Tiendrebeogo Ismael Lionnel 20102656

Ming-Xia Delvas 20104038

Question 1

1. Les lignes ont 16 octets = 24 octets, il y a donc 4 bits d’offset (déplacement)

4K/16 = (4\*1024) /16 = 256 = 28 lignes, donc 8 bits d’index

Il reste 28-9-4 = 16 bits d’étiquettes (tag)

|  |  |  |
| --- | --- | --- |
| étiquette : 16 bits | index : 8 bits | depl : 4 bits |

1. 3CE4FE216 = 0011 1100 1110 0100 1111 1110 0010

On découpe selon le résultat obtenu à la question précédente

0011 1100 1110 0100 1111 1110 0010

étiquette index depl

|  |  |  |
| --- | --- | --- |
| étiquette : 3CE4 | index : FE | depl : 2 |

1. 3CE4FE1
2. 3CE4FE2(deja la)
3. 3CE4FE3
4. 3CE4FE4
5. 3CE4FE5
6. 3CE4FE6
7. 3CE4FE7
8. 3CE4FE8
9. 3CE4FE9
10. 3CE4FEA
11. 3CE4FEB
12. 3CE4FEC
13. 3CE4FED
14. 3CE4FEE
15. 3CE4FEF

Question 2

1. Nombre de mots = 4 bits = 24 = 16 mots

Temps moyen de changement pour une ligne = 50 + 15 \* 10 = 200 ns pour 16 mots

1. Tmoyen = h\*Tc+(1-h)\*TM

Tmoyen = 0.8\*4+(1-0.8)\*(4+200) = 44 ns

1. Écriture différée

Question 3

1. x - (y\*((z\*y) - (x\*z)))

-

/ \

x \*

/ \

y -

/ \

/ \

/ \

\* \*

/ \ / \

z y x z

1. Représentation en notation postfixe : x y z y \* x z \* - \* -

Question 4

1. F(x+G(y\*z), x+z, H(G(y+x), x\*y))

F

/ | \

/ | \

/ | \

/ | \

/ + \

+ / \ H

/ \ x z / \

/ \ / \

x G G \*

| | / \

\* + x y

/ \ / \

y z y x

En représentation postfixe : x y z \* G + x z + y x + G x y \* H F

1. READIN # 0 lit x

READIN # 1 lit y

READIN # 2 lit z

PUSH K

CALL 3

PROUT

HALT

# On calcule

# K(x, y, z) = F(x+G(y\*z), x+z, H(G(y+x), x\*y))

# x y z \* G + x z + y x + G x y \* H F en postfixe

K: PUSH 0(FP) # empile une copie de x

PUSH 1(FP) # empile une copie de y

PUSH 2(FP) # empile une copie de z

MUL # x y z \*

PUSH G # appel de G

CALL 1 #G a un argument

ADD # x y z \* G +

PUSH 0(FP) # empile une copie de x

PUSH 2(FP) # empile une copie de z

ADD # x y z \* G + x z +

PUSH 1(FP)

PUSH 0(FP)

ADD # x y z \* G + x z + y x +

PUSH G

CALL 1

PUSH 0(FP)

PUSH 1(FP)

MUL # x y z \* G + x z + y x + G x y \*

PUSH H # appel de H

CALL 2 # H a deux arguments

PUSH F

CALL 3

# le code pour le return de la fonction K est

POP 0(FP) # met la valeur de K dans 0(FP)

RETFORM 3 # K a trois parametres

# On calcule

# F(x, y, z) = 2\*x + 3\*y + 4\*z

# 2 x \* 3 y \* + 4 z \* + en postfixe

F: PUSH 2 # 2

PUSH 0(FP) # empile une copie de x

MUL # 2 x \*

PUSH 3 # 3

PUSH 1(FP) # empile une copie de y

MUL # 2 x \* 3 y

ADD

PUSH 4

PUSH 2(FP)

MUL

ADD # 2 x \* 3 y \* + 4 z \* +

POP 0(FP)

RETFORM 3 # F a 3 parametres

# On calcule

# G(x) = 5\*x

# 5 x \* en postfixe

G: PUSH 5 # 5

PUSH 0(FP) # empile une copie de x

MUL # 5 x \*

POP 0(FP)

RETFORM 1

# On calcule

# H(x, y) = x\*y

# x y \* en postfixe

H: PUSH 0(FP) # x

PUSH 1(FP) # y

MUL # x y \*

POP 0(FP)

RETFORM 2