

Instituto Tecnológico y de Estudios Superiores de Occidente

Organización y Arquitectura de Computadoras



Práctica 2

Juego de la serpiente en RIPES

Profesor: Juan Pablo Ibarra Esparza

Integrantes:

Zamora Vargas Luis Santiago	735430
Abraham de León Gutiérrez	739343

SNAKE

Con esta práctica se planea implementar un juego de snake en el simulador de RIPES, el cual emula la arquitectura de RISC-V.

¿Qué hace?

El código es bastante sencillo, se divide en funciones para poder organizar mejor todo:

1. Primero se limpia el tablero apagando todos los leds
2. Posteriormente se prenden los dos leds de la serpiente, además de que hacemos que head y tail apunten a los inicios del arreglo, donde guardamos las direcciones de memoria de estas partes.
3. Imprimimos los límites marcados en un color beige, se hacen dos ciclos donde cada ciclo llena la width y height del color.
4. Se genera una manzana de una forma pseudorandom, ya que no se podía usar time.h, se hace un ciclo while donde se garantiza que no se genere una manzana en el borde o sobre la serpiente.
5. Se comienza con el juego en un while determinado por la variable game, que se convierte en falso cuando hay colisión.
6. Se comienza a mover la serpiente con la lógica de un array circular donde la head y la tail van avanzando, además se verifican colisiones y si se come la manzana entramos a eat apple.
7. En eatapple, hacemos que la manzana cambie al color de la head y que en este caso no se haga nada con la tail, con eso crece la serpiente.
8. Para detectar los dpads, simplemente manejamos dx y dy, (0, -1) arriba, (0, 1) abajo, (-1, 0) izquierda, (1, 0), derecha.
9. Al finalizar el juego limpiamos de nuevo el tablero.

```
#include "ripes_system.h"
#include <stdio.h>
#include <stdlib.h>

#define SW0 (0x01)
#define SW1 (0x02)
#define SW2 (0x04)
#define SW3 (0x08)
#define SW4 (0x10)
#define SW5 (0x20)
#define SW6 (0x40)
#define SW7 (0x80)

#define LED_MATRIX_0_SIZE (0xdac)
#define LED_MATRIX_0_WIDTH (0x23)
#define LED_MATRIX_0_HEIGHT (0x19)

volatile unsigned int * led_base = (volatile unsigned int *)LED_MATRIX_0_BASE;
```

```

volatile unsigned int * d_pad_up = (volatile unsigned int *)D_PAD_0_UP;
volatile unsigned int * d_pad_do = (volatile unsigned int *)D_PAD_0_DOWN;
volatile unsigned int * d_pad_le = (volatile unsigned int *)D_PAD_0_LEFT;
volatile unsigned int * d_pad_ri = (volatile unsigned int *)D_PAD_0_RIGHT;

#define MAX_SNAKE_SIZE (LED_MATRIX_0_WIDTH * LED_MATRIX_0_HEIGHT)

int game = 1;
int head = 1;
int tail = 0;

int randcounter = 50;

int size = 2;

int new_head_index;

int dx = 0;
int dy = 1;

volatile unsigned int snakeLEDs[MAX_SNAKE_SIZE];

void initSnake() {
    int st1 = (LED_MATRIX_0_WIDTH+2);
    int st2 = (LED_MATRIX_0_WIDTH*2+2);
    *(led_base+st1) = 0x00FF00;
    *(led_base+st2) = 0x00FF00;
    snakeLEDs[tail] = st1;
    snakeLEDs[head] = st2;
}

int moveSnake() {
    if (dy == 1) {
        new_head_index = (snakeLEDs[head] + LED_MATRIX_0_WIDTH);
    } else if (dy == -1) {
        new_head_index = (snakeLEDs[head] - LED_MATRIX_0_WIDTH);
    } else if (dx == 1) {
        new_head_index = (snakeLEDs[head] + 1);
    } else if (dx == -1) {
        new_head_index = (snakeLEDs[head] - 1);
    }
    //The game ends when in contact with the colors of the edges of the map or the snake
    itself
    if (*(led_base+new_head_index)==0xF9F6B9 || *(led_base+new_head_index)==0x00FF00 ) {
        return 0;
    } else if (*(led_base+new_head_index)==0xFF0000) {
        eatApple();
        generateApple();
        return 1;
    }
    head = (head + 1) % MAX_SNAKE_SIZE;
    snakeLEDs[head] = new_head_index;
    *(led_base + new_head_index) = 0x00FF00;

```

```

    *(led_base + snakeLEDs[tail]) = 0x0;
    tail = (tail + 1) % MAX_SNAKE_SIZE;
    return 1;
}

void printlimits(int color) {
    for (int i = 0; i < LED_MATRIX_0_WIDTH+1; i++)
    {
        *(led_base+i)=color;
        *(led_base+(LED_MATRIX_0_HEIGHT-1) * LED_MATRIX_0_WIDTH + i) = color;
    }
    for (int i = 0; i < LED_MATRIX_0_HEIGHT; i++) {
        *(led_base + i * LED_MATRIX_0_WIDTH) = color;
        *(led_base + i * LED_MATRIX_0_WIDTH-1) = color;
    }
}

void cleanBoard() {
    for (int i = 0; i < MAX_SNAKE_SIZE; i++) {
        *(led_base+i) = 0x0;
    }
}

void changeDirection(int dex, int dey) {
    dx = dex;
    dy = dey;
}

void eatApple() {
    head = (head + 1) % MAX_SNAKE_SIZE;
    snakeLEDs[head] = new_head_index;
    *(led_base + new_head_index) = 0x00FF00;
}

void generateApple() {
    int flag = 1;
    while (flag){
        randcounter += 5;
        srand(randcounter);
        int random_x = 2 + rand() % (LED_MATRIX_0_WIDTH -4);
        int random_y = 2 + rand() % (LED_MATRIX_0_HEIGHT -4);
        int position = random_y * LED_MATRIX_0_WIDTH + random_x;
        if (*(led_base+position) != 0xF9F6B9 || *(led_base+position) != 0x00FF00) {
            *(led_base+position) = 0xFF0000;
            flag = 0;
        }
    }
}

void main() {
    cleanBoard();
    initSnake();
    printlimits(0xF9F6B9);
    generateApple();
}

```

```

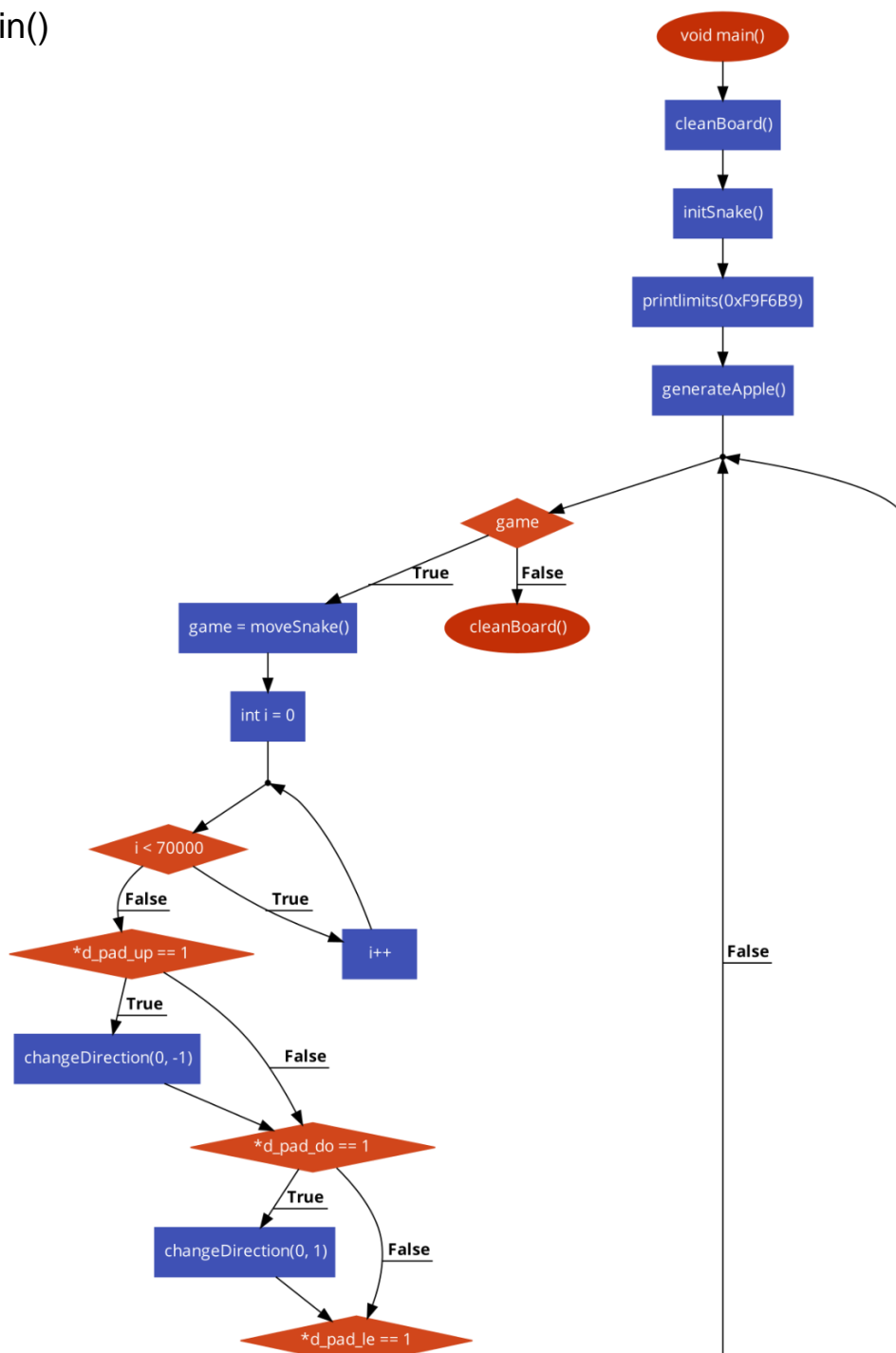
while (game){
    game = moveSnake();
    for (int i = 0; i < 10000; i++) {

    }
    if(*d_pad_up == 1) changeDirection(0, -1);
    if(*d_pad_do == 1) changeDirection(0, 1);
    if(*d_pad_le == 1) changeDirection(-1, 0);
    if(*d_pad_ri == 1) changeDirection(1, 0);
}
cleanBoard();
}

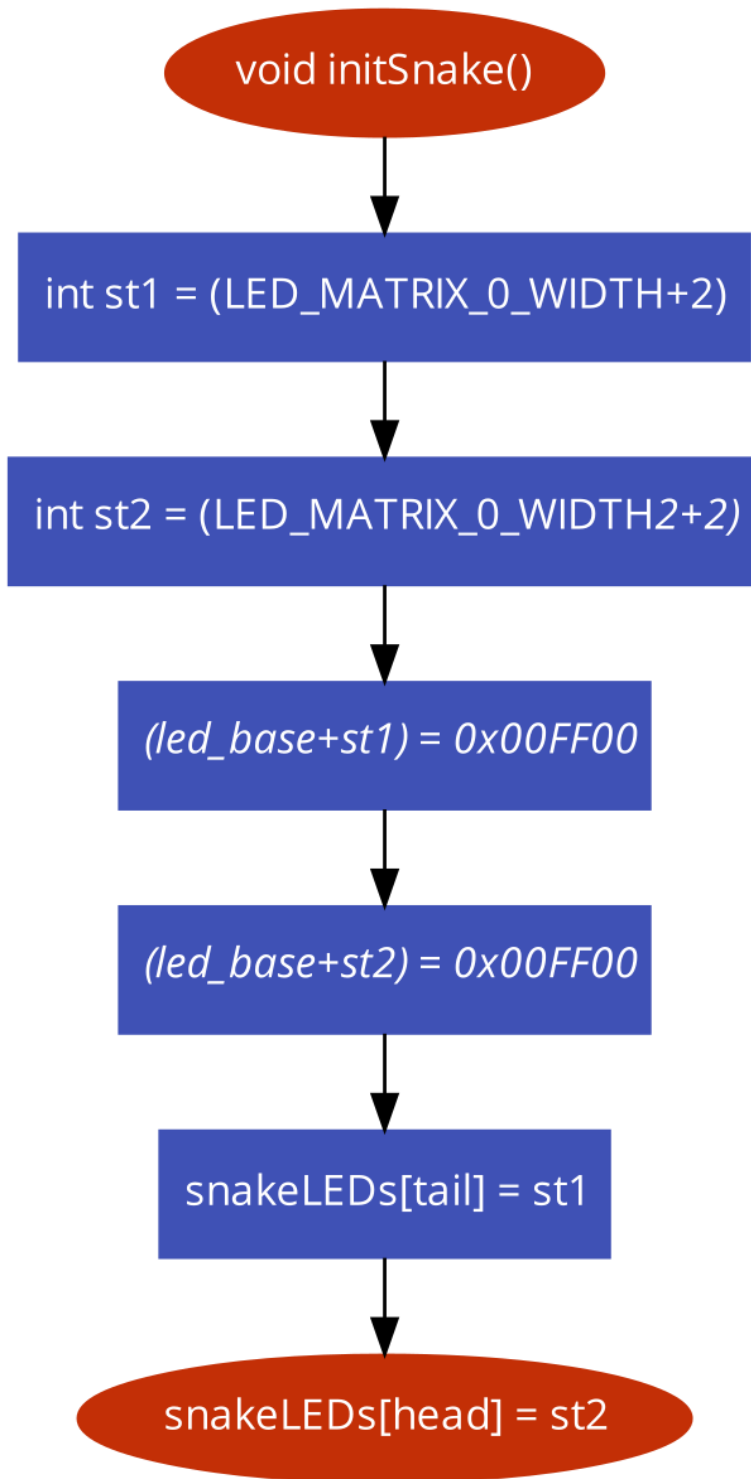
```

DIAGRAMA DE FLUJO

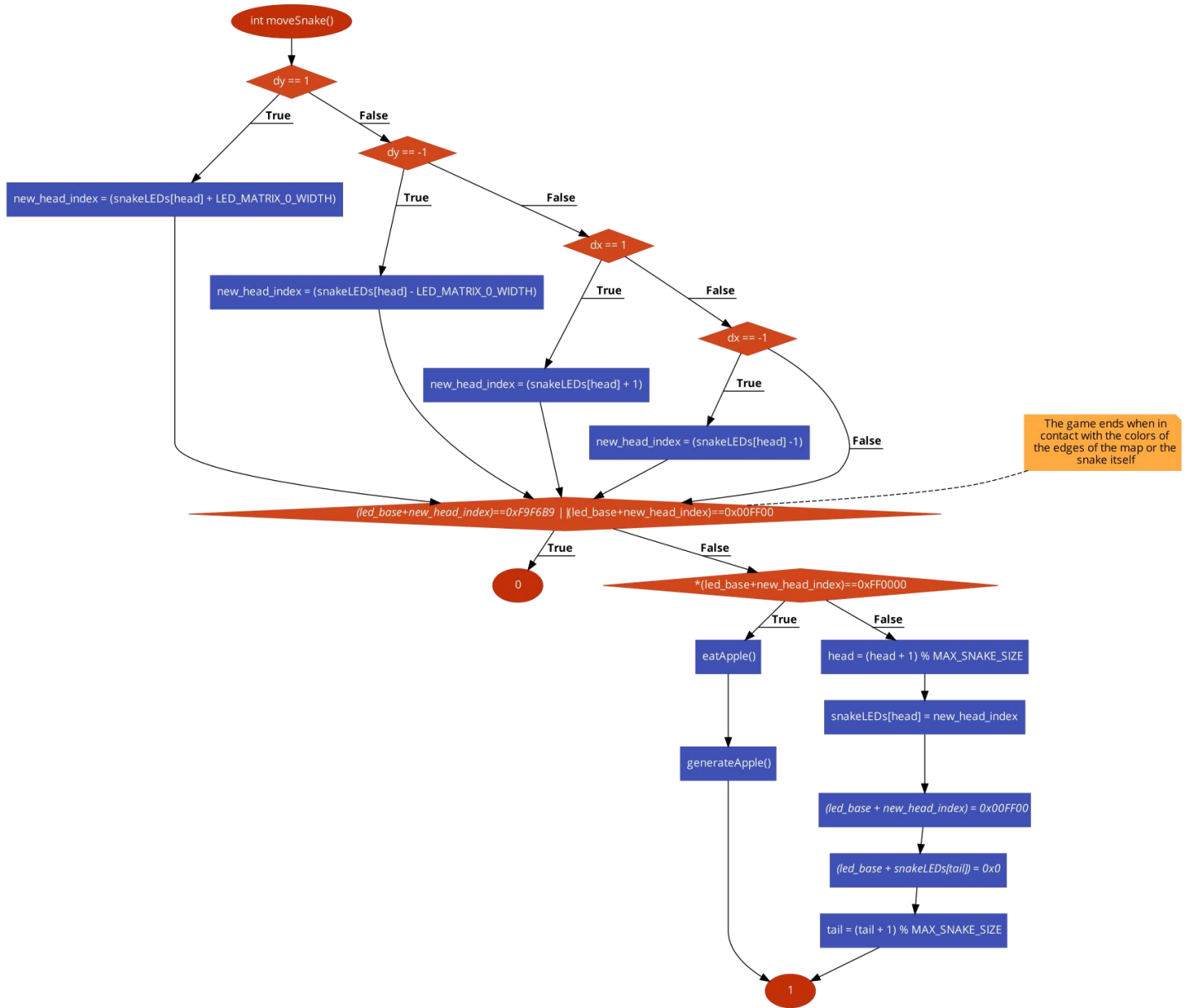
Main()



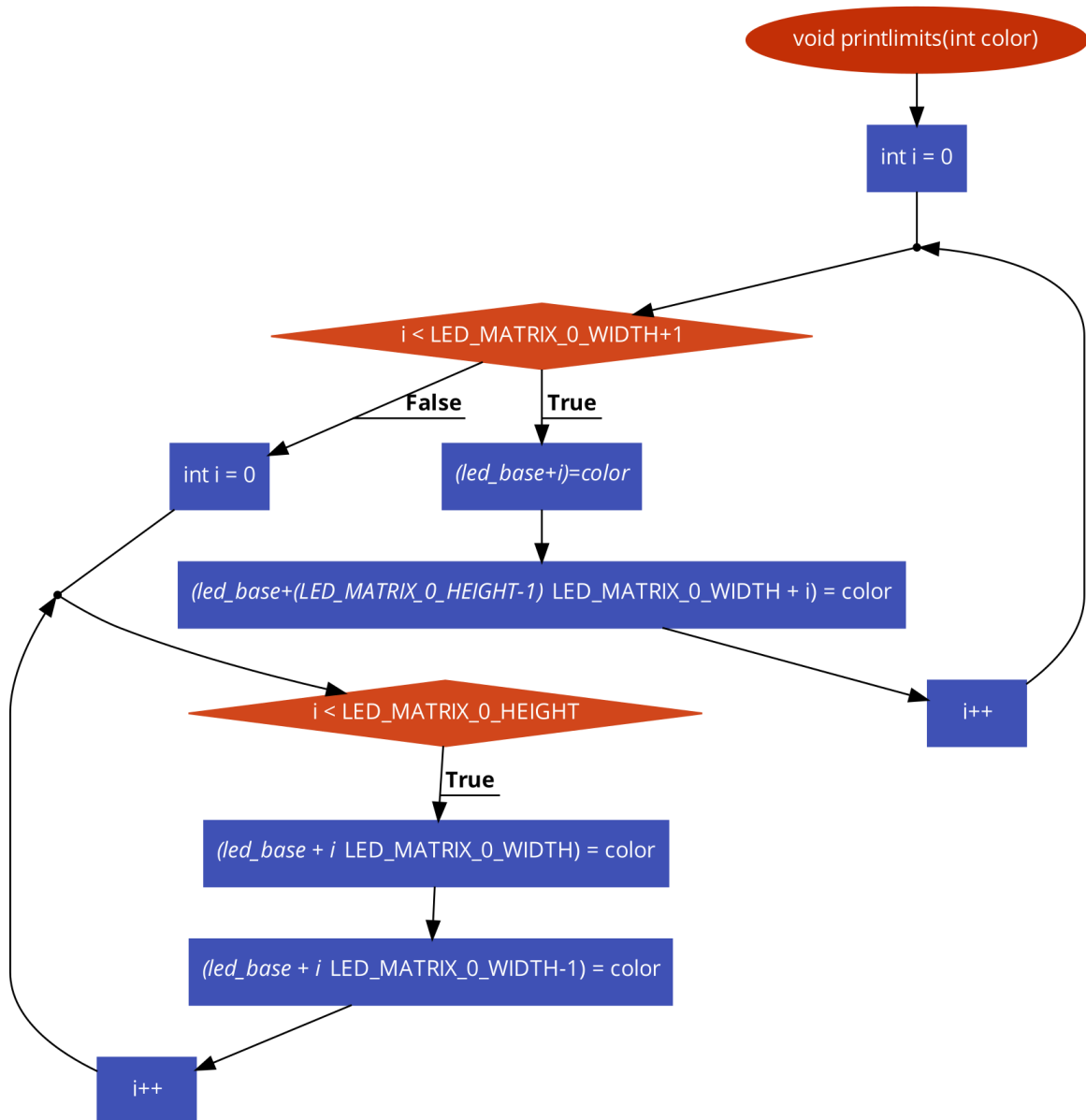
initSnake()



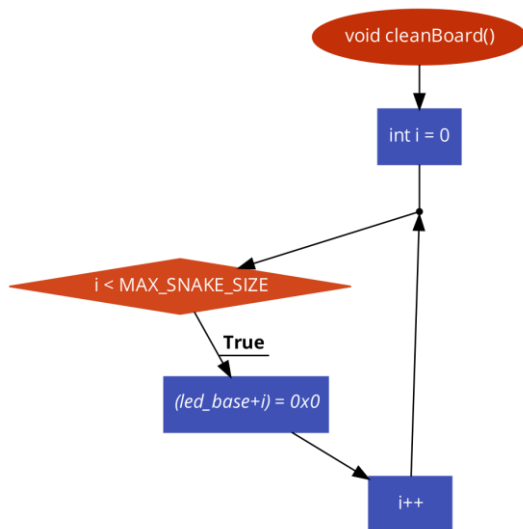
moveSnake()



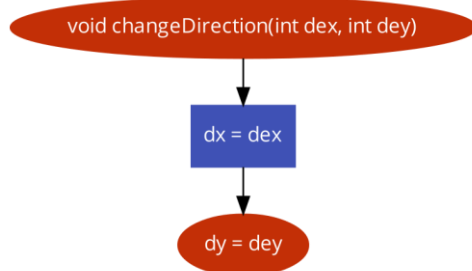
printLimits()



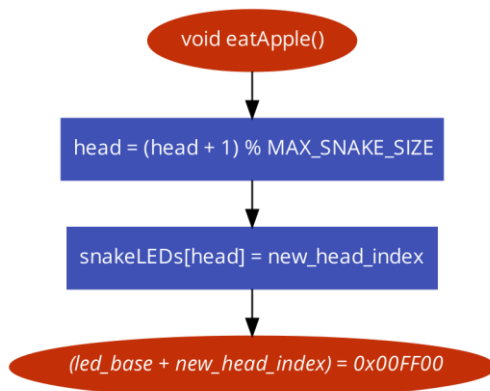
cleanBoard()



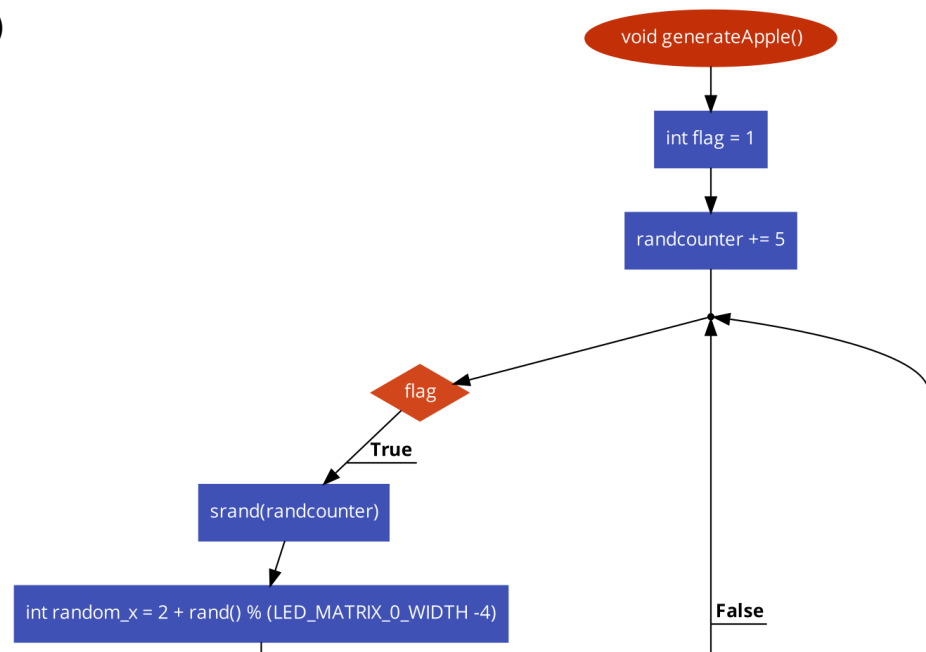
changeDirection()



eatApple()





generateApple()

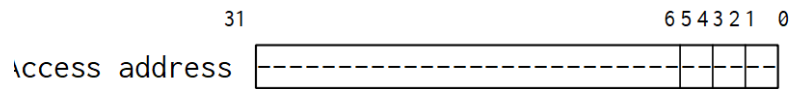


CACHÉ

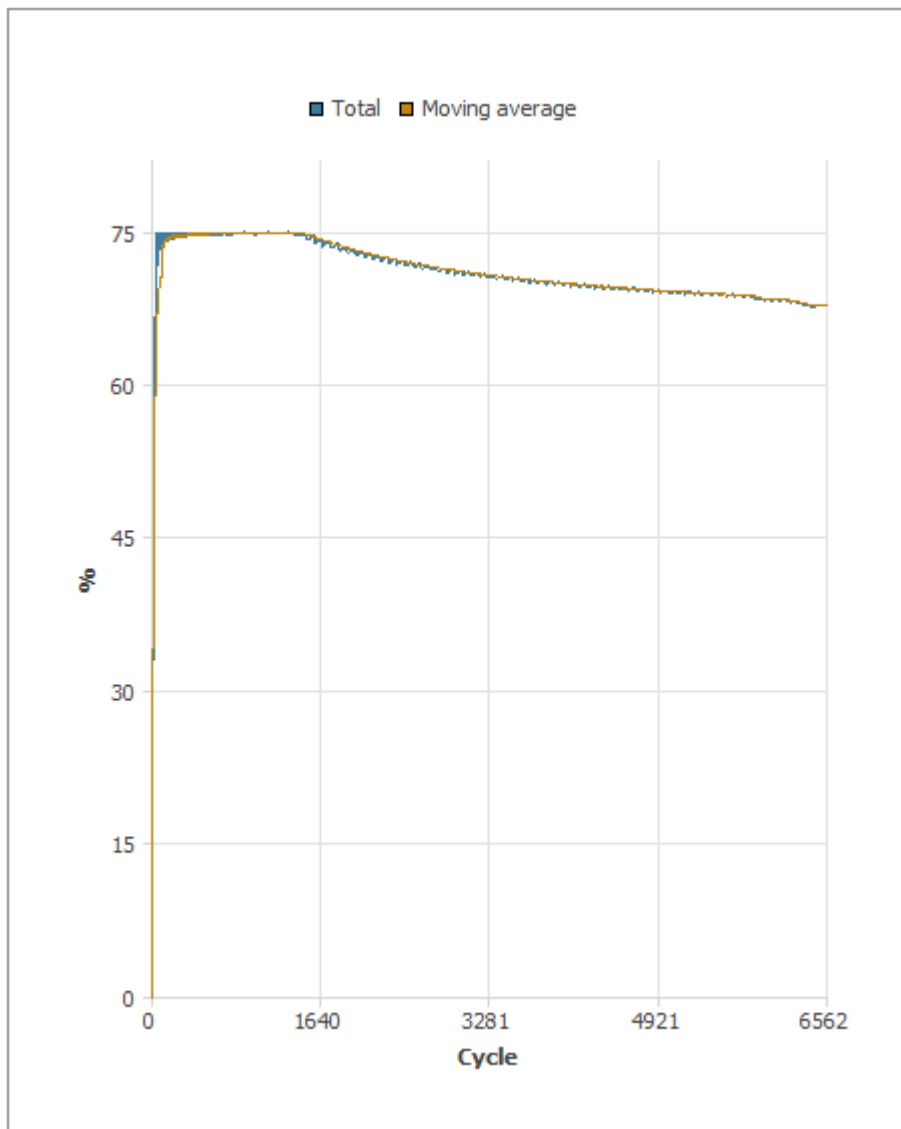
En base a las siguientes configuraciones de caché, ¿Cuál es el hit rate de cada configuración?

- Mapeo directo 4 líneas

L1 Data Cache		L1 Instr. Cache	
Cache configuration:			
Preset:		 	
2 ^N Lines:	2	Repl. policy:	LRU
2 ^N Ways:	0	Wr. hit:	Write-back
2 ^N Words/Line:	2	Wr. miss:	Write allocate
Plot configuration:		Statistics:	
Numerator	Hits	Size (bits):	624
Denominator	Access count	Hit rate:	0.6628
<input checked="" type="checkbox"/> Ratio		Writebacks:	1219
<input checked="" type="checkbox"/> Moving avg.	50 cyc.	Hits:	3563
		Misses:	1813



Index	V	D	Tag	Word 0	Word 1	Word 2	Word 3
0	1	0	0x0000046a	0x00000000	0x00000000	0x00000000	0x00000000
1	1	0	0x01ffffff	0x00000000	0x00000000	0x00000000	0x000105e0
2	1	0	0x0000046e	0x00000000	0x000105fc	0x00000000	0x00000000
3	1	0	0x00000479	0x00000000	0x00000000	0x00011a50	0x00000000





HR: 0.6628


- Asociativa con 2 conjuntos (lines) y 2 vías (ways)


L1 Data Cache


L1 Instr. Cache


Cache configuration:


Preset:  


2ⁿ Lines: 

2ⁿ Ways: 


2ⁿ Words/Line: 


Repl. policy: LRU 

Wr. hit: Write-back 


Wr. miss: Write allocate 

Plot configuration:


Numerator: Hits 

Denominator: Access count 

☒ Ratio

☒ Moving avg. 

Statistics:

Size (bits): 

Hit rate: Writebacks:

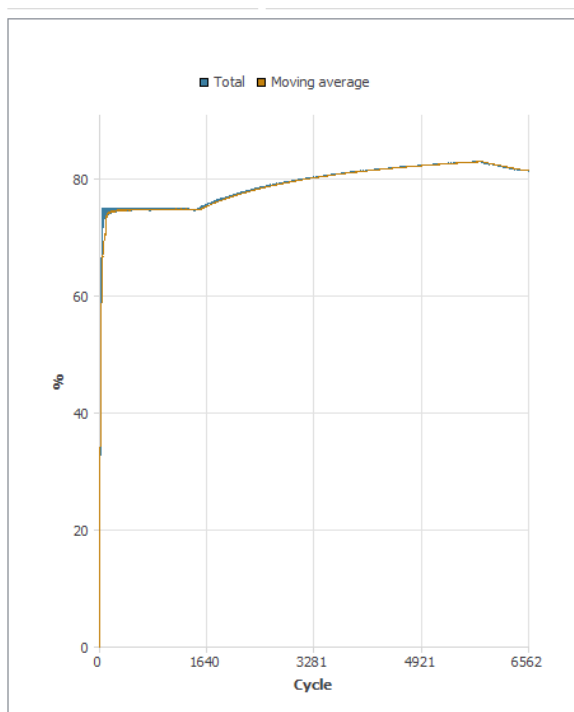
Hits: Misses:

31

5 4 3 2 1 0

Access address

Index	V	D	LRU	Tag	Word 0	Word 1	Word 2	Word 3
0	1	0	0	0x000008f4	0x00000000	0x00000000	0x00011a60	0x00000000
	1	0	1	0x03ffffffe	0x00000000	0x00000000	0x00000000	0x00000000
1	1	0	1	0x03ffffffd	0x00000000	0x00000000	0x00000000	0x00000000
	1	0	0	0x000008d4	0x00000000	0x00000000	0x00000000	0x00000000



HR: 0.808

- Totalmente asociativa con 4 vías (ways)

L1 Data Cache L1 Instr. Cache

Cache configuration:

Preset: H X

2nd Lines: 0 Repl. policy: LRU

2nd Ways: 2 Wr. hit: Write-back

2nd Words/Line: 2 Wr. miss: Write allocate

Plot configuration:

Numerator: Hits

Denominator: Access count

☒ Ratio

☒ Moving avg. 50 cyc.

Statistics:

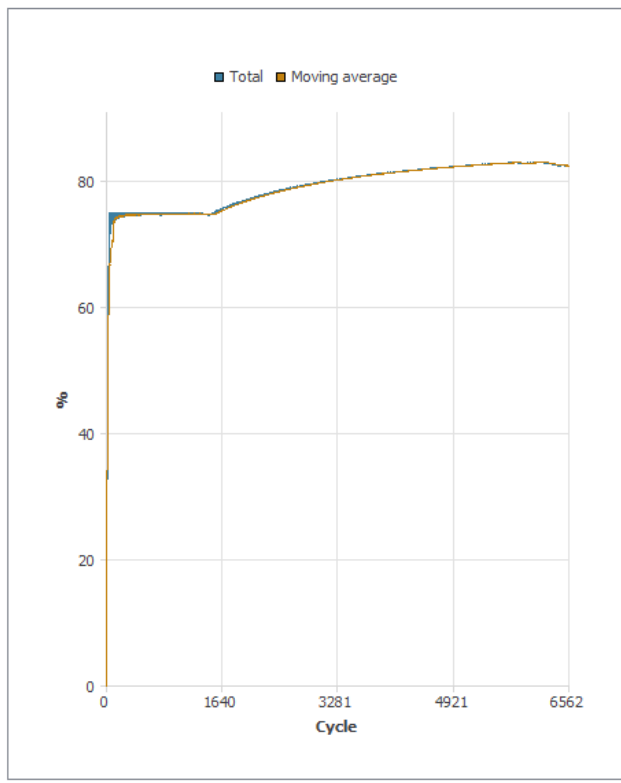
Size (bits): 640 ?

Hit rate: 0.8112 Writebacks: 877

Hits: 4361 Misses: 1015

Access address 31 4 3 2 1 0

Index	V	DLRU	Tag	Word 0	Word 1	Word 2	Word 3
1	0	0	0x000011a9	0x00000000	0x00000000	0x00000000	0x00000000
1	0	3	0x07fffffc	0x00000000	0x00000000	0x00000000	0x00000000
1	0	1	0x000011e8	0x00000000	0x00000000	0x00011a60	0x00000000
1	0	2	0x07fffffb	0x00000000	0x00000000	0x00000000	0x00000000



HR: 0.8112

- Maepo directo 16 líneas

L1 Data Cache
L1 Instr. Cache

Cache configuration:

Preset:

2^N Lines: 4
2^N Ways: 0
2^N Words/Line: 2
Repl. policy: LRU
Wr. hit: Write-back
Wr. miss: Write allocate

Plot configuration:

Numerator: Hits
Denominator: Access count
☒ Ratio
☒ Moving avg. 50 cyc.

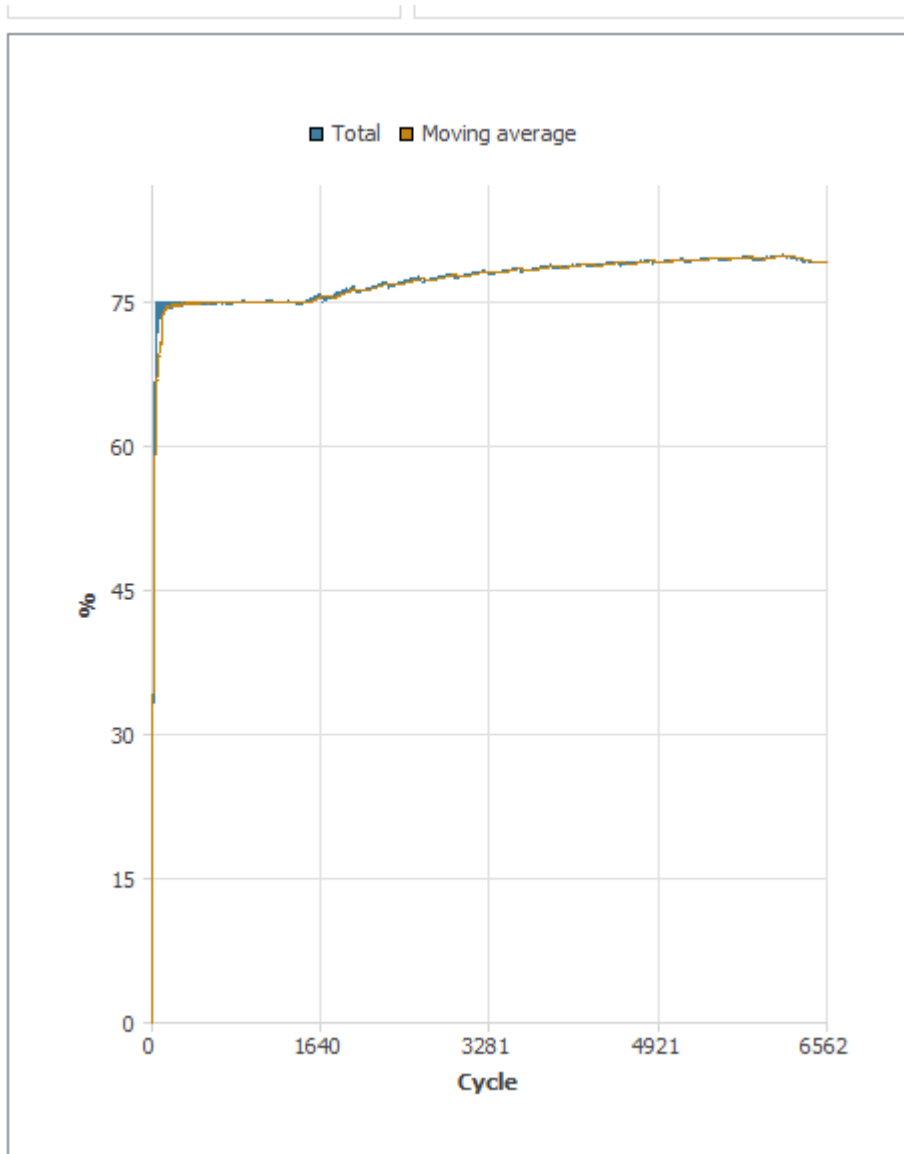
Statistics:

Size (bits): 2464
Hit rate: 0.8101
Writebacks: 855
Hits: 4355
Misses: 1021

31 8 7 4 3 2 1 0

ess address

Index	V	D	Tag	Word 0	Word 1	Word 2	Word 3
0	1	1	0x00f0000d	0x00000000	0x00000000	0x00000000	0x00000000
1	1	1	0x00f0000d	0x00000000	0x00000000	0x00000000	0x00000000
2	1	1	0x00f0000d	0x00000000	0x00000000	0x00000000	0x00000000
3	1	0	0x0000011d	0x00000000	0x00000000	0x00000000	0x00000000
4	1	1	0x00f0000d	0x00000000	0x00000000	0x00000000	0x00000000
5	1	0	0x0000011a	0x00000000	0x00010074	0x00010124	0x000100d4
6	1	1	0x00f0000d	0x00000000	0x00000000	0x00000000	0x00000000
7	1	1	0x00f0000d	0x00000000	0x00000000	0x00000000	0x00000000
8	1	0	0x0000011e	0x00000000	0x00000000	0x00011a60	0x00000000
9	1	0	0x0000011a	0x00000000	0x00000000	0x00000000	0x00000000
10	1	0	0x0000011b	0x00000000	0x00000000	0x00011bac	0x00000000
11	1	0	0x007ffffff	0x00000000	0x00000000	0x00000000	0x00000000
12	1	0	0x007ffffff	0x00000000	0x00000000	0x00000000	0x00000000
13	1	1	0x007ffffff	0x00000000	0x00000000	0x00000000	0x000105f0
14	1	1	0x007ffffff	0x00000000	0x00000000	0x00000000	0x000100d0
15	1	1	0x00f0000c	0x00000000	0x00000000	0x00000000	0x00000000



HR: 0.8101

- Asociativa con 4 conjuntos (lines) y 4 vías (ways)

L1 Data Cache

L1 Instr. Cache

Cache configuration:

Preset:

2^N Lines:

2

Repl. policy:

LRU

2^N Ways:

2

Wr. hit:

Write-back

2^N Words/Line:

2

Wr. miss:

Write allocate

Plot configuration:

Numerator

Hits

Denominator

Access count

☒ Ratio

☒ Moving avg.

50 cyc.

Statistics:

Size (bits):

2528

Hit rate:

0.857

Writebacks:

745

Hits:

4607

Misses:

769

31

6

5

4

3

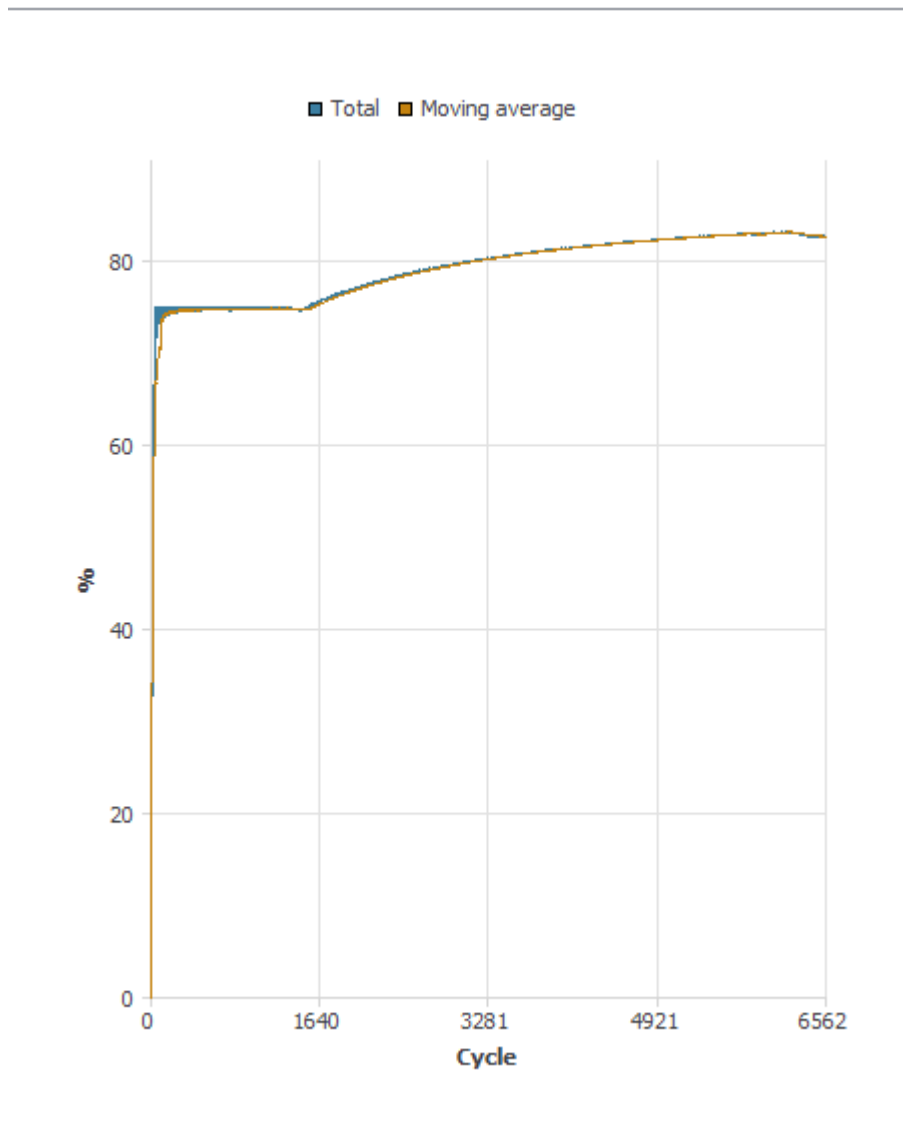
2

1

0

ccess address

Index	V	D	LRU	Tag	Word 0	Word 1	Word 2	Word 3
0	1	0	0	0x0000047a	0x00000000	0x00000000	0x00011a60	0x00000000
	1	1	3	0x03c00036	0x00000000	0x00000000	0x00000000	0x00000000
	1	1	1	0x01ffffff	0x00000000	0x00000000	0x00000000	0x00000000
	1	1	2	0x0000047b	0x00000015	0x00000001	0x00000000	0x00000000
1	1	0	0	0x0000046a	0x00000000	0x00000000	0x00000000	0x00000000
	1	1	1	0x01ffffff	0x00000000	0x00000000	0x00000000	0x000105f0
	1	1	3	0x03c00036	0x00000000	0x00000000	0x00000000	0x00000000
	1	0	2	0x00000469	0x00000000	0x00010074	0x00010124	0x000100d4
2	1	0	0	0x0000046e	0x00000000	0x00000000	0x00011bac	0x00000000
	1	1	1	0x01fffffe	0x00000000	0x00011bb0	0x00000000	0x000109bc
	1	1	2	0x01ffffff	0x00000000	0x00000000	0x00000000	0x000100d0
	1	1	3	0x03c00036	0x00000000	0x00000000	0x00000000	0x00000000
3	1	1	0	0x01fffffe	0x00000000	0x00000000	0x00000000	0x00000000
	1	1	1	0x0000046e	0x00000000	0x0001060c	0x00000000	0x00000000
	1	0	2	0x00000474	0x00000000	0x00000000	0x00000000	0x00000000
	1	1	3	0x03c00036	0x00000000	0x00000000	0x00000000	0x00000000





HR: 0.857




- Totalmente asociativa con 16 vías




L1 Data Cache




L1 Instr. Cache

Cache configuration:


Preset:  


2^N Lines: 0   Repl. policy: LRU 

2^N Ways: 4   Wr. hit: Write-back 



2^N Words/Line: 2   Wr. miss: Write allocate 

Plot configuration:


Numerator: Hits 

Denominator: Access count 

☒ Ratio

☒ Moving avg. 50 cyc.  

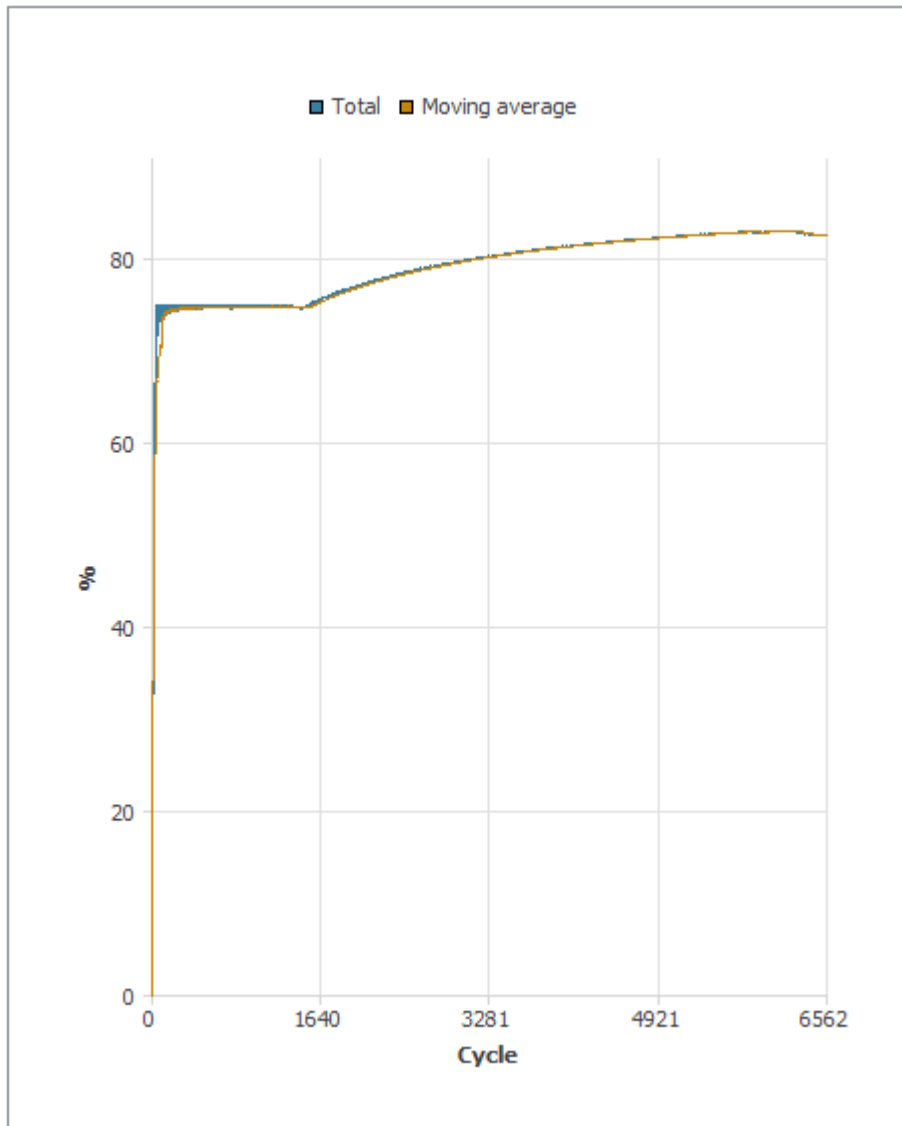
Statistics:

Size (bits): 2592 

Hit rate: 0.8566 Writebacks: 748

Hits: 4605 Misses: 771

Index	V	D	LRU	Tag	Word 0	Word 1	Word 2	Word 3
1	0	9		0x000011a5	0x00000000	0x00010074	0x00010124	0x000100d4
1	1	7		0x07ffffffa	0x00000000	0x00011bb0	0x00000000	0x000109bc
1	1	12		0x0f0000db	0x00000000	0x00000000	0x00000000	0x00000000
1	1	3		0x07ffffffc	0x00000000	0x00000000	0x00000000	0x00000000
1	0	0		0x000011a9	0x00000000	0x00000000	0x00000000	0x00000000
1	0	13		0x000011eb	0xf0000000	0xf0000014	0x00011a60	0x00000000
1	0	10		0x000011d3	0x00000000	0x00000000	0x00000000	0x00000000
1	1	14		0x0f0000da	0x00000000	0x00000000	0x00000000	0x00000000
1	0	1		0x000011e8	0x00000000	0x00000000	0x00011a60	0x00000000
1	1	6		0x000011bb	0x00000000	0x0001060c	0x00000000	0x00000000
1	1	8		0x000011ec	0x00000015	0x00000001	0x00000000	0x00000000
1	1	11		0x07ffffffe	0x00000000	0x00000000	0x00000000	0x000100d0
1	1	4		0x07ffffffd	0x00000000	0x00000000	0x00000000	0x000105f0
1	1	15		0x0f0000d9	0x00000000	0x00000000	0x00000000	0x00000000
1	0	5		0x000011ba	0x00000000	0x00000000	0x00011bac	0x00000000
1	1	2		0x07ffffffb	0x00000000	0x00000000	0x00000000	0x00000000



HR: 0.8566

¿Cuál de las siguientes configuraciones tiene el mejor hit rate?

La asociativa con 4 conjuntos (lines) y 4 vías (ways) y por poco la totalmente asociativa con 16 vías.