

## การทดลองที่ A

การทำงานของแคชชนิด Direct Mapped

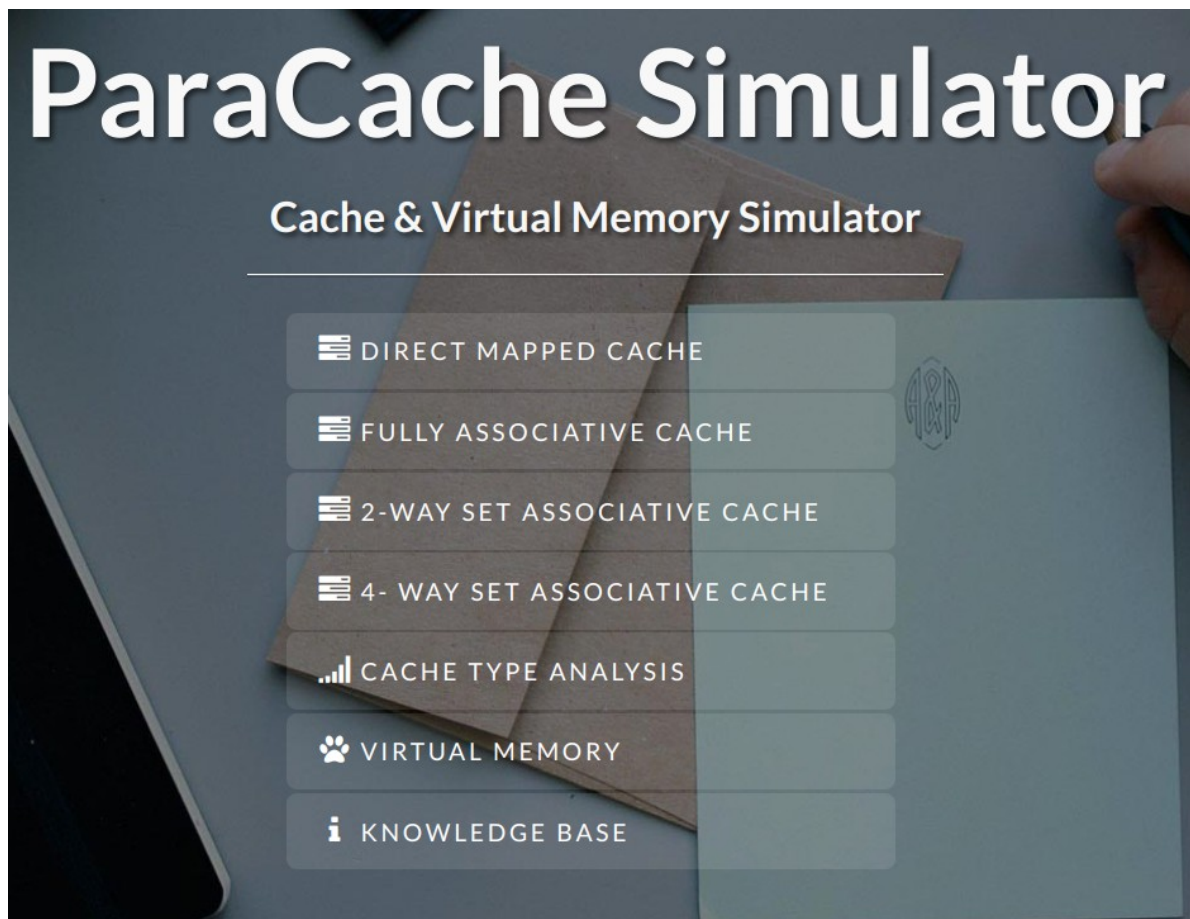
วิชา Computer Organization and Assembly Language

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใช้เว็บเบราว์เซอร์เปิดใช้งานซิมูเลเตอร์ ชื่อ Para Cache

<https://www3.ntu.edu.sg/home/smitha/ParaCache/Paracache/start.html>



เอกสารอธิบาย

<https://www3.ntu.edu.sg/home/smitha/ParaCache/Paracache/kb.pdf>

ทำการทดลอง ตามขั้นตอนต่อไปนี้

## 1. การทดลอง Direct Mapped Cache

กดเมนู เลือก Direct Mapped Cache ตั้งขนาดและ Write Policy ของแคช ดังรูป

ParaCache

**Write Policies**

☒ **Write Back**    ☐ **Write Through**  
☒ **Write On Allocate**    ☐ **Write Around**

**Cache Size (power of 2)**      
**Memory Size (power of 2)**      
**Offset Bits**

2. กด Submit แล้วสังเกตรายละเอียดของแคชที่อยู่ด้านขวา

## DIRECT MAPPED CACHE

➔ Instruction Breakdown

TAG	INDEX	OFFSET
2 bit	2 bit	0 bit

☐ Memory Block

B. A W. 0  
B. B W. 0  
B. C W. 0  
B. D W. 0  
B. E W. 0  
B. F W. 0

☐ Cache Table

Index	Valid	Tag	Data (Hex)	Dirty Bit
0	0	-	0	0
1	0	-	0	0
2	0	-	0	0
3	0	-	0	0

เลื่อนหน้าต่างลงไปด้านล่างสุดของ Memory Block โปรดสังเกตหมายเลขบล็อก (B.) มีค่าเท่ากับ 0 ถึง F และหมายเลขเวิร์ด (W.) เท่ากับ 0 เสมอ

## ☐ Memory Block

B. A W. 0  
B. B W. 0  
B. C W. 0  
B. D W. 0  
B. E W. 0  
B. F W. 0

เพราะเหตุใด

B มีค่า 0-F เพราะขนาด memory size = 16 และมี Offset Bits = 0

และเหตุผลที่ W.=0 เสมอ เพราะ Offset Bits = 0

3. การทดลองคำสั่ง Load Instruction ที่หมายเลขแอดเดรสที่ต้องการ หรือ ให้โปรแกรมสุ่มหมายเลขแอดเดรสให้  
 กรอก 4 ลงในหมายเลขฐานสิบหกที่มีอยู่ในกล่องข้อความด้านขวา  
 กรอกหมายเลข 7, 7, c, 4, 0, 4, 3, 5, 5 ในกล่องข้อความดังรูป

Instruction

Load

(in hex)#

4

7,7,c,4,0,4,3,5,5

Gen. Random

Submit

Information

Offset = 0 bits

Index bits =  $\log_2(4/1) = 2$  bits

Instruction Length =  $\log_2(16) = 4$  bits

Tag = 4 bits - 0 bits - 2 bits = 2

Next

Fast Forward

อธิบาย information ในรูปว่า Tag, Index และ Offset สัมพันธ์กับ Cache Size และ Memory Size ที่กรอก

$$\text{tag} = \text{cache size} - \text{offset} - \text{index}$$

$$\text{index} = \log_2(\text{cache size}) - \text{offset}$$

$$\text{offset} = 0 \quad \text{เนื่องจากเรากำหนดให้ offset bits} = 0$$

4. กดปุ่ม Submit หมายเลข 4 ที่กรอก โปรแกรมสังเกตและอ่านกล่องข้อความที่เป็นสีชมพู อธิบายตามความเข้าใจ

แปลง hex  $\rightarrow$  binary แล้วแบ่งเป็น tag, index, offset

5. กดปุ่ม Next และสังเกตกล่องข้อความที่เปลี่ยนเป็นสีเหลืองว่าเกี่ยวข้องกับหมายเลขที่ Submit ไปก่อนหน้านี้อย่างไร

➡ Instruction Breakdown

01	00	0
2 bit	2 bit	0 bit

☐ Memory Block

B. A W. 0
B. B W. 0
B. C W. 0
B. D W. 0
B. E W. 0
B. F W. 0

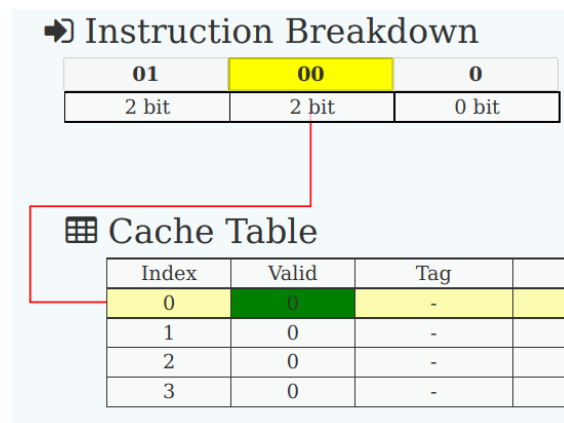
☐ Cache Table

Index	Valid	Tag	Data (Hex)	Dirty Bit
0	0	-	0	0
1	0	-	0	0
2	0	-	0	0
3	0	-	0	0

อธิบายความสัมพันธ์ระหว่าง Instruction Breakdown 01 00 และหมายเลข 4

ใช้งาน 4 bits โดย 00 และ 01 ใช้งานรวมแล้ว 2 bits

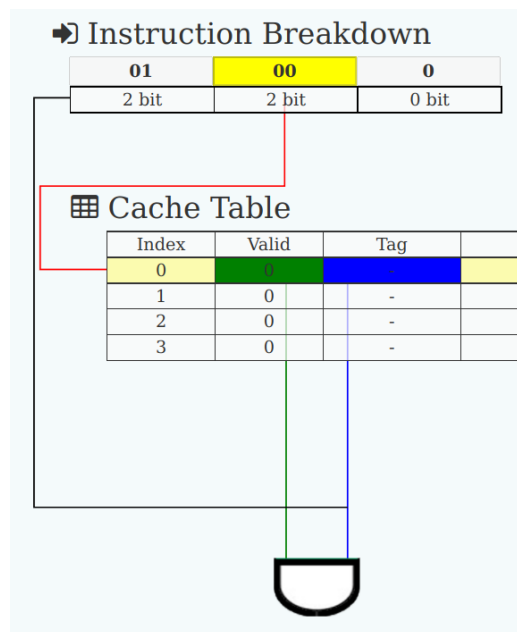
6. กดปุ่ม Next และสังเกตกล่องข้อความที่เปลี่ยนเป็นสีเขียว



อธิบายรูปนี้ และบิต Valid จึงเป็น 0

เพราะยังไม่มีการโหลดค่าเข้า

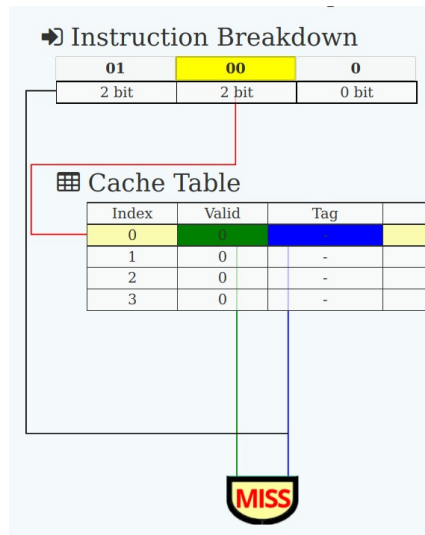
7. กดปุ่ม Next และสังเกตกล่องข้อความที่เปลี่ยนเป็นน้ำเงิน และ AND เกตว่าทำกระบวนการอะไรกัน



อธิบายว่า Tag จึงมีสัญลักษณ์ '-'

Tag → '-' : ยังไม่มีการโหลดค่าเข้า, AND ทำการ check ว่ามีค่าไหม

8. กดปุ่ม Next เพื่อดำเนินการต่อ โปรดสังเกตข้อความบน AND เกต



กดปุ่ม Next เพื่อดำเนินการต่อ โปรดสังเกต Cache Table ว่ามีการเปลี่ยนแปลงอย่างไร

### Cache Table

Index	Valid	Tag	Data (Hex)	Dirty Bit
0	1	01	BLOCK 4 WORD 0 - 0	0
1	0	-	0	0
2	0	-	0	0
3	0	-	0	0

อธิบาย Valid Tag และ Data (Hex) จึงเปลี่ยนเป็นรูปนี้

*index 0 ยังไม่มีข้อมูล จึง update ค่า 4 เข้าไปที่ index 0 ค่าของ valid จึงเป็น 1 เนื่องจากมีข้อมูลมาเก็บแล้ว*

9. กดปุ่ม Next เพื่อดำเนินการต่อ โปรดสังเกตข้อมูลสถิติสี่เหลี่ยมด้านล่าง

<b>Statistics</b>	
<b>Hit Rate :</b>	<b>0%</b>
<b>Miss Rate :</b>	<b>100%</b>
<b>List of Previous Instructions :</b>	
• Load 4 [Miss]	

อธิบายข้อมูลที่ได้

*ค่าที่ในลวดไม่ได้อยู่ใน cache → miss rate 100%*

10. โปรดสังเกตหมายเลขแอดเดรสถัดไปจะย้ายมาในกล่องข้อความด้านขวาบนของรูปนี้ กดปุ่ม Submit

11. กดปุ่ม Fast Forward เพื่อเร่งการทำงานของคำสั่งให้รวดเร็วขึ้น โปรดสังเกตการเปลี่ยนแปลงใน Cache Table และ Statistics หลังจากนั้น

01	11	0
2 bit	2 bit	0 bit

### Memory Block

### Cache Table

Index	Valid	Tag	Data (Hex)
0	1	01	BLOCK 4 WORD 0 - 0
1	0	-	0
2	0	-	0
3	1	01	BLOCK 7 WORD 0 - 0

### Statistics

**Hit Rate : 0%**

**Miss Rate : 100%**

**List of Previous Instructions :**

- Load 4 [Miss]
- Load 7 [Miss]

12. กด Submit และ Fast Forward เรื่อยๆ จนไม่เหลือหมายเลขแอดเดรส โปรดสังเกตการเปลี่ยนแปลงใน Statistics หลังจากนั้น

### Instruction

Load

▼

(in hex)#

List of next 10 Instructions

Gen. Random

Submit

### Information

The cycle has been completed.  
Please submit another instructions

Next

Fast Forward

Statistics	
Hit Rate :	20%
Miss Rate :	80%
List of Previous Instructions :	
<ul style="list-style-type: none"> <li>• Load 4 [Miss]</li> <li>• Load 7 [Miss]</li> <li>• Load 7 [Hit]</li> <li>• Load C [Miss]</li> <li>• Load 4 [Miss]</li> <li>• Load 0 [Miss]</li> <li>• Load 4 [Miss]</li> <li>• Load 3 [Miss]</li> <li>• Load 5 [Miss]</li> <li>• Load 5 [Hit]</li> </ul>	

อธิบายข้อมูลที่ได้ว่า Hit Rate และ Miss Rate คำนวณอย่างไร

$$\frac{\text{จำนวนครั้งที่ hit rate}}{\text{จำนวนครั้งที่ load}} \times 100, \quad \frac{\text{จำนวนครั้งที่ miss rate}}{\text{จำนวนครั้งที่ load}} \times 100 \quad \text{Example : Hit rate} = \frac{2}{10} \times 100 = 20\%$$

นักศึกษาควรจะได้ผลการทดลองใน Cache Table ตรงกับรูปนี้

### Cache Table

Index	Valid	Tag	Data (Hex)
0	1	01	BLOCK 4 WORD 0 - 0
1	1	01	BLOCK 5 WORD 0 - 0
2	0	-	0
3	1	00	BLOCK 3 WORD 0 - 0

13. กรอกหมายเลขบล็อกที่แคชมีอยู่ เพื่อจูงใจให้เกิด แคชฮิต ดังรูป

### Instruction

Load

▼

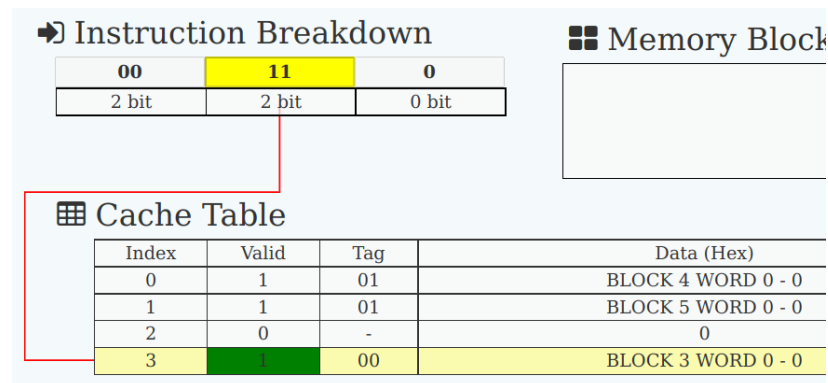
(in hex)#

3, 4, 5

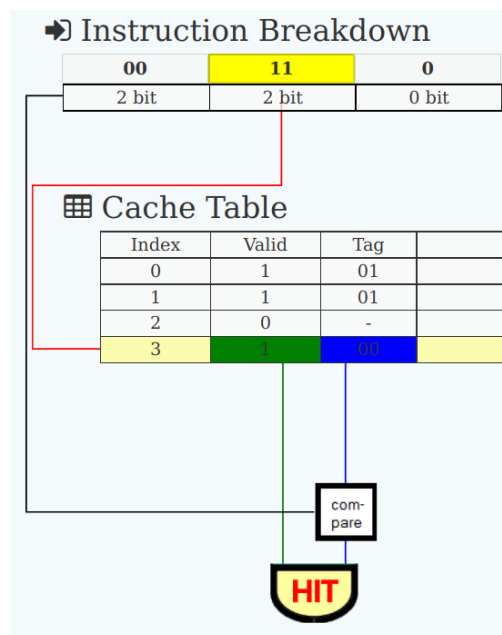
Gen. Random

Submit

กด Submit และ Next จนได้เหตุการณ์นี้



โปรดสังเกตคอลัมน์ Valid และ Tag ว่าตรงกันหรือไม่



14. กดปุ่ม Submit หมายเลขถัดไปจนหมด และแนบรูปตาราง Statistics ว่ามีการเปลี่ยนแปลงหรือไม่ อย่างไร

Statistics	
Hit Rate :	38%
Miss Rate :	62%
List of Previous Instructions :	
• Load 4 [Miss]	
• Load 7 [Miss]	
• Load 7 [Hit]	
• Load C [Miss]	
• Load 4 [Miss]	
• Load 0 [Miss]	
• Load 4 [Miss]	
• Load 3 [Miss]	
• Load 5 [Miss]	
• Load 5 [Hit]	
• Load 3 [Hit]	
• Load 4 [Hit]	
• Load 5 [Hit]	

มีการเปลี่ยนแปลง โดยมีค่า Hit rate เพิ่มขึ้น  
เพราะมีการโหลดค่าซ้ำที่มีการ cache อยู่แล้ว



**กิจกรรมท้ายการทดลอง**

1. ศึกษาการทำงานของ Load Instruction เช่นเดิม
  - ตั้งขนาดของแคชเท่ากับ 8 และ Memory Size เท่ากับ 16 OFFSET = 0 บิต แล้วเปรียบเทียบ
  - ตั้งขนาดของแคชเท่ากับ 4 และ Memory Size เท่ากับ 16 OFFSET = 1 บิต แล้วเปรียบเทียบ
2. ศึกษาการทำงานของ Load Instruction เช่นเดิม แต่ตั้ง Write Policy เป็น Write Through และ Write Around
  - ตั้งขนาดของแคชเท่ากับ 4 และ Memory Size เท่ากับ 16 OFFSET = 0 บิต
  - ตั้งขนาดของแคชเท่ากับ 8 และ Memory Size เท่ากับ 16 OFFSET = 0 บิต
  - ตั้งขนาดของแคชเท่ากับ 4 และ Memory Size เท่ากับ 16 OFFSET = 1 บิต
3. เปลี่ยน Instruction เป็น Store เพื่อศึกษาการทำงานของ Dirty Bit โดยตั้ง Write Policy เป็น Write Back และ Write on Allocate
  - ตั้งขนาดของแคชเท่ากับ 4 และ Memory Size เท่ากับ 16 OFFSET = 0 บิต
  - ตั้งขนาดของแคชเท่ากับ 8 และ Memory Size เท่ากับ 16 OFFSET = 0 บิต
  - ตั้งขนาดของแคชเท่ากับ 4 และ Memory Size เท่ากับ 16 OFFSET = 1 บิต
4. เปลี่ยน Instruction เป็น Store เพื่อศึกษาการทำงานของ Dirty Bit โดยตั้ง Write Policy เป็น Write Through และ Write Around
  - ตั้งขนาดของแคชเท่ากับ 4 และ Memory Size เท่ากับ 16 OFFSET = 0 บิต
  - ตั้งขนาดของแคชเท่ากับ 8 และ Memory Size เท่ากับ 16 OFFSET = 0 บิต
  - ตั้งขนาดของแคชเท่ากับ 4 และ Memory Size เท่ากับ 16 OFFSET = 1 บิต

① Cache Size = 8 , Memory Size = 16 , offset = 0

↳ Hit Rate : 30% , Miss Rate : 70%

Statistics	
Hit Rate :	30%
Miss Rate :	70%
List of Previous Instructions :	
<ul style="list-style-type: none"> <li>• Load 4 [Miss]</li> <li>• Load 7 [Miss]</li> <li>• Load 7 [Hit]</li> <li>• Load C [Miss]</li> <li>• Load 4 [Miss]</li> <li>• Load 0 [Miss]</li> <li>• Load 4 [Hit]</li> <li>• Load 3 [Miss]</li> <li>• Load 5 [Miss]</li> <li>• Load 5 [Hit]</li> </ul>	

Cache Size = 4 , Memory Size = 16 , offset = 1

↳ Hit Rate : 30% , Miss Rate : 70%

Statistics	
Hit Rate :	30%
Miss Rate :	70%
List of Previous Instructions :	
<ul style="list-style-type: none"> <li>• Load 4 [Miss]</li> <li>• Load 7 [Miss]</li> <li>• Load 7 [Hit]</li> <li>• Load C [Miss]</li> <li>• Load 4 [Miss]</li> <li>• Load 0 [Miss]</li> <li>• Load 4 [Miss]</li> <li>• Load 3 [Miss]</li> <li>• Load 5 [Hit]</li> <li>• Load 5 [Hit]</li> </ul>	

ต่างจากที่ทำตอนแรก โดยเหตุผลที่ต่างกันขึ้นอยู่กับค่า load

Cache Size = 4 , Memory Size = 16 , offset = 0

↳ Hit Rate : 20% , Miss Rate : 80%

Statistics	
Hit Rate :	20%
Miss Rate :	80%
List of Previous Instructions :	
<ul style="list-style-type: none"> <li>• Load 4 [Miss]</li> <li>• Load 7 [Miss]</li> <li>• Load 7 [Hit]</li> <li>• Load C [Miss]</li> <li>• Load 4 [Miss]</li> <li>• Load 0 [Miss]</li> <li>• Load 4 [Miss]</li> <li>• Load 3 [Miss]</li> <li>• Load 5 [Miss]</li> <li>• Load 5 [Hit]</li> </ul>	