

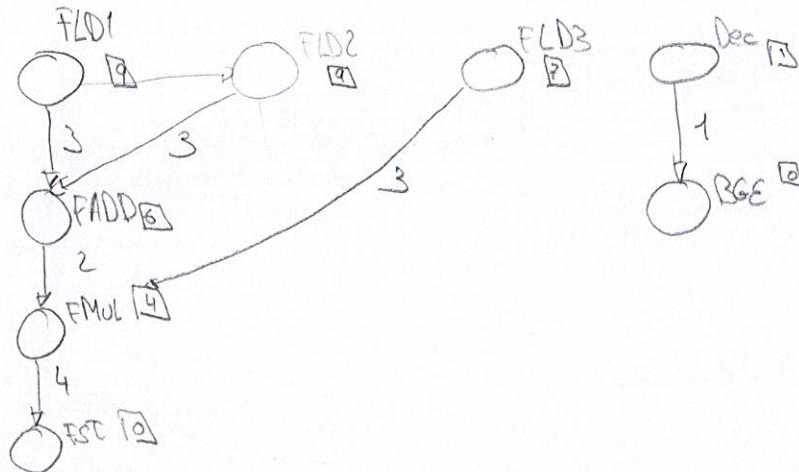
Konkurrenz-tellurisch Arbeitsteilung

Z. arithmetik

- a) Einheitscode behält den zuletzt eingesetzten 16 Bit Wortraum da.

FDI FD2	F+ FLD3	F*	FST DEC BGE X
-----------	-----------	----	---------------------

b)



FLD# F1, A(R4)

beg: FLD F2, B(R4)

FLD F4, C(R4)

Dec R4

FADD F3, F1, F2

FMUL F5, F3, F4

FST A+1(R4), F5

BGE R4, beg

FLD F1, A(R4)

beg

FLD1	FLD2	FLD3	Dec	F+	F*	FST	BGE	FLD1
------	------	------	-----	----	----	-----	-----	------

$$\text{Speed-up} = \frac{16}{12}$$

c)

FLD F1, A(R4)

FLD F2, B(R4)

FADD F3, F1, F2

FLD F4, C(R4)

FMUL F5, F3, F4

FST A(R4), F5

FLD F10 A-1(R4)

FLD F20 B-1(R4)

FADD F30 F10, F20

FLD F40, C-1(R4)

FMUL F50, F30, F40

FST A-1(R4), F50

Subi R4, R4, #2

BFE R4, beg

9- { FLD1, FLD2, FLD1', FLD2' }

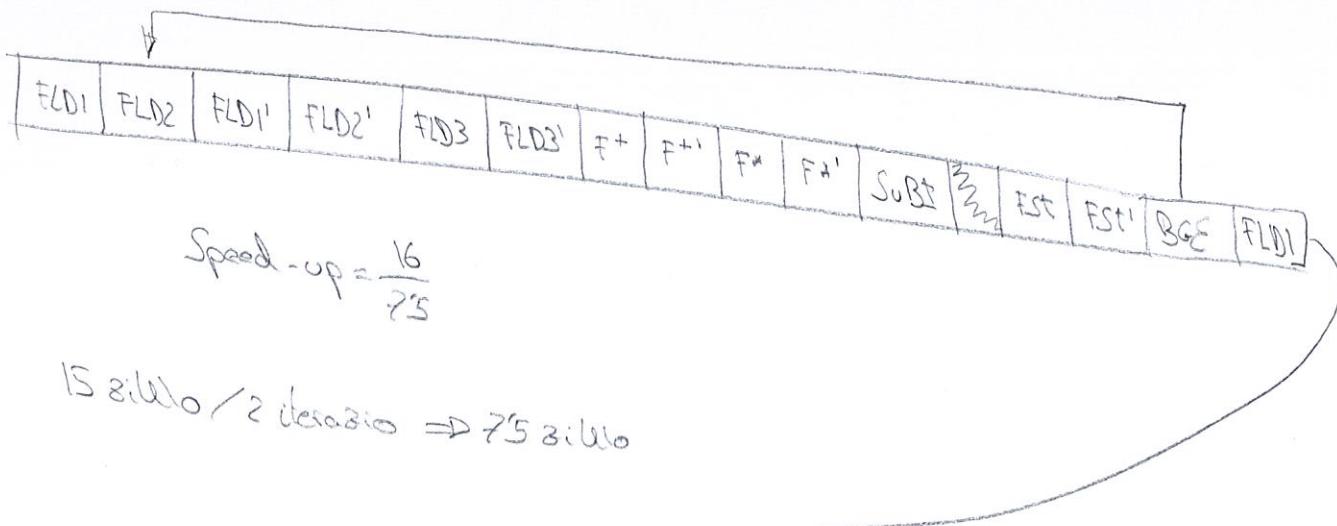
7- { FLD3, FLD3' }

6- { F+, F+1 }

4- { FA, FX }

2- { SUB }

0- { FST, BG/E, FST' }

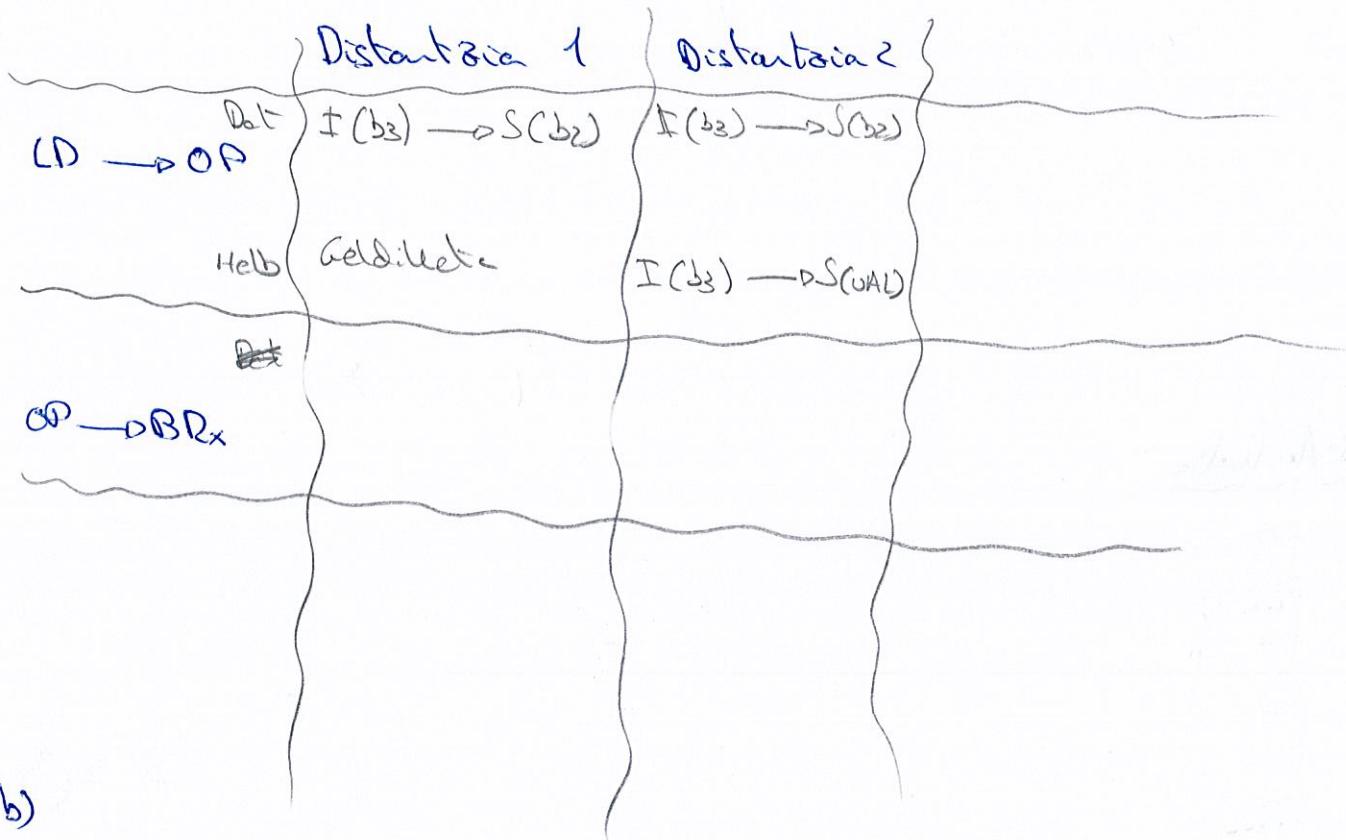


D_{deg}:
FLD1 F1, AC(R4)
FLD2 F2, AC(R4)
FLD1' F10, A-1(R4)
FLD2' F20 A-1(R4)
FLD F4, C(R4)
FLD F5, C-1(R4)
FADD F3, F1, F2
FADD F30, F10, F20
FMUL F5, F3, F4
FMUL F50, F30, F40

SUB R4, R4, H2
FST, A+2(R4), F5
FST, A+1(R4), F50
BG/E R4, deg
FLD F1, AC(R4)

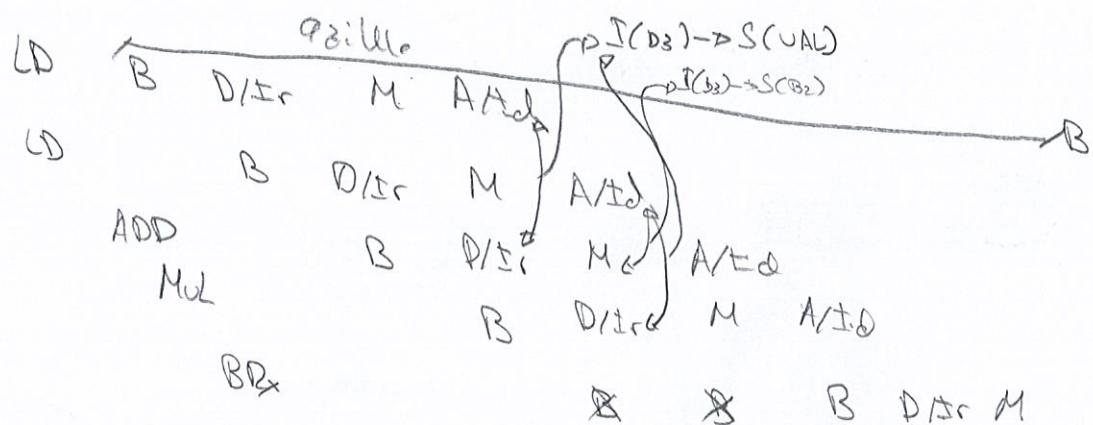
Z. Architektur

a)



b)

LD R₁ O(R₄)
 LD R₃ O(R₄)
 ADD R₅ O(R₁), R₃
 Mul R₄, O(R₁), R₂
 BR_x R₅, R₄, beg



Iterazione esecutata dalla qzillio

9)

820 BRx \rightarrow Java: anders.

820 } 2ziffer 920
} 1ziffer 960
0ziffer 910

12 ziffer

8-Anhänger

FLD 1

FLD 2

FDiv 4

FSt 8

FLD 9

FMUL 11

FLD 12

FADD 14

FSt 16

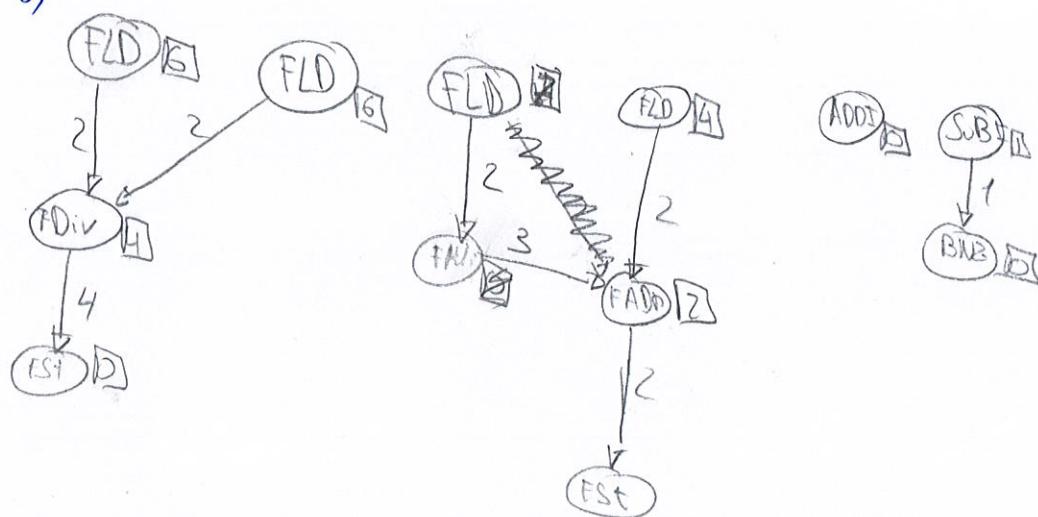
ADDI 17

SUBI 18

BNZ R2, beg 19

20 ziffer behält 3ten beziffer erledigt alle

b)



2. Gaten

Sarre

Adibidet

5ns 45ns 6ns 55ns 5ns

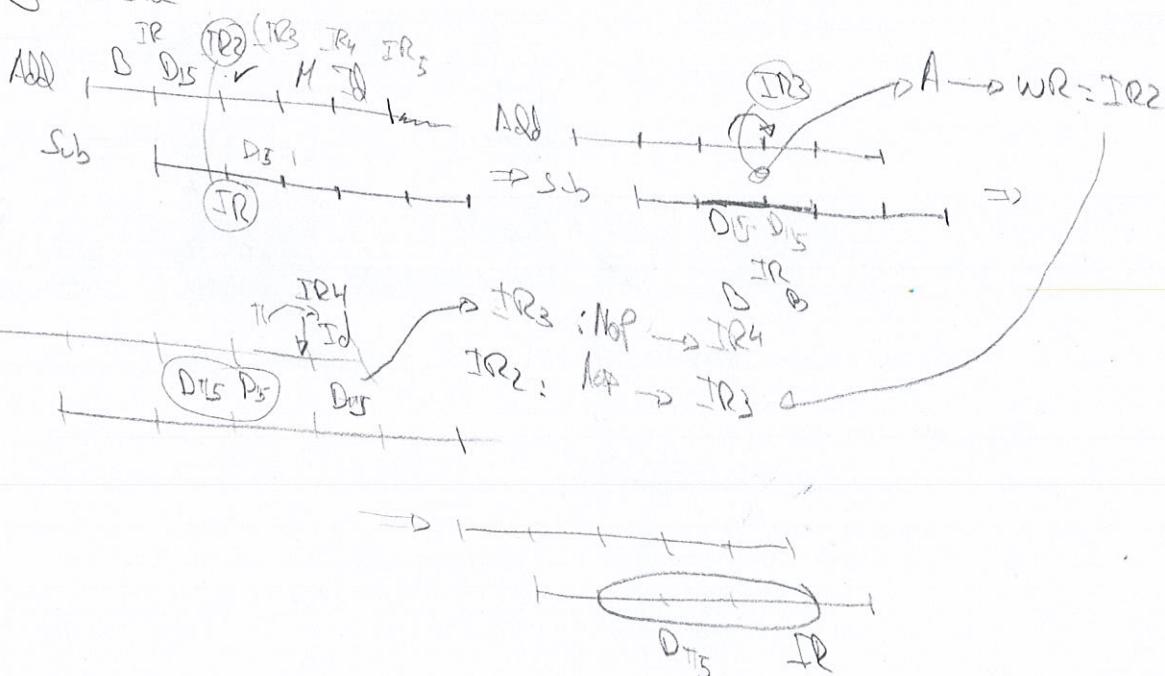
b)

6ns, 6ns, 6ns, 6ns, 6ns → $300\text{ns} + 5\text{ns} \cdot 5 = 325\text{ns}$

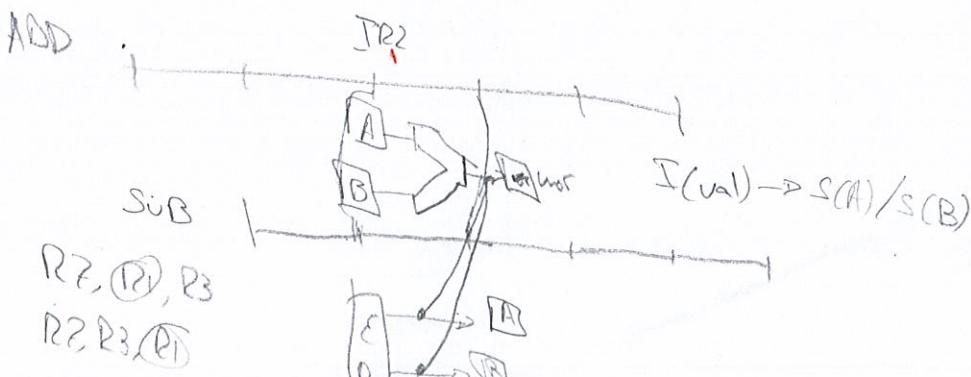
Rau

1)

geblüft



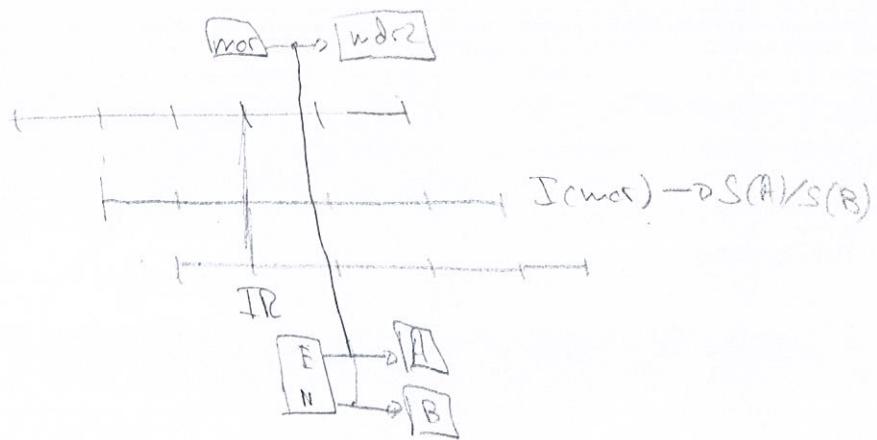
* Zirkulärbuffer:



ADD R1, R2, R3

Mul R8, R9, R10

Sub R7, R1, R3



ADD R1, R2, R3

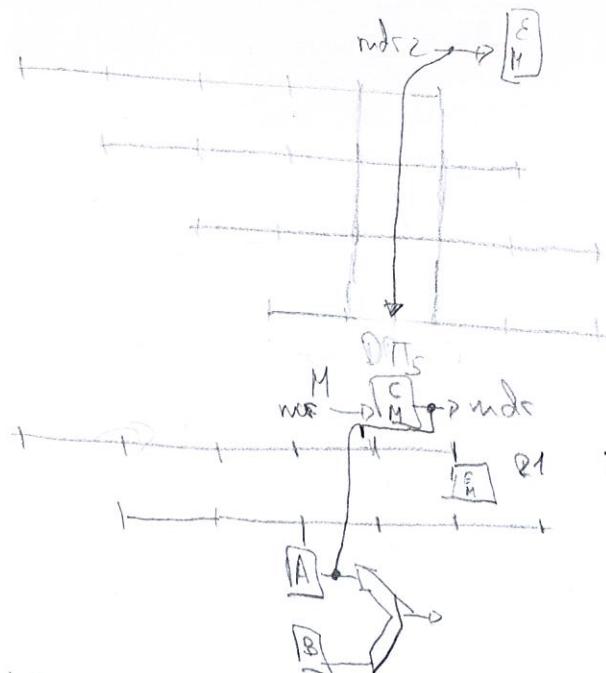
[Mul R8, R9, R10

Div R12, R3, R2

Sub R5, R3

LD R1, A(R2)

ADD R2, R1, R3



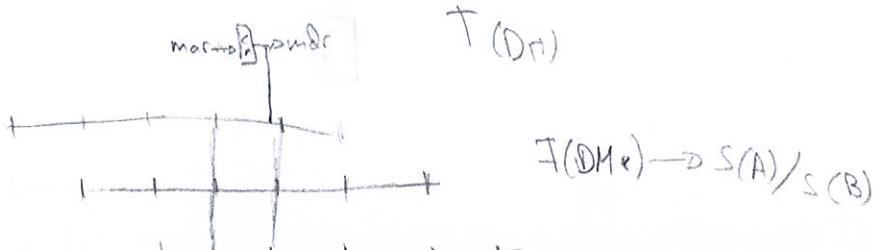
A creare registrul rezultat se ia valoarea sa se salveaza inregistrul de rezultat.

Bine ca din cauza debordarilor, UAL-ul are deosebita capacitate de a lucra cu egalele.

LD R1, A(R2)

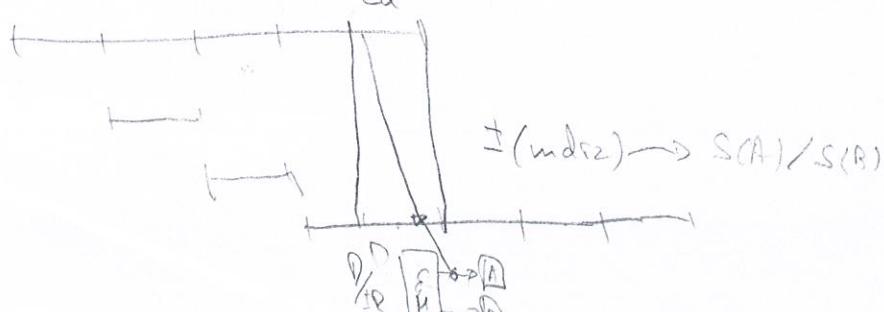
NOP

ADD R2, R1, R3



LD R1, B(R2)

—
—
ADD R2, R1, R3
R3, R1



① #include <stdio.h> // print & variables.

```

    int Main()
    {
        i=1;
        n=100;
        while (i < 100) {
            if (i % 2 == 0) {
                printf("%d\n", i*i);
            }
        }
        while (i <= n) {
            printf("%d\n", fib(i));
            i++;
        }
    }

```

②

```

for (j=0, j<10; j=j+4) K = k+j;
for (j=0; j < 10, j = j+4) K = k+j;
K = 0;
K = 0+4;
K = 4+8 = 12

```

⑥

```

void LogString (char S1[], char S2[])
{
    int k=0;
    int i=0;
    while (S2[k] != '\0') {k++;}
    for (int i=0; S1[i] != '\0'; i++) {
        S2[k+i] = S1[i];
    }
}

```

$$S2[k+i] = S1[i]$$

$$S2[k+i] = \text{null}$$



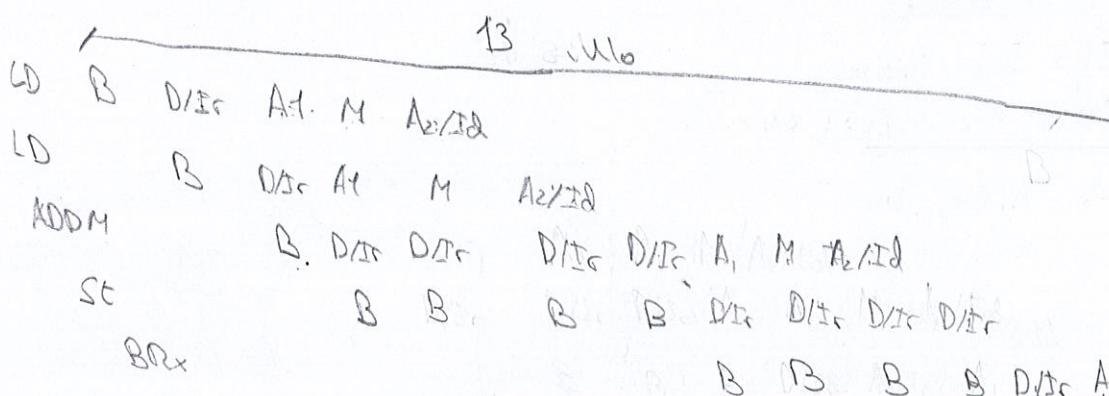
11. arkitetura

a)

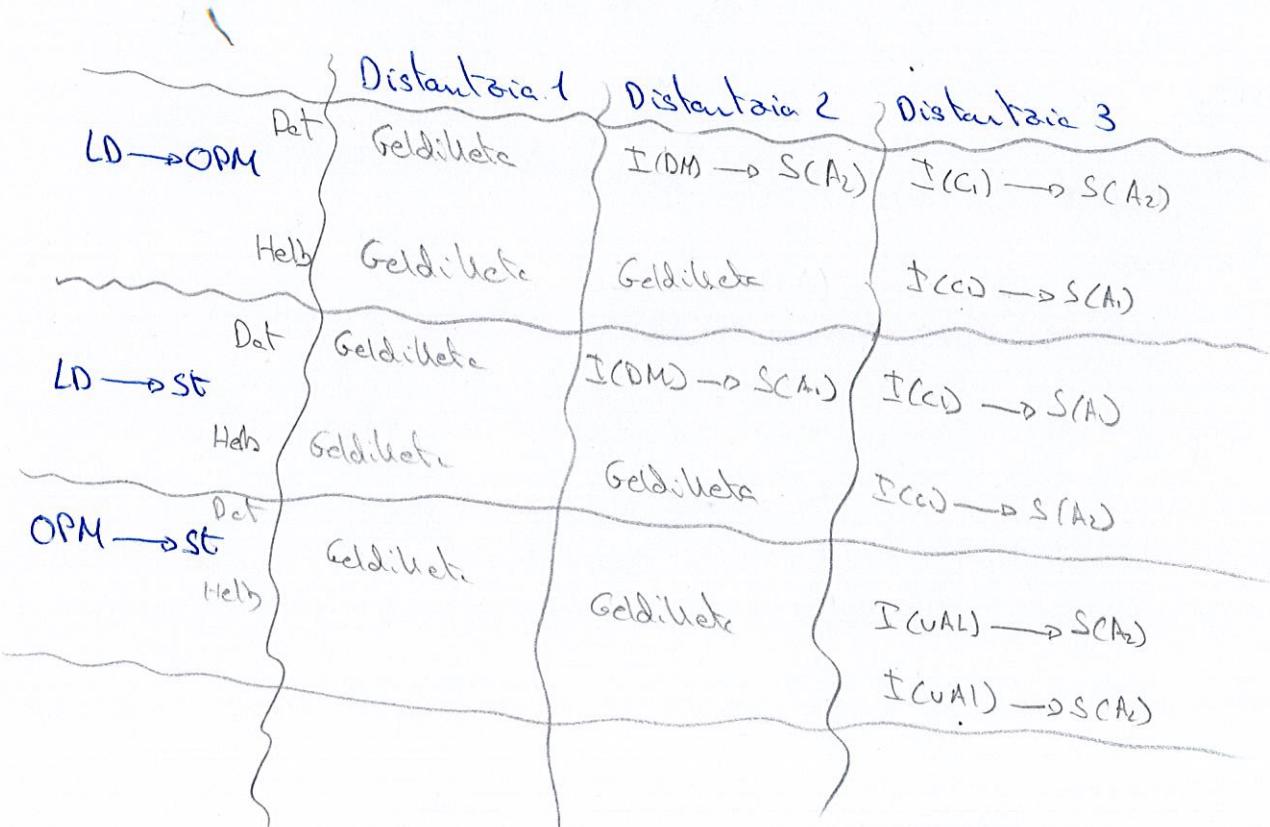
Prozesadoreak ez beditu erabiltzen ez datu - es kontro-dependentziarile, programan es da eza aldatuko.

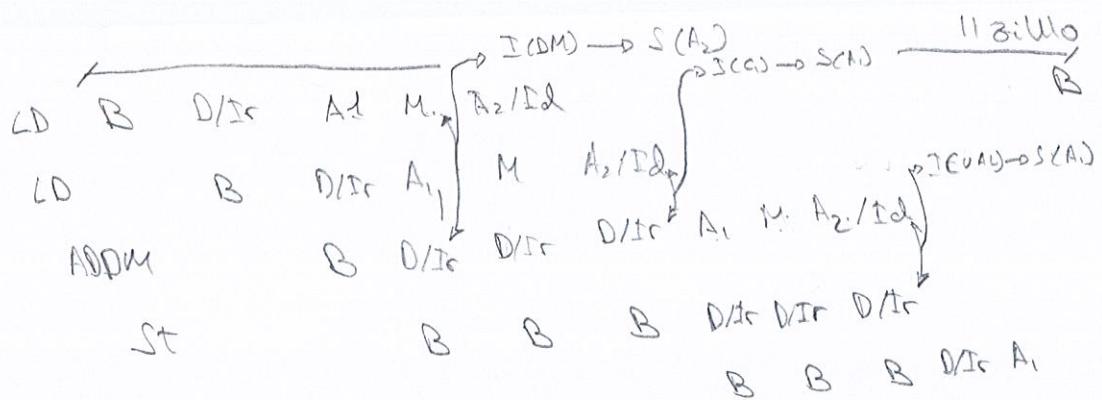
loop: LD R₃, A[R₁]
 LD R₄, B[R₁]
 ADDM R₁ [R₃], R₄
 ST A[R₁], R₄
 BRx R₃, loop
 NOP

Registren Bilbo klopurua:



b)





c)

Jauzi atzeretua teknike osabiliak
nolle. Horrela ADDM eta ST-ren arteko zirkuitu laburra atzeretua.
(A)

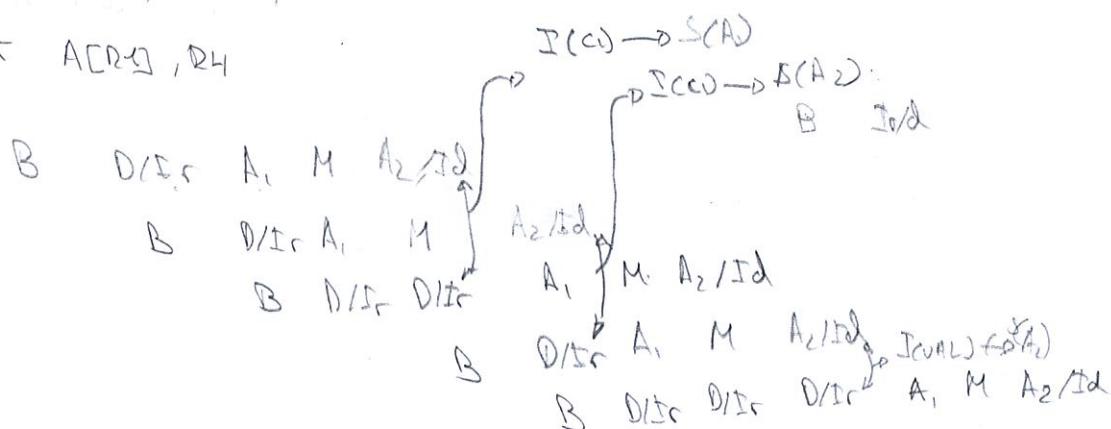
LD R3ACD1

LD D4 B[R3]

B Rx R3, loop

ADDM AD,, C[R3], R4

ST A[R3], R4



8zillo

Ondorioz 3 zilloa aurkeztu genuitak bera bezizten sortzen
den behatzen.

$$200 \cdot 3 = 600 \text{ zilloa aurkeztu}$$

1. Ariketa

$$P = 16$$

Programa:

$$0.5 \rightarrow 16p$$

$$0.25 \rightarrow 8p$$

$$0.25 \rightarrow 1p$$

Zeru da adelerazio-faktorea:

$$af = \frac{ts}{f \cdot p + (1-f)ts} = \frac{1}{0.5 \cdot \frac{1}{16} + 0.25 \cdot \frac{1}{8} + 0.25} = 32$$

2. Ariketa

$$P = 8$$

Programa:

$$50 \text{ seg} \rightarrow 8p$$

$$20 \text{ seg} \rightarrow 4p$$

$$30 \text{ seg} \rightarrow 1p$$

Zeru da adelerazio-faktorea:

$$af = \frac{ts}{f \cdot p + (1-f)ts} = \frac{1}{0.5 \cdot 1, \frac{1}{8} + 0.2, \frac{1}{4} + 0.3} = 286$$

$$frag = \frac{af}{P_{\max}} = \frac{286}{8} = 0.35$$

Gainera hainbat paralelotako klasuetan gainazaleko
exkluzio-deboratzen da 10.

3. Ariketa

$$P = 64$$

a)

$$1. Erag = 0.50 \Rightarrow Erag = 0.5$$

$$Erag = \frac{af}{P_{\max}} \Rightarrow 0.5 = \frac{af}{64} \Rightarrow af = 32$$

$$32 \cdot \left(\frac{f}{64} + 1 - f \right) = 1 \Rightarrow \frac{1}{2}f + 32 - 32f = 1 \Rightarrow$$

$$\Rightarrow -31.5f = -31$$

$$f = 0.984$$

$$af = \frac{ts}{f \cdot p + (1-f)ts} = \frac{ts}{\frac{ts}{P} + (1-f)ts} = \frac{ts}{\frac{ts}{64} + (1-f)ts} \Rightarrow 32 = \frac{1}{\frac{f}{64} + (1-f)}$$

$$2) f = 0'9$$

$$\alpha f = \frac{ts}{\frac{f}{4}tp + (1-f)ts} = \frac{1}{\frac{0'9}{64} + 0'1} = 8'767$$

b)

Gustafsson-en logaritmiskt metoder

$$\alpha f = \frac{ts}{tp} = (1-f) + fP \Rightarrow \alpha f = 0'1 + 0'964 = 578$$

6. Arithmetica

Seriellt sättia exekutatabello 100 sec $P=16$

Iteratio ballotaa 50 sec

16 iteratio

$$ts = 100 + \frac{50 \cdot 16}{16} = 900$$

$$tp = 100 + \frac{50 \cdot 16}{16} = 150$$

$$\alpha f = \frac{900}{150} = 6$$

1600 iteratio

$$tp = 100 + \frac{50 \cdot 1600}{16} = 5100$$

$$ts = 100 + 50 \cdot 1600 = 80100$$

$$\alpha f = \frac{80100}{5100} = 15'2$$

7. Arithmetica

100 iteratio $\Rightarrow 99 \cdot ts + 20s \cdot d$

$$a) ts = 119$$

$$tp = 119 + \frac{100}{4} \Rightarrow \alpha f = \frac{ts}{tp} = 2'71$$

b)

Berdine Bangs de

c)

Kasville olleisen

Atden bukkete ~~o~~ some

$$t_p = 44 + \frac{0.05 \cdot 100}{4} = 45.25 \rightarrow at = \frac{119}{45.25} = 2.629$$

Kasville auken

$$t_p = \frac{119 + 0.05 \cdot 100}{4} = 31.25 \quad at = \frac{119}{31} = 3.838$$

Konzeptarbeiten Architektura

Konzeptarbeiten parallelisch

1. Architektur

$$P = 16$$

$$50\% \rightarrow 16_p$$

$$25\% \rightarrow 8_p$$

$$25\% \rightarrow 1_p$$

$$Af = ?$$

$$\cancel{Af = \frac{1}{0'25 \cdot 8 + 0'5 \cdot 16 + 16 \cdot 0'25}}$$

$$Af = \frac{1}{0'25 \cdot \frac{1}{8} + 0'5 \cdot \frac{1}{16} + 0'25} = 32$$

2. Architektur

$$P = 8$$

$$8_p \rightarrow 50 \text{ Seg} \Rightarrow 50 \cdot 0'1 = 50 \text{ Seg}$$

$$4_p \rightarrow 20 \text{ Seg} \Rightarrow 20 \cdot 0'1 = 20 \text{ Seg}$$

$$1_p \rightarrow 30 \text{ Seg}$$

$$Af = \frac{1}{0'5 \cdot 0'1 \cdot \frac{1}{8} + 0'2 \cdot 0'1 \cdot \frac{1}{4} + 0'3} = 2'36$$

$$Eff = \frac{at}{P_{Max}} = \frac{2'36}{8} = 0'295$$

3. Arithmetik

64 prozessoren

$$a,1) \text{ Andahl: } af = \frac{ts}{tsp}, \quad tsp = f \cdot ts + (1-f) \cdot ts$$

$$\text{erg} = \frac{af}{64} \geq 0.5 \Rightarrow af \geq 32$$

$$af = 32 \text{ b-dc etc, } ts = 32 \cdot tsp \Rightarrow tsp = f \cdot ts + (1-f) \cdot ts \cdot 32 \Rightarrow$$

$$\Rightarrow tsp = \frac{1}{2} \cdot f \cdot ts + (1-f) \cdot 32 \cdot ts \Rightarrow tsp = \frac{1}{2} f \cdot ts + 32 \cdot ts + 32 \cdot ts = \\ = \left(\frac{1}{2} f \cdot ts - 32 \cdot ts \right) + 32 \cdot ts = 31.5 \cdot ts + 32 \cdot ts$$

$$32.5 \cdot ts \cdot f = 32 \cdot ts \Rightarrow f = \frac{32 \cdot ts}{32.5 \cdot ts} = 0.984 \quad t_p = \frac{ts}{64}$$

$$a,2) \quad af = ? \Rightarrow f = 0.9 \Rightarrow af = \frac{ts}{0.9 \cdot ts + 0.1 \cdot ts} \Rightarrow \frac{1}{af} = \frac{0.9 \cdot ts}{ts} + 0.1 = ?$$

4. Arithmetik

1, b) Gustafson:

$$P = 32 \quad af = (1-f)P + fP = 0.1 + 0.9 \cdot 64 = 57.2 \quad \Rightarrow \quad \frac{1}{af} = \frac{0.9}{64} + 0.1 \Rightarrow$$

$$0.1 \Rightarrow \frac{1}{32} \quad P = 1$$

$$0.2 \Rightarrow \frac{1}{16} \quad P = 1$$

$$0.2 \Rightarrow \frac{1}{P} \quad P = 1$$

$$Af = \frac{1}{0.1 \cdot \frac{1}{32} + 0.2 \cdot \frac{1}{16} + 0.2} = 4.05$$

$$af = 1 = \left(\frac{0.9}{64} + 0.1 \right) af \quad \cancel{af = 8.262}$$

$$\text{erg} = \frac{4.05}{32} = 0.1285 \Rightarrow \% 12.65$$

5. Arithmetik

$$P = 32$$

$$Af = \frac{1}{0.4 \cdot 12 \cdot \frac{1}{32} + 0.2 \cdot 11 \cdot \frac{1}{16} + 0.2} = 3.97$$

$$\text{erg} = \frac{3.97}{32} = 0.124$$

6. Arbeit

$$t_{\text{p}} = 100 \text{ sec}$$

$$t_{\text{interaktion}} = 50 \text{ sec}$$

16 Iterationen bilden die akzessorische Taktzone:

$$100 + 50 \cdot 16 = 900 \text{ sec}$$

$$t_p = \frac{16 \cdot 50}{16} = 50 \text{ s} + 100 = 150 \text{ sec}$$

$$t_s = 60 + 50 \cdot 16 = 900 \text{ sec}$$

$$at = \frac{t_s}{t_p} = \frac{900}{150} = 6$$

1600 Iterationen

$$t_p = \frac{1600 \cdot 50}{16} + 100 = 5100 \text{ sec}$$

$$t_s = 100 + 50 \cdot 1600 = 80100 \text{ sec}$$

$$at = \frac{t_s}{t_p} = \frac{80100}{5100} = 15'705$$

7. Arbeit

$$P = 4$$

100 Iterationen

$$\begin{aligned} qq &\rightarrow 1 \text{ sec} \\ l &\rightarrow 20 \text{ sec} \end{aligned}$$

$$\textcircled{2} t_{\text{Serie}} : qq + 20 \text{ sec} = 119 \text{ sec}$$

$$t_{\text{Periodenz}} : \frac{100+19}{4} = 44 \text{ sec}$$

$$at = \frac{119}{44} = 2'2$$

$$Eff = \frac{2'2}{4} = 0'6751$$

b)

Ez dio arak

c)

Kasu txerruenak:

Seriean: 119 seg

Perdelaun: $44 + 0.05 \cdot \frac{100}{4} = 45'25$

$\alpha_f = 2'63$

Kasuril omenak

$$\frac{119 + 0.05 \cdot 100}{4} = 31 \text{ seg}$$

F. Aribetakle

$P=4$

Sakaza independente

15, 10, 6, 15, 8, 3, 2, 1/ alazk. baliotzaera exclusio dubora

$$t_s = 5 + 10 + 6 + 15 + 8 + 3 + 2 + 1 = 49.50 \text{ seg}$$

d)

"Static, z" erabiliz edo

$$t_p = 21 \quad \alpha_f = \frac{50}{21} = 2'38$$

e)

Bandalera dinamikoa erabiliz

$$t_p = 15$$

$$\alpha_f = \frac{50}{15} = 3\frac{1}{3}$$

P. Antweber

min = int_Max

for ($i=0; i < N; i++$) if ($A[i] < \text{min}$) $\text{min} = A[i]$
for ($i=0; i < N; i++$) $A[i] = A[i] - \text{min}$

mini = int_Max

#pragma omp parallel private(min) shared(mini)
#pragma omp for private(i)

for ($i=0; i < N; i++$) if ($A[i] < \text{mini}$) $\text{mini} = A[i]$

#pragma critical
if ($\text{mini} < \text{min}$) $\text{min} = \text{mini}$

#pragma omp for reduction(-: A[i])

for ($i=0; i < N; i++$) $A[i] = A[i] - \text{min}$

10. Arriketa

```
printf ("Sartu x-ren balioa");
scanf ("%d", &x);
#pragma omp parallel schedule(dynamic)
#pragma omp for private(i) reduction(*; A[i])
for (i=0; i<N; i++) A[i] = A[i]*x;
#pragma omp for private(i) reduction(*; B[i])
for (i=0; i<N; i++) B[i] = B[i]*A[i]+x;
```

min = INT_MAX;

```
#pragma omp for private(i, mini) shared(mini)
for (i=0; i<N; i++) if (B[i] < mini) mini = B[i]
```

#pragma critical

if (mini < min) min = mini;

```
#pragma omp for private(i) reduction(-; B[i])
```

```
for (i=0; i<N; i++) B[i] = B[i]-min;
```

#pragma omp for private(i)

```
for (i=0; i<N; i++) printf ("%d\n", B[i]);
```

11. Arriketa

```
batura = 0.0
#pragma omp parallel
#pragma omp for private(i) reduction(+: batura)
for (i=0; i<N; i++) batura = batura + A[i]*A[i];
#pragma omp single
```

bb = batura/4;

```
#pragma omp for private(i)
for (i=0; i<N; i++) A[i] = sqrt(A[i]+bb)+pow(2, bb/4);
```

12. Arithete

```
main() {
    int i, j, histo[256];
    Struct image in-image;
    #pragma omp parallel
    #pragma omp for private(i)
    for (i=0; i<256; i++)
        histo[i]=0;
    #pragma omp for private(i) parallel
    for (i=0; i<in-image.h; i++) {
        #pragma omp for private(j) parallel
        for (j=0; j<in-image.w; j++)
            histo[in-image.im[i][j]]++;
    }
}
```


3. Arithmetik

P=64 processore

ω

$$E_{avg} \geq 0.5 \Rightarrow \frac{af}{64} = 0.5 \Rightarrow af = 32 \Rightarrow af = \frac{ts}{\frac{f}{t_p} + (1-f)t_s}$$

non ts sericho denbore

$$t_{sp} = \frac{f}{t_p} + (1-f)t_s$$

$$t_p = \frac{ts}{f}$$

$$af = \frac{ts}{f \cdot \frac{ts}{\alpha} + (1-f)ts} = \frac{ts}{ts \left(\frac{f}{\alpha} + (1-f) \right)} = \frac{1}{\frac{f}{\alpha} + (1-f)} = D$$

$$\Rightarrow 32\left(\frac{f}{64} + 1-f\right) = 1 \Rightarrow \frac{1}{2}f + 32 - 32f = 1 \Rightarrow 31.5f = 31 \Rightarrow f = 0.984$$

$$af = \frac{ts}{f \cdot ts + (1-f)ts} = \frac{ts}{\frac{f \cdot ts}{P} + (1-f)ts} = \frac{ts}{ts \left(\frac{0.9}{64} + 0.1 \right)} =$$

8.767

٦

$$af = \frac{ts}{tp} = (1 - 0.9) + 0.964 = 57.2$$

$$ts = (1 - f)ts + f t s p$$

$$t_p = (1-f)t_s + f t_s$$

$$1 - \alpha f = (1-f) + fP$$

— — — — — — — —



6. Arbeit

$$t_s = 100s$$

16 Iterationen

~~$$t_p = \frac{ts}{(1-f)} + \frac{f \cdot ts}{P} = \frac{100}{(1-f)} + \frac{f \cdot 100}{50} = 100 + \frac{16 \cdot 100}{16} = 150$$~~

$$\cancel{t_p} = \frac{100}{(1-f)} + \frac{f \cdot 100}{P} = 100 + \frac{16 \cdot 100}{16} = 150$$

$$t_p = (1-f) t_s + f \cdot \frac{t_s}{P} = 100 + \frac{16 \cdot 100}{16} = 150$$

$$at = \frac{ts}{t_p} = \frac{100}{150} = 0.666$$

1600 Iterationen

$$f =$$

$$ts = (1-f) \cancel{t_p} + f \cdot P \cdot ts = 100 + \frac{50 \cdot 1600}{16} = 80100$$

$$t_p = (1-f) ts + f \cdot \frac{ts}{P} = 100 + \frac{50 \cdot 1600}{16} = 80100$$

$$at = \frac{80100}{80100} = 1.0$$

7. Arbeit

P = 4 prozedure

160 Iterationen

Adress-seg etc bestehende 1seg

a) 99 adresssegs 1seg

1 " 20seg

Serieen: $99 + 20 = 119$ seg

Paralelben: $\frac{100}{4} + 19 = 44$ seg

Kasurili Queue:

$$\frac{ts + 50ms \cdot \text{Iteration}}{P} =$$

$$= \frac{119 + 0.05 \cdot 100}{4} = 31s$$

$$\text{Speed-up} = \frac{119}{44} = 2.7$$

b) Bandbreite Tabelle

E8 du axela \rightarrow Speed-up = 2.2

c) Kasurili Tarafrennen

Serieen: 119 seg

Paralelben: $44 + 0.05 \cdot 25 = 45.25s \rightarrow \text{Speed-up} = 2.629$

d) Kasurili Queueen

$$\frac{119 + 0.05 \cdot 100}{4} = 31s \quad \text{Speed-up} = \frac{119}{31} = 3.83$$

Südostniedersachsen

11. Arithmetik

$$b_{\text{ture}} = 0.0;$$

#pragma omp parallel for reduction(+:b_{ture})

for (i=0; i<N; ++i) b_{ture} = b_{ture} + A[i]*A[i];

$$bb = b_{\text{ture}}/N$$

#pragma omp parallel for private(i)

for (i=0; i<N; ++i) A[i] = sqrt(A[i] + bb) + pow(bb, 4);

4. Arithmetik

$$P = 32$$

$$f = 0.1 \cdot 32 + 0.7 \cdot 16$$

$$1-f = 0.2$$

$$\alpha_f = \frac{t_s / t_p}{1} = \frac{1}{\frac{1}{32} \cdot 0.1 + \frac{1}{16} \cdot 0.7 + 1 \cdot 0.2} = \\ = 4.05$$

$$E_{\text{frag}} = \frac{\alpha_f}{P_{\text{max}}} = \frac{4.05}{32} = \% 12.8$$

5. Arithmetik

$$\alpha_f = \frac{t_s}{1.2 \left(\frac{t_s}{32} \right) \cdot 0.1 + 1.1 \cdot \left(\frac{t_s}{16} \right) \cdot 0.7 + t_s \cdot 0.2} \\ = \frac{1}{1.2 \left(\frac{1}{32} \right) \cdot 0.1 + 1.1 \left(\frac{1}{16} \right) \cdot 0.7 + 1 \cdot 0.2} \approx 3.97$$

$$E_{\text{frag}} = \frac{\alpha_f}{P_{\text{max}}} = \frac{3.97}{32} \approx \% 12.4$$

⑪

loop:

```

LD    R3 A[R1]
LD    R4 B[R1]
ADD M R1, C[R3], R4
ST    A[R1], R4
BRx  R3, loop

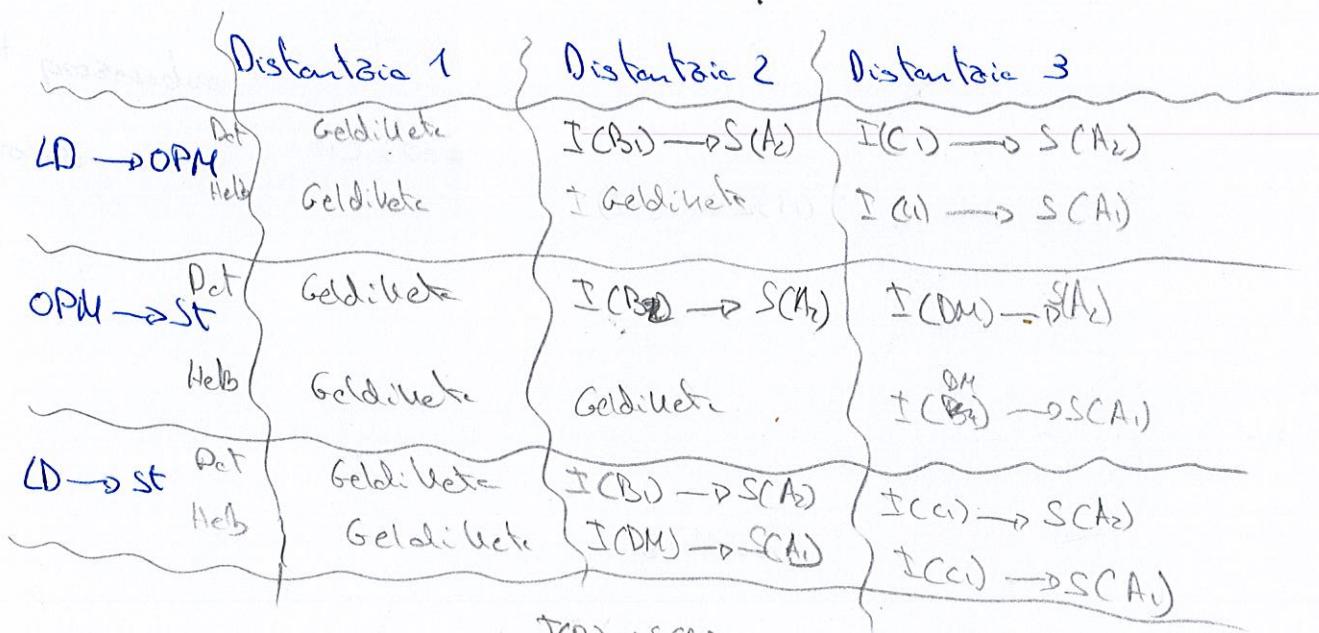
```

a)

Esimme luke additioonille esimme programmi
11 bittiä

LD	B	D/Ir	A ₁	M	A ₂ /Id	B
LD	B	D/Ir	A ₁	M	A ₂ /Id	
ADD M	B	D/Ir	D/Ir	D/Ir	D/Ir D/Ir A ₁ M A ₂ /Id	
ST	B	B	B	B	D/Ir D/Ir D/Ir A M A ₂ /Id	
	B	B	B	B	B D/Ir A ₁	

b)



LD	B	D/Ir	A ₁	M	A ₂ /Id
LD	B	D/Ir	A ₁	M	A ₂ /Id
ADD M	B	D/Ir	A ₁		

S. Arribalzaga

a)

Mahine honetan VAL kopuruak 2 izango da.

batzuetako kopuruak 3 da, eta kopurategailuak egingo dira.

b)

		Distantzia 1	Distantzia 2	Distantzia 3	Distantzia 4
OP → ST	dat	$I(UAH) \rightarrow S(a_1)$	$I(b_1) \rightarrow S(a_1)$	$I(c_1) \rightarrow S(a_1)$	$I(d_1) \rightarrow S(a_1)$
	Heib	$I(UAH) \rightarrow S(a_2)$	$I(b_1) \rightarrow S(a_2)$	$I(c_2) \rightarrow S(a_2)$	$I(d_1) \rightarrow S(a_2)$
$(I) \rightarrow OOP$		Geldiketa	$I(b_1) \rightarrow S(e_1, a_2)$	$I(c_1) \rightarrow S(e_1, a_2)$	$I(d_1) \rightarrow S(e_1, a_2)$
	dat	Geldiketa	$I(b_1) \rightarrow S(OOP)$	$I(c_1) \rightarrow S(OOP)$	$I(d_1) \rightarrow S(OOP)$
OPM → OPM	dat	Geldiketa	Geldiketa	$I(c_2) \rightarrow S(a_2)$	$I(d_2) \rightarrow S(a_2)$
	Heib	Geldiketa	Geldiketa	$I(UAH_2) \rightarrow S(e_1, a_2)$	$I(d_1) \rightarrow S(e_1, a_2)$

c)

JMP_x aginduak %20 eta %15 IMP

%20 → %50 agindu bat

→ %10 agindu bi

$$\%15 \cdot 0'5 = \%25$$

JMP_x : Lehenetako 3

$$\%20 \cdot (0'50 \cdot 2 + 0'10 \cdot 1 + 0'40 \cdot 3) = \%46$$

⑩

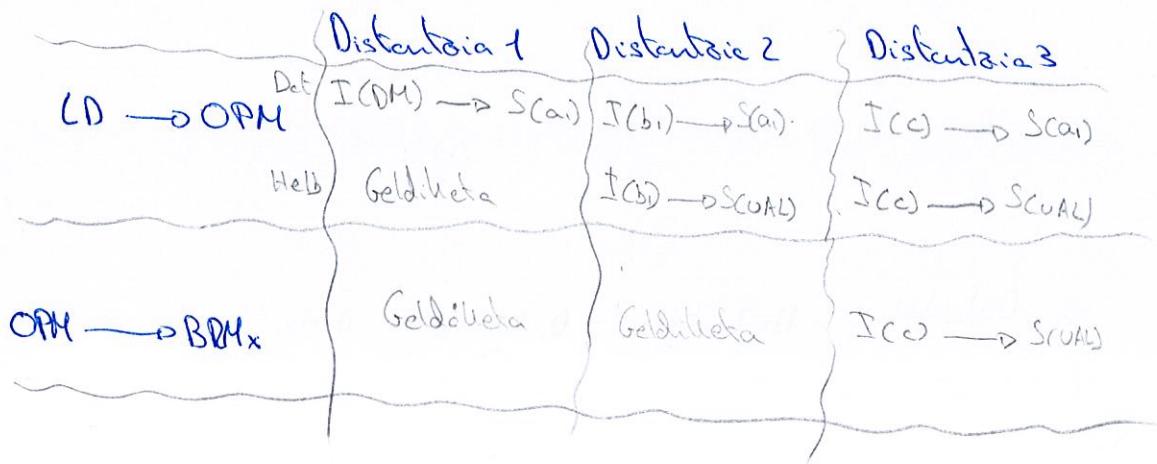
$$60 \cdot 0'2 = 12$$

Bide jarrizun $28 \text{ Mio. } 10 = 28 \text{ Mio. }$

$$12 \cdot 3 = 36 \Rightarrow 36 + 20 + 30 = 86 \text{ Mio.}$$

Z. Asilente

a)



Jauzi-aginduren latente horago agindua exekutatzen hasteko behar duen denbora da.

b)

① beg: LD R3, AC(R1) +

ADDI R1, R1, #12

B Jr

B

LD R5 BC(R2) 3

B Jr

MULM R4, R3, C(R5)

B R M A Id

B Jr Jr Jr M A Id

BRMx D(R4), beg

B Jr Jr Jr Jr

100 aldiz exekutatu, zirkulu laburrik erabiliz gabe 11 zilito ibango
ezker zilitolo,

$$\text{Ber. de, } 11 \text{ zilito} \cdot 100 = 1100 \text{ zilito}$$

②

~~(2)~~ beg: LD R3, AC(R1)

LD B Jr/AM M A Id

$\rightarrow I(DM) \rightarrow S(ai)$

ADDI R1, R1, #12

ADDI B Jr/AM M A Id

LD R5 BC(R2)

LD

B Jr/AM M A Id

MULM R4, R3, C(R5)

MULM

B Jr/AM M A Id

BRMx D(R4), beg

B Jr/AM M A Id

LD R3 AC(R1)

B

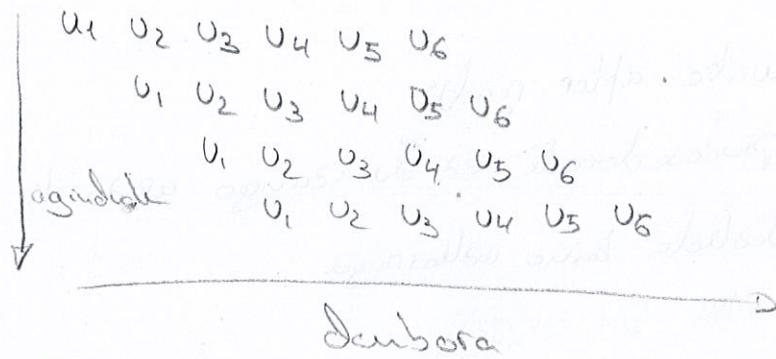
$$5 \text{ zilito} \cdot 100 = 500 \text{ zilito}$$

2. Gaia Prozesadoreko muntaketa-klaseak erabiltzen ditu.

Prozesadore Seguentatzailea

Segmentazio lineala

Aginduen exekuzioen muntaketa - kate bedala antolatu
(Pipeline).



Agindu bat lehen fasean bigarrenera pasatzeari hurreago
agindua exekutatzeko hasten da. Helburua urrats bat zirkulo
exekutatzea da.

Fase guztiek denbora beretikoa definitu. Eta agindu
guztiek fase guztietatik igaro behar dute.

Segmentazio erregistroak edo latch-ak erabilizten dira
urratsen arteko komunikazioe uztentzela, agindu batetan emaitza
hurrengoen erregirako izan daiteke.

Dlx prozesadore

RISC prozesadoreen da, eragikete guztiekin erregistro osakor
melen cogiten dira.

Exekuzio-urratsak: Bildaketa(B), Desbuketaka etc Irakurketa(D/Ir),
VAL-a (A), Memoria(M), Emaitza ideaketa(Id)

Baino aginduak ean dira beti gainjarrak exekutatu.

Adibide:

LD R₂, d(R₁) → B | D/I_r | A | M | Id
ADD R₃, R₂, #1 → B | D/I_r | - - -

Addit. eaneko du R₂ erregistroa ikurrira? zikloan datuei ez
dego prest.

- WAR (write-after-read)

Dlx prozesadorede edo izango aezorile, irakurketa beti
izango de idazteko baino aterrago.

ADD R₁, R₄, R₂

SUBI R₂, R₆, #1

Zurrunbait baino egiteko, irakurketa idazteko baino aterrago egiten
dute.

- WAW (write-after-write)

Bi aginduak erregistro bera idazti uhi dute.

ADD R₂, R₄, R₁

SUBI R₂, R₆, #1

Zurrunbait baino de ordenenak betetzen del prozesor.

- WAW eta WAR dependentziak

WAR: Baskoleta atzeratu aurkeko aginduak datuei irakurri
ante.

WAW: Lehenengo aginduaren baskoleta baliogabeak edo
atzeratu.

= R/W (read-after-write)

Agindu biler aurreko beste agindu bat irakurri behar denezan.

RAU dependentziak

Datuak behar duen agindua geldizazia, beste aginduak ermitza EM utzi arte itzaron, erabiltzeko zuzenea izateko.

Bi modu:

- Software-z (kompiladoreak)

Bi aginduen distantzia handitzeko behar dira NOP-all sortzen ditu.

- Hardware-z (kontrol-unitateak)

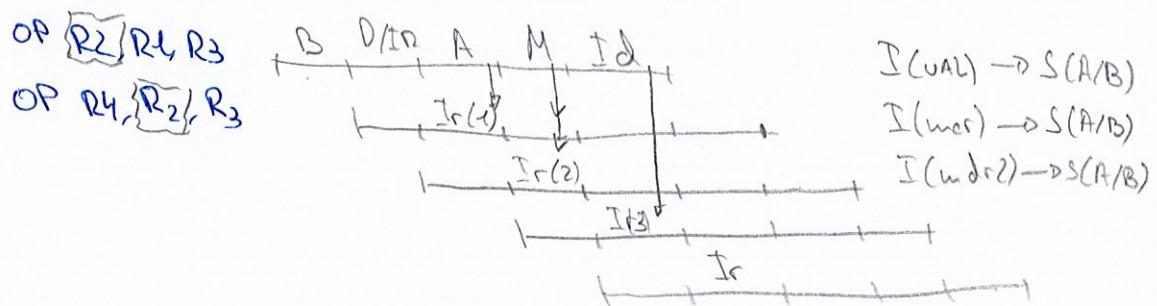
Hauke dependentziak antzematen du, eta aginduaren distantzia geldizten du, behar den zileko kopuruak.

Zirkuitu laburrak (Forwarding)

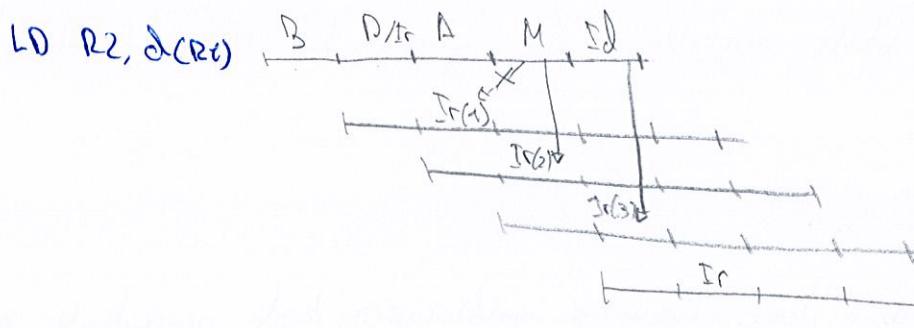
Datuak sortzen den UF-tik zuzenean lor daiteke, EM-tile irakurri beharraren.

Agindu bilako gosakide astekoa RAU dependentziak antzemanak.

OP-OP zirkuitu laburrak



LD-OP Zirkulationskunst



Gelditur!

$$\Sigma(DM) \rightarrow S(A/B)$$

$$\Sigma(\text{Indr2}) \rightarrow S(A/B)$$

1. Grafito honean, prozesadore segmentatu bat erabiliz lor dantzeen azelerazio-faktorea ageri da. "Segmentazio-hodiaren" urrats kopuruaen arabera.
- Azal eza zu grafikoan ageri den portaera.
-

2. Aplikazio jakin batean, bi bektoaren arteko biderkadura eskalalarra kalkulatzentz da, honako begizta honen bidez:
- ```

ADDI R1,R0,#0
ADDI R2,R0,#1024
LD R4,A(R1)
LD R5,B(R1)
MUL R6,R4,R5
ADD R2,R2,R6
ADDI R1,R1,#1
SUBI R3,R3,R1
BNZ R3,b1
ST B,ES(R0),R2

```
- Bezia horti bi prozesadore desberdinaren ezerkutatu da. Lehenbiziakoa, prozesadore sekuentzial bat da, non oinarrizko ergangien latenziaek honako hauek batirat: biaketa, 10 ns; deskodeketa eta eragilegaitan irakurketa, 4 ns; ergiketa aritmético bat, 7 ns; memoria trakurra edo idazte, 10 ns; erregistro-multzoan idazte, 4 ns. Bigarrena DLX prozesadorea da, zirkuitulaburak gehituta; erloju-maiztasuna 90 MHz da.
- a. Prozesadore sekuentzialerako, kalkula eza zu agindu mota (OP, LD, ST, BNZ) bakoitzan ezerkuzio-denboran, ondorioz, kalkula eza zu programa osoa ezerkutatzeko behar den ziklo kopurua;
- b. DLX prozesadorearen kasuan, kalkula eza zu begizta-iterazio bat ezerkutatzeko behar den denborra.
- c. Kalkula eza zu bigarren prozesadorearen "load aceleration" eta jauzi azterazio-faktorea (zenbat alidiz azkarrago ezerkutatzeko den programa). Atxarrioz erantzuna.

3. Hartu kontuan DLX prozesadorearen erreserba-taula eta ebaz itzazu aginduen arteko honako datu-dependentsia hauek (kasu debergarria): OP-LD; LD-ST; LD-LD; LD-BR. Gero, kalkula eza zu begizta hau ezerkutatzeko behar den denborra kasu haustan: (1) zirkuitulaburak ez dira erabilten; eta (2) DLXren zirkuitulabur guiziak erabiltzen dira, (C) zirkuitulaburak kontuan hartua, jauziaren aldeko apustua erabiliz da, eta (d) zirkuitulaburak kontuan hartua "load aceleration" eta jauzi azterazia erabiliz dira.
- Kalkula eza zu kasuarekin konfrontatu eta ebaz itzazu aginduen arteko honako datu-guztietan.

```

ADDI R1,R0,#63
beg: LD R2,X(R1)
 LD R3,Z(R1)
 MUL R3,R3,R3
 SUB R3,R3,R2
 ADDI R3,R3,#1
 ST X(R1),R3
 SUBI R1,R1,#1
 BNE R1,beg

```

4. Hartu kontuan oinarrizko lau agindu hauek:

```

OP Rh,Ri,Rj
Rh := M[A(Ri)]
LD Rh,A(Ri)
N[A(Ri)] := Rj
ST A(Ri),Rj
balign (R1 - 0) PC := PC + desp
BCK Ri,desp

```

- Prozesadore segmentatu lineal batzuk honako hiru urrats hauetan segmentatzaten du agindu horien ezerkuzioa:
- B/D, Ir/A eta M/Id.
- a. Egin eza zu aginduen datogilekien erreserba-taula. Horren arabera, zirriboratu prozesadorearen datu-bidea.
- b. Analizatu aginduen arteko datu-dependentsiaz, eta proposatu dagozkienn zirkuitulaburak.

### 5. Lau urratsek prozesadore segmentatu bat diseinatu behar da, erreserba-taula hau kontutan hartuta.

|             | B                                         | D/Ix                         | M/Id                                              |
|-------------|-------------------------------------------|------------------------------|---------------------------------------------------|
| OP Rh,R1,R2 | IR := AM(PC)<br>PC := PC + 4              | b1 := EM[R1]<br>b2 := EM[R2] | c1 := b1 OP b2<br>EM[Rh] := c1                    |
| LD Rh,A(R1) | IR := AM(PC)<br>PC := PC + 4              | b1 := EM[R1]                 | c1 := b1 + @A<br>EM[Rh] := DM(c1)                 |
| ST A(R1),R2 | IR := AM(PC)<br>PC := PC + 4              | b1 := EM[R1]<br>b2 := EM[R2] | c2 := b2<br>DM(c2) := c2                          |
| BR1,R1,help | IR := AM(PC)<br>PC := PC + 4              | b1 := EM[R1]<br>b2 := EM[R2] | c1 := b1 - 1<br>EM[R1] := c1                      |
| BN2,R1,R2   | IR := AM(PC)<br>PC := PC + 4<br>PC1 := PC | b1 := EM[R1]<br>b2 := EM[R2] | haldir (b1-1 = 0)<br>PC := PC2 + b2<br>PC2 := PC1 |

Ebatz itzazu aginduen arteko dependentsia (RAW) hauek: OP → OP; OP → ST; LD → ST; BR1 → LD; etab.

Ebatz itzazu aginduen arteko dependentsia (RAW) hauek: OP → OP; OP → ST; LD → ST; BR1 → LD; etab.  
BR1 → BR2. Adiskizti, zehatz-mehaz, egun daitzekeen zirkuitulaburru guztiek.

- Jauzi latenzia dela eta, NOP aginduak erabilizten dira. Zenbat NOP idatziz behar dira BR1 jauzien ondoren? eta BR2 jauzien ondoren?
6. Aurreko arketadorek sortu duten kodea (R1 enregistroaren hasierako balioa 99 da):
- ```

LD R3,A(R1)
beg: LD R3,A(R1)
      ST B(R1),R2
      BR1 R1,beg
      NOP
      LD R3,C(R1)
      ...
    
```
- a. Kalkula eza zu begizta ezerkutzeko behar den ziklo kopurua. (1) zirkuitulaburak erabilizten ez badira; eta (2) aurreko arketadoko zirkuitulaburak erabilizten badira.
- b. Demagun "jauzi azterata" teknika erabil daietikela (zirkuitulaburru gain); aldatu begiztaren ordezko faktorearen ordena ezerkuzio eraginkorrakorraga lortu ahal izateko. Kalkula eza zu lortutako begiztaren ezerkuzio-denborra? Azkenik, jauzien aldeko apustuak erabiliko baginu, aldatuko lituzakeen ezerkuzio-denborra.

7. Ereserba-taula honetan, prozesadore batzen lau aginduren ezerkuzio segmentatuaren ageri da.

	B/D	Ix/AM	M	A	Id
OP Rh,R1,R2	IR := AM(PC) PC := PC + 4	a1 := EM[R1] a2 := EM[R2]	b1 := a1 b2 := a2	c = b1 OP b2 EM[Rh] := c	EM[Rh] := c
LD Rh,A(R1)	IR := AM(PC) PC := PC + 4	a1 := EM[R1]	b1 := EM[R1] b2 := a2	c := b1 OP b2 EM[Rh] := c	EM[Rh] := c
OPM Rh,A(R1),R2	IR := AM(PC) PC := PC + 4	a1 := EM[R1] a2 := EM[R2]	b1 := EM[R1] b2 := a2	c := b1 OP b2 EM[Rh] := c	EM[Rh] := c
BNEC A(R1),desp	IR := AM(PC) PC := PC + desp	a1 := EM[R1] PC2 := PC1	b1 := EM[R1] PC := PC2	b2 := EM[R1] + desp PC3 := PC + desp	-

- a. Ebatz itzazu LD → OPM eta OPM → BRM aginduen arteko dependentsiaz (RAW). Adierazi, zehatz-mehaz, egun behar diren zirkuitulaburak. Zein da jauzi-aginduaren latenzia? Zergatik?
- b. Begizta honen 100 iterazioa ezerkutatu behar dira:
- beg: LD R3,A(R1)
- ADDI R1,R1,#1
- LD R5,B(R2)
- MULM R4,R3,C(R5)
- BNEC D(R4),desp

1. Kalkula eza zu 100 iterazioak ezerkutzatzeko behar den ziklo kopurua. a. ataleko zirkuitulaburru kontuan hartu gabe.
2. Erabil eza zu jauzien aldeko apustua, eta berriatzi begizta. Kalkula eza zu begizta ezerkutzatzeko behar den ziklo kopurua. a. ataleko zirkuitulaburru kontuan hartu gabe.
3. Azkenik, nola idatziz behar da begizta jauzi aztertua teknika erabilii nahi badu?

8.

Prozesadore segmentatu lineal batek sei urratu erabiltzen ditu aginduen exekuzioa segmentatzeko, honako erreserba-taula honetan ageri den moduan.

B	D/Tx	A1	M	A2	Id
OP PB, R11, R12	IR := AM(PC)	a1 := EM(R11)	b1 := a1 OP a2	c1 := b1	d1 := c1
	PC := PC + 4	a2 := EM(R12)			EM(R11) := d1
OPM RB, R11, A(R12)	IR := AM(PC)	a1 := EM(R11)	b1 := a1	c1 := b1	d1 := c1 OP c2
	PC := PC + 4	a2 := EM(R12)	b2 := a2	c2 := DM(b2)	EM(R11) := d1
LD Rh, A(R12)	IR := AM(PC)	a2 := PC + 4	b2 := EM(R12)	c2 := DM(b2)	d1 := c2
ST A(R12), R11	IR := AM(PC)	a1 := EM(R11)	a2 := a2 + 6A	c2 := DM(b2)	EM(R11) := d1
JMPX R11, R12, dp	IR := AM(PC)	a1 := EM(R11)	b1 := a1	Dm(R12) := b1	-
	PC := PC + 4	a2 := EM(R12)	b2 := a2 + 6A	Dm(R12) := b1	-
JMP R11, R12, dp	IR := AM(PC)	a1 := PC + 4	b1 := EM(R12)	c1 := PC + dp	-
	PC := PC + 4	a2 := PC + 4	b2 := PC2	c2 := PC + dp	-

9.

- Izan bedi honako begizia hau:
- a. Zerbat UAL batzuk eta komparagailu batek dira makina horretan?
- b. Ebazititzazu OP → ST, LD → OP, OPM → OPM eta OPM → OP aginduen arteko datu-dependenteziak, adierazi, zehatz-mehatz, egin dientezen zirkuitulaburuz giztak.
- c. Hartu kontuan datu hauek; batz beste, JMPX agindutako exekuzio-mailtasuna % 20 da, eta JMP agindutako % 15, hardwarek ez ditu jauzi ondorenko aginduak balisgabezen; jauzien latenzia-zikloak betetzeko, kompliadoreak agindu bat lortzen du kusuen erdian, eta bi agindu kusuen % 10ean. Datu horien arabera, zein izango da eraginkortasunaren galera jauzien eraginez?

B	D/Tx	A1	M	A2/Id
LD Rh, V[R1]	IR := AM(PC)	PC := PC + 4	A1 := EM(R1)	BL := A1 + 6V
OPM RB, V[R1], R1	IR := AM(PC)	PC := PC + 4	A1 := DECODE(R1)	C1 := DM(B1)
BRX R1, d	IR := AM(PC)	PC := PC + 4	A1 := EM(R1)	EM(R1) := C1 OP C2

10. Prozesadore segmentatu lineal batzuk sortutako programa horren nahi da:

loop: LD R3,A[R1]
LD R4,B[R1]
ADD M[R1,C[R3],R4]
ST A[R1],R4
BRX R3,loop

a) Prozesadoreak ez badu hardwaren datu- eta kontrol dependentziak ebazteko, zein izango da egiteko?

b) Programa horretan dauden RAW dependentziiek sortzen dituzten gelditak antzeko behar diren zirkuitulaburak proposatu. Zirkuitulaburak bakoizko adierazi zein agindu parentez den.

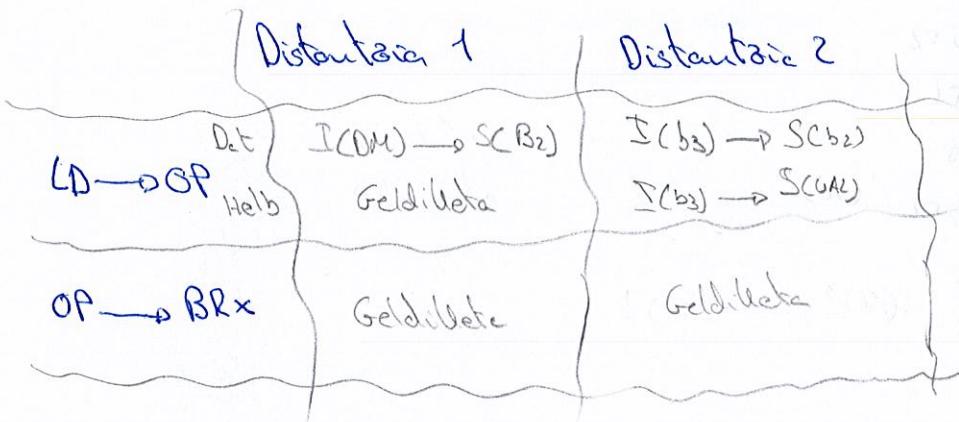
c) Kontrol dependentziak ebazteko, zein teknika inuditzen zaitu egoitza gara programaren exekuzioan?. Kalkulatu zerbat ziklo aurrezten diren begizta 200 aldiz exekutatzenean.

Betiztaren hiru bertsio idatzi ditugu non jauziak modu desberdinietan tratatzaten baitira. Jauzien latenzia esan zer teknika erabiliz den jauzi bakoitzera tratatzeko.

1. bertsioa	2. bertsioa	3. bertsioa
P0: LD R5,A(R1)	P0: LD R5,A(R1)	P0: LD R5,A(R1)
ADD R6,R5,#1	ADD R6,R5,#1	ADD R6,R5,#1
BZ R6,P1	ST A(R1),R6	ST A(R1),R6
ST A(R1),R6	R6,P1	R6,P1
ST B(R1),R0	NOP	ST B(R1),R0
JMP R2	ST B(R1),R0*	JMP R2
P1: NOP	P2: ADDI R1,R1,#1	P1: ADDI R1,R1,#1
LD R7,B(R1)	LD R7,B(R1)	LD R7,B(R1)
ADDI R8,R7,#1	ADDI R8,R7,#1	ADDI R8,R7,#1
ST B(R1),R8	ST B(R1),R8	ST B(R1),R8
ADDI R1,R1,#1	ADDI R1,R1,#1	ADDI R1,R1,#1
BNZ R2,P0	BNZ R2,P0	BNZ R2,P0
SUBI R2,R1,#N	SUBI R2,R1,#N	SUBI R2,R1,#N
NOP	NOP	NOP
	LD R5,A(R1)	LD R5,A(R1)

①

a)



b)

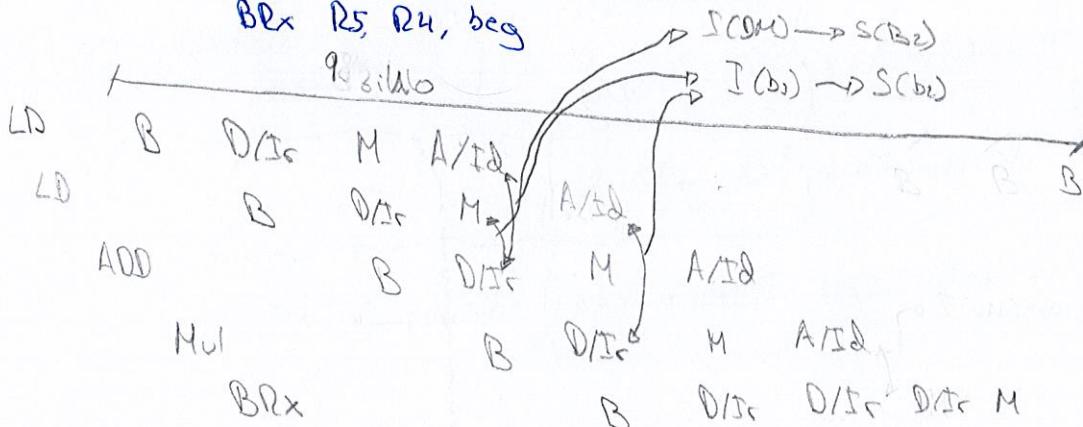
beg, LD R1, O(R4)

LD R3, O(R4)

ADD R5, O(R1), R3

Mul R4, O(R1), R2

BRx R5, R4, beg



c)

BRx → %20 Latenzzeit: 2

Agiindu bollerse %60an eta bi agiindu %10

bi agiindu sareze: 0.8:00 · 80 = 10

agiindu bollerse: 0.8:00 · 860 = 60 / 12
28:00 · 830 = 60

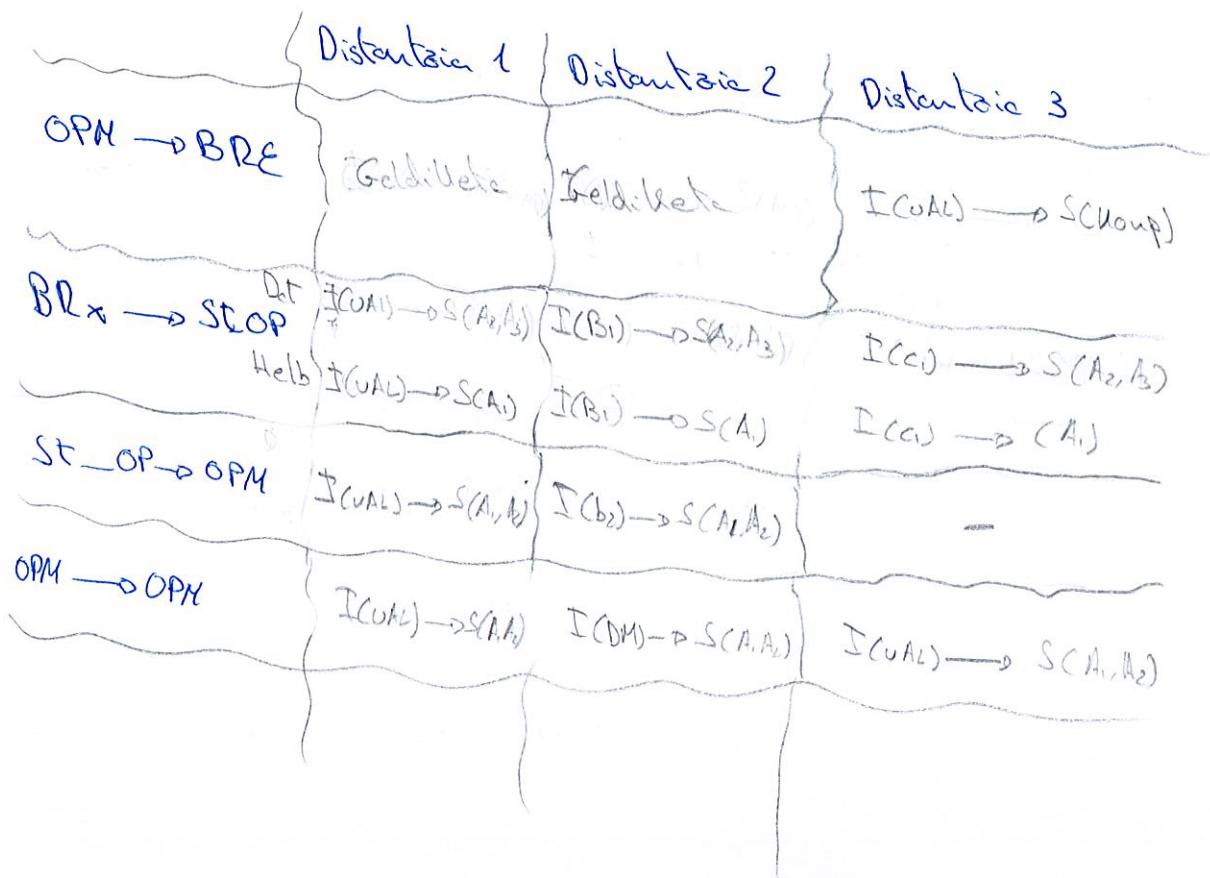
$$820 \cdot (830 \cdot 2 + 1 \cdot 860) = 824$$

②

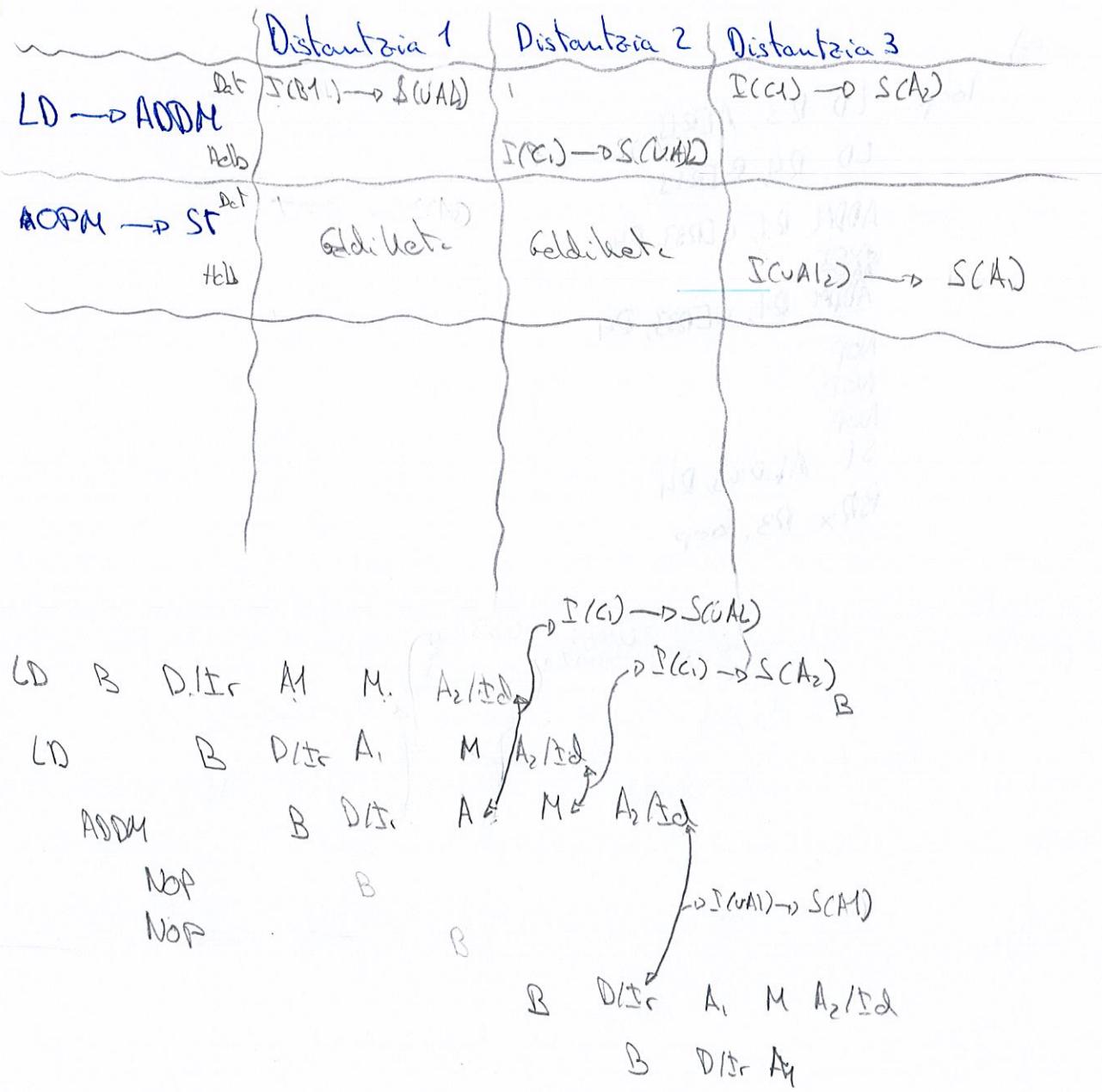
ADD M	r_5, r_2, r_3
BRE	$r_5, \#16$
ST-ADD	$O(r_4), r_5, r_2$
Div M	r_4, r_4, r_1
Sub M	r_3, r_8, r_8
ST-Sub	$O(r_6), r_5, r_1$
Mult M	r_6, r_8, r_8
ADD M	r_6, r_9, r_8

$OPM \rightarrow BRE$
 $BRE \rightarrow ST_OP$
 $ST_OP \rightarrow OPM$
 $OPM \rightarrow OPM$

a)



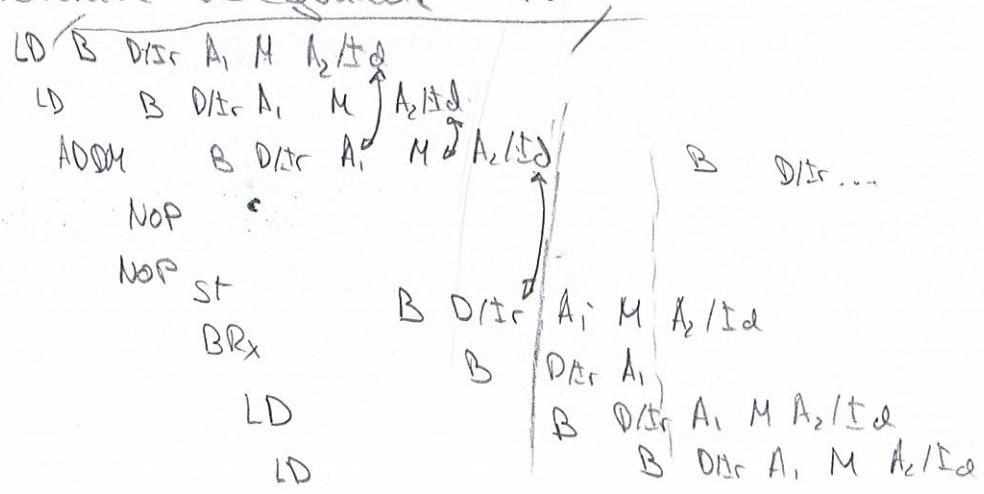
b)



Iteraziōn 9 zilito behar ditu exekutatzun. $200 \cdot 2 = 1400$ zilito

c) Jauzi atzerterre teknike de egorriene 7 zilito

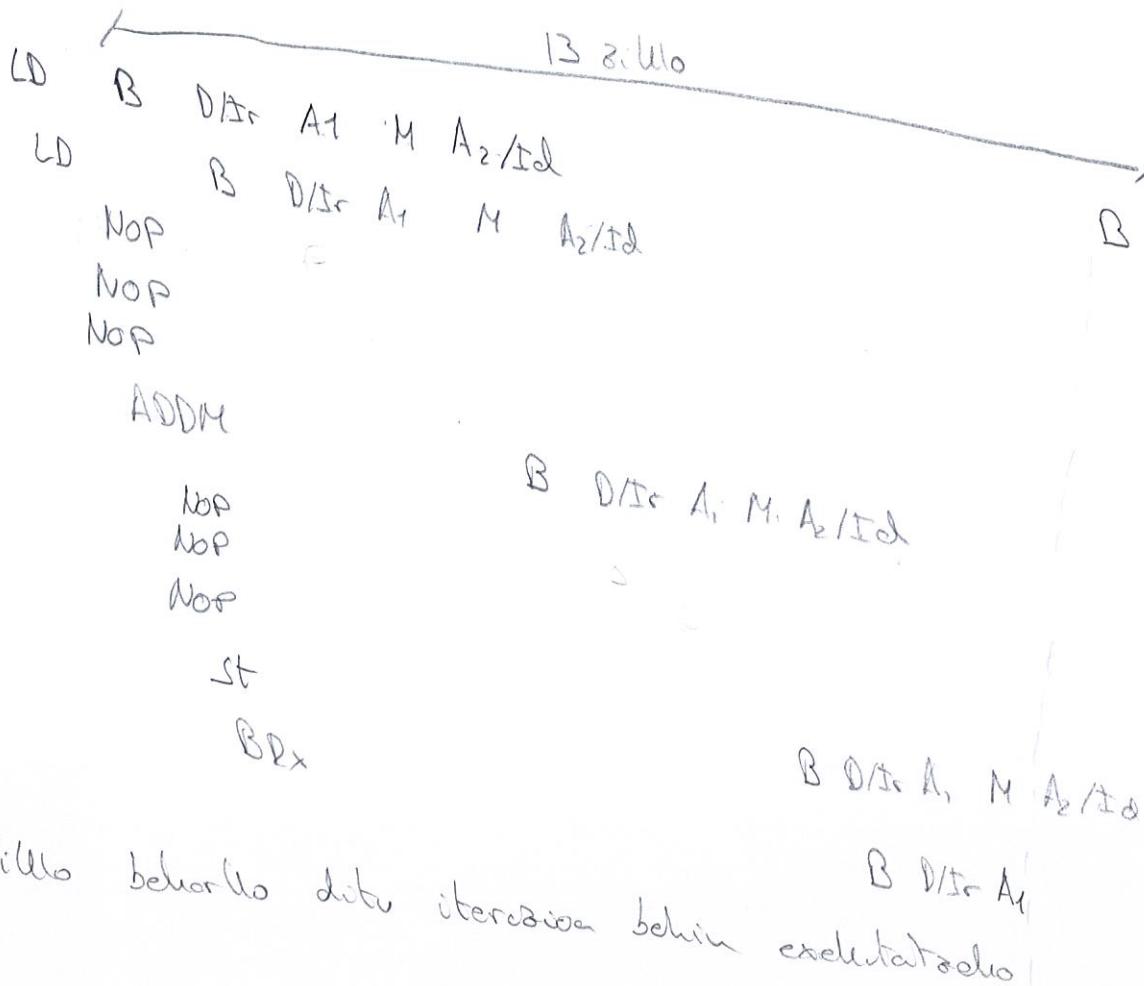
loop: ADDM
 LD R3, A[R1]
 LD R4, B[R1]
 LD R1, ([R3], R4)
 NOP
 NOP
 ST A[R1], R4
 BRx R3, loop
 LD R3, A[R1]
 LD R4, B[R1]



II. Arithmetik

a)

loop: LD R3, A[R1]
LD R4, B[R1]
NOP
NOP
ADDM R1, C[R3], R4
NOP
NOP
NOP
ST A[R1], R4
BRx R3, loop

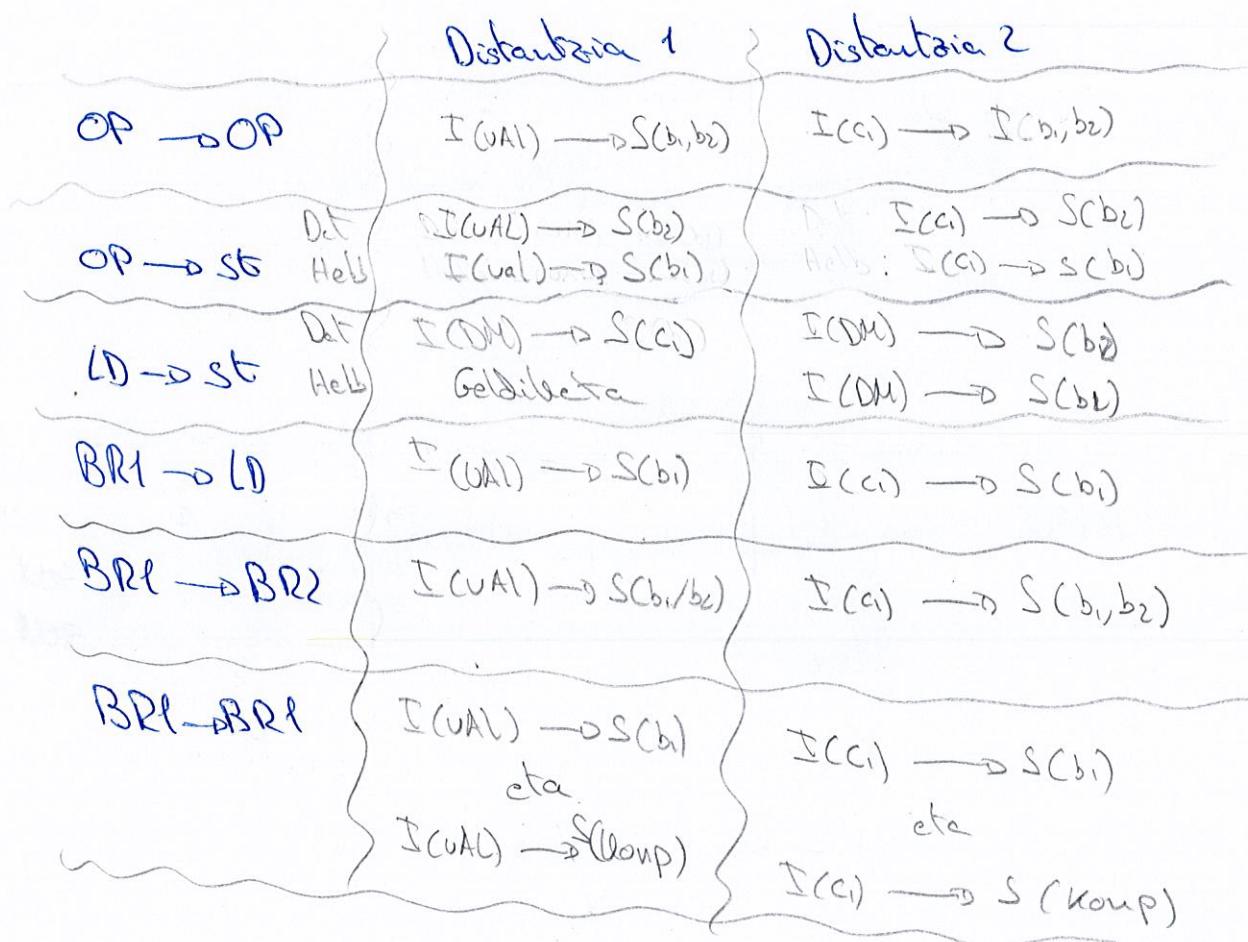


Ariveltal - Hauptzweigten Architektur

5. Ariveltal

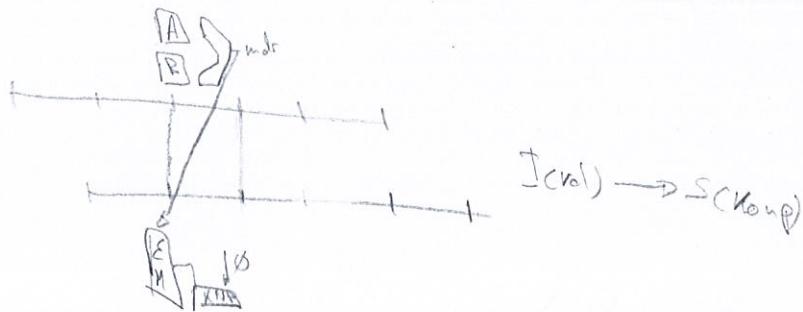
Row dependency

4 arivels dardane, izan daiteken distantzia agirdu batet
eta bestearren artean 2 izango da hor de 2 distantzia daude.



ADD R1, R2, R3

BLx, R1, etik

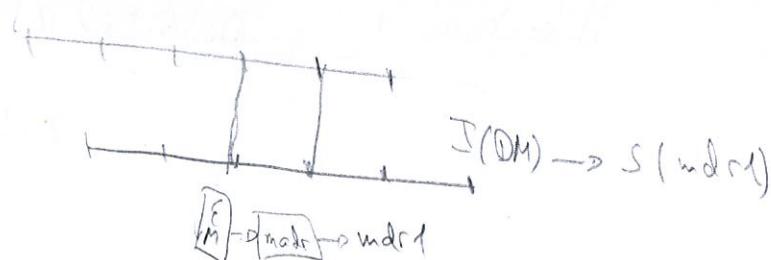


Umprogrammierbare hardware ablauffen

LD-ST

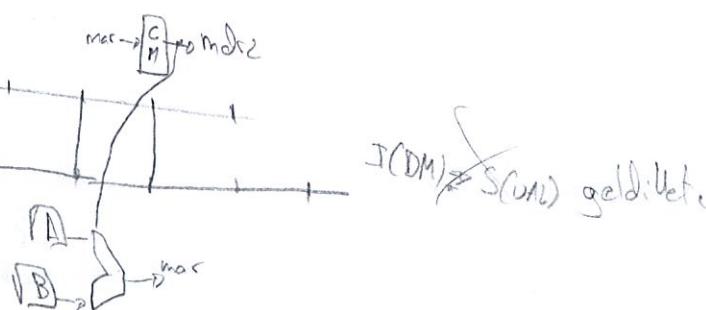
LD R1, A(R2)

ST R2, B(R3)



LD R1, A(R2)

ST R5, B(R1) = ADD



E. caribae

Ankündigung MA

$$R_1 = 99$$

deg: LD R₃, A(₀₁)

st B(R₁), R₃

BR1 R₁, deg

Nop

LD R₃, C(R₀)

a)

1) Zillenkalk Zirkuitlabour mit erzielbarer gabe:

LD ~~B, D/Ir, A, M/Id~~ B ~~D/Ir~~

st ~~B, D/Ir, D/Ir, A, M/Id~~ B

BR1 ~~B, B, B, D/Ir, A, M/Id~~

Nop

$$\text{Zillenkalk} = 7 \text{ Zillo} \times 100 + 4 = 704 \text{ Zillo}$$

Begleit

2) Zillenkalk ~~zweig~~ direkt & Berria LD erzielbarer kosten decken, Zillenkalk getrennt abrechnen

LD ~~B, D/Ir, A, M/Id~~ B $\xrightarrow{I(DM) \rightarrow S(C_2)}$ $\xrightarrow{I(C_1) \rightarrow S(b_1)}$

st ~~B, D/Ir, A, M/Id~~

BR1 ~~B, D/Ir, A, M/Id~~

$$\text{Zillenkalk} = 43 \text{ Zillo} \times 100 + 4 = 404 \text{ Zillo}$$

b)

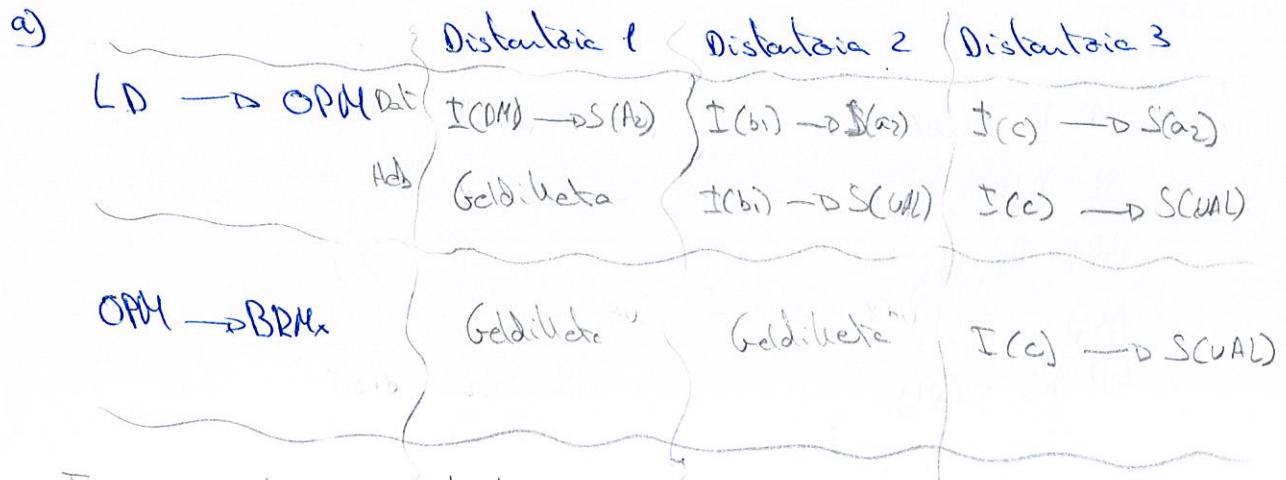
Zirkuitlabour mit + jahresatmosphäre

LD ~~B, D/Ir, A, M/Id~~ $\xrightarrow{I(DM) \rightarrow S(b_2)}$

BR1 ~~B, D/Ir, A, M/Id~~ $\xrightarrow{I(C_1) \rightarrow S(b_1)}$

st ~~B, D/Ir, A, M/Id~~ $\xrightarrow{I(C_1) \rightarrow S(u_1)}$

Zaribela



Jauzi-ag-i-duren letantaisi: 3 (angga da, Distantaic 3 an
pedalien batios klioe valari,

5)

$$11 \text{ zillo} \cdot 100 \text{ iterdice} = 1100 \text{ zillo}.$$

2

② LD R3, A(R1)
 beq; ADDI R4, R1, #1
 LD R5, B(R2)
 MULR R4, R3, C(R5)
 BZMx D(R4), beq

6. arrikete segida

LD R ₃ A(R ₁)	B	D/I _r	A	M/I _d	
st B(R ₁) R ₃		B	D/I _r	A	M/I _d
B/R ₁ R ₁ , beg			B	A/I _d	
LP R ₃ A(C ₁)				R	D/I _r
LD R ₃ A(R ₃)					B

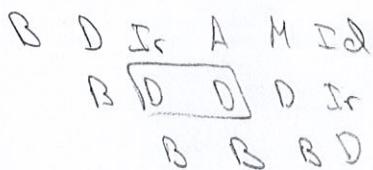
(+1)

Lehenergo adien apostu bat egindu non LD R₃ A(R₁) sartu dugu
 LD R₃ C(R₃) -ren tollakoa, auditoria saltoa ~~egin beharreko~~ baitu. Auditoria
 ixti egindako diagno

$$\delta = 3 \cdot 100 + 1 + 4 = 305$$

10. arriketa

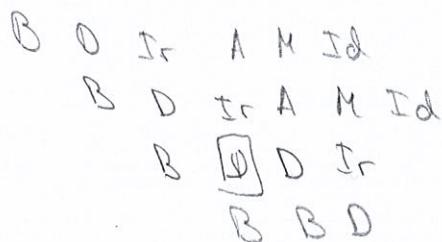
1. aukera biak jarrizan:



10 adiz egin behar leku bako

$$28110 \cdot 10 = 281100$$

2. aukera 2 distantzia



30 adiz zitiko bat gelditxo da.

$$2110 \cdot 30 = 30300$$

60 jauzi agindu exekutiboa, %20 huts egin da

$$12 \cdot 3 = 36$$

Galerak: $20 + 30 + 36 = 86$ zitiko

Anleitalk Nonpotagaiwen Arkitektur

Wien 1990

Lariketa

a) $UAL = 2$

Betragswert = 3

Konspiegelwerte = 1

b) . . .
OP \rightarrow St
St agendek Dative etc agendek
bereisten dito. Audoris banatu egingo
ditug.

Dat:

$$D_1: I(UAL_1) \rightarrow S(a_1)$$

$$D_2: I(b_1) \rightarrow S(a_1)$$

$$D_3: I(c_1) \rightarrow S(a_1)$$

$$D_4: I(d_1) \rightarrow S(a_1)$$

$$D_1: I(UAL_1) \rightarrow S(a_2)$$

$$D_2: I(b_1) \rightarrow S(a_2)$$

$$D_3: I(c_1) \rightarrow S(a_2)$$

$$D_4: I(d_1) \rightarrow S(a_2)$$

OPM \rightarrow OPA

Dat:

$$D_1: I(UAL_1) \rightarrow S(c_1)$$

$$D_2: I(UAL_2) \rightarrow S(b_1)$$

$$D_3: I(UAL_2) \rightarrow S(a_1)$$

$$D_4: I(d_1) \rightarrow S(a_1)$$

Heb:

$$D_1: Gedächte S(c_2)$$

$$D_2: Gedächte S(b_2)$$

$$D_3: I(UAL_2) \rightarrow S(a_2)$$

$$D_4: I(d_1) \rightarrow S(a_2)$$

OPM \rightarrow OP

D₁: Geldilleta

D₂: Geldilleta

D₃: I (iak) \rightarrow S(a₁, a₂)

D₄: I (di) \rightarrow S(a₁, a₂).

LD \rightarrow OP

Dat: D₁: Geldilleta

D₂: I(DH) \rightarrow S(a₁, a₂)

D₃: I(C8) \rightarrow S(a₁, a₂)

D₄: I(D8) \rightarrow S(a₁, a₂)

④

Lehenengo bertsioa gainditu egongo da Subir BN2-rei
azkenean jasaten duelako. eta, 12-rei baliar ea de itzango berdin
aldebatz lori egiten badugu.

Bigoarra bertsioa, gauko egongo da lehenengo aldiari nola
berdula den, bigorreraen beti ADDi exekutatu behar da, berria
azkena pizteko saldu egiten dugunean egongo da ADDi bat gertatu
exekutatuko baitugu.

Hirugarren bertsioa, ondo den. 3 apurtu egitea ditzu.

⑤

c)

JMP

PC = PC + d₈

Latentzia: 1

$$\text{Olera: } 80\% \cdot 1 \cdot 0.5 = 40\%$$

Berro bideratu galdean
badugu

JMPc_x

Latentzia: 3 Hirugarren biltzen

kerroetzen baiten PC.

$$\text{Olera: } 80\% \cdot 3 = 24\%$$

$$80\% + 40\% + 24\% = 144\%$$

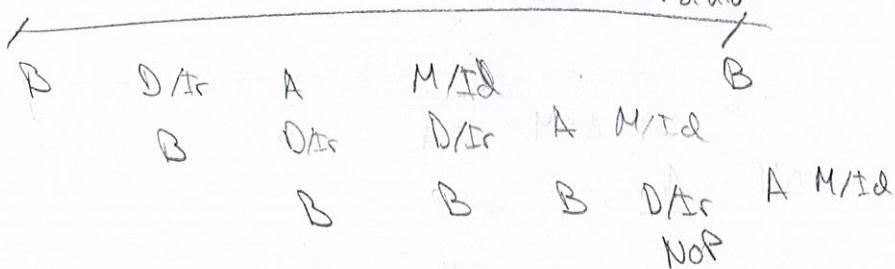
5. Anliegen

		Distanz 1	Distanz 2
OP → OP		I(UA) → S(b ₁ , b ₂)	I(c ₁) → S(b ₁ , b ₂)
OP → st	Dat Hab	I(UA) → S(b ₂)	I(c ₁) → S(b ₂)
LD → st	Dat Hab	I(DA) → S(c ₁)	I(DM) → S(b ₂) I(DM) → S(b ₁)
BR1 → LD		I(UA) → S(b ₁)	I(c ₁) → S(b ₁)
BR1 → BR2	Re	I(UA) → S(b ₁) etc I(UA) → S(komp)	I(c ₁) → S(b ₁) etc I(UA) → S(komp)

b) BR1 passieren undurch NOP hat keinen Einfluss da.

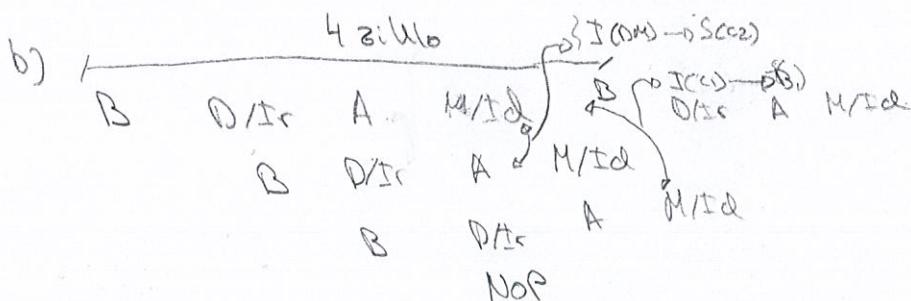
6. Anliegen

a) zirkular labornah erreichbar geben. τ_{ZU}



? zirk. Behar de begreife keinen exklusivz. z.

$$\text{Iterazio. z. } = \tau_{ZU} \cdot 100 + 4 = 204 \text{ z. }$$



$$\text{Iterazio. z. } = 4 \text{ z. } \cdot 100 + 4 = 404 \text{ z. }$$

b)

~~beg~~ LD R3, A(Q1)
 BR1 B(Q1), ~~beg~~] S(C1) → S(b2)
 ST B(Q1), R3] I(DM) → S(b2)
 LD R3, C(Q1)

LD B D/Ic A M/Ia] S(C1) → S(b2)
 BR1 B D/Ic A] I(DM) → S(b2)
 ST B D/Ic A M/Ia] I(C1) → S(VAL)
 B D/Ic A M/Ia

LD R3, A(Q1)

~~beg~~: ST B(Q1), R3
 BR1 R1 ~~beg~~

LD R3, A(Q1)

LD R3 C(Q3)

3 z. u. o.
 LD B D/Ic A] D/I(UA1)
 ST B D/Ic A M/Ia
 BR1 B D/Ic A M/Ia
 LD B D/Ic A] I(UA1) → S(b1)
 LD B

$$3 \text{ z. u. o.} \cdot 100 + 4_{+1} = 305 \text{ z. u. o.}$$

8. Hartu kontuan honako begizta hau eta aginduen latenzia:

```

1 beg:    FLD   F1,A,(R1) 1
2          FLD   F2,B,(R1) 1
3          FDIV  F3,F1,F2 2
4          FST   A(R1),F3 2
5          FLD   F4,C(R1) 3
6          FMUL  F5,F4,5.0 3
7          FLD   F6,D(R1) 3
8          FADD  F7,F5,F6 4
9          FST   C(R1),F7 4
10         ADDI  R1,R1,#8 4
11         SUBI  R2,R2,#1 4
12         BNZ   R2,beg 4

```

a. Kalkulatu begizaren iterazio batzen ezezungo-denbora (ziklotan).

b. Marraztu dependentez-grafoa eta berrordenatu kodea *list scheduling* algoritmoa erabiliz. Jauzia tratzeko, erabili ezazu jauzi aitzera teknika. Idatzi begizta berria eta kalkula izazu iterazio baten ezezungo-denbora eta lortu den azelerazio-faktorea.

Emititzu kontuan harria, arrezoitu lortuko den azelerazio-faktorea begizta bi aldiz zabaldu eta kodea berrordenatzan bada. Limiteean, begizta *n* aldiz zabaldua, zein izango litzateke lor zitzekeen azelerazio-faktore maxima?

9. Ziklo anitzeko prozesadore batean, honako begizta hau ezeektu behar da. Prozesadoreak $Id \rightarrow Ir$ zirkuitulaburraak egiten ditu, eta mota bakoitzeko unitate funtzional bana du, guztiak segmentatuak.

```

Aginduen segmentazio-eskema
a: FLD   F1,A,(R1)
     FLD   F2,B,(R1)
     FMUL  F3,F2,F1
     FST   C(R1),F3
     FLD   F5,-C(R1)
     FDIV  F6,F5,F1
     FST   A(R1),F6
     SUBI  R1,R1,#8
     BEQ   R1,b

```

a. Azier ezazu begizta eta saia zaitetx kodea optimizatzen. Idatzi kode optimizatua eta kalkula ezazu begizaren iterazio batzen ezezungo-denbora (ziklotan).

b. Berrordenatu kodea *list scheduling* algoritmoa eta jauzi aitzera erabiliz. Idatzi begizta berria eta kalkula izazu iterazio batzen ezezungo-denbora eta lortu den azelerazio-faktorea.

c. Zabaldu ezazu begizta bi aldiz, eta errepetuki b. atala.

10. Begizta hau ziklo anitzeko konputagailu arrunt batean ezeektu behar da. Mota bakoitzeko unitate funtzional bana diago, guztiak segmentatuak.

```

Aginduen latenzia
b: FLD   F1,A,(R1)
     FLD   F2,B,(R1)
     FMUL  F3,F2,F1
     FST   A(R1),F3
     FLD   F4,C(R1)
     FADD  F5,F4,F3
     FST   C(R1),F4
     FLD   F6,D(R1)
     FADD  F7,F6,F5
     FST   B(R1),F7
     ADDI  R1,R1,#8
     SUBI  R2,R2,#1
     BNZ   R2,b

```

a. Kalkula ezazu begizaren iterazio bat ezeektuzeko behar den ziklo kopurua.

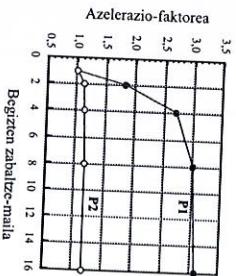
b. Marraztu dependentez-grafoa eta berrordenatu kodea *list scheduling* algoritmoari jarraituz, erabili ezazu jauzi aitzera teknika. Idatzi begizta berria, eta kalkula ezazu jatorrizko begizarekiko azelerazio-faktorea.

c. Zabaldu begizta bi aldiz eta berrordenatu kodea. Kalkulu a. eta b. atalekiko azelerazio-faktorea.

1. Irudiko grafikoan, begiztak n aldiz zabalduz gero

programa baten ekzekuzioa hor daitekeen azelerazio-faktorearen ageri da, bi programa jakinetarako: P1 eta P2.

Azelitzazu bi programa horien portera eta portera hori justifikatzeko duden programen ezagunariak.

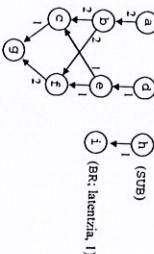


2.

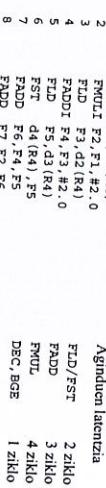
Honako bi grafo hauean, bi begiztakoa aginduak arteko dependenziak ageri dira. Ekzekuzio-denbora

- ziklo kopurua begizta-iterazio aginduak eta erabil ezazu jauzi atzeratua. Kasu bakoitzean, adieraz:
- begiztak zabalzten direnean, ziklo kopurua laburzearren zergatia.

Begiztak zabalzten direnean, begiztak 2, 3 edo 4 aldiz zabalten direnean.



3. Begizta hau prozesadore segmentatu batean ejecutatu behar da:



Aginduenean latenzia

1. beg: FLD F1, G1(R4)

2. beg: FMUL F2, E1 #2.0

3. FLD F3, G2(R4)

4. FADD F4, F3, #2.0

5. FST F5, G3(R4)

6. FST F6, G4(R4), F5

7. FADD F7, F4, F6

8. FADD F8, F7, F6

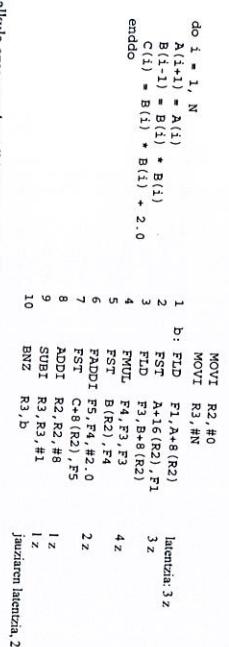
9. FST d5(R4), F7

10. DEC R4

11. BEG R4

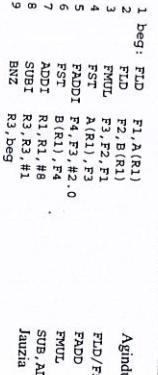
12. R4, beg

7. Begizta hau prozesadore segmentatu batean ejecutatu behar da:



- a. Kalkula ezazu zorbat ziklo behar diren begiztaren iterazio bat exekuzio-denbora. Zein da lortuko den azelerazio-faktorea (speed-up)?
- b. Berrodenatu begizta *list scheduling* algoritmoa eta jauziaren kalkula ezazu iterazio baten exekuzio-denbora. Zein da lortuko den azelerazio-faktorea?
- c. Kalkula ezazu dependenzi-a-grafoa eta berrodena izazu begiztaren aginduak exekuzio-denbora minimoa izan daidin. Erabil ezazu jauzi azeratua teknika. Kalkula ezazu begiztaren exekuzio-denboran lortuko den aurrepena, etgunekotan, ordenazio berria dela eta.

4. Begizta hau prozesadore segmentatu batean ejecutatu behar da:



Aginduenean latenzia

1. beg: FLD F1, A(R4)

2. FLD F2, B(R4)

3. FMUL F3, C(R4)

4. FLD F4, F3, #2.0

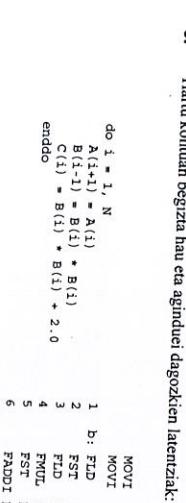
5. FST B(R4), F4

6. ADD R1, R1 #8

7. SUB R3, R2, #1

8. BEG R3, beg

Mazaz ezazu begiztaren dependenzi-a-grafoa, eta berrodena izazu begiztaren aginduak exekuzio-denbora minimoa izan daidin. Erabil ezazu jauzi azeratua teknika. Kalkula ezazu begiztaren exekuzio-denboran lortuko den aurrepena, etgunekotan, ordenazio berria dela eta.

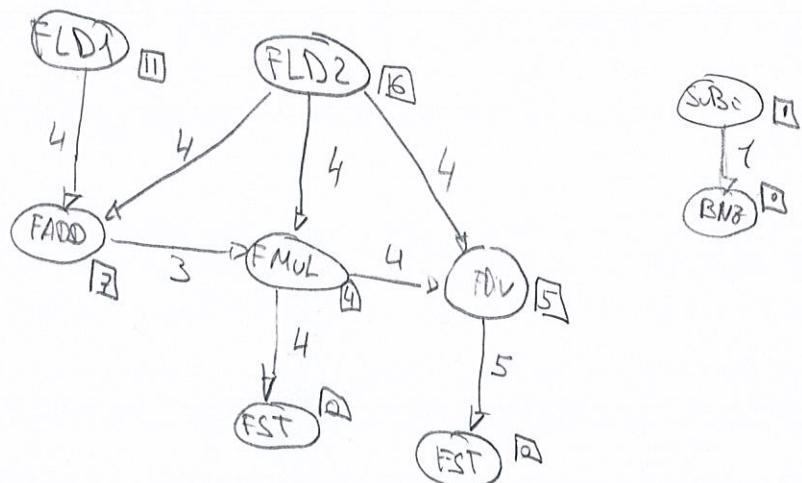


- a. Kalkula ezazu iterazio bat exekuzteko behar den ziklo kopurua.
- b. Egia ezazu dependenzi-a-grafoa eta berrodenatu koden (exekuzio-denbora laburzearren). Jauziak exekuzio-denbora eta lortu den azelerazio-faktorea.
- c. Zabaldu ezazu begizta bi aldiz, eta errepikau b. atala.

- a. Kalkula ezazu zenbat ziklo behar diren begizta-iterazio bat exekuzatzeko.
- b. Erabil itzazu berrodenatez-algoritmoa eta jauzi azeratua, eta berridatz begizia. Zenbat ziklo behar dira orain iterazio-kot?

3. Architettonico

a)



23 zillioli iterazione escluso tutto

FLD1	FLD2	.	.	FADD	.	FMUL	.	FST / FDIV	.	.	FST / SUBF	.	BNZ	.
------	------	---	---	------	---	------	---	------------	---	---	------------	---	-----	---

c)

FLD1	FLD2	FLD10	FLD20	SUBF	FADD	FADD'	FMUL	FMUL'	FST / FDIV	FST' / FDiv'	.	.	FST / BNZ / FST'
------	------	-------	-------	------	------	-------	------	-------	------------	--------------	---	---	------------------

b)

FLD1	FLD2	.	.	FADD	.	FDIV / FMUL / SUB	.	FST	BNZ	FST	.	.	.
------	------	---	---	------	---	-------------------	---	-----	-----	-----	---	---	---

7 zillioli speed-up-a

