามารถเปิดได้ 3 รูปแบบ ได้แก่ ไฟล์ภาพ ไฟล์ swf และไฟล์หนัง โดยการแสดงภาพจาก desktop, โปรแกรมที่ใช้งานอยู่ และไฟล์ที่เลือกแสดงในส่วนไฟล์ภาพและไฟล์ swf หลังจากเลือกแล้วระบบจะเริ่มการทำงานในส่วนจับภาพและส่งข้อมูลทันที ส่วนไฟล์ภาพยนตร์จะรอคำสั่งเล่น (play) ก่อนจึงจะเริ่มจับภาพและส่งข้อมูล

4. การจับภาพ จะเก็บข้อมูลภาพจากหน้าจอจะใช้วิธีรับค่าจาก Device Context (DC) ซึ่งเป็นส่วนเชื่อมต่อกับอุปกรณ์ต่าง เช่น การแสดงผล (Display) , การพิมพ์งาน (printer) เป็นต้น หลังจากจับภาพแล้ว จะแปลงข้อมูลที่ได้ให้อยู่ในรูปข้อมูลสีแดง , สีเขียว และสีน้ำเงิน ความละเอียดสีละ 8 บิต โดยข้อมูลที่ได้จะเรียงจากจุดภาพทางด้านซ้ายไปขวา และเรียงแถวจากล่างขึ้นบน (ลักษณะการเรียงข้อมูลจะอยู่ในรูปข้อมูลภาพบิตแมป) การจับภาพเลือกจับในพื้นที่ที่กำหนดไว้ตามขนาดของภาพที่ต้องการจะส่งไปแสดงบนแผงแสดงภาพทั้งหมด เช่น ต้องการส่งข้อมูลภาพขนาด 32 x 32 จุดภาพไปยังแผงแสดงภาพ 2 แผง ขนาด 32 x 16 จุดภาพ ระบบที่ออกแบบจะให้มีการจับภาพขนาด 32 x 32 และแบ่งการส่งข้อมูลให้กับแต่ละแผง ทั้งนี้เนื่องจาก การแบ่งจับข้อมูลขนาด 32 x 16 จุดภาพ 2 ครั้ง จะเสียเวลามากกว่าจับภาพทั้งหมดในครั้งเดียว

5. การส่งข้อมูล เนื่องจากแต่ละแผงไม่มีการเชื่อมต่อกัน ทำให้เกิดปัญหาในการแสดงภาพแต่ละแผงให้พร้อมกัน ดังนั้นจึงแบ่งย่อยข้อมูลที่จะส่งไปแต่ละแผง โดยแบ่งออกตามจำนวนแถวของจุดภาพ เช่นสำหรับแผงขนาด 32x16 จุดภาพ (กว้าง 32 จุดภาพ สูง 16 จุดภาพ) จะแบ่งข้อมูลออกเป็น 16 ชุด ในแต่ละชุดจะมีข้อมูลสีของจุดภาพแต่ละแถว การส่งข้อมูลจะสลับส่งไปทุกๆแผงโดยส่งข้อมูลที

ละแถว ดังแสดงในรูปที่ 4.3 จะทำให้เวลาในการรับข้อมูลไม่ต่างกันมากนัก ทำให้ดูเสมือนแสดงผลพร้อมกันทุกแผงได้



รูปที่ 4.3 ลำดับการส่งข้อมูลสำหรับแผงแสดงภาพขนาด 2 แผง

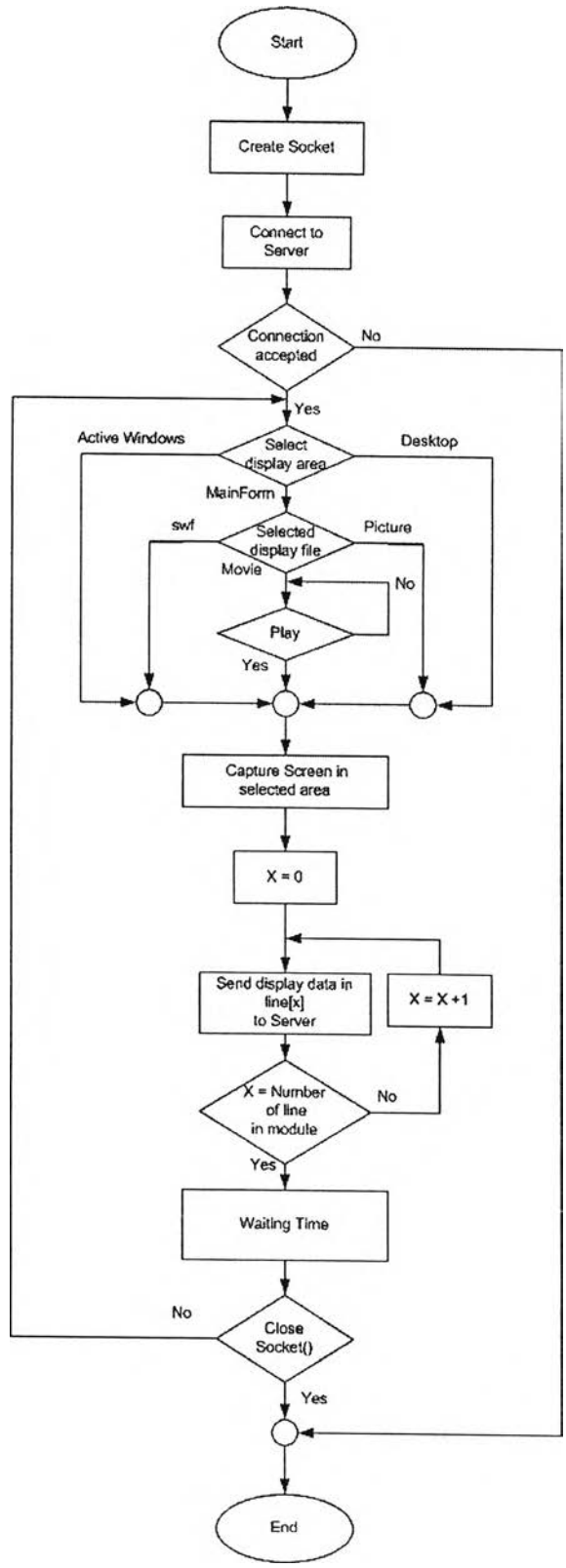
6. หลังจากส่งข้อมูลครบแล้ว จะรอจนถึงการจับภาพครั้งต่อไป จึงเริ่มจับภาพและส่งข้อมูลอีกครั้ง โดยหากภาพที่ส่งไปแสดงบนแผงแสดงภาพไดโอดเปล่งแสงมีความเร็วการเปลี่ยนภาพสูงพอ จะทำให้มองเห็นเป็นภาพเคลื่อนไหวได้

7. ปิดช่องทางเชื่อมต่อระหว่างไคลเอนท์กับ เซิฟเวอร์ เพื่อยุติการส่งข้อมูล

การส่งข้อมูลกำหนดให้ทำทุกๆ 40 ms หรือส่งข้อมูล 25 เฟรมต่อวินาที ซึ่งเป็นความถี่ที่ใช้แสดงภาพเคลื่อนไหวของโทรทัศน์ โดยข้อมูลที่ส่งจะเป็นข้อมูลสีของจุดภาพขนาด 8 บิต 3 สี ดังนั้น

ข้อมูลที่ส่งภายใน 1 วินาที = ความกว้างภาพ (จุดภาพ) x ความยาวภาพ (จุดภาพ) x ขนาดความละเอียดของสี (บิต) x 3 (จำนวนสี) x 25 (ความเร็วข้อมูลภาพต่อ 1 วินาที)

หากภาพมีขนาด 256 x 192 จุดภาพ จะมีข้อมูลทีต้องส่งทั้งหมด 28 Mbps และภาพขนาด 320 x 240 จะมีขนาดข้อมูลที่ต้องส่ง 43.94 Mbps ซึ่งระบบเครือข่ายท้องถิ่นขนาด 100 Mbps น่าจะสามารถรองรับการใช้งานนี้ได้ แม้จะเพิ่มข้อมูลในส่วน header ของข้อมูลที่รับส่งเข้าไป



X = แถวของภาพที่ถูกส่งไปยังโมดูลแสดงภาพ

รูปที่ 4.4 ขั้นตอนการส่งข้อมูลไปยังส่วนส่งข้อมูล

2. **ส่วนควบคุมการแสดงภาพ**

ในส่วนนี้จะแบ่งการทำงานออกเป็น 2 ส่วนหลักดังที่กล่าวไว้แล้ว คือ ไมโครคอนโทรลเลอร์ และส่วนวงจรขับ โดยแต่ละส่วนมีรายละเอียดดังนี้

ส่วนรับข้อมูล

ในส่วนรับข้อมูล เลือกใช้ไมโครคอนโทรลเลอร์ระบบปฏิบัติการลีนุกส์ฝังตัว uClinux เพื่อให้สะดวกต่อการติดต่อกับอุปกรณ์ต่างๆ และการเขียนโปรแกรม โดยเลือกไมโครคอนโทรลเลอร์ระบบปฏิบัติการลีนุกส์ฝังตัว uClinux หน้าที่หลักในส่วนนี้ คือ รับข้อมูลจากเครือข่ายท้องถิ่น , จัดข้อมูลสีของแต่ละจุดภาพและสร้างสัญญาณควบคุมแผงแสดงภาพไดโอดเปล่งแสงเพื่อส่งให้ส่วนวงจรขับต่อไป

ระบบปฏิบัติการลีนุกส์ฝังตัว uClinux เป็นระบบปฏิบัติการลีนุกส์ฝังตัวแบบหนึ่งที่มีขนาดเล็กประมาณ 2 MB เป็นตัวควบคุม โดยระบบปฏิบัติการลีนุกส์ฝังตัว (Embedded Linux) เป็นระบบปฏิบัติการฝังตัวรูปแบบหนึ่งที่นำระบบปฏิบัติการลีนุกส์ (Linux Operating System : Linux OS) มาใช้งาน เพื่อให้ง่ายต่อการออกแบบโปรแกรมและใช้งาน

ระบบปฏิบัติการลีนุกซ์ สร้างขึ้นโดย Linus Torvalds ขณะยังเป็นนักศึกษาอยู่ที่มหาวิทยาลัย Helsinki ประเทศฟินแลนด์ พัฒนามาจากระบบปฏิบัติการยูนิกซ์ (UNIX) โดยสร้างตามแบบของระบบปฏิบัติการมินิกซ์ (Minix) เป็นระบบปฏิบัติการแบบเปิด (Open source) ทำให้การใช้งานไม่เสียค่าลิขสิทธิ์ทางซอฟต์แวร์ ระบบปฏิบัติการลีนุกซ์อยู่ภายใต้ลิขสิทธิ์ GPL (GNU General Public License) ซึ่งอนุญาตให้นำรหัสต้นฉบับมาทำการแก้ไข ปรับปรุงและแจกจ่ายได้ตามความต้องการอย่างเป็นอิสระ โดยซอฟต์แวร์ที่ถูกปรับปรุงนั้นจะเป็นลิขสิทธิ์ GPL ทำให้มีผู้สนใจใช้งานและร่วมพัฒนากันอย่างแพร่หลายทั้งในภาคธุรกิจ และในมหาวิทยาลัยต่างๆ ในปัจจุบันนี้สามารถหาระบบปฏิบัติการลีนุกซ์ในรูปแบบต่างๆ มาใช้ได้สะดวกขึ้น เช่น Fedora , Slackware, Red Hat, Debian, SuSE , Ziif เป็นต้น ระบบปฏิบัติการลีนุกส์ฝังตัวก็เช่นกัน มีการดัดแปลงออกมาในหลายรูปแบบ เช่น uClinux , MontaVista , RTOS เป็นต้น

รูปแบบการทำงานของโปรแกรมที่ออกแบบ จะสามารถแบ่งออกเป็น 2 ส่วนหลัก คือ

1. ส่วนรับข้อมูลสีจากเครือข่ายท้องถิ่น ออกแบบให้ทำหน้าที่เป็นเซิฟเวอร์รอรับข้อมูลสีจากคอมพิวเตอร์ส่วนบุคคลที่ทำหน้าที่เป็นไคลเอนท์

2. ส่วนจัดข้อมูลสี และสร้างสัญญาณควบคุม เพื่อส่งไปยังส่วนวงจรขับ ในส่วนนี้จะแยกสัญญาณสีแต่ละสีของจุดภาพออกจากกัน และส่งออกไปทีละจุดภาพ รวมทั้งสร้างสัญญาณควบคุมขึ้นเพื่อให้แผงแสดงภาพไดโอดเปล่งแสงสามารถแสดงภาพได้ถูกต้อง โดยสัญญาณที่ส่งออกไปยังส่วนวงจรขับมีดังนี้

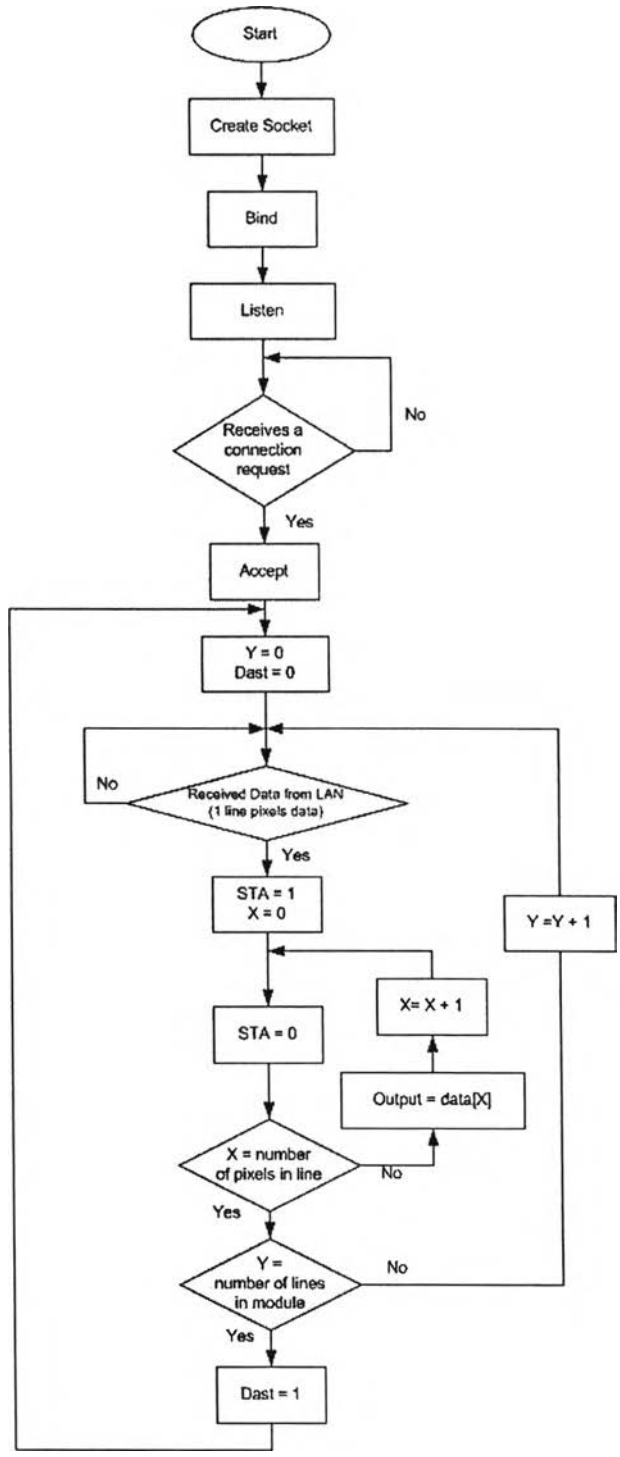
   - สัญญาณข้อมูลสี 3 สี สีละ 8 บิต
   - สัญญาณนาฬิกา สำหรับกำหนดข้อมูลสี
   - สัญญาณ STA ใช้เลื่อนแถวรับข้อมูลสีในแต่ละแถวแผง
   - สัญญาณ STB ใช้เลื่อนแถวโมดูลที่ต้องการแสดงภาพ กรณีควบคุมโมดูลมากกว่า 1 แถว
   - สัญญาณ DAST เป็นสัญญาณให้แผงแสดงข้อมูลที่รับเข้าไป

<u>ขั้นตอนการทำงานของส่วนรับข้อมูล</u>

ขั้นตอนการทำงานในส่วนนี้แสดงในไดอะแกรมรูปที่ 4.5 โดยมีรายละเอียดการทำงานดังนี้

1. สร้างช่องทางสำหรับเชื่อมต่อ
2. รอการเชื่อมต่อจากส่วนส่งข้อมูลที่ทำหน้าที่เป็นไคลเอนท์
3. เมื่อยืนยันการเชื่อมต่อแล้ว จะเริ่มรับข้อมูลสีจากส่วนส่งข้อมูลทีละ 1 แถวจุดภาพ
4. ส่วนรับข้อมูลสร้างสัญญาณ STA ขึ้น เพื่อให้ส่วนวงจรขับเริ่มรับข้อมูลแถวใหม่
5. ส่งข้อมูลสีของจุดภาพไปยังวงจรขับจนครบทั้งแถว แล้วส่วนรับข้อมูลจึงเปิดให้รับข้อมูลในแถวถัดไปจากระบบเครือข่ายอีกครั้ง
6. กรณีควบคุมโมดูลแสดงภาพมากกว่า 1 แถว จะส่งสัญญาณ STB ออกไป เพื่อเลื่อนแถวการรับข้อมูลของโมดูล แล้วจึงเริ่มส่งข้อมูลให้กับแถวโมดูลใหม่

7. เมื่อส่งข้อมูลออกไปครบทุกจุดภาพแล้ว (ส่งข้อมูลไปครบทุกจุดภาพในแผง) จะส่งสัญญาณ DAST ไปยังส่วนวงจรขับเพื่อให้แสดงข้อมูลที่ได้รับออกมา

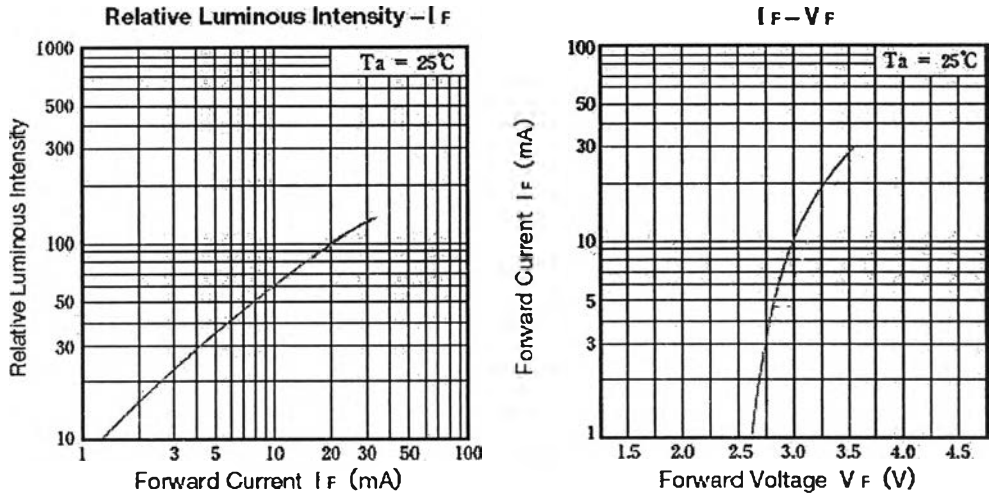


X = ตำแหน่งจุดภาพในแนวนอน , Y = ตำแหน่งแถวจุดภาพ

รูปที่ 4.5 ขั้นตอนการทำงานของส่วนรับข้อมูลที่ส่งให้แต่ละโมดูล

ส่วนวงจรขับ (Driver Circuit)

ในส่วนนี้จะทำหน้าที่ขับไดโอดเปล่งแสงที่อยู่บนแผง และเนื่องจากความต้องการระดับสีที่แตกต่างกันออกไป ทำให้ต้องการปรับความเข้มแสงของไดโอดเปล่งแสงที่เปล่งออกมา จากความสัมพันธ์ของความสว่างกับแรงดันและกระแสของไดโอดเปล่งแสง ดังแสดงในรูปที่ 4.6 จึงเลือกการขับหลอดไดโอดเปล่งแสงด้วยกระแสคงตัว



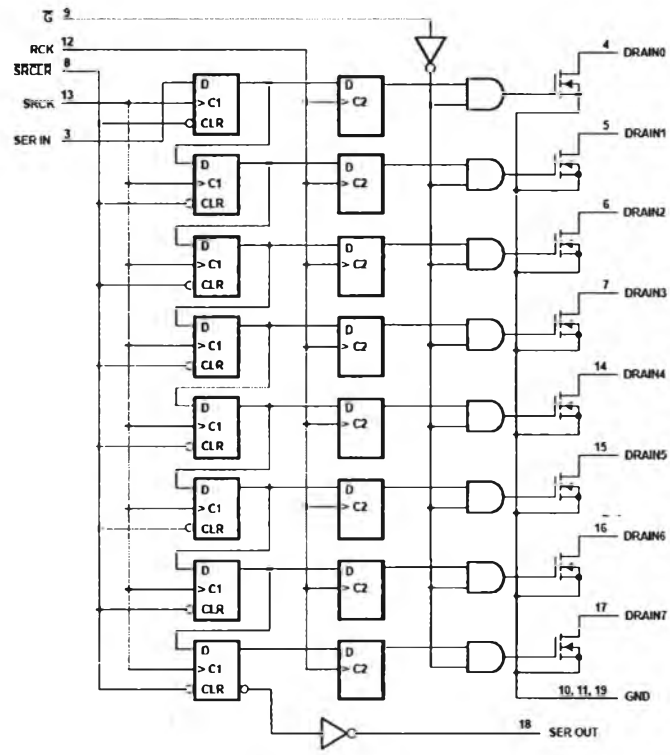
รูปที่ 4.6 ความสัมพันธ์ระหว่างความสว่าง , แรงดัน และกระแสของไดโอดเปล่งแสง

การออกแบบให้ระบบสามารถสร้างสัญญาณกระแสคงที่ที่สามารถเปลี่ยนแปลงค่าได้ตามความต้องการโดยตรงทำได้ลำบาก จึงนำการสร้างสัญญาณกระแสคงตัวด้วย PWM มาใช้งาน ซึ่งใช้วิธีเปิดปิดการจ่ายกระแส ทำให้เสมือนว่าระบบจ่ายกระแสคงที่ในระดับต่างๆออกมาได้

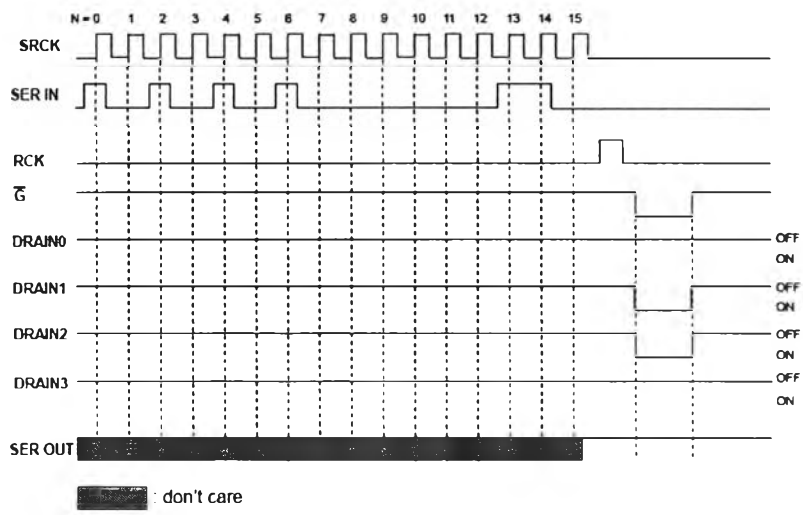
ในการขับไดโอดเปล่งแสงหากขับโดยตรง 1 หลอดต่อ 1 ชุดขับ จะทำให้วงจรมีขนาดใหญ่มาก และมีการเชื่อมต่อที่ยุ่งยาก (3 สี สีละ 1 จุดภาพเป็นอย่างต่ำ) จึงได้เลือกใช้ชิปสำเร็จรูปที่สร้างสัญญาณกระแสคงตัวสำหรับขับไดโอดเปล่งแสงมาใช้งาน โดยชิปดังกล่าวจะรับข้อมูลเข้ามาในรูปสัญญาณอนุกรมของข้อมูลสี และเลื่อนค่าที่รับเข้ามาออกไปแสดงยังขาต่างๆ ดังรูปที่ 4.7 และมีสัญญาณการทำงานดังรูปที่ 4.8 อีกทั้งยังสามารถเชื่อมต่อกันระหว่างชิปดังกล่าวเองได้ ทำให้สามารถลดพื้นที่ในส่วนวงจรไปได้มาก

แม้ว่าการสร้างสัญญาณกระแสคงตัวจะง่ายขึ้น แต่การส่งข้อมูลไปยังชิปดังกล่าวเพื่อสร้างสัญญาณกระแสคงที่ด้วย PWM จะยุ่งยากขึ้น เนื่องจากข้อมูลมีลักษณะอนุกรมกัน ทำให้ข้อมูลที่ส่งเข้าต้องเปลี่ยนค่าตลอดเวลา การออกแบบวงจรต้องเพิ่มส่วนการวนค่าขึ้น ทำให้ขนาดวงจรมี

ขนาดใหญ่ ดังนั้นจึงได้นำ FPGA (Field-Programmable Gate Array) เข้ามาช่วยในเรื่องนี้ เพื่อลดขนาดของวงจรลง และทำให้วงจรขับมีขนาดเหมาะสมกับแผงแสดงภาพ



รูปที่ 4.7 องค์ประกอบภายในของชิปสำเร็จรูปสร้างกระแสคงที่สำหรับขับไดโอดเปล่งแสงแบบขาออก 8 ขา



รูปที่ 4.8 ลำดับสัญญาณเข้าและออกของชิปสร้างกระแสคงที่

FPGA (Field-Programmable Gate Array) เป็นอุปกรณ์ PLD (Programmable Logic Device) ที่มีความซับซ้อนอีกระดับหนึ่ง ภายในประกอบด้วยอาเรย์ของลอจิกเกตต่างๆ มีขนาดให้เลือกตั้งแต่ระดับพันเกต จนถึงระดับล้านเกต ทำให้สามารถลดขนาดวงจรลงได้ และรองรับวงจรที่มีความสลับซับซ้อนได้อย่างดี สามารถออกแบบและพัฒนาได้ง่าย

การแบ่งประเภทของ FPGA ตามรูปแบบการโปรแกรม แบ่งได้ 2 แบบ คือ

1. การโปรแกรมโดยทำให้เกิดการเปลี่ยนแปลงทางกายภาพ

   - Fuse หลังจากโปรแกรมจุดเชื่อมต่อจะขาดจากกัน โปรแกรมได้เพียงครั้งเดียว

   - Anti Fuse การโปรแกรมคล้ายแบบ Fuse แต่จะใช้การเชื่อมต่อถึงกัน แทนการทำให้จุดเชื่อมต่อขาดจากกัน

2. การโปรแกรมโดยใช้หน่วยความจำ

   - EEPROM Based FPGA มักเรียกว่า CPLD (Complex Programmable Logic Device) ใช้เทคโนโลยีเหมือน EEPROM มีความจุของเกตต่ำ แต่สามารถเก็บข้อมูลที่โปรแกรมได้โดยไม่ต้องใช้ไฟเลี้ยง

   - SRAM Based FPGA ใช้เทคโนโลยีเหมือน SRAM ทำให้โปรแกรมซ้ำได้ไม่จำกัด ความจุเกตสูง ไม่สามารถเก็บโปรแกรมโดยไม่มีไฟเลี้ยงได้ ส่วนมากจึงใช้ ROM เก็บโปรแกรม และให้โหลดลง FPGAในตอนเริ่มใช้งาน

การออกแบบ FPGA ไม่จำเป็นต้องทราบถึงโครงสร้างภายในชิป ใช้เพียงความรู้การออกแบบลอจิก ต่างจากไมโครโพรเซสเซอร์ที่ต้องรู้ถึงโครงสร้างภายในรวมถึงภาษา Assembly ของไมโครโพรเซสเซอร์นั้น การออกแบบบน FPGA สามารถใช้ภาษา HDL (Hardware Description Language) ออกแบบได้ โดย ภาษา HDL มีความยืดหยุ่นสูง ทำได้รวดเร็ว และไม่จำเป็นต้องทราบลักษณะของวงจรว่าเชื่อมต่อกันอย่างไร เพียงกำหนดลักษณะการทำงาน จากนั้นซอฟต์แวร์จะ Synthesis และ Optimize ให้ทั้งหมด

สัญญาณเข้าและสัญญาณออกจาก FPGA ที่ออกแบบในส่วนวงจรขับ

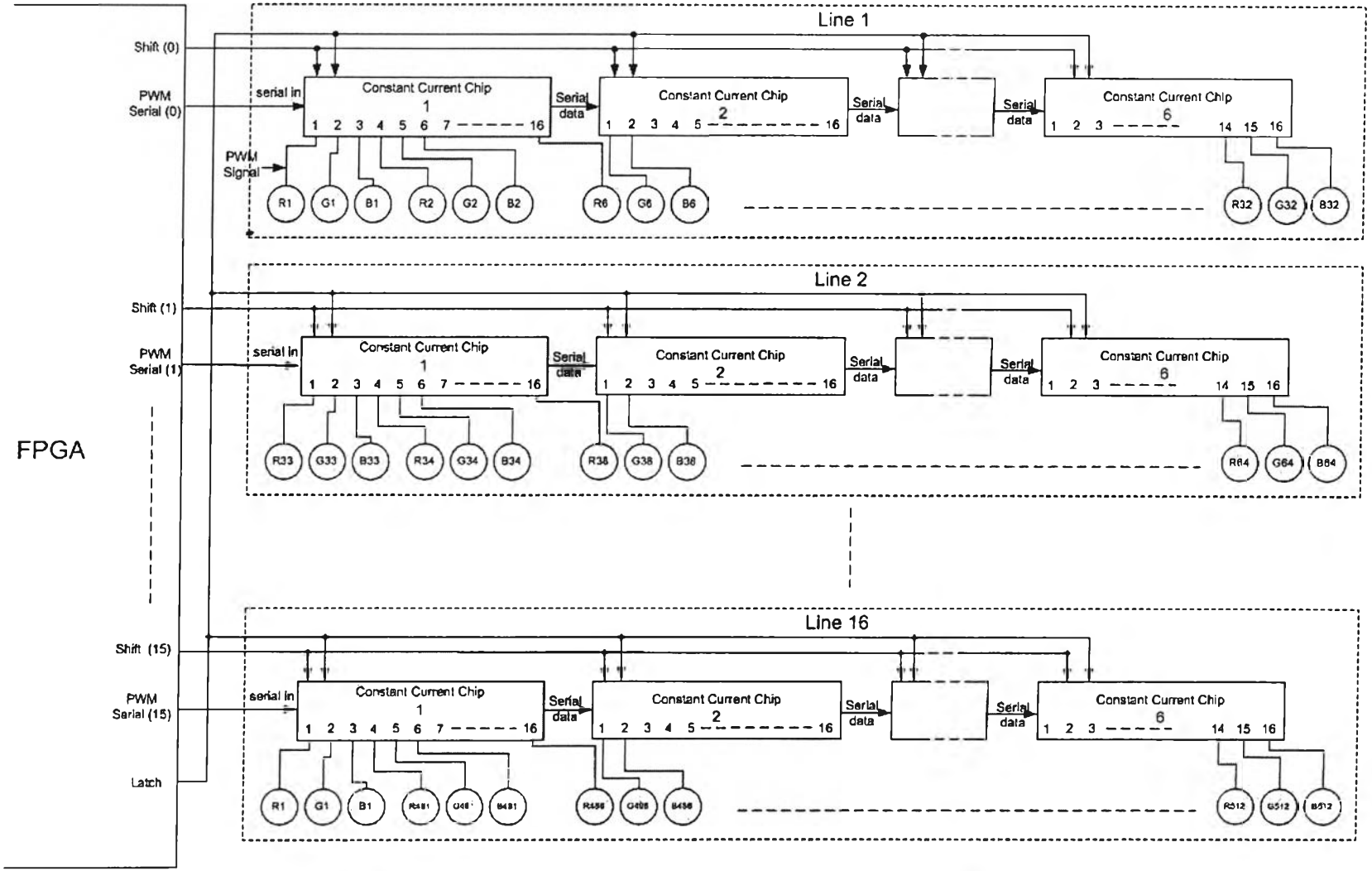
สัญญาณเข้า FPGA จะมาจากส่วนรับข้อมูล เป็นสัญญาณควบคุมและสัญญาณข้อมูลสี ที่ใช้ในการแสดงผล ส่วนสัญญาณออก จะเป็นสัญญาณที่จ่ายให้ชิปสร้างกระแสคงที่ มีการเชื่อมต่อดังรูปที่ 4.9

สัญญาณขาเข้าของ FPGA

- สัญญาณสี (Color Data) แบ่งออกเป็น สีแดง สีเขียว และสีน้ำเงิน สีละ 8 บิต
- สัญญาณนาฬิกา (CLK) ใช้จำแนกชุดข้อมูลสีแต่ละชุดออกจากกัน
- สัญญาณนาฬิกาภายใน (CLK_inner) ใช้สัญญาณนาฬิกาภายในของ FPGA ใช้ในส่วนสร้างสัญญาณ PWM
- STA เป็นสัญญาณเลื่อนแถวจุดภาพ
- STB เป็นสัญญาณเลือกให้โมดูลรับสัญญาณต่างๆที่ส่งเข้ามาภายในวงจรหรือไม่
- DAST ใช้แสดงค่าข้อมูลชุดใหม่ที่แผงเก็บไว้ออกมา

สัญญาณขาออกของ FPGA

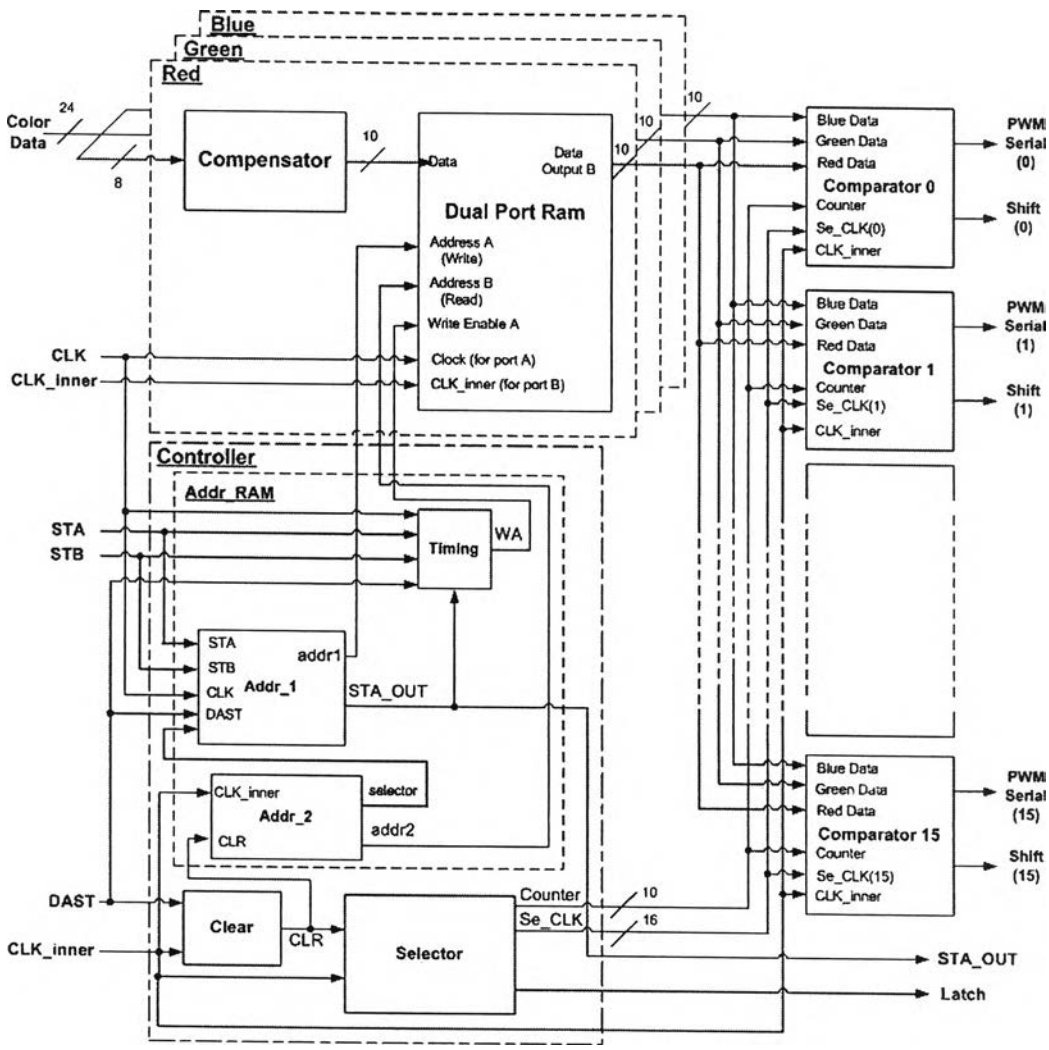
- PWM เป็นสัญญาณข้อมูลสีที่ส่งให้ชิปสำเร็จรูปสร้างกระแสคงที่ โดยแยกเป็นแต่ละแถวของแผงแสดงภาพไดโอดเปล่งแสง ข้อมูลที่ส่งจะสลับกันไปตามรูปแบบการเชื่อมต่อไดโอดเปล่งแสง ในวิทยานิพนธ์นี้ออกแบบให้ส่งข้อมูลสีทั้ง 3 ไปบนสัญญาณออกขาเดียวกัน โดยการส่งจะส่งข้อมูลสีแดง , เขียว และน้ำเงินของจุดภาพที่ 1 ต่อจากนั้นจะเป็น สีแดง , เขียว และน้ำเงิน ของจุดภาพที่ 2 เรื่อยไปจนหมดแถว และวนกลับมาจุดภาพที่ 1 อีกครั้ง
- Shift เป็นสัญญาณที่ใช้เลื่อนการแสดงค่าออกยังขาต่างๆของชิปสร้างสัญญาณกระแสคงที่ (รูปที่ 4.7)
- Latch เป็นสัญญาณใช้แสดงค่าในแต่ละขาที่เลื่อนไว้ ให้ออกมาพร้อมกัน กล่าวอีกอย่างว่าเป็นสัญญาณแสดงภาพนั้นเอง



รูปที่ 4.9 การเชื่อมต่อระหว่าง FPGA กับชิปสร้างกระแสคงที่สำเร็จรูป

การออกแบบวงจรขับโดยใช้ FPGA

มีองค์ประกอบดังรูปที่ 4.10 และมีขั้นตอนการทำงานแยกออกเป็น 2 ส่วนหลักๆ คือส่วนรับค่าจากส่วนรับข้อมูล และส่วนสร้างสัญญาณด้วย PWM โดยมีรายละเอียดของสัญญาณเข้าออก และรูปแบบการทำงานดังนี้



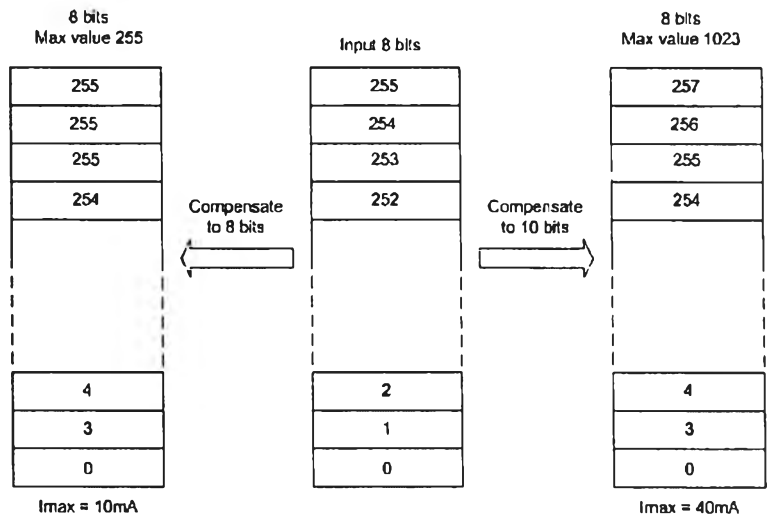
รูปที่ 4.10 องค์ประกอบของโปรแกรมภายใน FPGA ที่ออกแบบ

ส่วนประกอบต่างๆและการทำงานภายใน FPGA

ระบบที่ออกแบบสามารถแบ่งออกเป็น 5 ส่วนใหญ่ๆ คือ ส่วนสีแดง (Red part) , ส่วนสีเขียว (Green Part) , ส่วนสีน้ำเงิน (Blue Part) , ส่วนเปรียบเทียบค่า (Comparator) และส่วนควบคุม (Controller) โดยในส่วนของสีต่างๆจะมีฟังก์ชั่นเหมือนกัน แต่แยกจากกัน เพื่อความสะดวกในการปรับแต่งค่าและการเรียกใช้งาน ส่วนควบคุมจะเป็นส่วนที่รับสัญญาณควบคุมจากส่วนส่งข้อมูลมาประมวลผล และสร้างสัญญาณต่างๆ ที่ใช้ในโปรแกรม

ภายในส่วนควบคุมจะแบ่งหลักเป็น 3 ส่วนคือส่วนควบคุมหน่วยความจำ รับหน้าที่สร้างสัญญาณกำหนดตำแหน่ง (Address) ทั้ง 2 ค่า , ส่วนสร้างสัญญาณ Clear , และส่วนเลือกการทำงาน (Selector) โดยแต่ละส่วนมีหน้าที่ดังนี้

1. ส่วนปรับระดับสี (Compensator) ออกแบบเป็น Look up Table เมื่อมีค่าเข้ามาให้เปรียบเทียบกับตารางที่กำหนดไว้ โดยจะเปลี่ยนระดับสีจาก 8 บิตเป็น 10 บิต เพื่อให้สามารถปรับระดับสีให้เหมาะสมกับไดโอดเปล่งแสงที่ใช้งาน และสามารถแสดงสีได้หลังจากปรับค่าแล้วได้ 256 ระดับ ในส่วนปรับระดับสีแต่ละสีจะแยกออกจากกัน เพราะแต่ละสีจะมีการปรับค่าที่ต่างกันออกไป



รูปที่ 4.11 การเปรียบเทียบการปรับระดับสี

จากรูปที่ 4.11 จะเห็นว่าการปรับระดับจาก 8 บิตเป็น 8 บิต จะทำให้การปรับระดับของค่า 253-255 มีค่าซ้ำกัน ซึ่งทำให้ไม่สามารถแสดงค่าสีได้ 256 ระดับตามที่ต้องการ ดังนั้นจึงออกแบบให้ปรับระดับสีเป็น 10 บิต และเพิ่มค่ากระแส

สูงสุดของชิปสร้างกระแสคงที่ ขึ้น 4 เท่า จาก 10mA เป็น 40mA ทำให้สามารถแสดงค่าได้ 256 ระดับ และจ่ายกระแสออกได้ตรงตามความต้องการ

2. ส่วนเก็บค่าข้อมูล (Dual Port RAM) เป็นส่วนเก็บค่าข้อมูลที่มาจากส่วนส่งข้อมูล ภายในหน่วยความจำจะแบ่งออกเป็น 2 ส่วน ส่วนหนึ่งทำหน้าที่เก็บค่า และอีกส่วนสำหรับสร้างสัญญาณ PWM โดยทั้ง 2 ส่วนจะสลับหน้าที่กันทุกครั้งที่มีสัญญาณ DAST เข้ามา การเก็บข้อมูลนี้จะแยกข้อมูลแต่ละสีออกจากกัน โดยแต่ละส่วนมีขนาด 512 ตำแหน่ง ในส่วนที่ 1 จะมีตำแหน่ง 0x0000000000 ถึง 0x0111111111 และส่วนที่ 2 จะอยู่ในช่วง 0x1000000000 ถึง 0x1111111111 ทั้งนี้เพื่อลดพื้นที่การใช้งานของ FPGA

3. ส่วนเปรียบเทียบค่า (Comparator) ใช้สร้างสัญญาณเพื่อไปขับชิปสร้างกระแสคงที่โดยตรง โดยรับค่าที่เก็บอยู่ในหน่วยความจำมาเปรียบเทียบกับค่าอ้างอิง (counter) ที่เปลี่ยนแปลงไปเรื่อยๆ เพื่อสร้างสัญญาณ PWM ขึ้นมา ในระบบที่ออกแบบจะมีส่วนนี้ 16 ตัว โดยรับผิดชอบแถวจุดภาพที่แสดงแยกจากกัน

4. ส่วนสร้างสัญญาณ WA (Timing) เป็นส่วนกำหนดเวลาการเปิดปิดการเขียนข้อมูลเข้าไปใน RAM

5. ส่วนสร้างสัญญาณกำหนดตำแหน่ง A (Addr_1) เป็นส่วนสร้างค่าตำแหน่ง A เพื่อใช้ในการเก็บข้อมูลเข้าสู่ RAM

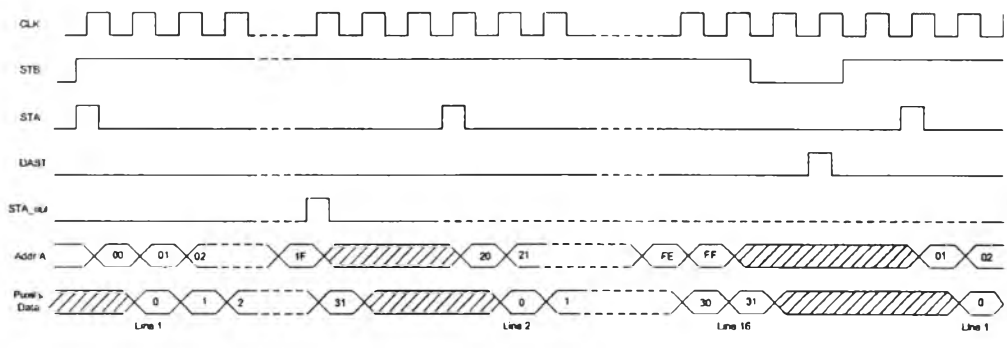
6. ส่วนสร้างสัญญาณกำหนดตำแหน่ง B (Addr_2) เป็นส่วนสร้างค่าตำแหน่ง B ขึ้นมาเพื่อให้ระบบส่งค่าออกจากหน่วยความจำไปยังส่วนเทียบค่า

7. ส่วนเลือกการทำงาน (Selector) ในส่วนนี้จะเป็นตัวกำหนดการทำงานของส่วนเทียบค่า เนื่องจากข้อมูลจากส่วนเก็บค่าข้อมูล (Dual Port RAM) จะเข้าไปยังส่วนเทียบค่าทุกตัวพร้อมกัน จึงต้องมีส่วนกำหนดว่าค่าที่ออกจากหน่วยความจำในช่วงเวลานั้นเป็นของส่วนเทียบค่าตัวใด นอกจากนี้ในส่วนนี้ยังเป็นส่วนสร้างค่าอ้างอิง (counter) อีกด้วย

8. ส่วนสร้างสัญญาณ CLR (Clear) ใช้สร้างสัญญาณ CLR จากสัญญาณ DAST มีขนาดกว้าง 1 สัญญาณนาฬิกาของ CLK_inner ใช้เป็นสัญญาณล้างค่ากับส่วนที่โปรแกรมบางส่วนที่ใช้สัญญาณนาฬิกา CLK_inner

การทำงานของส่วนรับค่าจากส่วนรับข้อมูล

ในระบบนี้ได้ออกแบบให้ใช้กับแผงขนาด 32 x 16 จุดภาพ ลำดับของสัญญาณที่ใช้ในการทำงานแสดงในรูปที่ 4.12 ค่าที่รับเข้ามาจะเก็บในหน่วยความจำ ซึ่งค่าตำแหน่งของหน่วยความจำจะถูกเปลี่ยนค่าจะสัญญาณ CLK เช่นเดียวกับการเก็บค่า แต่ใช้ผลของเวลาประมวลผลใน FPGAทำให้สามารถเก็บค่าลงในหน่วยความจำได้

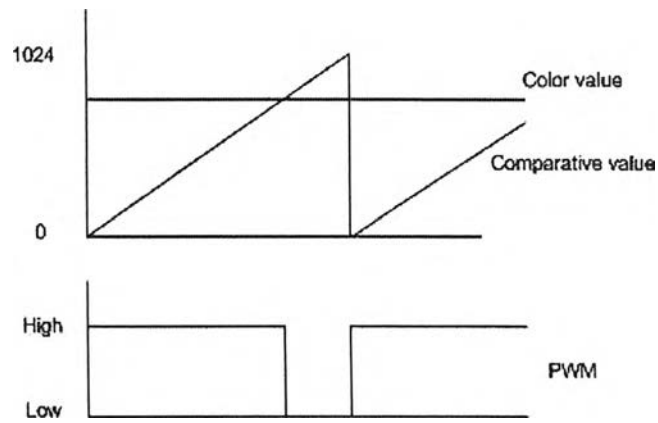


รูปที่ 4.12 ลำดับการส่งสัญญาณขาเข้าของ FPGA

จาก Timing diagram ในรูปที่ 4.12 การทำงานจะเริ่มจากสัญญาณ STB เป็น High เพื่อให้วงจรทำงาน (การทำงานคล้ายสัญญาณ Enable) เมื่อมีสัญญาณ CLK ขาขึ้นตรงกับสัญญาณ STA ที่เป็น High จะเริ่มต้นการเก็บข้อมูลต่อจากตำแหน่ง address เดิม เป็นจำนวนเท่ากับความยาวของจุดภาพบนแผงแสดงภาพ (ในวิทยานิพนธ์นี้ ค่าความยาวเป็น 32 จุดภาพ หรือ 0x1F) การเก็บข้อมูลจะใช้สัญญาณ CLK เป็นตัวเลื่อนค่า address ที่ระบุตำแหน่ง และเป็นสัญญาณที่ใช้ในการเก็บค่าเช่นกัน โดยจะเหลื่อมกัน 1 สัญญาณ CLK เมื่อจำนวนข้อมูลเข้ามาเท่ากับจำนวนจุดภาพในด้านยาวแล้ว วงจรจะส่งสัญญาณ STA_out เพื่อแจ้งให้แผงข้างเคียงที่เชื่อมต่ออยู่เปิดรับค่าข้อมูล โดยในการทำงานในแผงนี้จะหยุดรับข้อมูล แม้ว่ามีสัญญาณ CLK เข้ามาในช่วงนี้ก็จะไม่มีการเก็บค่าลงหน่วยความจำ รอจนกว่าจะมีสัญญาณ STA เป็น High เข้ามาใหม่ เพื่อแจ้งว่ามีข้อมูลแถวถัดไปเข้ามา จึงเริ่มเก็บข้อมูลต่อจากตำแหน่งเดิมอีกครั้ง จนข้อมูลจุดภาพเข้ามาครบทุกจุดภาพบนแผงแล้ว (ตำแหน่ง address จะเริ่มไปที่ 0xFF ซึ่งเป็นตำแหน่งของจุดภาพที่ 32x16) จะหยุดรับค่า เมื่อมีสัญญาณ DAST เข้ามาระบบจะเปลี่ยนหน้าที่ของหน่วยความจำในส่วนรับค่าและสวนสร้างสัญญาณด้วย PWM เพื่อนำข้อมูลภาพที่รับเสร็จแล้วไปแสดงค่า และเก็บค่าข้อมูลภาพใหม่ทับข้อมูลภาพที่ไม่ใช่แสดงผลแล้ว

การทำงานของส่วนสร้างสัญญาณด้วย PWM

การสร้างสัญญาณด้วย PWM ขนาด 10 บิต วิธีที่เลือกใช้คือการเปรียบเทียบค่าข้อมูลสีกับค่าเปรียบเทียบ ดังรูปที่ 4.13 โดยเมื่อค่าข้อมูลสีมีค่ามากกว่าค่าเปรียบเทียบจะให้ส่งค่า 1 ออกมา แต่ถ้าน้อยกว่าหรือเท่ากับจะให้ส่งค่า 0 ออกมา หลังจากนั้นจะเปลี่ยนค่าเปรียบเทียบ แล้วสร้างสัญญาณใหม่ไปเรื่อยๆ จนครบทุกค่าใน 0 – 1023 จะทำให้สามารถสร้างสัญญาณด้วย PWM ขนาด 10 บิตออกมาได้



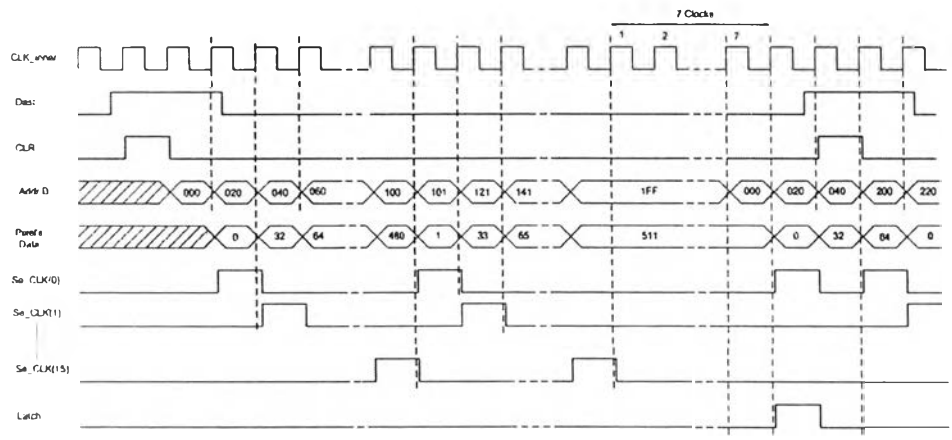
รูปที่ 4.13 การสร้างสัญญาณด้วย PWM

จากการออกแบบทางด้านการเชื่อมต่อของชิปสร้างกระแสคงที่ ที่ออกแบบให้ข้อมูลสีของไดโอดเปล่งแสงสีแดง , เขียว และน้ำเงินอยู่บนเส้นเดียวกัน ทำให้หากส่งเรียงลำดับจากจุดภาพแรกไปจุดภาพสุดท้ายจะทำให้เสียเวลามาก เพราะในแต่ละจุดภาพต้องรอการเลื่อนค่าแสดงผลของข้อมูล 3 สี ซึ่งจะทำให้ใช้ความถี่ในการทำงานอย่างต่ำมากกว่าเดิม 3 เท่า ดังนั้นจึงออกแบบให้การส่งข้อมูลสลับแถวกันไปเรื่อยๆ โดยเรียงลำดับดังรูปที่ 4.14 ซึ่งจะทำให้การเลื่อนค่าข้อมูลในแต่ละจุดภาพสามารถทำพร้อมกับการส่งข้อมูลได้

1	17	33	49	65		481	497
2	18	34	50	66		482	498
3	19	35	51	67		483	499
4	20	36	52	68		484	500
5	21	37	53	69		485	501
6	22	38	54	70		486	502
7	23	39	55	71		487	503
15	31	47	63	79		495	511
16	32	48	64	80		496	512

รูปที่ 4.14 ลำดับการส่งข้อมูลของส่วนสร้างสัญญาณด้วย PWM ที่ออกแบบ

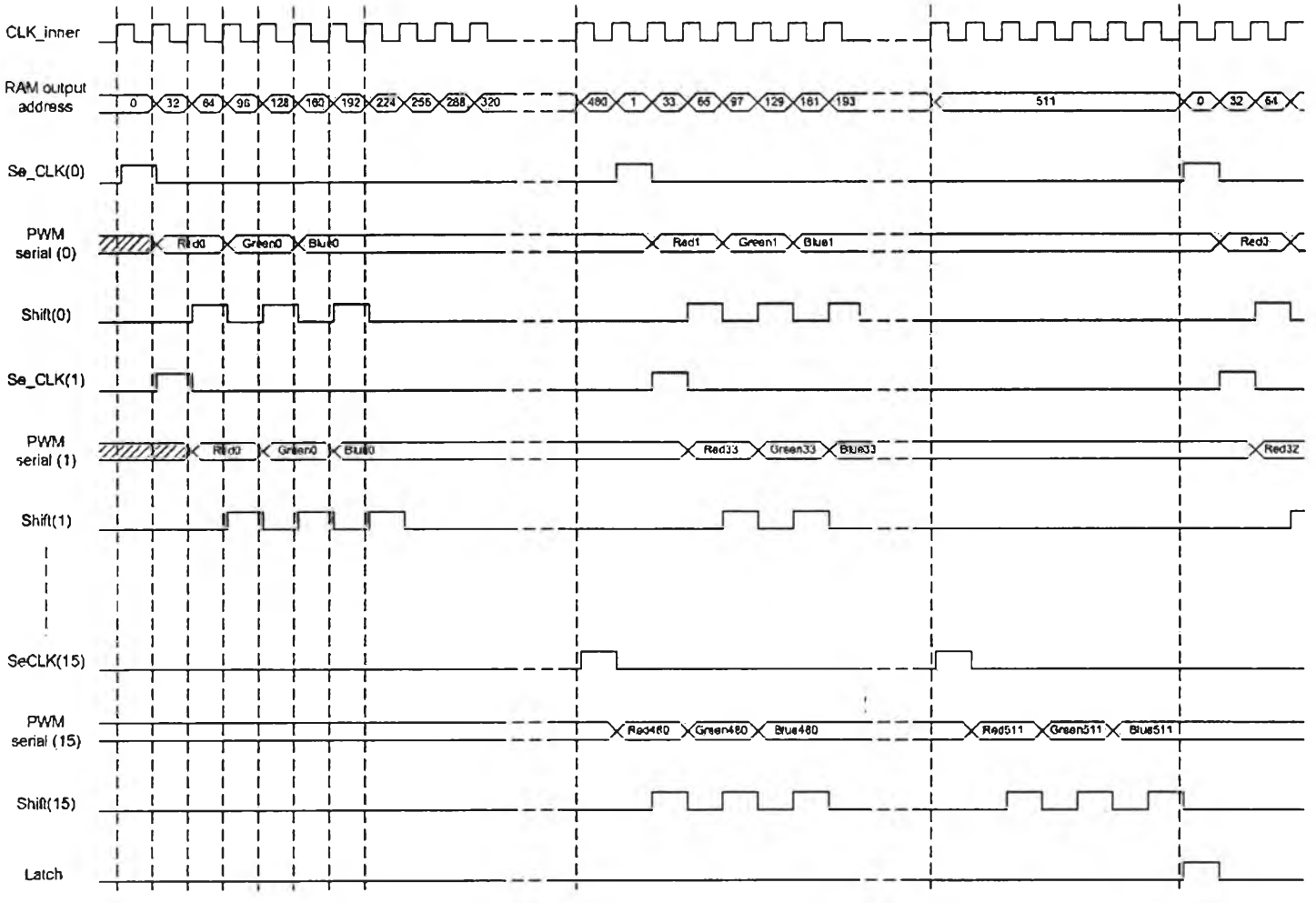
การออกแบบแบบนี้ 1 รอบการแสดงผลจะใช้เวลา 519 สัญญาณนาฬิกา (32 x 16 จุดภาพ + เวลาการเลื่อนค่าของจุดสุดท้าย 7 สัญญาณนาฬิกา) ซึ่งจะลดระยะเวลาการวนรอบไปได้มากทำให้การวนรอบของสัญญาณ PWM เพิ่มขึ้นด้วย ซึ่งหากใช้วิธีส่งเรียงลำดับจะใช้เวลารวม 1536 สัญญาณนาฬิกา (32 x 16 จุดภาพ x 3 สี)



รูปที่ 4.15 ลำดับสัญญาณในการสร้างสัญญาณ PWM

จาก Timing diagram ในรูป 4.15 การทำงานในส่วนนี้จะใช้สัญญาณ CLK_inner ที่มีความถี่ 25 MHz เป็นสัญญาณนาฬิกา เมื่อมีสัญญาณ DAST เข้ามาก จะสร้างสัญญาณ CLR ขนาด 1 ความยาว CLK_inner ขึ้นเพื่อใช้ล้างค่า หลังจากนั้นจะสร้างค่า address เพื่อเรียกข้อมูลออกจากหน่วยความจำ โดยนำข้อมูลของจุดภาพด้านซ้ายสุดของแถวบนออกมาก่อน ตามด้วยข้อมูลจุดภาพซ้ายสุดของแถวที่ 2 ไล่ลงไปเรื่อยๆจนถึงแถวสุดท้าย จึงเริ่มดึงข้อมูลจุดภาพที่ 2 จาก

ซ้ายของแถวแรกออกมา (ดังรูปที่ 4.14) โดยตำแหน่ง address และข้อมูลที่ออกมาจะเหลื่อมกัน 1 สัญญาณนาฬิกา ในระหว่างการดึงข้อมูลออกจากหน่วยความจำ จะมีการสร้างสัญญาณ se_CLK เพื่อบอกไปยังส่วนเปรียบเทียบค่า (comparator) เพื่อบอกว่าข้อมูลที่ออกมาจากหน่วยความจำนั้นเป็นของแถวจุดภาพใด (1 ใน 16 แถวจุดภาพ) ในระหว่างที่ส่วนเปรียบเทียบค่าแต่ละตัวรอข้อมูลจุดภาพถัดไปมาสร้างสัญญาณ (รอ 16 สัญญาณนาฬิกา) จะทำการจัดเรียงค่า PWM ของแต่ละสีในจุดภาพนั้น (สีแดง , สีเขียว และสีน้ำเงิน ตามลำดับ) ดังแสดงใน Timing diagram ในรูป 4.16 และส่งออกไปยังชิปสร้างกระแสคงที่ พร้อมกับสัญญาณ shift เพื่อใช้แยกค่าในแต่ละสีออกจากกัน โดยการจัดเรียงและส่งค่านี้จะใช้เวลาทั้งหมด 7 สัญญาณนาฬิกา จึงทำให้การทำงานในแต่ละแถวขนานกับไปได้ โดยไม่ต้องรอกัน เมื่อเรียกค่าออกมาครบทุกจุดภาพแล้วจะรอเวลาอีก 8 สัญญาณนาฬิกา เพื่อให้ค่าสุดท้ายเปรียบเทียบค่ากับค่าเปรียบเทียบ , จัดเรียง และส่งค่าออกไปจนเสร็จ จึงสร้างสัญญาณ LATCH ออกเพื่อให้แผงแสดงค่าข้อมูลที่รับเข้าไป หลังจากนั้นจะเริ่มวนค่าที่เรียกออกจากหน่วยความจำอีกครั้ง และทำงานวนไปเรื่อยๆจนกว่าจะมีสัญญาณ CLR เกิดขึ้นจึงเริ่มนำข้อมูลชุดใหม่มาแสดงผล



รูปที่ 4.16 สัญญาณ PWM serial และสัญญาณ shift ที่สร้างขึ้นของส่วนเทียบค่า