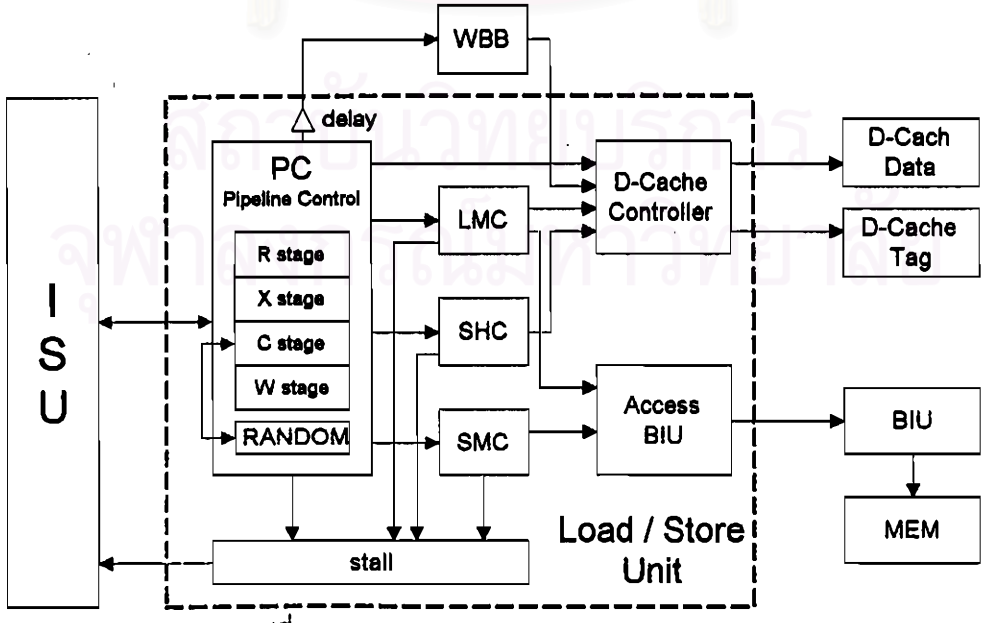
บทที่ 4
การออกแบบ Load / Store Unit

ในบทนี้จะกล่าวถึงการออกแบบ Load / Store Unit (LSU) ซึ่งใช้ในการประมวลผลคำสั่ง Load และคำสั่ง Store โดยจะกล่าวถึงการทำงานและวิธีการออกแบบของแต่ละหน่วยประมวลผลย่อยภายใน LSU, สัญญาณการเชื่อมต่อระหว่างหน่วยประมวลผลย่อย และสัญญาณที่ติดต่อกับหน่วยอื่นที่อยู่ภายนอก LSU

4.1 หน่วยประมวลผลย่อยภายใน LSU (Load / Store Unit)

ผู้วิจัยได้ออกแบบ LSU มีลักษณะดัง Block Diagram ดังรูปที่ 4.1 จะเห็นว่าการทำงานของ LSU มีการสัญญาณติดต่อกับ ISU, BIU, WBB และ D-Cache Data / Tag ซึ่งได้กล่าวไว้แล้วในบทที่ 3 หน่วยย่อยของ LSU ได้แก่

1. PC (Pipeline Control)
2. LMC (Load Miss Control) Unit
3. SHC (Store Hit Control) Unit
4. SMC (Store Miss Control) Unit
5. DCCtrl (D-Cache Controller) Unit
6. AccessBIU (Access Bus Interface Unit) Unit
7. Delay Unit
8. Stall Unit
9. RANDOM Unit



รูปที่ 4.1 block diagram ของ Load / Store Unit

โดยแต่ละหน่วยประมวลผลย่อยมีหน้าที่การทำงาน และการออกแบบดังนี้

4.1.1 PC (Pipeline Control)

PC ทำงานเป็น 4-Stage Pipeline ดังรูป 4.2 โดยการทำงานของแต่ละ Stage เป็นดังนี้

รูปที่ 4.2 pipeline stage ของหน่วย LSU

1) Read register and instruction decode stage

ใน Stage นี้ PC จะทำการพิจารณาว่าจะประมวลผลที่คำสั่งใด ระหว่างคำสั่ง InstO[31:0] และคำสั่ง InstE[31:0] ที่ได้รับมาจากหน่วย ISU โดยพิจารณาจากสัญญาณ LsuOEnpและ LsuEEnp กล่าวคือถ้า LsuOEnp = '1' หน่วย LSUจะประมวลผลตามคำสั่ง InstO และถ้า LsuEEnp = '1' จะประมวลผลตามคำสั่ง InstE แต่ถ้าสัญญาณทั้งสองเป็น '1' พร้อมกัน ก็จะเลือกประมวลผลในคำสั่ง InstO เมื่อเลือกคำสั่งที่จะประมวลผลได้แล้ว ก็จะถอดรหัสคำสั่งตามภาคผนวก ก.

2) eXecution stage

ภายใน Stage นี้มี Full 32-bit Adder เพื่อใช้ในการคำนวณค่า Virtual Effective Address โดยการนำค่า Register Source บวกกับค่า offset ซึ่งถ้าหากหน่วย LSU ไม่มีการประมวลผลคำสั่ง Load หรือคำสั่ง Store ทำให้ตัว Adder ใน Stage นี่ว่าง ดังนั้นจึงได้เสริมประสิทธิภาพการประมวลผลโดยการอนุญาตให้หน่วย LSU สามารถประมวลผลคำสั่ง ADD ได้ด้วย โดยการประมวลผลคำสั่ง ADD คำนวณได้จาก Register Source บวกกับ Register Target แล้วส่งผลการคำนวณที่ได้ออกทางสัญญาณ LSEXD[31:0] ไปให้หน่วย ISU เพื่อเขียนค่าลงรีจีสเตอร์ต่อไป

3) Cache read stage

ใน Stage นี้ PC จะอ่านข้อมูลมาจาก D-Cache Data ทางสัญญาณ DC0DataDop[63:0] และ DC1DataDop[63:0] และอ่าน D-Cache Tag ทางสัญญาณ DC0TagDop[23:0] และ DC1TagDop[23:0] จากนั้นนำค่า Tag มาตรวจสอบดูว่า ข้อมูลจากแคชที่อ่านได้ดังกล่าวเป็นตำแหน่งตรงกับที่ต้องการอ่านหรือไม่ โดยตรวจสอบจากค่าแอดเดรส Dpaddrp[31:13] ที่ได้รับจากหน่วย CP0 ว่าตรงกันกับค่า Tag ที่เก็บไว้หรือไม่ ถ้าตรงกันแสดงว่าการอ่านข้อมูลจากแคช Hit แต่ถ้าไม่ตรงกันเรียกว่า Miss

ในกรณีที่เกิด Load Hit นั้น PC จะส่งข้อมูลที่เป็นผลจากการทำงาน (ตามในภาคผนวก ก.) คำสั่ง Load ที่ได้ออกทางสัญญาณ LSCRD[31:0] เพื่อให้หน่วย ISU ทำการเขียนค่าลงรีจีสเตอร์ต่อไป

ส่วนในกรณีที่เกิด Load Miss นั้น PC จะส่งสัญญาณ LsuMissp ไปบอก ISU เพื่อให้ ISU ทราบว่าข้อมูลจากการโหลดที่ส่งไปให้นั้นไม่ถูกต้อง และส่งสัญญาณ CRVAddr[31:0], LMp, Cap, Dtyp และ Refill1p ไป

ให้หน่วย LMC (Load Miss Control) เพื่อให้หน่วย LMC จัดการรติดต่อกับหน่วยความจำเพื่อ Refill ข้อมูลลงในแคช หลังจากที่หน่วย LMC ติดต่อกับหน่วยความจำเสร็จแล้วมันจะส่งสัญญาณ LMCFixUpp พร้อมกับข้อมูลที่ได้รับจากหน่วยความจำทางสัญญาณ DData[63:0] กลับมาที่หน่วย PC และข้อมูลที่ได้จากการประมวลผลคำสั่ง Load ก็จะถูกส่งออกไปให้หน่วย ISU ทาง LSCRD[31:0] พร้อมทั้งสัญญาณ LsuFixUpp เพื่อบอกว่าปัญหา Load Miss ได้ถูกแก้ไขแล้ว

ในกรณีที่เกิด Store Hit นั้น PC จะไม่ทำการเขียนข้อมูลลง D-Cache ในทันที เพื่อป้องกันการเกิด stall pipeline อันเนื่องมาจากการขออ่านแคช (ในช่วง X stage) และเขียนแคชพร้อมกัน แต่จะส่งสัญญาณ DC0SHitp, DC1SHitp, SDirtyp, SDCBytep และ SDataHit ไปให้กับหน่วย SHC (Store Hit Control) เพื่อให้จัดการเขียนข้อมูลลงแคชในเวลาที่เหมาะสมต่อไป

ส่วนในกรณีที่เกิด Store Miss นั้น PC จะไม่เขียนข้อมูลลงในหน่วยความจำทันทีอีกเช่นกัน เพื่อป้องกันการเกิด stall pipeline อันเนื่องมาจาก มีหน่วยประมวลผลอื่นต้องการติดต่อกับหน่วยความจำในขณะนั้น แต่จะส่งสัญญาณ RFIFOWrp, FDataIn[31:0] และ FSizeIn[1:0] ไปให้หน่วย SMC (Store Miss Control) เพื่อจัดการเขียนข้อมูลลงหน่วยความจำในเวลาที่เหมาะสมต่อไป

4) Write back stage

หน่วย PC ไม่มีการประมวลผลใน Stage นี้ แต่สำหรับหน่วย LSU แล้ว Stage นี้จะเป็นการทำงานต่อจากการประมวลผล Load Miss, Store Hit และ Store Miss โดยมี ISU ทำการประมวลผล Load Hit ต่อใน Stage เดียวกันนี้

- สัญญาณที่ส่งไปให้หน่วย LMC

CRVAddr[31:0] เป็นตำแหน่งของข้อมูลที่ต้องการติดต่อกับหน่วยความจำ

LMp เป็นสัญญาณที่เกิดขึ้นเมื่อ C stage ตรวจพบว่าเกิด Load Miss ซึ่งจะเป็นค่าเดียวกับ LsuMissp

Cap เป็นสัญญาณที่บอกว่าในระบบมีแคชหรือไม่ ซึ่ง Cap = ~DUnCachep ซึ่งสัญญาณ DunCachep ได้รับมาจากหน่วย CP0

Dtyp ใช้บอกว่าแคชบรรทัดที่ถูกแทนที่นั้น ค่า WB bit = '1' หรือไม่ โดยค่า Dtyp ได้จาก DCTagDop[0] หรือ DC1TagDop[0] ขึ้นกับว่าเซ็ตไหนจะถูกแทนที่

Refill1p ใช้บอกว่าต้องการแทนที่แคชในเซ็ตไหน โดยพิจารณาตามหลักการที่ได้กล่าวไว้แล้วในบทที่ 2 เรื่อง Cache Lines Update

- สัญญาณที่ส่งไปให้หน่วย SHC

DC0Shitp สัญญาณนี้จะเกิดขึ้นเมื่อเกิด store hit ที่ค่า Tag จาก D-Cache Set 0 ตรงกับตำแหน่งของหน่วยความจำที่ต้องการติดต่อ

DC1Shitp สัญญาณนี้จะเกิดขึ้นเมื่อเกิด store hit ที่ค่า Tag จาก D-Cache Set 1 ตรงกับตำแหน่งของหน่วยความจำที่ต้องการติดต่อ

SDirtyp เป็นสัญญาณที่บอกถึงความจำเป็นว่าต้องตั้งค่าให้ dirty bit ใน D-Cache Tag เป็น '1' หรือไม่ โดยจะไปตั้งค่าก็ต่อเมื่อเกิด store hit ขึ้นและค่า dirty bit = '0' และโหมดการเขียนเป็น write back

SDCBytep[3:0] เป็นสัญญาณที่ระบุความต้องการเขียนข้อมูลลงแคชในไบต์ใด

SDataHit[31:0] เป็นข้อมูลที่ต้องการเขียนลงแคช

- สัญญาณที่ส่งไปให้หน่วย SMC

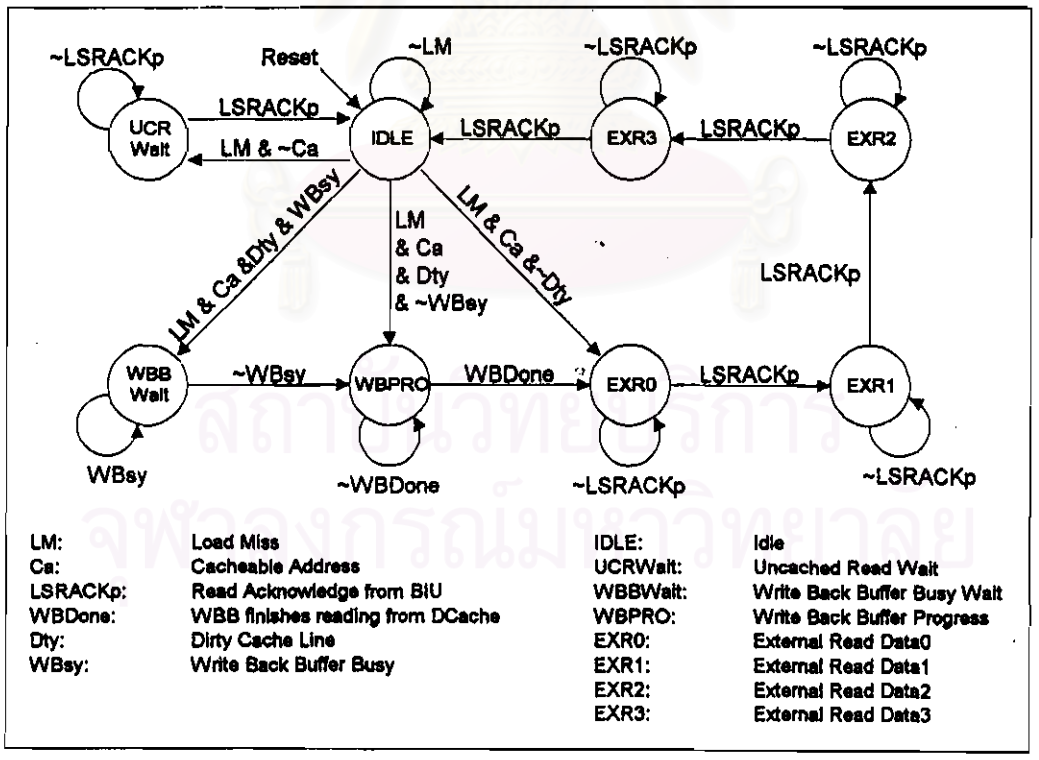
RFIFOWrp สัญญาณขอเขียนข้อมูลลง FIFO เกิดขึ้นเมื่อเกิด store miss

FDataIn[31:0] เป็นข้อมูลที่ต้องการเขียนลงหน่วยความจำ

FAddrIn[31:0] เป็นตำแหน่งที่ต้องการเขียนข้อมูลลงหน่วยความจำ

4.1.2 LMC (Load Miss Control) Unit

หน่วย LMC ถูกใช้ในการควบคุมการทำงานในกรณีที่เกิด Load Miss ขึ้น โดย LMC จะขอข้อมูลที่ต้องการจากหน่วยความจำหลักเพื่อเขียนลงใน D-Cache ผู้วิจัยได้ออกแบบ State Machine ของ LMC เพื่อควบคุมการทำงานดังรูปที่ 4.3

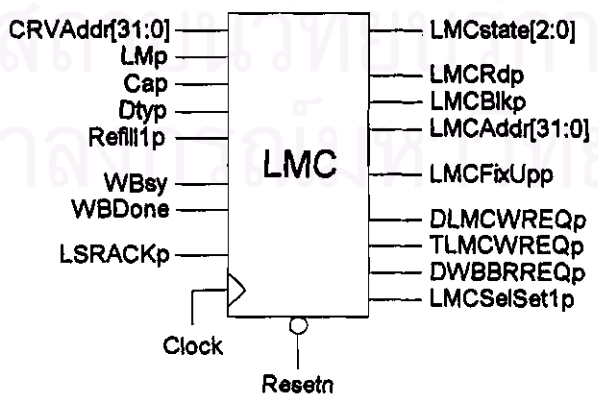


รูปที่ 4.3 Load Miss Control Unit State Diagram

ในขณะเริ่มการทำงานจะเกิดสัญญาณ Reset ขึ้น ทำให้สถานะอยู่ที่ "IDLE" State แล้วจะมีการเปลี่ยนแปลงสถานะดังนี้

1. สถานะจะเปลี่ยนไปที่ "Uncached Wait" State เมื่อเกิด Load Miss ขึ้นและภายในระบบไม่มีแคชและจะกลับไปที่ "iDLE" state อีกครั้งเมื่อได้รับข้อมูลจากหน่วยความจำแล้ว
2. สำหรับกรณีที่เกิด Load Miss และภายในระบบมีแคช ก็จะพิจารณาว่าข้อมูลในบรรทัดที่ต้องการเขียนทับนั้น Dirty หรือไม่ (ถ้า Dirty ค่า WB bit = '1') บรรทัดจะ Dirty เมื่อ ข้อมูลในแคชมีการเปลี่ยนแปลงทำให้มีข้อมูลไม่ตรงกับหน่วยความจำหลัก (WB bit มีค่าเท่ากับ '1') ซึ่งมีผลมาจากคำสั่งที่เกิด Store Hit (สำหรับโหมดการเขียนแบบ Write Back)
   2.1 ถ้าหากข้อมูลในบรรทัดนั้นไม่ Dirty มันจะเปลี่ยนจาก "iDLE" State ไปเป็น "EXR0" State และรอจนกระทั่งได้ข้อมูลที่ต้องการโหลดมาจากหน่วยความจำหลัก แล้วจึงเปลี่ยนไปที่ "EXR1" State และ "EXR2" State และ "EXR3" State ทำจนได้ข้อมูลครบ 4 doubleword ก็จะกลับไปอยู่ที่ "iDLE" State เหมือนเดิม โดยในขณะที่อยู่ "EXR3" State นั้นจะมีการเปลี่ยนค่าภายใน DCache Tag ด้วย
   2.2 สำหรับกรณีที่เกิด Load Miss และข้อมูลในบรรทัดนั้น Dirty ก็จำเป็นต้องนำข้อมูลในบรรทัดนั้นเขียนกลับไปที่หน่วยความจำหลัก โดยผู้วิจัยได้ออกแบบ Write Back Buffer (WBB) ขนาด 4 doubleword เพื่อเก็บข้อมูลในบรรทัดนั้นพักเอาไว้ก่อน แล้วจึงเขียนลงหน่วยความจำหลักเมื่อว่าง จากนั้น LMC จะทำงานต่อไปเหมือนในข้อ 2.2
   2.3 สำหรับกรณีที่เกิด Load Miss และข้อมูลในบรรทัดนั้น Dirty และ WBB ยังมีข้อมูลที่รอการเขียนลงหน่วยความจำค้างอยู่ สถานะจะเปลี่ยนจาก "iDLE" State ไปที่ "WBB Wait" State เพื่อรอให้ WBB เขียนข้อมูลลงหน่วยความจำให้เรียบร้อยเสียก่อน เมื่อ WBB ว่างแล้วสถานะย้ายไปอยู่ "WBPRO" State แล้ว LMC จึงนำข้อมูลในบรรทัดที่จะถูกเขียนทับ เขียนลงใน WBB จากนั้น สถานะเปลี่ยนไปที่ "EXR0" State เพื่อทำการอ่านข้อมูลจากหน่วยความจำจนครบ 4 doubleword

สัญญาณที่ติดต่อกับหน่วย LMC เป็นดังรูปที่ 4.4



รูปที่ 4.4 สัญญาณที่ติดต่อกับหน่วย LMC

- สัญญาณเข้า

CRVAddr[31:0] เป็นแหน่งของข้อมูลที่เกิดการ Load Miss ขึ้น

LMp สัญญาณที่บอกว่าเกิด Load Miss ขึ้น

Cap สัญญาณที่บอกว่าในระบบมีแคชอยู่หรือไม่

Dtyp สัญญาณที่บอกว่าบรรทัดของแคชที่จะถูกแทนที่นั้น ค่า WB bit ถูกตั้งค่าไว้

Refill1p สัญญาณระบบว่าต้องการแทนที่แคชในเซ็ตใด

WBsy หน่วย WBB ส่งมาบอกว่าบัฟเฟอร์ว่าง (0) หรือไม่ว่าง (1)

WBDone หน่วย WBB ส่งมาบอกว่านำข้อมูลจากแคชเขียนลงบัฟเฟอร์เรียบร้อยแล้ว

LSRACKp สัญญาณตอบกลับการขออ่านข้อมูลจากหน่วยความจำ

- สัญญาณออก

LMCstate[2:0] สัญญาณบอกสถานะการทำงานของหน่วย LMC ว่าเป็นอย่างไร

LMCRdp สัญญาณร้องขอการอ่านข้อมูลจากหน่วยความจำที่ตำแหน่ง *LMCAddr[31:0]* โดยในกรณีที่มีแคชในระบบ LMC จะขออ่านข้อมูลครั้งละ 4 doubleword โดยส่งสัญญาณ *LMCBlkp* ไปขอ

LMCFixUpp สัญญาณขอแก้ไขข้อมูลที่เคย Load Miss ซึ่งจะเกิดขึ้นเมื่อสถานะของ LMC อยู่ที่ EXR0 และได้รับการตอบกลับของสัญญาณ LSRACKp

DLMCWREQp หน่วย LMC ร้องขอการเขียนข้อมูลลง D-Cache Data โดยตรวจสอบจากสถานะของหน่วย LMC อยู่ที่ EXR0, EXR1, EXR2 หรือ EXR3 และได้รับการตอบกลับของสัญญาณ LSRACKp

TLMCWREQp หน่วย LMC ร้องขอการเขียนข้อมูลลง D-Cache Tag โดยตรวจสอบจากสถานะของหน่วย LMC อยู่ที่ EXR3 และได้รับการตอบกลับของสัญญาณ LSRACKp

DWBBRREQp หน่วย WBB ร้องขอการอ่านข้อมูลจาก D-Cache Data โดยตรวจสอบจากสถานะของหน่วย LMC อยู่ที่ WBPRO

LMCSelSet1p สัญญาณที่หน่วย LMC ใช้ในการเลือกเซ็ตของ D-Cache ที่จะถูกแทนที่

4.1.3 SHC (Store Hit Control)

หน่วย SHC ใช้ในการควบคุมการทำงานในกรณีที่เกิด Store Hit โดยภายใน SHC มีบัฟเฟอร์ (SHB – Store Hit Buffer) ใช้สำหรับเก็บข้อมูลที่ต้องการเขียนลงแคช โดยข้อมูลภายใน SHB นี้จะถูกเขียนลง D-Cache เมื่อ D-Cache ว่าง (ไม่มีหน่วยใดร้องขอการเขียนหรืออ่านข้อมูลกับ D-Cache) ผู้วิจัยได้ออกแบบ State Machine ของ SHC ตามขนาดของบัฟเฟอร์ เพื่อควบคุมการทำงานดังรูปที่ 4.5 โดยขนาดของบัฟเฟอร์ที่ใช้เป็น 1, 2, 4 หรือ 8 บัฟเฟอร์

หน่วย SHC จะเริ่มต้นการทำงานที่ "sempty" Stage และจะเปลี่ยนสถานะไปที่ "stbuf1" Stage ก็ต่อเมื่อเกิดเหตุการณ์ Store Hit ขึ้น ซึ่งสามารถตรวจสอบได้จากสัญญาณ DC0SHitp หรือ DC1SHitp หน่วย SHC จะเปลี่ยนสถานะกลับไปที่ "sempty" Stage ได้ก็ต่อเมื่อได้รับสัญญาณตอบกลับการเขียนข้อมูลลงแคช (SHCWACKp) โดยไม่มีเหตุการณ์ Store Hit เกิดขึ้นอีก และบัฟเฟอร์ภายใน SHC จะเต็มก็ต่อเมื่อสถานะเปลี่ยนไปอยู่ที่สถานะสุดท้าย เช่นขนาดของ SHB เท่ากับ 4 บัฟเฟอร์ ก็จะเต็มเมื่อสถานะไปอยู่ที่ "stbuf4" State เป็นต้น

สัญญาณที่ติดต่อกับหน่วย SHC เป็นดังรูปที่ 4.6 โดยสัญญาณเข้าได้กล่าวมาแล้วในหน่วย PC

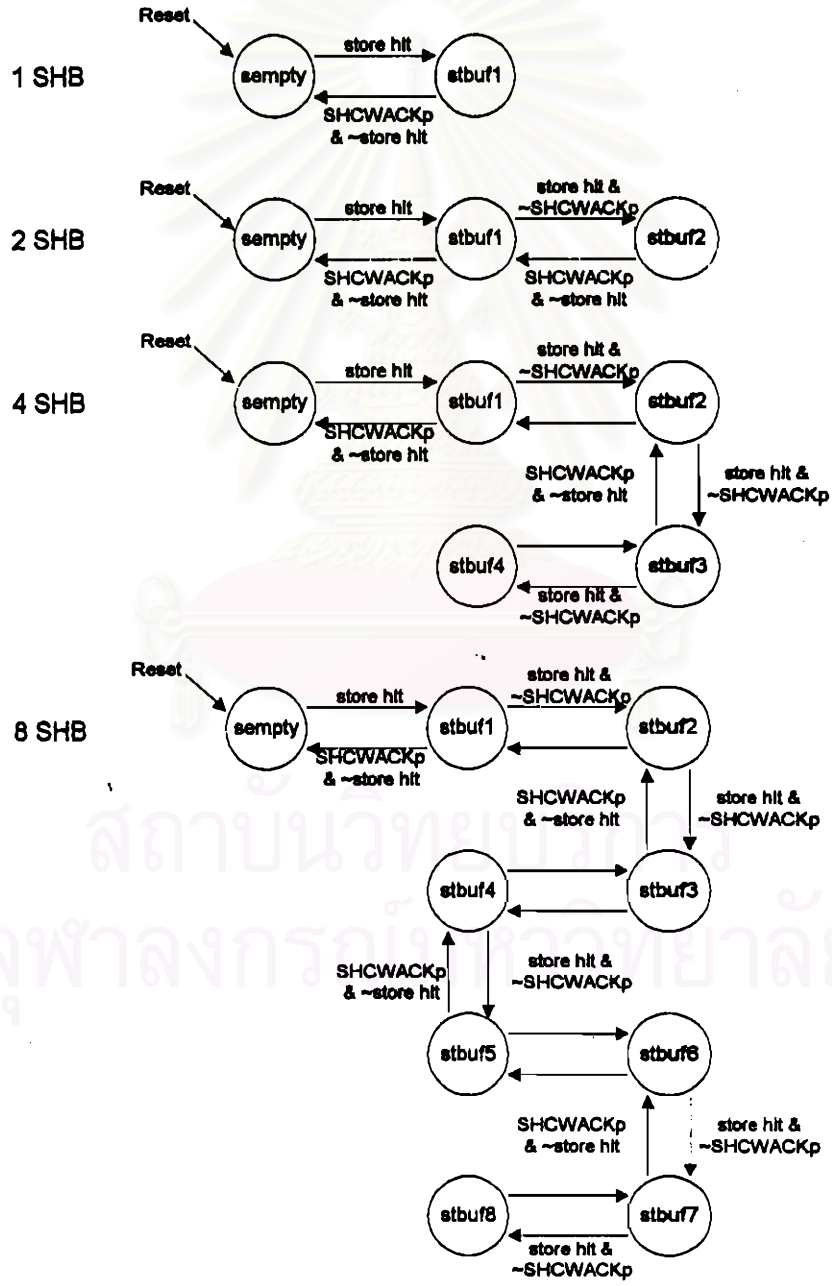
- สัญญาณออก

SHCFull สัญญาณนี้ใช้บอก Pipeline C Stage ว่าบัฟเฟอร์ในหน่วย SHC เต็มแล้ว

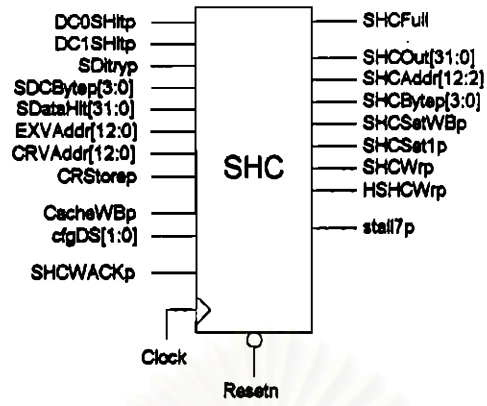
SHCWrp สัญญาณนี้ส่งไปที่ DCCtrl เพื่อร้องขอการเขียนข้อมูลลงแคชที่ตำแหน่ง SHCAddr[12:2] ใน Set SHCSet1p โดยข้อมูลที่ต้องการเขียนคือ SHCOut[31:0] และมีสัญญาณ SHCBytep [3:0] เป็นตัวระบุว่าต้องการเขียนข้อมูลลงแคชในไบต์ใด

SHCSetWBp สัญญาณนี้ส่งไปที่ DCCtrl เพื่อร้องขอการตั้งค่า write back bit ในแคช

HSHCWrp สัญญาณนี้ส่งไปที่ DCCtrl เพื่อขอเพิ่มความสำคัญในการเขียนข้อมูลลงแคช



รูปที่ 4.5 Store Hit Control Unit State Diagram



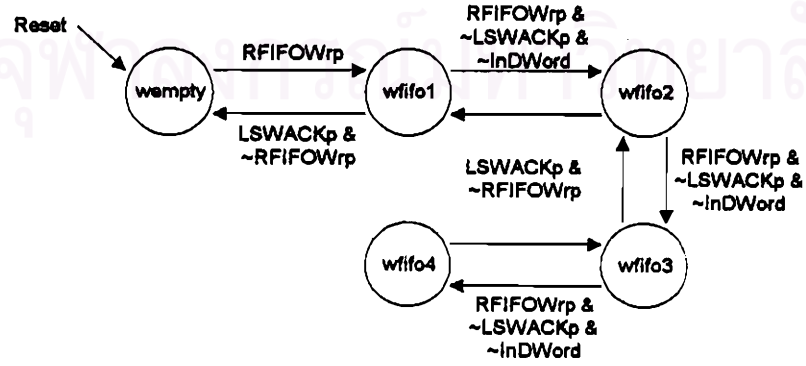
รูปที่ 4.6 สัญญาณที่ติดต่อกับหน่วย SHC

4.1.4 SMC (Store Miss Control)

หน่วย SHC ใช้ในการควบคุมการทำงานในกรณีที่เกิด Store Miss โดยภายใน SMC มี FIFO ใช้สำหรับเก็บข้อมูลที่ต้องการเขียนลงหน่วยความจำ โดยข้อมูลภายใน FIFO นี้จะถูกเขียนลงหน่วยความจำเมื่อบัสที่ติดต่อกับหน่วยความจำว่าง ผู้วิจัยได้ทำการออกแบบ State Machine ของ SMC ตามขนาดของ FIFO ที่ใช้เพื่อควบคุมการทำงานดังรูปที่ 4.7 โดย State Diagram ในรูปมีขนาด FIFO เป็น 4 doubleword แต่ขนาดของ FIFO ที่ใช้เป็น 2, 4 หรือ 8 word และ doubleword

สำหรับ FIFO ชนิด doubleword การทำงานเริ่มต้นที่ "wempty" Stage และสถานะเปลี่ยนไปเป็น "wfifo1" State เมื่อเกิดการเขียนข้อมูลลง FIFO (เขียนเมื่อเกิด Store Miss ถ้าเป็นโหมดการเขียนแบบ Write Through จะเพิ่มการเขียน FIFO เมื่อเกิด Store Hit ด้วย) และสำหรับการเกิด Store Miss ครั้งต่อไปจะต้องตรวจสอบดูก่อนว่าสามารถเขียนข้อมูลลงใน doubleword เดียวกับข้อมูลที่อยู่ใน FIFO เดิมหรือไม่ (InDWord) ถ้าหากได้สถานะก็ยังคงอยู่ที่ "wfifo1" State แต่ถ้าไม่ได้สถานะจะเปลี่ยนไปเป็น "wfifo2" State

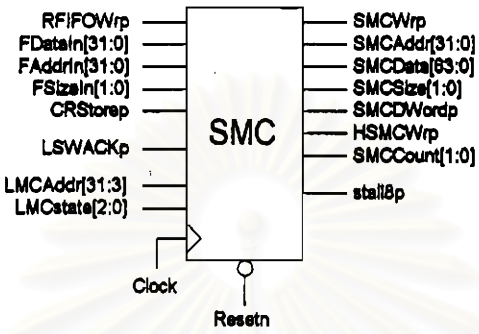
FIFO จะเต็มเมื่อสถานะอยู่ที่ "wfifo4" State (ถ้าหากขนาดของ FIFO เปลี่ยนเป็น 2 doubleword จะเต็มที่สถานะ "wfifo2" State และถ้าขนาด FIFO เปลี่ยนเป็น 8 doubleword จะเต็มที่สถานะ "wfifo8" State)



รูปที่ 4.7 Store Miss Control Unit State Diagram

สำหรับ FIFO ชนิด word การทำงานเริ่มต้นที่ "wempty" State และสถานะเปลี่ยนไปเป็น "wfifo1" State เมื่อเกิดการเขียนข้อมูลลง FIFO (เขียนเมื่อเกิด Store Miss) และสำหรับการเกิด Store Miss ครั้งต่อไป สถานะจะเปลี่ยนไปที่ "wfifo2" State

สัญญาณที่ติดต่อกับหน่วย SMC เป็นดังรูปที่ 4.8 โดยสัญญาณเข้าได้กล่าวมาแล้วในหน่วย PC และสัญญาณออกจะได้กล่าวถึงในหน่วย AccessBIU ต่อไป



รูปที่ 4.8 สัญญาณที่ติดต่อกับหน่วย SMC

4.1.5 DCCtrl (D-Cache Controller)

DCCtrl เป็นหน่วยที่จัดลำดับความสำคัญของการติดต่อกับ D-Cache Data และ D-Cache Tag ว่าขณะนั้นการติดต่อใดควรได้ทำก่อน เพื่อให้การประมวลผลที่สำคัญกว่าสามารถติดต่อกับแคชได้ก่อน ซึ่งส่งผลให้เกิดการ "STALL" Pipeline น้อยที่สุด

โดยมีการจัดลำดับความสำคัญของการติดต่อกับ D-Cache Data เรียงไว้ดังนี้

1) LMC ขอเขียน D-Cache Data โดยขอมาทางสัญญาณ DLMCWREQp ที่ตำแหน่ง LMCAddr [31:0] และข้อมูลที่ต้องการเขียนได้มาจากการอ่านหน่วยความจำทางสัญญาณ Ddata[63:0]
2) WBB ขออ่าน D-Cache Data โดยขอมาทางสัญญาณ DWBBRREQp ที่ตำแหน่ง WBBAddr [12:3]
3) SHC ขอเขียน D-Cache Data ในกรณีที่เกิดการ "STALL" Pipeline อันเนื่องมาจากการที่ Pipeline C Stage ไม่สามารถเขียนข้อมูลลง SHB ได้เนื่องจากบัฟเฟอร์เต็มแล้ว โดยตำแหน่งของแคชที่ต้องการเขียนคือ SHCAddr[12:2] และข้อมูลที่ต้องการเขียนคือ SHCData[31:0], SHCBytep[3:0] คือตำแหน่งของไบต์ที่ต้องการเขียน และ SHCSet1p คือเซ็ตของแคชที่ต้องการติดต่อด้วย
4) X stage ขออ่าน D-Cache Data โดยขอมาทางสัญญาณ EXLoadp ที่ตำแหน่ง EXVAddr[12:0]
5) SHC ขอเขียน D-Cache Data โดยมีสัญญาณอื่นๆ ประกอบเหมือนในข้อ 3 แต่มีความสำคัญในการร้องขอต่ำกว่าเพราะยังพอมีที่ว่างในบัฟเฟอร์อยู่

ส่วนการจัดลำดับความสำคัญของการติดต่อกับ D-Cache Tag ถูกเรียงไว้ดังนี้

1) LMC ขอเขียน D-Cache Tag โดยขอมาทางสัญญาณ TLMCWREQp ที่ตำแหน่ง LMCAddr [31:0] และข้อมูลที่ต้องการเขียนคือ { LMCAddr[31:10], 1, 0 } อันได้แก่ค่า Tag, valid bit และ write back bit ตามลำดับ

2) SHC ขอเขียน D-Cache Tag ในกรณีที่เกิด "STALL" Pipeline จากหน่วย SHC โดยขอเขียนเพียงเพื่อตั้งค่า WB bit ให้เป็น '1' เท่านั้น ตำแหน่งที่ต้องการติดต่อคือ SHCAddr[12:2] และเซ็ตของแคชที่ต้องการเขียนคือ SHCSet1p

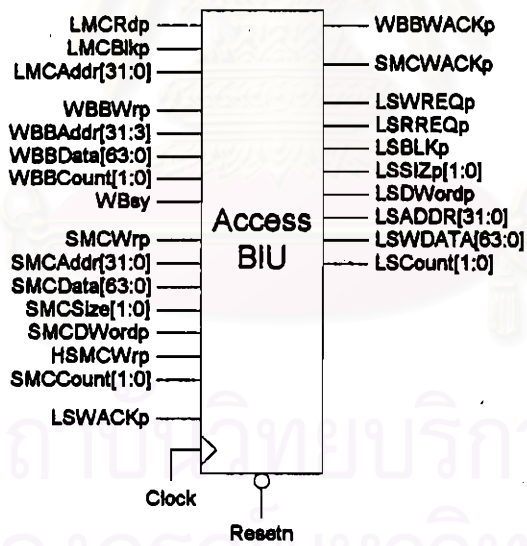
3) X stage ขออ่าน D-Cache Tag โดยขอมาทางสัญญาณ EXLoadp หรือ EXStorep ที่ตำแหน่ง EXVAddr[12:0]

4) SHC ขอเขียน D-Cache Tag โดยมีสัญญาณอื่นๆ ประกอบเหมือนในข้อ 2 แต่มีความสำคัญในการร้องขอต่ำกว่าเพราะยังพอมีที่ว่างในบัฟเฟอร์อยู่

4.1.6 AccessBIU (Access Bus Interface Unit)

เป็นหน่วยที่จัดการในส่วนที่ LSU ติดต่อกับ BIU เมื่ออ่านหรือเขียนข้อมูลกับหน่วยความจำ เพื่อให้การประมวลผลที่สำคัญกว่าได้ติดต่อกับหน่วยความจำไปก่อน

โดยสัญญาณที่ติดต่อกับหน่วย AccessBIU เป็นดังรูป 4.9



รูปที่ 4.9 สัญญาณที่ติดต่อกับหน่วย AccessBIU

- สัญญาณเข้า

LMCRdp หน่วย LMC ร้องขอการอ่านข้อมูลจากหน่วยความจำที่ตำแหน่ง *LMCAddr[31:0]* โดยถ้าต้องการขออ่านคราวละ 4 doublewords ต้องส่งสัญญาณ *LMCBlkp* = '1'

WBBWrp หน่วย WBB ร้องขอการเขียนข้อมูลลงหน่วยความจำที่ตำแหน่ง *WBBAddr[31:3]* โดยข้อมูลที่ต้องการเขียนคือ *WBBData[63:0]*, จำนวนที่หน่วย WBB ต้องการเขียนลงหน่วยความจำคือ *WBBCount[1:0]* และมีสัญญาณ *Wbsy* เป็นตัวบอกว่าบัฟเฟอร์ในหน่วย WBB ไม่ว่าง

SMCWrp หน่วย SMC ร้องขอการเขียนลงหน่วยความจำที่ตำแหน่ง *SMCAddr[31:0]* โดยข้อมูลที่ต้องการเขียนคือ *SMCData[63:0]* , ขนาดของข้อมูลที่ต้องการเขียนคือ *SMCSize[1:0]* โดยระบุเป็นจำนวนไบต์ และถ้าต้องการเขียนเป็น doubleword จะระบุที่สัญญาณ *SMCDWordp*, จำนวนที่หน่วย SMC ต้องการเขียนลงหน่วยความจำคือ *SMCCount[1:0]*

HSMCWrp เป็นสัญญาณเพิ่มความสำคัญของการขอเขียนจากหน่วย SMC

- สัญญาณภายในที่สำคัญ

HWBBWrp เป็นสัญญาณเพิ่มความสำคัญของการขอเขียนจากหน่วย WBB โดยเกิดขึ้นเมื่อตำแหน่งที่ WBB ขอเขียนกับ LMC ขออ่านเป็นตำแหน่งเดียวกันจึงจำเป็นต้องให้หน่วย WBB ได้ทำงานก่อน เพื่อป้องกันการอ่านข้อมูลที่ผิดพลาดไป และ WBsy = '1'

- สัญญาณออก

WBBWACKp สัญญาณตอบรับการเขียนของหน่วย WBB

SMCWACKp สัญญาณตอบรับการเขียนของหน่วย SMC

LSRREQp สัญญาณขออ่านข้อมูลจากหน่วยความจำที่ตำแหน่ง *LSADDR[31:0]* โดยถ้าต้องการอ่านคราวละ 4 doublewords จะบอกทางสัญญาณ LSBLKp

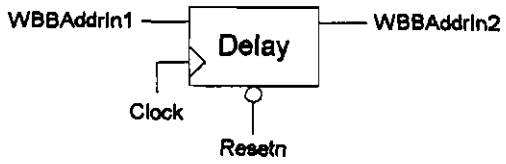
LSWREQp สัญญาณขอเขียนข้อมูลลงหน่วยความจำที่ตำแหน่ง *LSADDR[31:0]* ข้อมูลที่ต้องการเขียนคือ *LSWDATA[63:0]*, ขนาดของข้อมูลที่ต้องการเขียนระบุทางสัญญาณ *LSSIZp[1:0]* (จำนวนไบต์), *LSDWordp* (ขนาด doubleword) และจำนวนครั้งที่ต้องการขอเขียนคือ *LSCount[1:0]*

ลำดับความสำคัญของการติดต่อกับหน่วยความจำถูกเรียงไว้ดังนี้

1. HWBBWrp
2. HSMCWrp
3. LMCRdp
4. WBBWrp
5. SMCWrp

4.1.7 Delay

หน่วยนี้ใช้ในการ delay ข้อมูลไป 1 สัญญาณนาฬิกา ซึ่งทำเพื่อให้ข้อมูลที่ส่งไปเก็บที่หน่วย WBB ไปพร้อมกับค่าแอดเดรส โดยสัญญาณที่ติดต่อกับหน่วย Delay เป็นดังรูปที่ 4.10



รูปที่ 4.10 แสดงสัญญาณที่ติดต่อกับหน่วย Delay

4.1.8 Stall

หน่วยนี้ใช้ในการตรวจสอบเหตุการณ์ "STALL" Pipeline อันเนื่องมาจากการทำงานในหน่วย LSU ทั้งหมด โดยกรณีที่เกิดขึ้นทั้งหมดมี 9 กรณี ดังนี้

1. LMC=busy and X=load

การ "STALL" Pipeline กรณีนี้เกิดขึ้นเมื่อหน่วย LMC ยังมีการทำงานค้างอยู่ และที่ Pipeline X Stage ต้องการประมวลผลคำสั่ง Load จึงจำเป็นต้อง "STALL" Pipeline เพื่อป้องกันการเกิด Load Miss ในขณะที่หน่วย LMC ไม่ว่าง

2. Load miss and X=load or C index=X index of store

สำหรับกรณีนี้เกิดขึ้นเมื่อ Pipeline C stage ตรวจเจอว่าเกิด Load Miss ขึ้น และในขณะนั้นที่ Pipeline X stage กำลังประมวลผลคำสั่ง Load อยู่ หรือที่ Pipeline X stage กำลังประมวลผลคำสั่ง Store อยู่นั้น มีค่า index ในการชี้ D-Cache ที่เดียวกับของ Pipeline C stage ที่เกิด Load Miss ก็จำเป็นต้อง "STALL" Pipeline ไป 1 ช่วงสัญญาณนาฬิกา เพื่อป้องกันการอ่านค่า D-Cache Tag ที่ไม่ใช่ค่าจริง

3. Fixup data

การ "STALL" Pipeline แบบนี้เกิดขึ้นในช่วงที่สัญญาณ LsuFixUpp มีค่าเป็น '1' เพื่อให้ ISU มีช่วงเวลาในการแก้ไขข้อมูลในรีจีสเตอร์

4. LMC=busy and LMC index = X index of store

กรณีนี้เกิดขึ้นในหน่วย LMC ยังมีการทำงานค้างอยู่ และที่ Pipeline X Stage ต้องการประมวลผลคำสั่ง Store ในตำแหน่งของ D-Cache เดียวกัน ซึ่งจำเป็นต้องหยุดการทำงานคำสั่ง Store ไว้เพื่อป้องกันการอ่านค่า D-Cache Tag ที่ไม่มีอยู่จริง

5. X index of load = C index of store

สำหรับกรณีนี้เกิดขึ้นเมื่อ Pipelie X Stage ประมวลผลคำสั่ง Load อยู่ ต้องการอ่าน D-Cache Tag ในตำแหน่งเดียวกับที่ Pipeline C Stage กำลังทำงานคำสั่ง Store

6. SHB full and (C=store-hit or X index = SHB index)

การ "STALL" Pipeline ในกรณีนี้เกิดขึ้นเมื่อบัฟเฟอร์ในหน่วย SHC เต็มและที่ Pipeline C Stage ต้องการเขียนข้อมูลลงในบัฟเฟอร์ หรือในกรณีที่ตำแหน่งที่ Pipeline X Stage ต้องการอ่าน D-Cache Tag ตรงกับที่ SHB ต้องการเขียน D-Cache Tag

7. FIFO in SMC full, BIU can't service SMC, and C=store

เกิดขึ้นเนื่องจาก FIFO ในหน่วย SMC เต็มในขณะเดียวกัน BIU ยังไม่นำข้อมูลจาก SMC ไปเขียนลงหน่วยความจำหลัก และที่ C stage ประมวลผลคำสั่ง store

8. X stage will access the same cache as SHC

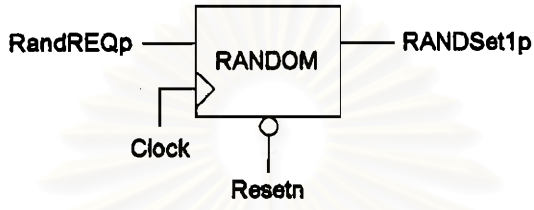
กรณีนี้เกิดขึ้นที่ Pipeline X Stage ต้องการติดต่อกับ D-Cache Tag หรือ D-Cache Data ตรงกับที่ SHC ต้องการติดต่อ เพื่อหยุด Pipeline X Stage เอาไว้เพราะใน Stage นั้นไม่สามารถอ่าน D-Cache ได้

9. X stage will access the cache as LMC and WBB

และในกรณีนี้เกิดขึ้นที่ X stage ต้องการติดต่อกับ D-Cache Tag หรือ D-Cache Data ตรงกับที่ LMC หรือ WBB ต้องการติดต่อ เพื่อหยุด X stage เอาไว้เพราะใน Stage นั้นไม่สามารถอ่าน D-Cache ได้

4.1.9 RANDOM

หน่วยนี้ใช้ในการสุ่มสัญญาณขึ้นมา เพื่อใช้เลือกเซ็ตที่จะต้องถูกแทนที่ข้อมูลในกรณีที่เป็นการจัดแคชแบบ 2-way set associative เนื่องจากผู้ออกแบบต้องการทราบด้วยว่าค่าที่สุ่มได้เป็นค่าอะไร เพื่อให้สามารถทดสอบความถูกต้องของการทำงานได้โดยการเปรียบเทียบกับ Output Vector ที่ได้จากภาษา C จึงจำเป็นต้องใช้ค่าที่ได้สุ่มจากการเขียนฟังก์ชั่น rand() ของภาษา C มาเก็บไว้ ซึ่งสามารถใช้กรรมวิธีอื่นแทนได้ โดยสัญญาณที่ติดต่อกับหน่วย RANDOM เป็นดังรูปที่ 4.11



รูปที่ 4.11 สัญญาณที่ติดต่อกับหน่วย RANDOM

- สัญญาณเข้า

RandREQp สัญญาณร้องขอข้อมูลจากการสุ่ม

- สัญญาณออก

RANDSet1p สัญญาณที่ได้จากการสุ่ม

4.2 สรุป

ในบทนี้ได้กล่าวถึงหน้าที่การทำงาน และวิธีการออกแบบหน่วยประมวลผลย่อยภายใน Load/Store Unit (LSU) โดยมีหน่วย PC จัดการในเรื่องของการทำงานเป็น pipeline, หน่วย LMC ทำงานเมื่อเกิด load miss ,หน่วย SHC ทำงานเมื่อเกิด store hit, หน่วย SMC ทำงานเมื่อเกิด store miss มีหน่วย DCCtrl จัดการในเรื่องของการจัดลำดับความสำคัญในการเข้าติดต่อกับ D-Cache และหน่วย AccessBIU จัดการจัดลำดับความสำคัญในการเข้าติดต่อกับ BIU เพื่อขออ่านและเขียนข้อมูลกับหน่วยความจำ