บทที่ 3

การออกแบบบัสระบบแบบอสมวาร

ในบทนี้กล่าวถึงการออกแบบบัสระบบแบบอสมวาร ด้วยวิธีการเข้ารหัสหนึ่งในสี่ ซึ่งใช้เป็นช่องทางในการรับส่งข้อมูลไปยังอุปกรณ์ต่างๆที่เชื่อมต่อถึงกันในระบบ รายละเอียดของการออกแบบบัสระบบแบ่งออกเป็น คุณสมบัติของบัสระบบ โครงสร้างของบัสระบบ และการทำงานของบัสระบบ โดยมีรายละเอียดดังต่อไปนี้

3.1 คุณสมบัติของบัสระบบ

บัสระบบแบบอสมวารที่ออกแบบ มีคุณสมบัติโดยสรุปดังนี้

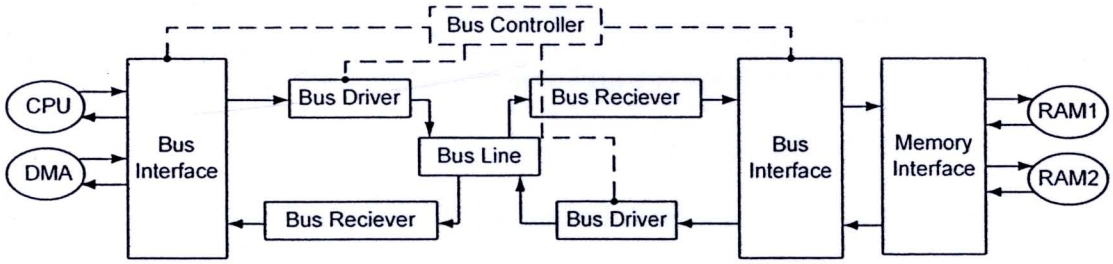
- ทำงานแบบมัลติเพล็กซ์ เพื่อลดจำนวนการใช้สายสัญญาณของบัส
- บัสมีความกว้าง 20'1of4* บิต ซึ่งเพิ่มจากบัสเวอร์ชันเดิม [4] ที่มีความกว้างเพียง16'dual-rail** บิต เพื่อรองรับการใช้งานรับส่งเลขที่อยู่ขนาด 20'1of4 บิต ซึ่งเพิ่มจากบัสเวอร์ชันเดิมที่รองรับการใช้งานรับส่งเลขที่อยู่เพียง 16'dual-rail บิต และรองรับการใช้งานรับส่งข้อมูลขนาด 16'1of4 บิต
- เชื่อมต่อกับไมโครโพรเซสเซอร์ ดีเอ็มเอ และหน่วยความจำขนาด 1Kx8bits จำนวน 2 ตัว เพื่อเพิ่มการติดต่อกับหน่วยความจำ จากบัสเวอร์ชันเดิม ที่เชื่อมต่อกับหน่วยความจำขนาด 256x8bits จำนวนเพียง 1 ตัว
- เข้ารหัสหนึ่งในสี่กับสัญญาณอาณัติแบบ 4 ขั้น แทนการเข้ารหัสรางคู่กับสัญญาณอาณัติแบบ 4 ขั้นของบัสเวอร์ชันเดิม เพื่อลดการเปลี่ยนสถานะสัญญาณของบัส
- ไม่ใช้สถานะไฮ-อิมพีแดนซ์ (High Impedance: Hi-Z) ในการทำงาน ใช้เพียงสัญญาณดิจิตอล คือ สถานะ 0 และสถานะ 1 เพื่อให้ทำงานแบบอสมวารได้

---

* '1of4 คือ จำนวนบิตของข้อมูลเข้ารหัสหนึ่งในสี่

** 'dual-rail คือ จำนวนบิตของข้อมูลเข้ารหัสรางคู่

3.2 โครงสร้างของบัสระบบ



รูปที่ 3.1 โครงสร้างของบัสระบบ

โครงสร้างของบัสระบบที่ออกแบบแสดงดังรูปที่ 3.1 ปรับปรุงจากบัสระบบเข้ารหัสรางคู่เวอร์ชันเดิม [4] ประกอบด้วย บัส ส่วนติดต่อกับไมโครโพรเซสเซอร์และดีเอ็มเอ (Bus Interface) ส่วนติดต่อกับหน่วยความจำแบบสมวาร (Memory Interface) และตัวควบคุมบัส (Bus Controller) โดยมีรายละเอียดของแต่ละส่วนประกอบดังนี้

3.2.1 บัส

บัสแบบอสมวารเข้ารหัสหนึ่งในสี่ประกอบด้วย ตัวขับบัส (Bus Driver) สายสัญญาณบัส (Bus Line) และตัวรับบัส (Bus Receiver) โดยปรับปรุงจากโครงสร้างบัสแบบอสมวารเข้ารหัสรางคู่เวอร์ชันเดิม [4] ดังรูปที่ 3.2(ก) เป็นเวอร์ชันปรับปรุงดังรูปที่ 3.2(ข) ซึ่งได้ลดรูปโครงสร้างบัสส่วนที่ซ้ำซ้อนออก จากนั้นปรับปรุงให้บัสเข้ารหัสหนึ่งในสี่แทนการเข้ารหัสรางคู่ได้ดังรูปที่ 3.2(จ) รายละเอียดของการออกแบบและปรับปรุงบัสทั้งหมด มีดังนี้

1. ตัวขับบัส: ออกแบบโดยใช้ลอจิกสามสถานะ (Tri-state Buffer) เช่นเดียวกับบัสเวอร์ชันเดิม ซึ่งลอจิกสามสถานะมีลักษณะการทำงานเหมือนสวิทช์ โดยเมื่อเปิดสวิทช์ (Enable=1) ลอจิกสามสถานะจะยอมให้ค่าลอจิกของอินพุทผ่านออกไปยังเอาท์พุท กล่าวคือค่าลอจิกของเอาท์พุทจะเท่ากับอินพุท ซึ่งมีความหมายว่าบัสถูกใช้งาน แต่เมื่อปิดสวิทช์ (Enable=0) ลอจิกสามสถานะจะตัดการเชื่อมต่อระหว่างอินพุทและเอาท์พุท กล่าวคือค่าลอจิกของเอาท์พุทจะมีค่าเท่ากับไฮ-อิมพีแดนซ์ ซึ่งมีความหมายว่าบัสว่าง ลอจิกสามสถานะสามารถต่อร่วมกันบนสายสัญญาณบัสเส้นเดียวได้ ซึ่งคุณสมบัตินี้ตรงกับความสามารถของบัสที่ต้องรองรับการใช้งาน

จากหลายอุปกรณ์ร่วมกันบนบัสเดียว ลอจิกสามสถานะจึงถูกใช้เป็นสวิทช์เปิด-ปิดให้สัญญาณของแต่ละอุปกรณ์ผ่านเข้าสู่บัสในการออกแบบทั่วไป

บัสเวอร์ชันเดิมยังประกอบด้วยส่วนตรวจสอบความพร้อมของสัญญาณก่อนเริ่มส่งข้อมูลเข้าบัส ซึ่งสามารถตัดออกได้ เนื่องจากบัสต้องใช้งานควบคู่กับส่วนพักข้อมูลต้นทางของบัส (Source Latch) ซึ่งทำหน้าที่เก็บข้อมูลก่อนเริ่มส่งเข้าสู่บัส ส่วนพักข้อมูลดังกล่าวประกอบด้วยส่วนตรวจสอบความพร้อมของสัญญาณเช่นกัน ดังนั้นในบัสเวอร์ชันปรับปรุงจึงตัดส่วนตรวจสอบความพร้อมของสัญญาณก่อนเริ่มส่งข้อมูลเข้าบัสออก เพื่อลดความซ้ำซ้อนของโครงสร้าง

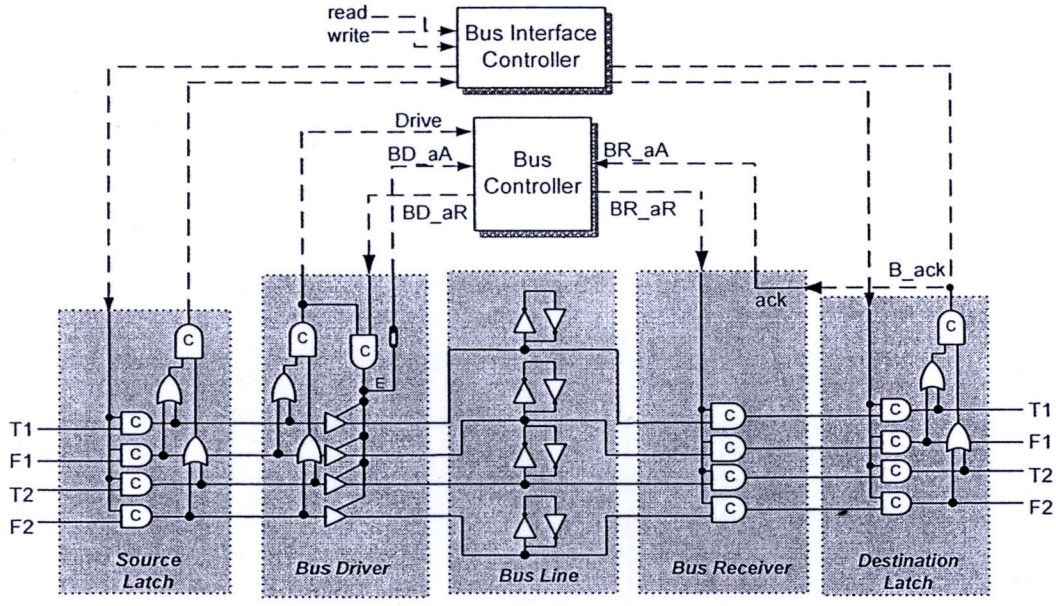
2. สายสัญญาณบัส: การทำงานของลอจิกสามสถานะจากตัวขับบัสทำให้เกิดค่าไฮ-อิมพีแดนซ์เมื่อบัสว่าง แต่ในโพรโทคอลแบบอสมวารใช้ลอจิก 0 และลอจิก 1 ในการติดต่อระหว่างวงจรอสมวารเท่านั้น บัสเวอร์ชันเดิมจึงใช้ Weak Inverter ซึ่งประกอบด้วยเกตผกผัน (Inverter) จำนวนสองตัวต่อกลับหัวกัน บนสายสัญญาณบัสแต่ละเส้น เพื่อเปลี่ยนสถานะไฮ-อิมพีแดนซ์เป็นลอจิก 0 แต่การออกแบบโดยใช้ Weak Inverter ดังกล่าวเป็นการออกแบบในระดับทรานซิสเตอร์ (Transistor Level) ซึ่งโปรแกรม Xilinx ที่งานวิจัยนี้ใช้งาน ออกแบบได้ในระดับเกต (Gate Level) เท่านั้น Weak Inverter จึงถูกตัดออกและแทนที่ด้วยแอนด์เกต (And-gate) ซึ่งเป็นการออกแบบระดับเกตแทน

เมื่อต่อขาที่หนึ่งของแอนด์เกตไว้บนสายสัญญาณบัส และต่อขาที่สองของแอนด์เกตร่วมกับขาสวิทช์ (Enable) ของลอจิกสามสถานะ ได้ดังรูปที่ 3.2(ค) ซึ่งหากพิจารณาจากบัสทิศทางเดียวซึ่งประกอบด้วยลอจิกสามสถานะตัวเดียว เมื่อขาสวิทช์ของลอจิกสามสถานะปิด (Enable=0) ซึ่งเป็นปัญหาคือทำให้เกิดค่าไฮ-อิมพีแดนซ์ที่เอาท์พุทของลอจิกสามสถานะ ขาสองของแอนเกตที่ต่อร่วมกับขาสวิทช์ของลอจิกสามสถานะ จะมีค่าลอจิกเท่ากับ 0 (เท่ากับEnable) เช่นกัน ดังนั้นแอนเกตจะรับค่าไฮ-อิมพีแดนซ์ที่เข้ามาทางขาหนึ่ง และเปลี่ยนเป็นค่าลอจิก 0 ออกไปทางขาเอาท์พุทของแอนเกต

ในทางปฏิบัติ บัสต้องรับส่งข้อมูลได้สองทิศทาง ซึ่งโครงสร้างบัสสองทิศทางจะประกอบด้วยลอจิกสามสถานะสองตัวต่อร่วมกัน เพื่อทำหน้าที่เป็นตัวขับบัสโดยให้สัญญาณจากสองทิศทาง คือทิศทางซ้ายและทิศทางขวา ผ่านเข้าสู่บัส ดังนั้นต้องเพิ่มแอนเกตเป็นสองตัวตามจำนวนของลอจิกสามสถานะ เพื่อแก้ปัญหาสถานะไฮ-อิมพีแดนซ์เช่นกัน โดยเป็นดังรูปที่ 3.2(ง) และรูปที่ 3.2(จ)

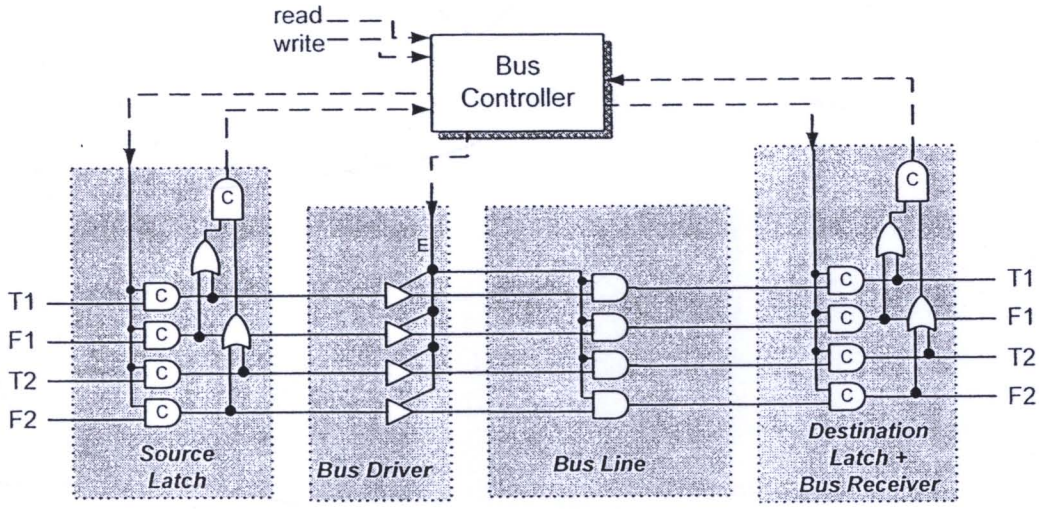
นอกจากนี้อาจต่อตัวต้านทานพูลดาวน์ (Pull-down Resistor) แทนการต่อแอนด์เกตบนสายสัญญาณบัสได้ เนื่องจากสามารถเปลี่ยนสถานะไฮ-อิมพีแดนซ์เป็นลอจิก 0 ได้เช่นกัน อีกทั้งมีขนาดเล็กกว่าแอนด์เกต แต่การออกแบบบัสซึ่งประกอบด้วยตัวต้านทานพูลดาวน์ดังกล่าวจะไม่สามารถทดลองบนเอฟพีจีเอ ที่ประกอบด้วยตัวต้านทานพูลอัพ (Pull-up Resistor) และพูลดาวน์เป็นโครงสร้างหลักภายในวงจร I/O Block ของเอฟพีจีเออยู่แล้วได้ ดังนั้นการออกแบบบัสระบบและทดลองผ่านเอฟพีจีเอ Xilinx SPARTAN 3E ในงานวิจัยนี้ จึงเลือกใช้แอนด์เกตต่อบนสายสัญญาณบัส

3. ตัวรับบัส: บัสเวอร์ชันเดิมใช้อุปกรณ์ชนิดซีเป็นตัวรับบัส โดยอุปกรณ์ชนิดซีจะเก็บค่าที่ส่งผ่านบัสไว้เพื่อส่งต่อไปยังปลายทางของบัส ซึ่งสามารถตัดอุปกรณ์ชนิดซีนี้ออกได้ เนื่องจากบัสต้องใช้งานควบคู่กับส่วนพักข้อมูลปลายทางของบัส (Destination Latch) ซึ่งทำหน้าที่เก็บข้อมูลที่บัสส่งเข้ามาเพื่อส่งต่อไปยังปลายทางเช่นกัน ส่วนพักข้อมูลดังกล่าวประกอบด้วยอุปกรณ์ชนิดซีเช่นกัน ดังนั้นในบัสเวอร์ชันปรับปรุงจึงตัดอุปกรณ์ชนิดซีในตัวรับบัสออก เพื่อลดความซ้ำซ้อนของโครงสร้าง

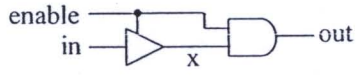


(ก) บัสเข้ารหัสรางคู่ทิศทางเดียวเวอร์ชันเดิมขนาด 4 บิต และส่วนพักข้อมูล

รูปที่ 3.2 การปรับปรุงโครงสร้างบัส

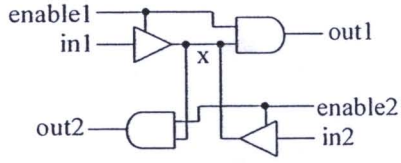


(ข) บัสเข้ารหัสรางคู่ทิศทางเดียวเวอร์ชันปรับปรุงขนาด 4 บิต และส่วนพักข้อมูล



Input		Inout	Output
enable	*in*	*x*	*out*
0	0	hi-Z	0
0	1	hi-Z	0
1	0	0	0
1	1	1	1

(ค) บัสทิศทางเดียวเวอร์ชันปรับปรุง และตารางค่าความจริง

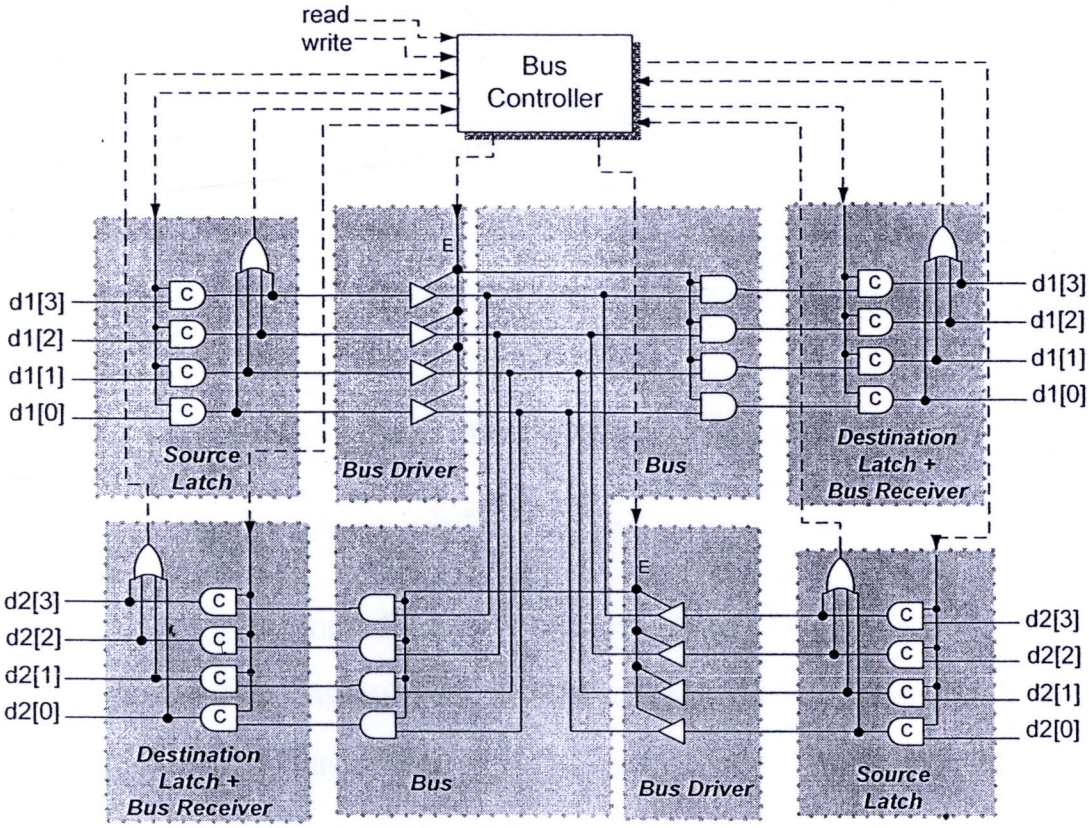


Input				Inout	Output	
enable1	*in1*	*enable2*	*in2*	*x*	*out1*	*out2*
0	0	0	0	hi-Z	0	0
0	1	0	1	hi-Z	0	0
1	0′	0	0″	0′	0′	0
1	1′	0	1″	1′	1′	0
0	0′	1	0″	0″	0	0″
0	1′	1	1″	1″	0	1″

หมายเหตุ: (′) หมายถึงค่าลอจิกจาก in1, (″) หมายถึงค่าลอจิกจาก in2

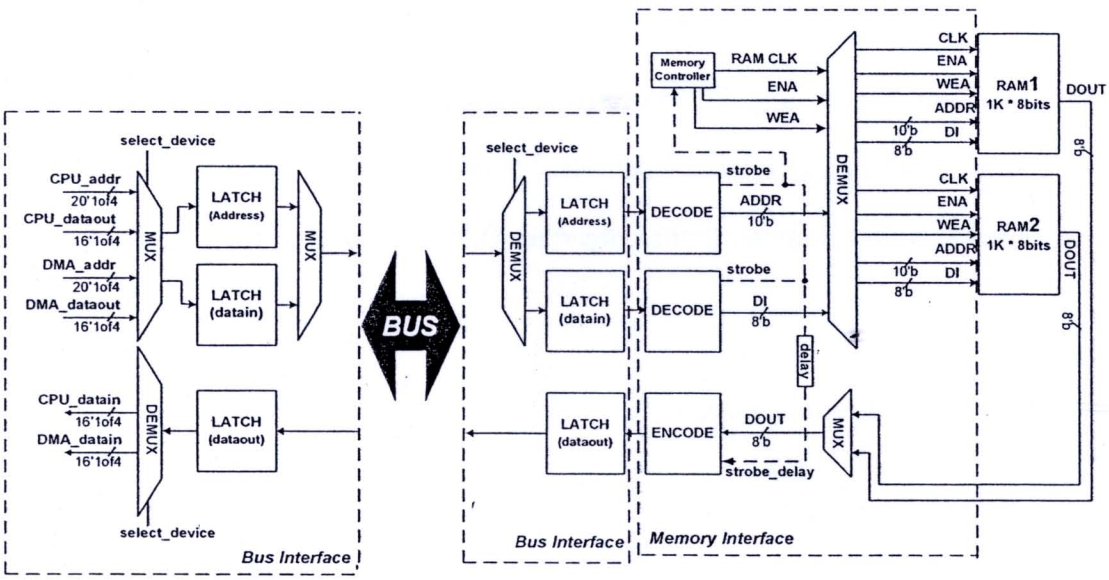
(ง) บัสสองทิศทางเวอร์ชันปรับปรุง และตารางค่าความจริง

รูปที่ 3.2 การปรับปรุงโครงสร้างบัส (ต่อ)



(จ) บัสเข้ารหัสหนึ่งในสี่สองทิศทางขนาด 4 บิต และส่วนพักข้อมูล

รูปที่ 3.2 การปรับปรุงโครงสร้างบัส (ต่อ)



รูปที่ 3.3 ส่วนติดต่อของบัส

3.2.2 ส่วนติดต่อของบัส

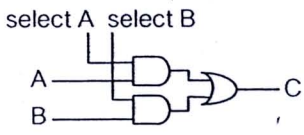
ส่วนติดต่อของบัส ใช้ติดต่อระหว่างบัสกับอุปกรณ์ที่ใช้บัส คือ ไมโครโพรเซสเซอร์ ดีเอ็มเอ และหน่วยความจำ ดังรูปที่ 3.3 ฝั่งซ้ายของบัสคือส่วนติดต่อของบัสกับไมโครโพรเซสเซอร์และดีเอ็มเอ ฝั่งขวาของบัสคือส่วนติดต่อของบัสกับหน่วยความจำแบบสมวาร มีรายละเอียดการออกแบบดังนี้

- **ส่วนติดต่อกับไมโครโพรเซสเซอร์และดีเอ็มเอ**

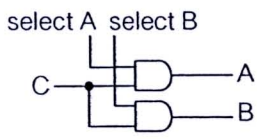
ส่วนติดต่อของบัสกับไมโครโพรเซสเซอร์และดีเอ็มเอ ประกอบด้วย

1. ตัวเลือกรวมสัญญาณ หรืออุปกรณ์สหสัญญาณ (Multiplexer) : ทำหน้าที่เลือกสัญญาณจากไมโครโพรเซสเซอร์ หรือดีเอ็มเอเข้าสู่บัส ตัวเลือกรวมสัญญาณมีโครงสร้างดังรูปที่ 3.4(ก) เมื่อมีสัญญาณเลือกข้อมูลอินพุท A (select A = 1) อินพุท A จะถูกเลือกไปเป็นเอาท์พุทของวงจร กล่าวคือ เอาท์พุท C จะมีค่าเท่ากับอินพุท A ในทางกลับกันเมื่อมีสัญญาณเลือกข้อมูลอินพุท B (select B = 1) อินพุท B จะถูกเลือกไปเป็นเอาท์พุทของวงจร กล่าวคือ เอาท์พุท C จะมีค่าเท่ากับอินพุท B

2. ตัวเลือกแยกสัญญาณ (Demultiplexer) : ทำหน้าที่เลือกสัญญาณออกจากบัสไปยังไมโครโพรเซสเซอร์ หรือดีเอ็มเอ ตัวเลือกแยกสัญญาณมีโครงสร้างดังรูปที่ 3.4(ข) เมื่อมีสัญญาณเลือกข้อมูลเอาท์พุท A (select A = 1) อินพุท C จะถูกเลือกไปเป็นเอาท์พุทของขา A กล่าวคือเอาท์พุท A จะมีค่าเท่ากับอินพุท C ในทางกลับกันเมื่อมีสัญญาณเลือกข้อมูลเอาท์พุท B (select B = 1) อินพุท C จะถูกเลือกไปเป็นเอาท์พุทของขา B กล่าวคือเอาท์พุท B จะมีค่าเท่ากับอินพุท C



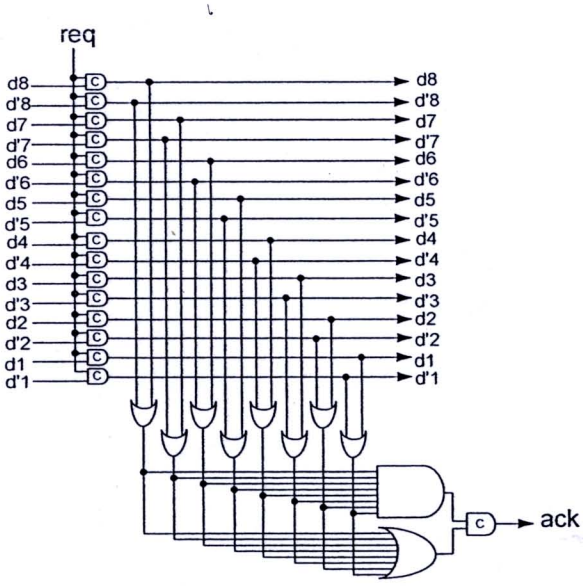
(ก) ตัวเลือกรวมสัญญาณ



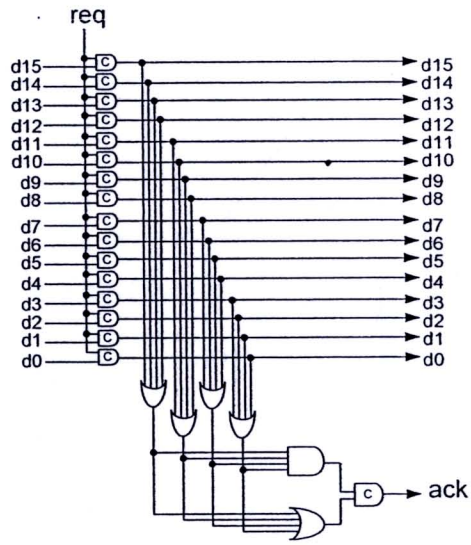
(ข) ตัวเลือกแยกสัญญาณ

รูปที่ 3.4 ตัวรวมสัญญาณและตัวแยกสัญญาณ

3. ส่วนพักข้อมูล: ทำหน้าที่รับข้อมูลและเก็บข้อมูลไว้เพื่อเตรียมส่งต่อไปยังส่วนถัดไป โดยใช้สัญญาณร้องขอและสัญญาณตอบรับในวงจรอสมวารเป็นตัวควบคุมจังหวะการรับส่งข้อมูลในส่วนพักข้อมูล ส่วนพักข้อมูลมีโครงสร้างดังรูปที่ 3.5 [16] หากมีข้อมูลอินพุทเข้าและสัญญาณร้องขอมีค่าเป็น 1 ส่วนพักข้อมูลจะจำข้อมูล เมื่อจำข้อมูลครบแล้วสัญญาณตอบรับจะมีค่าเป็น 1 แต่หากไม่มีข้อมูลเข้าหรือข้อมูลว่างและสัญญาณร้องขอมีค่าเป็น 0 ส่วนพักข้อมูลจะลบข้อมูล เมื่อลบข้อมูลทั้งหมดแล้วสัญญาณตอบรับจะมีค่าเป็น 0 โครงสร้างของส่วนพักข้อมูลสำหรับข้อมูลเข้ารหัสรางคู่แสดงดังรูปที่ 3.5(ก) และสำหรับข้อมูลเข้ารหัสหนึ่งในสี่แสดงดังรูปที่ 3.5(ข) ซึ่งจะสังเกตได้ว่าส่วนพักข้อมูลสำหรับข้อมูลเข้ารหัสรางคู่มีขนาดใหญ่กว่าส่วนพักข้อมูลสำหรับข้อมูลเข้ารหัสหนึ่งในสี่



(ก) ส่วนพักข้อมูลเข้ารหัสรางคู่ขนาด 16 บิต



(ข) ส่วนพักข้อมูลเข้ารหัสหนึ่งในสี่ ขนาด 16 บิต

รูปที่ 3.5 ส่วนพักข้อมูล

- **ส่วนติดต่อกับหน่วยความจำแบบสมวาร**

ส่วนติดต่อของบัสกับหน่วยความจำแบบสมวาร แบ่งออกเป็น ส่วนเลือกสัญญาณซึ่งประกอบด้วย ตัวเลือกรวมสัญญาณ ตัวเลือกแยกสัญญาณ และส่วนพักข้อมูล เช่นเดียวกับส่วนติดต่อของบัสกับไมโครโพรเซสเซอร์และดีเอ็มเอดังเช่นอธิบายข้างต้น และ

เช่นเดียวกับส่วนติดต่อของบัสกับไมโครโพรเซสเซอร์และดีเอ็มเอดังเช่นอธิบายข้างต้น และนอกจากนี้ยังประกอบด้วยส่วนสำหรับติดติดต่อกับหน่วยความจำโดยเฉพาะ คือ วงจรเข้ารหัสและถอดรหัสข้อมูล วงจรสร้างสัญญาณควบคุมหน่วยความจำแบบสมวาร และวงจรหน่วงเวลา โดยมีรายละเอียดการออกแบบในส่วนดังกล่าวรวมทั้งตัวหน่วยความจำดังนี้

1. หน่วยความจำ

การออกแบบส่วนติดต่อของบัสกับหน่วยความจำแบบสมวาร ต้องพิจารณาคุณสมบัติของหน่วยความจำที่บัสจะติดต่อด้วย เพื่อสร้างส่วนติดต่อที่สอดคล้องกับการทำงานของหน่วยความจำ โดยหน่วยความจำแบบอสมวารที่ใช้กับบัสระบบนี้สร้างจากโปรแกรมย่อย Core Generator บนโปรแกรม Xlinx ISE 11 มีโครงสร้างดังรูปที่ 3.6 [13] และมีคุณสมบัติเช่นเดียวกับหน่วยความจำที่ใช้กับไมโครโพรเซสเซอร์เข้ารหัสรางคู่ [17] ดังนี้

<u>คุณสมบัติของหน่วยความจำ</u>

i) หน่วยความจำเป็นแบบสมวาร เก็บข้อมูลรหัสฐานสอง (Binary Code) สูงสุดได้ 1K*8bits กล่าวคือมี 2^{10} = 1024 ช่อง (ความกว้างเลขที่อยู่ขนาด 10'b* บิต) แต่ละช่องเก็บข้อมูลได้ช่องละ 8'b บิต (ความกว้างข้อมูลขนาด 8'b บิต)

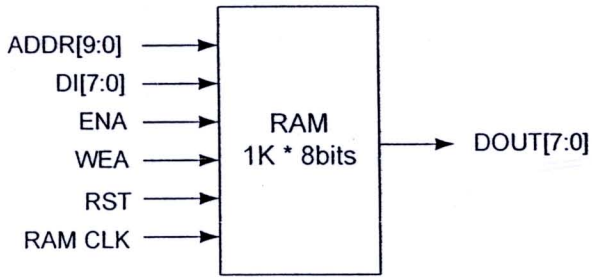
ii) ประกอบด้วยพอร์ทดังรูปที่ 3.6(ก) พอร์ทอินพุทคือ พอร์ท ADDR (Address) ขนาด 10'b บิต พอร์ท DI (Data Input) ขนาด 8'b บิต พอร์ท ENA (Enable) พอร์ท WEA (Write Enable) พอร์ท RST (Reset) พอร์ท CLK (Clock) พอร์ทเอาท์พุทคือ พอร์ท DOUT (Data Output) ขนาด 8'b บิต

iii) ทำงานเมื่อมีสัญญาณนาฬิกา (RAM CLK) ขอบขาขึ้นเข้ามา โดยรองรับการทำงานร่วมกับสัญญาณนาฬิกาความถี่สูงสุดได้ไม่เกิน 200 เมกกะเฮิร์ท (1 Clock Cycle Time ≥ 5 ns)

---

* 'b คือ จำนวนบิตของข้อมูลเข้ารหัสฐานสอง

iv) มีโหมดการทำงานเป็นแบบ WRITE FIRST Mode หรือ Transparent Mode ซึ่งมีลักษณะการทำงานคือ ขาเอาท์พุท DOUT ของหน่วยความจำ จะแสดงค่าข้อมูลปัจจุบันที่เก็บไว้ใน ตำแหน่งหน่วยความจำตามที่ขา ADDR ระบุไว้เสมอ และหน่วยความจำจะจำค่าข้อมูลใหม่จะจำค่าข้อมูลใหม่จากขา DI ต่อเมื่อสัญญาณอนุญาตให้เขียนข้อมูลคือ WEA มีค่าเท่ากับ 1 เท่านั้น โดยลักษณะการทำงานที่แสดงข้อมูลปัจจุบันเสมอนี้ ทำให้การตรวจสอบความถูกต้องของค่าภายในหน่วยความจำทำได้ง่าย



รูปที่ 3.6 หน่วยความจำ

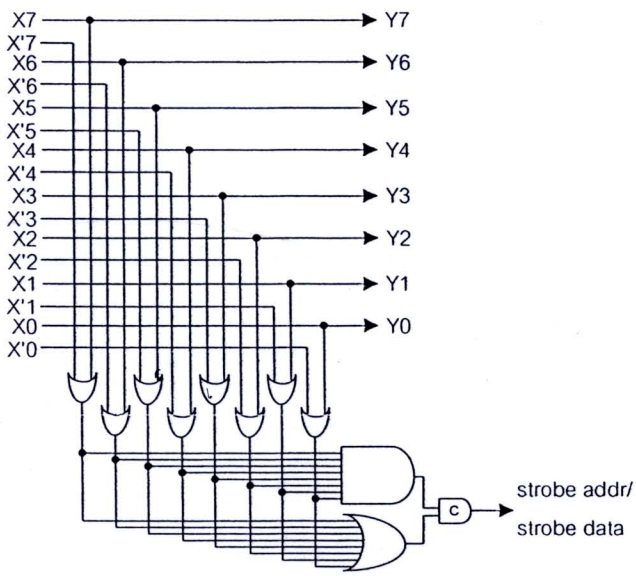
2. วงจรเข้ารหัสและถอดรหัสข้อมูล

การรับส่งข้อมูลจากบัสเข้ารหัสรางคู่เวอร์ชันเดิม ไปที่หน่วยความจำซึ่งเก็บข้อมูลในรูปแบบรหัสฐานสอง จำเป็นต้องมีวงจรถอดรหัส (Decode Circuit) ข้อมูลรางคู่จากบัส เป็นรหัสฐานสองเพื่อใช้บนหน่วยความจำ ดังรูปที่ 3.7(ก) [8] และวงจรเข้ารหัสฐานสองจากหน่วยความจำเป็นรหัสรางคู่เพื่อใช้บนบัส (Encode Circuit) ดังรูปที่ 3.7(ข) [8]

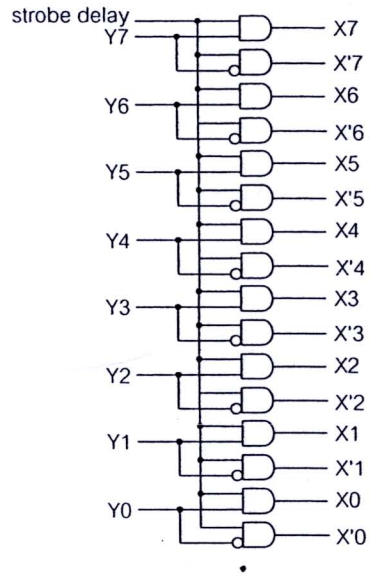
สำหรับเวอร์ชันปรับปรุง การรับส่งข้อมูลจากบัสเข้ารหัสหนึ่งในสี่เวอร์ชันปรับปรุงไปที่หน่วยความจำซึ่งเก็บข้อมูลในรูปแบบรหัสฐานสอง จำเป็นต้องมีวงจรถอดรหัสข้อมูลหนึ่งในสี่จากบัส เป็นรหัสฐานสองเพื่อใช้บนหน่วยความจำดังรูปที่ 3.7(ค) และวงจรเข้ารหัสข้อมูลฐานสองจากหน่วยความจำ เป็นรหัสหนึ่งในสี่เพื่อใช้บนบัสดังรูปที่ 3.7(ง) เช่นกัน

จากรูปที่ 3.7 ให้ค่า X เป็นรหัสรางคู่หรือรหัสหนึ่งในสี่ และให้ค่า Y เป็นรหัสฐานสอง สำหรับวงจรถอดรหัส เมื่อถอดรหัสข้อมูลสมบูรณ์แล้ว สัญญาณเอาท์พุท strobe addr หรือ strobe data จะมีค่าเป็น 1 โดยออกแบบให้สัญญาณ strobe addr เป็นสัญญาณบ่งบอกว่า

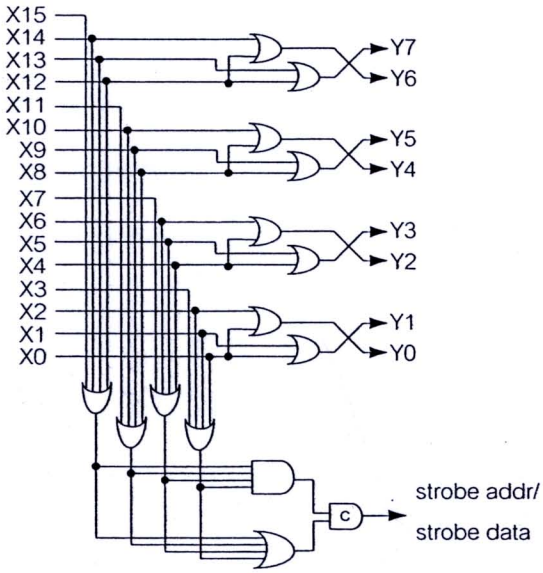
ถอดรหัสเลขที่อยู่จากบัสสมบูรณ์แล้ว ส่วนสัญญาณ strobe data เป็นสัญญาณบ่งบอกว่าถอดรหัสข้อมูลจากบัสสมบูรณ์แล้ว และสำหรับวงจรเข้ารหัส เมื่อข้อมูลมีความสมบูรณ์พร้อมที่จะเข้ารหัสสู่บัส สัญญาณอินพุท strobe delay จะมีค่าเป็น 1



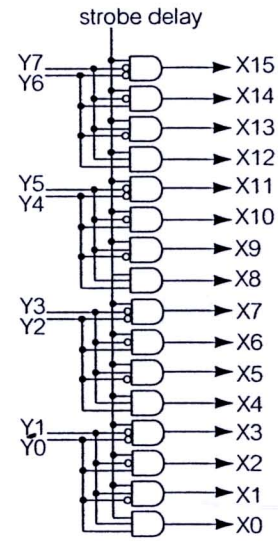
(ก) วงจรถอดรหัสรางคู่ขนาด 16 บิต



(ข) วงจรเข้ารหัสรางคู่ขนาด 16 บิต



(ค) วงจรถอดรหัสหนึ่งในสี่ขนาด 16 บิต



(ง) วงจรเข้ารหัสหนึ่งในสี่ขนาด 16 บิต

รูปที่ 3.7 วงจรเข้ารหัสและวงจรถอดรหัส

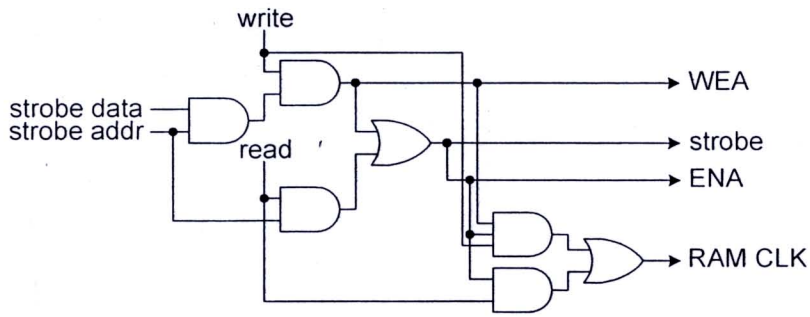
3. วงจรสร้างสัญญาณควบคุมหน่วยความจำแบบสมวาร

วงจรสร้างสัญญาณควบคุมหน่วยความจำแบบสมวาร (Memory Controller) ทำหน้าที่สร้างสัญญาณนาฬิกาและสัญญาณควบคุมคือ ENA และ WEA เข้าสู่หน่วยความจำแบบสมวาร บัสระบบแบบอสมวารจะติดต่อกับหน่วยความจำแบบสมวารได้ ต้องจำลองสัญญาณนาฬิกาจากสัญญาณภายในวงจรอสมวารขึ้นมาเพื่อใช้ติดต่อกับวงจรสมวาร โครงสร้างเป็นดังรูปที่ 3.8 โดยมีหลักการทำงานดังนี้

สำหรับคำสั่งเขียนข้อมูล เมื่อสัญญาณเขียนข้อมูล write เท่ากับ 1 และการถอดรหัสข้อมูลเลขที่อยู่รวมถึงการถอดรหัสข้อมูลมีความสมบูรณ์ คือ สัญญาณอินพุท strobe addr และ strobe data มีค่าเป็น 1 วงจรจะส่งสัญญาณเอาท์พุท RAM_CLK ENA WEA ไปที่หน่วยความจำแบบสมวารเพื่อกระตุ้นให้หน่วยความจำเริ่มต้นทำงานเขียนข้อมูล โดยสัญญาณ RAM_CLK นี้คือสัญญาณนาฬิกาที่จำลองขึ้นมาเพื่อใช้ติดต่อกับหน่วยความจำแบบสมวาร

สำหรับคำสั่งอ่านข้อมูล เมื่อสัญญาณอ่านข้อมูล read เท่ากับ 1 และการถอดรหัสข้อมูลเลขที่อยู่มีความสมบูรณ์ คือ สัญญาณอินพุท strobe addr มีค่าเป็น 1 วงจรจะส่งสัญญาณเอาท์พุท RAM_CLK ENA ไปที่หน่วยความจำแบบสมวารเพื่อกระตุ้นให้หน่วยความจำเริ่มต้นทำงานอ่านข้อมูล

นอกจากนี้วงจรจะส่งสัญญาณเอาท์พุท strobe ไปที่วงจรหน่วงเวลาเพื่อใช้เป็นสัญญาณเริ่มต้นการทำงาน ซึ่งจะอธิบายในหัวข้อถัดไป

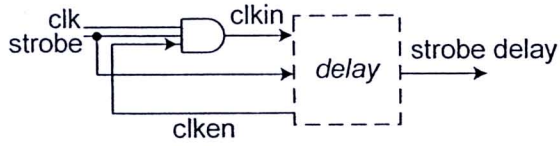


รูปที่ 3.8 วงจรสร้างสัญญาณควบคุมหน่วยความจำแบบสมวาร

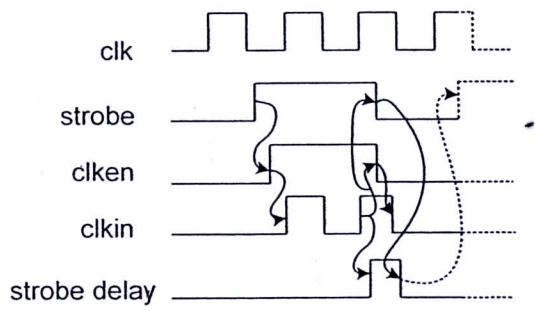
4. วงจรหน่วงเวลา

วงจรเข้ารหัสฐานสองเป็นรหัสหนึ่งในสี่ จะรับค่ารหัสฐานสองจากหน่วยความจำมาเข้ารหัส ซึ่งหากข้อมูลที่รับเข้ามาเข้ารหัสยังไม่สมบูรณ์ กล่าวคือเป็นข้อมูลที่หน่วยความจำยังเขียนหรืออ่านไม่เสร็จ จะทำให้การเข้ารหัสข้อมูลเกิดความผิดพลาด และส่งต่อค่าผิดพลาดดังกล่าวไปยังบัสและส่วนอื่นๆ วงจรหน่วงเวลาสามารถแก้ปัญหานี้ได้โดยหน่วงเวลารอให้หน่วยความจำเขียนหรืออ่านข้อมูลเสร็จสมบูรณ์ก่อน (Delay Time > Memory Access Time) [8] แล้วจึงค่อยส่งสัญญาณ strobe delay ซึ่งเป็นสัญญาณบ่งบอกว่าข้อมูลมีความพร้อมที่จะรับการเข้ารหัสแล้ว ไปที่วงจรเข้ารหัสเพื่อเริ่มต้นเข้ารหัสข้อมูล

วงจรหน่วงเวลาออกแบบโดยใช้กราฟบรรยายการเปลี่ยนสัญญาณดังรูปที่ 3.9 โดยมีโครงร่างวงจรดังรูปที่ 3.9(ก) และมีพฤติกรรมวงจรดังรูปที่ 3.9(ข) โดยเมื่อมีค่าอินพุท strobe เท่ากับ 1 ซึ่งหมายถึงข้อมูลถูกถอดรหัสเข้าหน่วยความจำโดยสมบูรณ์และหน่วยความจำได้รับการกระตุ้นให้เริ่มต้นทำงาน วงจรจะหน่วงเวลา 1 ลูกคลื่นสัญญาณนาฬิกาที่ความถี่ไม่เกิน 400 เมกกะเฮิร์ท (1 Clock Cycle Time = ½ RAM Clock Cycle Time ≥ 2.5ns) เพื่อรอเวลา

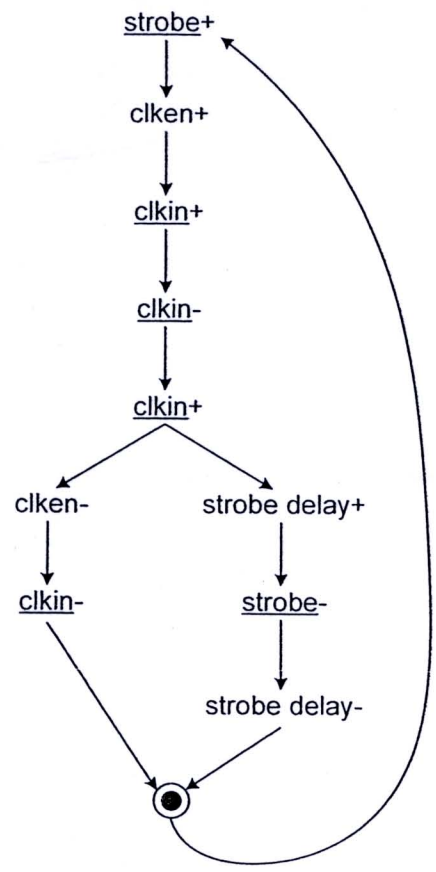


(ก) โครงร่างวงจร



(ข) แผนผังแสดงการเปลี่ยนแปลงสถานะลอจิกของอินพุทและเอาท์พุท ในแต่ละช่วงเวลา

รูปที่ 3.9 การออกแบบวงจรหน่วงเวลาโดยใช้กราฟบรรยายการเปลี่ยนสัญญาณ



(ค) กราฟบรรยายการเปลี่ยนสัญญาณ

รูปที่ 3.9 การออกแบบวงจรหน่วงเวลาโดยใช้กราฟบรรยายการเปลี่ยนสัญญาณ (ต่อ)

อย่างน้อย 5 ns ให้หน่วยความจำเขียนหรืออ่านข้อมูลจนเสร็จสมบูรณ์ แล้วจึงให้ค่าเอาท์พุท strobe delay เท่ากับ 1 เพื่อเข้ารหัสข้อมูลเข้าบัสต่อไป กราฟบรรยายการเปลี่ยนสัญญาณจากพฤติกรรมนี้แสดงดังรูปที่ 3.9 (ค)

หลังจากสร้างกราฟบรรยายการเปลี่ยนสัญญาณเสร็จสิ้นแล้ว ขั้นตอนของการสร้างวงจรต่อไปคือ สร้างกราฟแสดงสถานะ การพิจารณาค่าสัญญาณที่ได้จากกราฟแสดงสถานะลงในแผนที่คาร์นอฟ และการลดทอนสมการลอจิกโดยใช้ทฤษฎีบูลีน สามารถใช้โปรแกรม Petrify ช่วยในขั้นตอนเหล่านี้ได้ โดยโปรแกรม Petrify จะตรวจสอบและลดความซ้ำซ้อนของสถานะสัญญาณต่างๆในกราฟบรรยายการเปลี่ยนสัญญาณที่ออกแบบไว้ รวมถึงลดรูปสมการบูลีนด้วย ดาวน์โหลดโปรแกรมดังกล่าวได้ที่ http://www.lsi.upc.edu/~jordicf/petrify และดาวโหลดคู่มือ

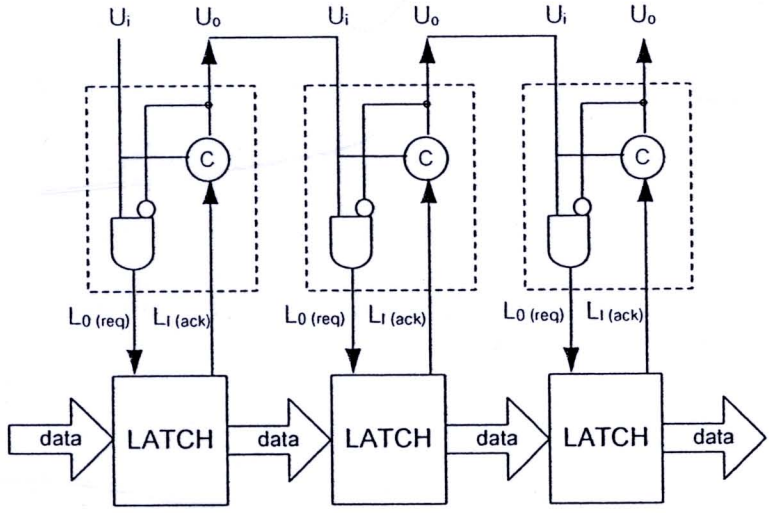
การใช้โปรแกรมได้ที่ http://www.cs.unc.edu/~montek/teaching/spring-04/petrify-tutorial.ps วิธีการใช้โปรแกรม Petrify ออกแบบวงจรหน่วงเวลา จะอธิบายในบทที่ 6 ต่อไป

3.2.3 ตัวควบคุมบัส

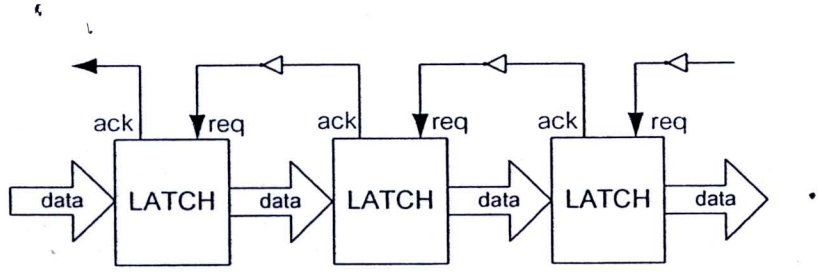
สัญญาณร้องขอและสัญญาณตอบรับในบัสระบบเวอร์ชันเดิม ถูกควบคุมด้วยอุปกรณ์ออโต้สวีป (Autosweeping Module : ASM) ดังรูปที่ 3.10(ก) [16] โดยใช้สัญญาณอาณัติแบบ 4 ขั้น สำหรับติดต่อรับส่งข้อมูลในแต่ละส่วน ซึ่งมีหลักการคือเมื่อตัวส่งข้อมูลต้องการส่งข้อมูล สัญญาณ Ui จะเปลี่ยนจาก 0 เป็น 1 เพื่อให้สัญญาณร้องขอของตัวรับข้อมูล คือ L0 เปลี่ยนจาก 0 เป็น 1 เพื่อเข้าสู่ขั้นทำงาน จากนั้นเมื่อตัวรับข้อมูลรับข้อมูลเสร็จสมบูรณ์ สัญญาณตอบรับของตัวรับข้อมูลคือ Li จะเปลี่ยนจาก 0 เป็น 1 ส่งผลให้ L0 เปลี่ยนจาก 1 เป็น 0 เพื่อเข้าสู่ขั้นว่าง จะเห็นได้ว่าการรับส่งข้อมูลจะมีการทำงานในขั้นทำงานและขั้นว่างสลับกัน ซึ่งเป็นไปตามรูปแบบของสัญญาณอาณัติแบบ 4 ขั้น

สัญญาณร้องขอและสัญญาณตอบรับในบัสระบบเวอร์ชันปรับปรุง ถูกควบคุมด้วยอุปกรณ์สายท่อ หรืออุปกรณ์ไปป์ไลน์ (Asynchronous Pipeline Module) [16] ดังรูปที่ 3.10(ข) ซึ่งใช้กับสัญญาณอาณัติแบบ 4 ขั้นเช่นกัน แต่มีขนาดวงจรเล็กกว่าอุปกรณ์ออโต้สวีป และใช้สัญญาณควบคุมน้อยกว่าอุปกรณ์ออโต้สวีปอีกด้วย อุปกรณ์ไปป์ไลน์มีหลักการคือ เมื่อตัวส่งข้อมูลต้องการส่งข้อมูล สัญญาณร้องขอของตัวรับข้อมูล คือ req จะเปลี่ยนจาก 0 เป็น 1 เพื่อเข้าสู่ขั้นทำงาน จากนั้นเมื่อตัวรับข้อมูลรับข้อมูลเสร็จสมบูรณ์ สัญญาณตอบรับของตัวรับข้อมูลคือ ack จะเปลี่ยนจาก 0 เป็น 1 ส่งผลให้ req เปลี่ยนจาก 1 เป็น 0 เพื่อเข้าสู่ขั้นว่าง

เส้นทางการรับส่งข้อมูลภายในบัส ถูกควบคุมโดยวงจรควบคุมซึ่งสร้างจากกราฟบรรยายการเปลี่ยนสัญญาณ และใช้โปรแกรม Petrify สร้างวงจร เช่นเดียวกับการออกแบบวงจรหน่วงเวลาที่ได้กล่าวไปแล้ว



(ก) Autosweeping Module : ASM



(ข) Asynchronous Pipeline Module

รูปที่ 3.10 ส่วนควบคุมการเปลี่ยนระดับสัญญาณร้องขอและสัญญาณตอบรับ

3.3 การทำงานของบัสระบบ

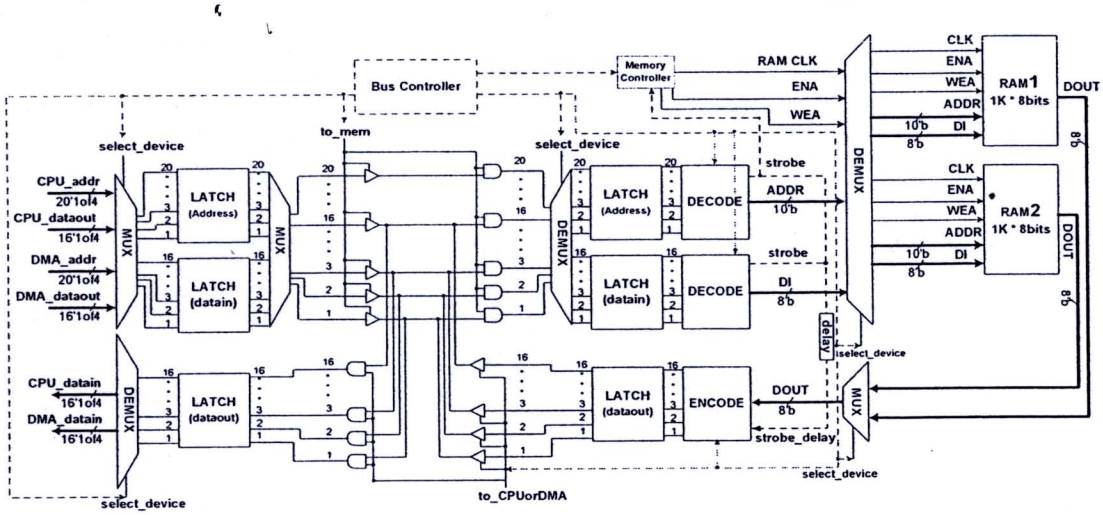
บัสระบบเข้ารหัสหนึ่งในสี่เวอร์ชันปรับปรุง มีโครงสร้างดังรูปที่ 3.11 โดยบัสระบบเชื่อมต่อไมโครโพรเซสเซอร์ ดีเอ็มเอ และหน่วยความจำเข้าด้วยกัน เพื่อรับส่งข้อมูลระหว่างอุปกรณ์ทั้ง 3 ตามคำสั่งที่ได้รับ คำสั่งที่เรียกใช้งานบัสระบบมีดังนี้

- คำสั่งโหลด (Load): ไมโครโพรเซสเซอร์จะส่งสัญญาณอ่าน และเลขที่อยู่ ผ่านบัสไปยังหน่วยความจำ เมื่ออ่านค่าจากหน่วยความจำเสร็จสิ้นจะส่งค่าที่อ่านได้ผ่านบัสกลับมายังไมโครโพรเซสเซอร์
- คำสั่งสโตร์ (Store): ไมโครโพรเซสเซอร์จะส่งสัญญาณเขียน เลขที่อยู่ และข้อมูล ผ่านบัสไปยังหน่วยความจำ เพื่อเขียนค่าลงหน่วยความจำ

- คำสั่งอิน (IN): แบ่งออกเป็นคำสั่งย่อยที่เรียกใช้งานบัสระบบคือ

- IN @K2, @K : ดีเอ็มเอจะส่งสัญญาณอ่าน และเลขที่อยู่ตำแหน่งที่ต้องการอ่าน ผ่านบัสไปยังหน่วยความจำต้นทาง เมื่ออ่านค่าจากหน่วยความจำต้นทางเสร็จสิ้นจะส่งค่าที่อ่านได้ผ่านบัสกลับมายังดีเอ็มเอ จากนั้นดีเอ็มเอจะส่งสัญญาณเขียน เลขที่อยู่ตำแหน่งที่ต้องการเขียน และข้อมูลที่อ่านได้ดังกล่าว ผ่านบัสไปยังหน่วยความจำปลายทาง เพื่อเขียนค่าลงหน่วยความจำปลายทาง

- IN I/O,@K : ดีเอ็มเอจะส่งสัญญาณอ่าน และเลขที่อยู่ตำแหน่งที่ต้องการอ่าน ผ่านบัสไปยังหน่วยความจำต้นทาง เมื่ออ่านค่าจากหน่วยความจำต้นทางเสร็จสิ้นจะส่งค่าที่อ่านได้ผ่านบัสกลับมายังดีเอ็มเอ เพื่อส่งต่อไปยังอุปกรณ์ต่อพ่วงปลายทาง



รูปที่ 3.11 บัสระบบเข้ารหัสหนึ่งในสี่ขนาด 10 บิต

- คำสั่งเอาท์ (OUT): แบ่งออกเป็นคำสั่งย่อยที่เรียกใช้งานบัสระบบคือ

- OUT @K, @K2 : ดีเอ็มเอจะส่งสัญญาณอ่าน และเลขที่อยู่ตำแหน่งที่ต้องการอ่าน ผ่านบัสไปยังหน่วยความจำต้นทาง เมื่ออ่านค่าจากหน่วยความจำต้นทางเสร็จสิ้นจะส่งค่าที่อ่านได้ผ่านบัสกลับมายังดีเอ็มเอ จากนั้นดีเอ็มเอจะส่งสัญญาณเขียน เลขที่อยู่ตำแหน่งที่ต้องการเขียน และข้อมูลที่อ่านได้ดังกล่าว ผ่านบัสไปยังหน่วยความจำปลายทาง เพื่อเขียนค่าลงหน่วยความจำปลายทาง

- OUT I/O,@K : ดีเอ็มเอจะรับค่าจากอุปกรณ์ต่อพ่วงไว้ และส่งสัญญาณเขียนเลขที่อยู่ตำแหน่งที่ต้องการเขียน และข้อมูลที่รับจากอุปกรณ์ต่อพ่วงดังกล่าว ผ่านบัสไปยังหน่วยความจำ เพื่อเขียนค่าลงหน่วยความจำ

จะเห็นได้ว่างานหลักของบัสระบบคือ อ่านข้อมูลจากหน่วยความจำและส่งต่อกับเขียนข้อมูลที่หน่วยความจำ การทำงานอ่านเขียนข้อมูลของบัสระบบดังกล่าว มีขั้นตอนการทำงานดังต่อไปนี้

<u>การอ่านข้อมูลของบัสระบบ</u>

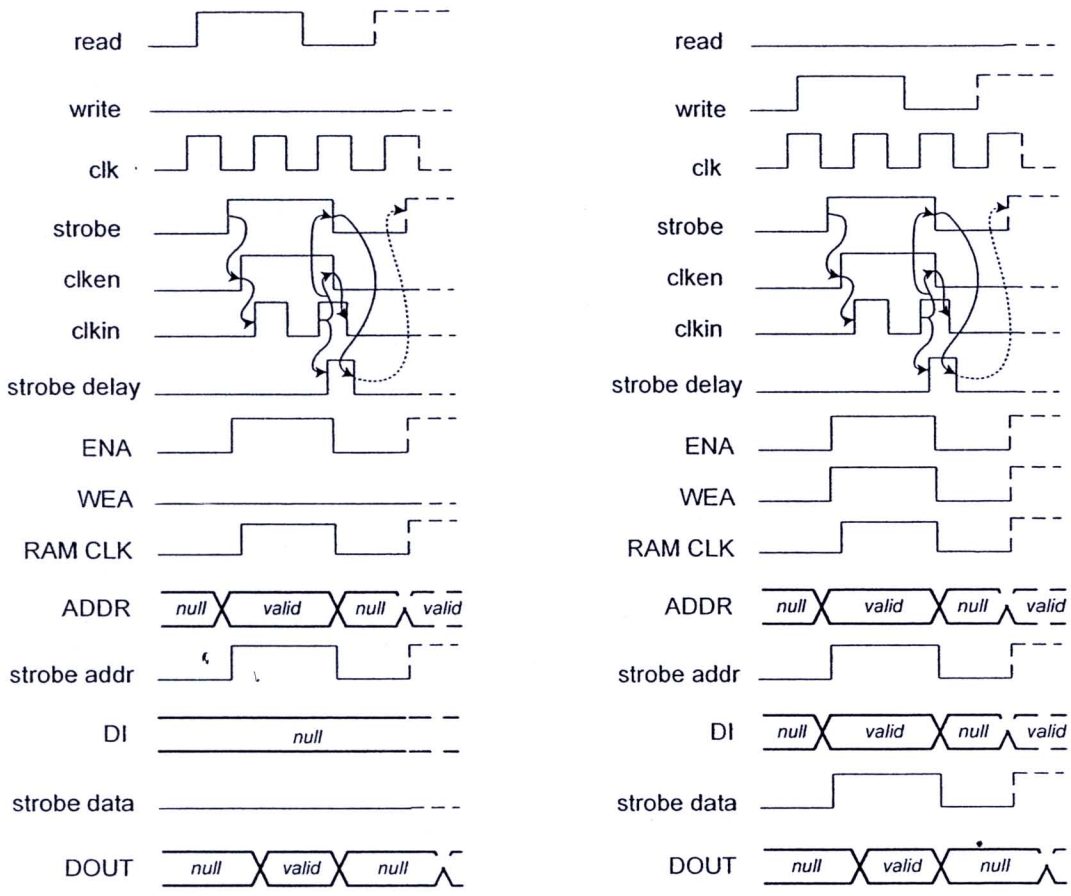
1. เมื่ออุปกรณ์ได้สิทธิ์ใช้งานบัสระบบ เส้นทางการรับส่งข้อมูลภายในบัสจะถูกกำหนดให้เชื่อมต่อระหว่างตัวส่งข้อมูลและตัวรับข้อมูลตามคำสั่งที่ได้รับ เมื่อบัสได้รับสัญญาณอ่านข้อมูล และเลขที่อยู่ ครบแล้วจะเข้าสู่ขั้นทำงาน โดยจะส่งค่าเลขที่อยู่จากตัวส่งข้อมูลผ่านบัสไปยังหน่วยความจำ

2. ส่วนติดต่อหน่วยความจำจะถอดรหัสค่าเลขที่อยู่จากบัสจนสมบูรณ์ (strobe addr = strobe = 1) แล้วจึงส่งสัญญาณควบคุมซึ่งประกอบด้วย สัญญาณเปิดการทำงานคือ ENA สัญญาณนาฬิกา คือ RAM_CLK และเลขที่อยู่ที่ถอดรหัสแล้วคือ ADDR ไปยังหน่วยความจำ

3. เมื่อหน่วงเวลารอให้อ่านข้อมูลจากหน่วยความจำเสร็จสิ้น (strobe delay = 1) ส่วนติดต่อหน่วยความจำจะเข้ารหัสข้อมูลที่อ่านได้จากหน่วยความจำคือ DOUT จนสมบูรณ์ และส่งต่อข้อมูลดังกล่าวผ่านบัสไปยังตัวรับข้อมูล จากนั้นสัญญาณควบคุมทั้งหมดจะเปลี่ยนเป็น 0 เพื่อเข้าสู่ขั้นว่าง

<u>การเขียนข้อมูลของบัสระบบ</u>

1. เมื่ออุปกรณ์ได้สิทธิ์ใช้งานบัสระบบ เส้นทางการรับส่งข้อมูลภายในบัสจะถูกกำหนดให้เชื่อมต่อระหว่างตัวส่งข้อมูลและตัวรับข้อมูลตามคำสั่งที่ได้รับ เมื่อบัสได้รับสัญญาณอ่านข้อมูล เลขที่อยู่ และข้อมูลที่ต้องการเขียนครบแล้วจะเข้าสู่ขั้นทำงาน โดยจะส่งค่าเลขที่อยู่และค่าข้อมูลจากตัวส่งข้อมูล ผ่านบัสไปยังหน่วยความจำ



(ก) อ่านข้อมูล (ข) เขียนข้อมูล

รูปที่ 3.12 แผนผังแสดงการเปลี่ยนแปลงสถานะลอจิกของส่วนติดต่อหน่วยความจำในคำสั่งอ่านและเขียนข้อมูล

2. ส่วนติดต่อหน่วยความจำจะถอดรหัสค่าเลขที่อยู่และค่าข้อมูลจากบัสจนสมบูรณ์ (strobe addr * strobe data = strobe = 1) แล้วจึงส่งสัญญาณควบคุมซึ่งประกอบด้วยสัญญาณอนุญาตเขียนข้อมูลคือ WEA สัญญาณเปิดการทำงานคือ ENA สัญญาณนาฬิกา คือ RAM_CLK เลขที่อยู่ที่ถอดรหัสแล้วคือ ADDR และข้อมูลที่ถอดรหัสแล้วคือ DI ไปยังหน่วยความจำ

3. เมื่อหน่วงเวลารอให้เขียนข้อมูลบนหน่วยความจำเสร็จสิ้น (strobe delay = 1) สัญญาณควบคุมทั้งหมดจะเปลี่ยนเป็น 0 เพื่อเข้าสู่ขั้นว่าง