DIRO
IFT 1215

DÉMONSTRATION N° 7

– Correction –

Max Mignotte

DIRO, Département d'Informatique et de Recherche Opérationelle, local 2377.
http: //www.iro.umontreal.ca/∼mignotte/ift1215/
E-mail: mignotte@iro.umontreal.ca

Excercice 7.2

```
20 LDA 50 # 550
21 ADD 51 # 151
50 724
51 006
```

a.

LDA 50:

PC(20)	→	MAR(20)
MDR(550)	→	IR(550)
IR [address](50)	→	MAR(50)
MDR(724)	→	A(724)
PC + 1	→	PC(21)

b.

ADD 51:

PC(21)	→	MAR(21)
MDR(151)	→	IR(151)
IR [address](51)	→	MAR(51)
A(724) + MDR(006)	→	A(730)
PC + 1	→	PC(22)

Excercice 7.7

a.

unsigned number : remplacer les bits par 0.

- shift 2 bits à gauche : Si au moment de chaque schift la valeur du bit déplacé est différent de 1 alors multiplication par 4. Autrement débordement.
- shift 1 bit à droite : division par 2.

b.

signed number : remplacer les bits par 0. Résultats erronés. Il faut tenir compte du bit de signe.

c.

signed number : garder toujours la même valeur du bit de signe. Pour un shift à droite insérer la valeur du bit de signe au lieu de 0.

- shift 2 bits à gauche : Si au moment de chaque schift(2 en tout) la valeur du bit de signe n'est pas changée par rappor à la valeur initiale alors le résultat est une multiplication par 4. Autrement il y a débordement.
- shift 1 bit à droite : division par 2.

Excercice 7.12

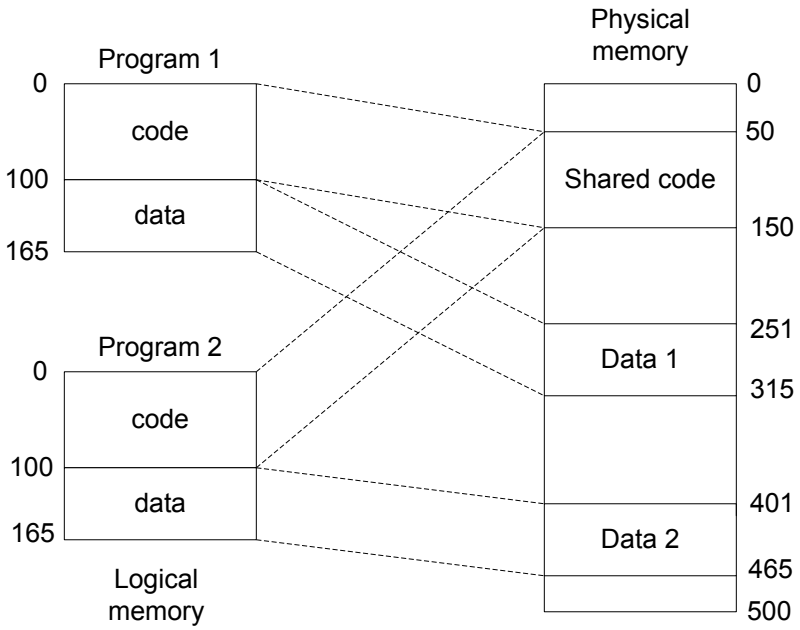
	PC	→	MAR
	MDR	→	IR
	IR [address](XX)	→	MAR
	MDR	→	IAR
0XX			
0YY	PC + 1	→	PC
	PC	→	MAR
	MDR	→	IR
	IR [address](YY)	→	MAR
	IAR	→	MDR
	PC + 1	→	PC

Excercice 7.16

MOV REG1 REG2 :			**LDA XX** :		
PC	→	MAR	PC	→	MAR
MDR	→	IR	MDR	→	IR
IR [REG1]	→	IR [REG1]	IR [address]	→	MAR
			MDR	→	A
PC + 1	→	PC	PC + 1	→	PC

On gagne une étape qui consiste à lire la valeur de la mémoire (généralement c'est l'étape la plus lente comparée aux autre étapes que le CPU exécute).

Excercice 8.15



Excercice 8.16

	Page	Frame
A	0	31
	1	32
	2	33
	3	34
	4	35
	5	36
	6	37
	7	38
	8	39
	9	40
B	10	45
	11	46
	12	47
	13	48
	14	49
C	15	08
	16	09
	17	10
	18	11
	19	12

CPU & Memory (1)

Exercice 7.2

a. (BL2-) Working from the F/E cycle for instruction 20,

PC → MAR	
MDR → IR	← final value of IR = 550
IR [address] → MAR	← final value of MAR = 50
MDR → A	← final valur of MDR and A = 724
PC + 1→ PC	← final value of PC = 21

b. (BL2-)

	PC	MAR	MDR	IR	A
PC → MAR	21	21	724	550	724
MDR → IR	21	21	351	151	724
IR [address] → MAR	21	51	351	151	724
A + MDR → A	21	51	006	151	730
PC + 1 → PC	21	51	006	151	730

Exercice 7.7

(all BL2)

a. Shifting an unsigned number two bits to the left multiplies it by four, unless the value overflows. Overflow occurs if either of the two leftmost bits is a 1.

b. Any mix of 1's and 0's in the three leftmost bits will cause an overflow, and possibly a sign change as well. If all three leftmost bits are the same, the value is algebraically quadrupled.

c. The left shift still requires the leftmost bits to be the same, otherwise overflow occurs. However, at least the sign is now correct. Right shifts will divide the value in half for each bit shifted, provided that the sign bit is carried to the right each time. This assures a succession of 1's or 0's at the leftmost bit positions, and preserves the sign.

Exercice 7.12

(BL3) There are five basic steps to be performed:

a. fetch the instruction
b. retrieve the data from memory location XX
c. save the retrieved data in IAR
d. fetch the next location to get address YY
e. store the data from IAR to address YY

The following F-E cycle will do the job. The steps above are identified at the right:

PC → MAR	
MDR → IR	(step a)
IR [add] → MAR	(step b)
MDR → IAR	(step c)
PC + 1 → PC	
PC → MAR	
MDR → IR	(step d)
IR [add] → MAR	
IAR→ MDR	(step e)
PC + 1 → PC	

Exercice 7.16

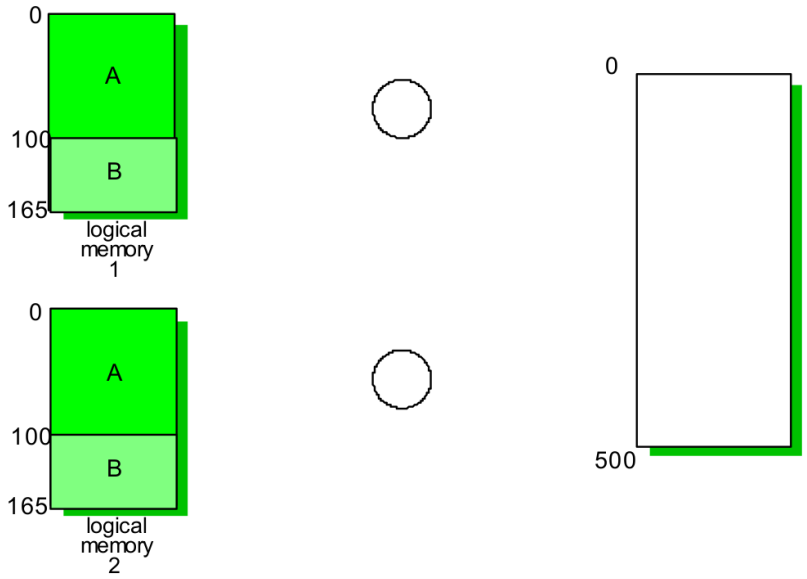
(BL3) For the MOVE instruction,

PC → MAR
MDR → IR
GPR (IR[Addr1]) → GPR (IR[Addr2])
PC + 1 → PC

Comparing this F-E cycle with that of the LOAD instruction, it it is apparent that the MOVE instruction eliminates the second memory access required to obtain the data from memory with the LOAD instruction. Since memory accesses are the slowest operation, (See 7.15), the MOVE operation will be significantly faster than the LOAD.

CPU & Memory (2)

Exercice 8.15



Exercice 8.16

(BL2)

logical page	physical page	logical page	physical page
0	31	10	45
1	32	11	46
2	33	12	47
3	34	13	48
4	35	14	49
5	36	15	8
6	37	16	9
7	38	17	10
8	39	18	11
9	40	19	12