

Assignment 1:

Create a project, type in the program of Laboratory 7, Home Assignment 4 , build it and upload it to the cache simulator.

Code:

```
#Laboratory Exercise 7, Home Assignment 4

.data
    Message: .asciiz "Ket qua tinh giai thua la: "

.text
main:
    jal WARP
print:
    add $a1, $v0, $zero # $a0 = result from N!
    li $v0, 56
    la $a0, Message
    syscall
quit:
    li $v0, 10 #terminate
    syscall
endmain:
#-----
#Procedure WARP: assign value and call FACT
#-----

WARP:
    sw $fp,-4($sp) #save frame pointer (1)
    addi $fp,$sp,0 #new frame pointer point to the top (2)
    addi $sp,$sp,-8 #adjust stack pointer (3)
    sw $ra,0($sp) #save return address (4)
```

```

    li $a0, 10          #load test input N
    jal FACT #call fact procedure
    nop

    lw $ra,0($sp) #restore return address (5)
    addi $sp,$fp,0 #return stack pointer (6)
    lw $fp,-4($sp) #return frame pointer (7)
    jr $ra

wrap_end:
#-----
#Procedure FACT: compute N!
#param[in] $a0 integer N
#return $v0 the largest value
#-----

FACT:
    sw $fp,-4($sp) #save frame pointer
    addi $fp,$sp,0 #new frame pointer point to stack's top
    addi $sp,$sp,-12 #allocate space for $fp,$ra,$a0 in stack
    sw $ra,4($sp) #save return address
    sw $a0,0($sp) #save $a0 register

    slti $t0,$a0,2 #if input argument N < 2
    beq $t0,$zero,recursive #if it is false ((a0 = N) >=2)
    nop
    li $v0,1 #return the result N!=1
    j done
    nop

recursive:
    addi $a0,$a0,-1 #adjust input argument

```

```
jal FACT #recursive call
nop
lw $v1,0($sp) #load a0
mult $v1,$v0 #compute the result
mflo $v0
```

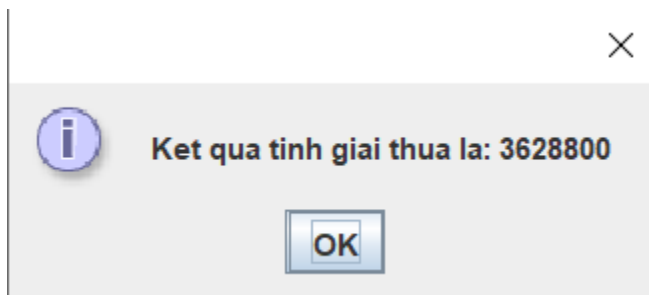
done:

```
lw $ra,4($sp) #restore return address
lw $a0,0($sp) #restore a0
addi $sp,$fp,0 #restore stack pointer
lw $fp,-4($sp) #restore frame pointer
jr $ra #jump to calling
```

fact_end:

Kết quả:

Tính giai thừa 10



Simulate and illustrate data cache performance

Data Cache Simulation Tool, Version 1.2

Simulate and illustrate data cache performance

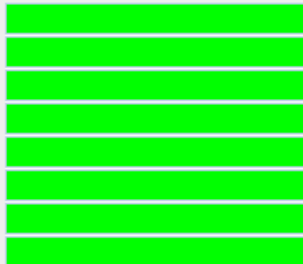
Cache Organization

Placement Policy: Direct Mapping Number of blocks: 8

Block Replacement Policy: LRU Cache block size (words): 4

Set size (blocks): 1 Cache size (bytes): 128

Cache Performance

Memory Access Count: 355 Cache Block Table: 

Cache Hit Count: 331 (block 0 at top)

Cache Miss Count: 24 ☐ = empty

Cache Hit Rate: 93% ☒ = hit

☐ = miss

Runtime Log

☐ Enabled

Tool Control

Disconnect from MIPS Reset Close

Assignment 2:

Run the program in the cache simulator and study how the instruction cache works.

Then give full answers to the following questions.

- How is the full 32-bit address used in the cache memory?
 - Lấy địa chỉ 32 bit, dịch trái 2 bit, và lưu vào phần tag của block cache.
- What happens when there is a cache miss?
 - Khi bộ xử lý không tìm thấy vị trí bộ nhớ, trong bộ nhớ cache thì bộ nhớ cache đã miss.
 - Khi lỗi bộ nhớ cache, bộ đệm sẽ phân bổ một mục nhập mới rồi sao chép dữ liệu từ bộ nhớ chính, sau đó yêu cầu được thực hiện từ nội dung của bộ nhớ cache.
- What happens when there is a cache hit?

- Khi bộ xử lý nhận thấy rằng vị trí bộ nhớ nằm trong bộ nhớ cache, thì dữ liệu sẽ được đọc từ bộ nhớ cache

■ What is the block size?

- Bộ nhớ cache được chia làm nhiều block. Block size là kích thước của 1 block trên cache.

■ What is the function of the tag?

- Tag là phần để lưu trữ địa chỉ của dữ liệu.

Assignment 3:

The parameters of the cache memory can be changed to test the effects of different cases.

Investigate the effects of different parameter settings.

■ Explain the following: cache size, block size, number of sets, write policy and replacement policy.

- Kích thước của bộ nhớ đệm (cache size) là lượng dữ liệu bộ nhớ chính mà nó có thể chứa. Kích thước này có thể được tính bằng số byte được lưu trữ trong mỗi khối dữ liệu nhân với số khối được lưu trữ trong bộ nhớ đệm.
- Block size là kích thước các khối chia ra từ bộ nhớ đệm. VD: Nếu tăng kích thước khối trong khi vẫn giữ nguyên kích thước bộ nhớ cache, thì sẽ giảm số khối mà bộ nhớ đệm có thể chứa.
- Bộ nhớ đệm được chia thành các nhóm khối (block), được gọi là sets.
- Write policy and replacement policy: Bộ đệm cải thiện hiệu suất bằng cách giữ các mục dữ liệu gần đây hoặc thường sử dụng ở các vị trí bộ nhớ truy cập nhanh hơn hoặc rẻ hơn về mặt tính toán so với các bộ nhớ lưu trữ thông thường. Khi bộ nhớ đệm đầy, thuật toán phải chọn mục nào cần loại bỏ để nhường chỗ cho các mục mới.

■ If a cache is large enough that all the code within a loop fits in the cache, how many cache misses will there be during the execution of the loop? Is this good or bad?

- Số lần miss = Số lần đọc và ghi vào 1 địa chỉ mới
- Nếu như số lần đọc, ghi vào cùng 1 địa chỉ nhiều thì đây là điều tốt.

■ What should the code look like that would benefit the most from a large block size?

- Với blocksize lớn hơn, ta có thể lưu nhiều hơn trên 1 block nhưng số block bị giảm đi.