

# 7.1. Encodage des instructions

- Code opération (*opcode*) : permet d'identifier l'instruction

*Opcode* ADD (AL, EAX), donnée

0000010w

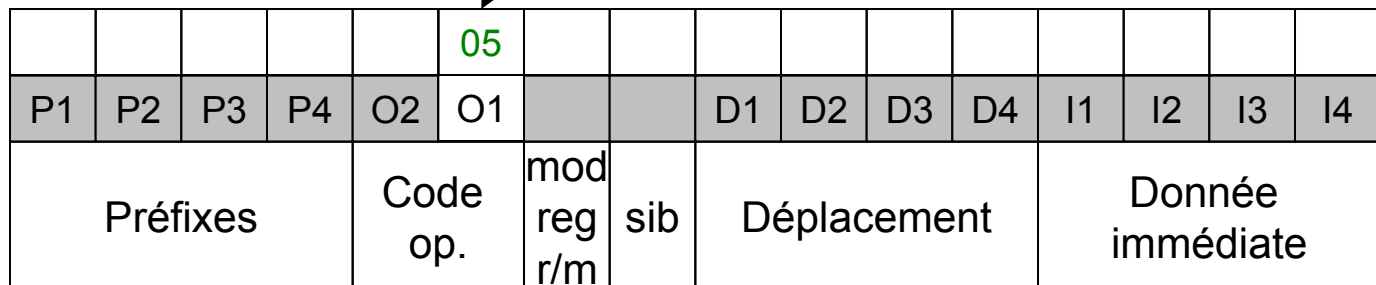
w = 0 → opérandes 8 bits

w = 1 → opérandes 32 bits

On encode ADD EAX, 2 → opérandes 32 bits, w = 1.

*Opcode* ADD EAX, donnée

00000101 = 05h



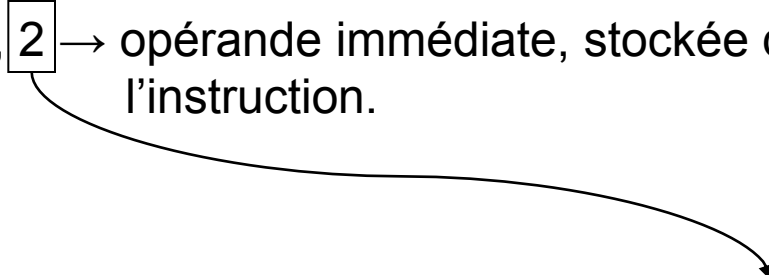
# 7.1. Encodage des instructions

- Opérande destination : identifie l'adresse ou sera rangé le résultat

On encode ADD EAX, 2 → l'opérande destination est implicite

- Opérande source : identifie ou trouver la première opérande de l'opération

On encode ADD EAX, 2 → opérande immédiate, stockée dans l'instruction.



					05							02	00	00	00
P1	P2	P3	P4	O2	O1			D1	D2	D3	D4	I1	I2	I3	I4
Préfixes				Code op.		mod reg r/m	sib	Déplacement				Donnée immédiate			

# 7.1. Encodage des instructions

- On obtient pour l'encodage de ADD EAX,2 :  
05 02 00 00 00
- Seuls les octets significatifs (utilisés dans l'instruction) sont effectivement stockés en mémoire.

					05							02	00	00	00
P1	P2	P3	P4	O2	O1			D1	D2	D3	D4	I1	I2	I3	I4
Préfixes				Code op.		mod reg r/m	sib	Déplacement				Donnée immédiate			

# 7.1. Encodage des instructions

- **ADD ECX, [EBX + ESI \* 4 + 12]**

*Opcode* ADD reg/mém, reg/mém

000000dw

d = 0 → destination = reg/mém  
source = reg

d = 1 → destination = reg  
source = reg/mém

w = 0 → opérandes 8 bits

w = 1 → opérandes 32 bits

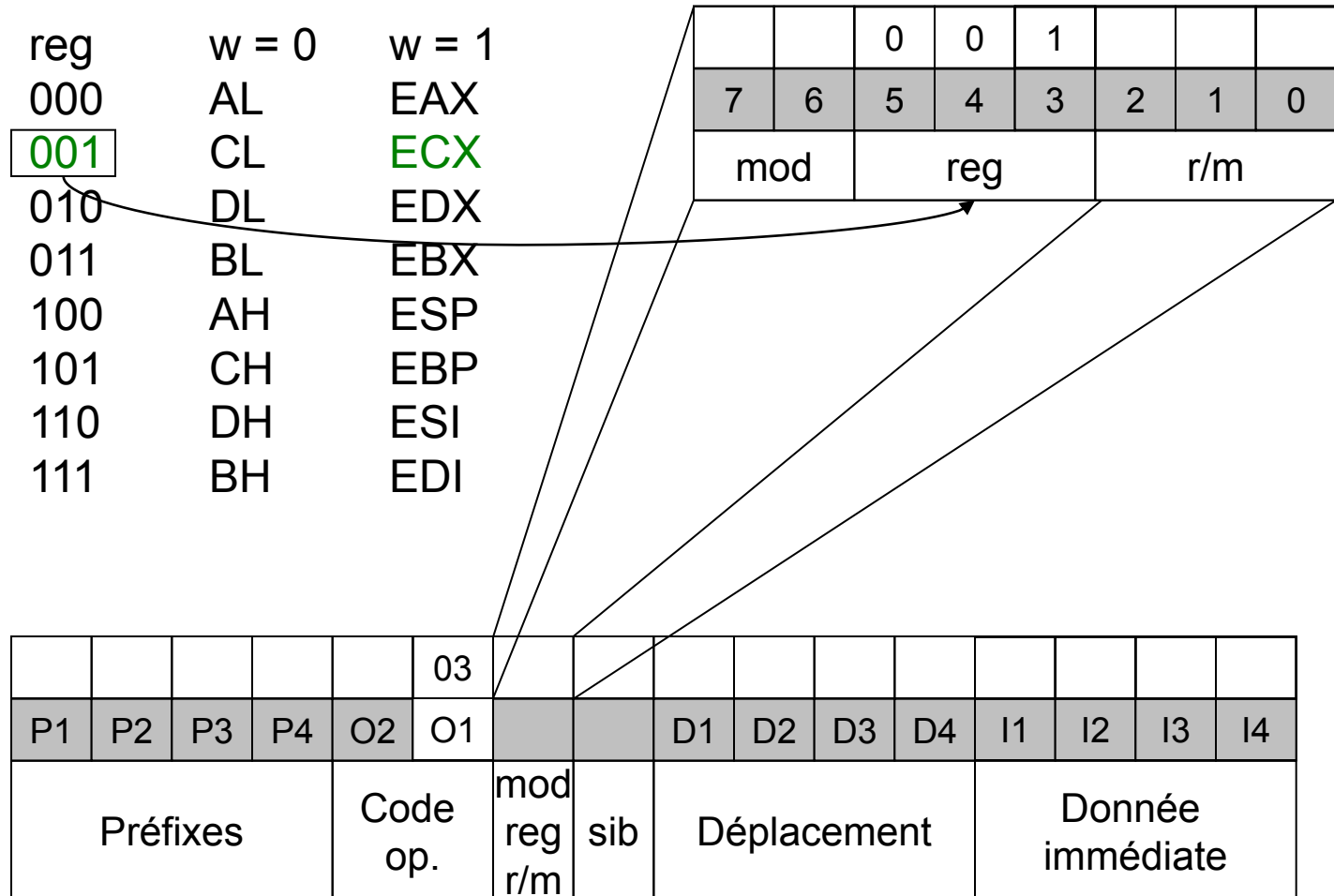
*Dans ce cas, opérandes 32 bits,  
ECX ← [EBX + ESI \* 2 + 12] + ECX  
donc d ← 1, w ← 1.*

00000011 = 03h

					03										
P1	P2	P3	P4	O2	O1			D1	D2	D3	D4	I1	I2	I3	I4
Préfixes				Code op.		mod reg r/m	sib	Déplacement				Donnée immédiate			

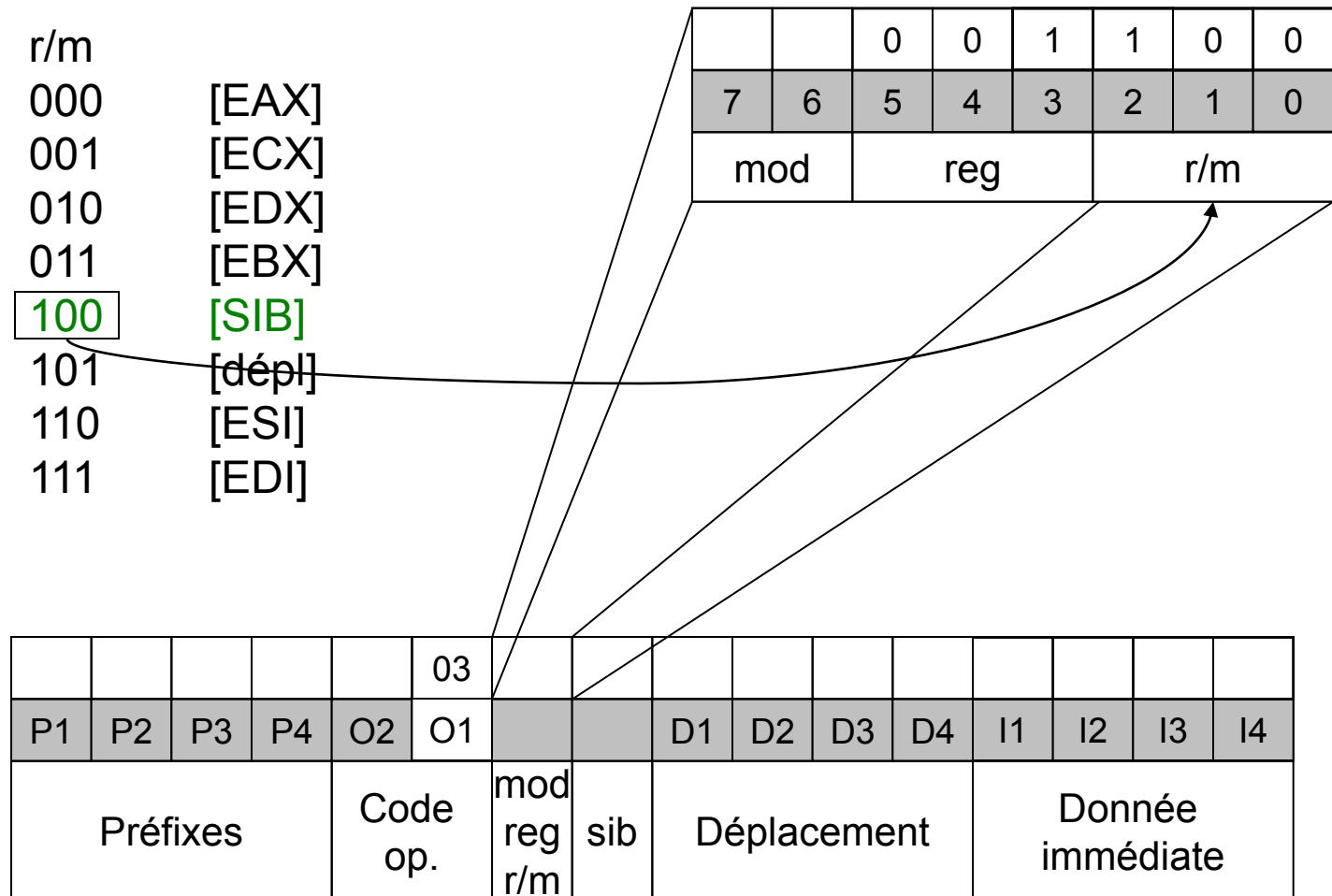
# 7.1. Encodage des instructions

- Opérande destination : registre ECX



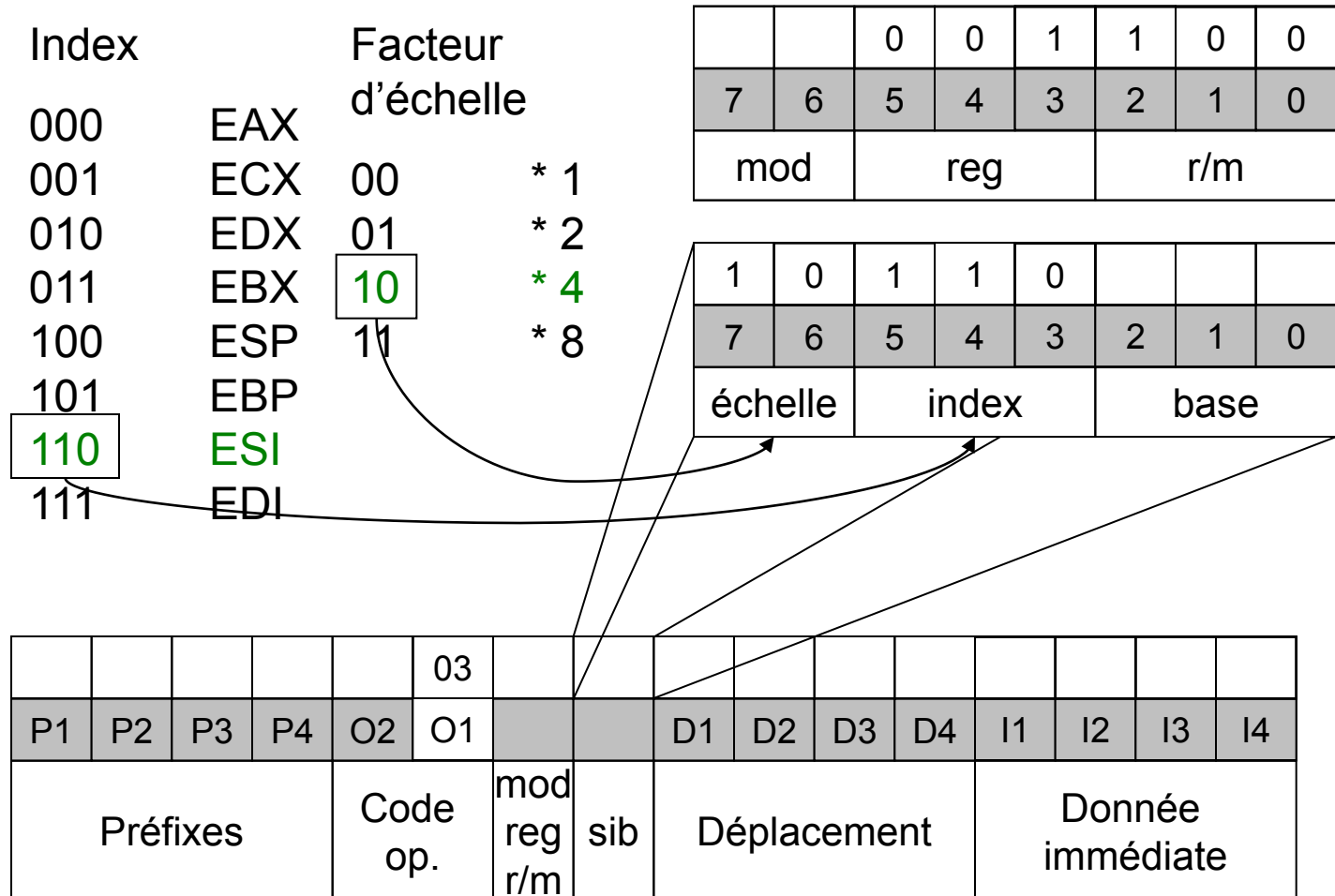
# 7.1. Encodage des instructions

- Opérande source :  $[EBX + ESI * 4 + 12]$



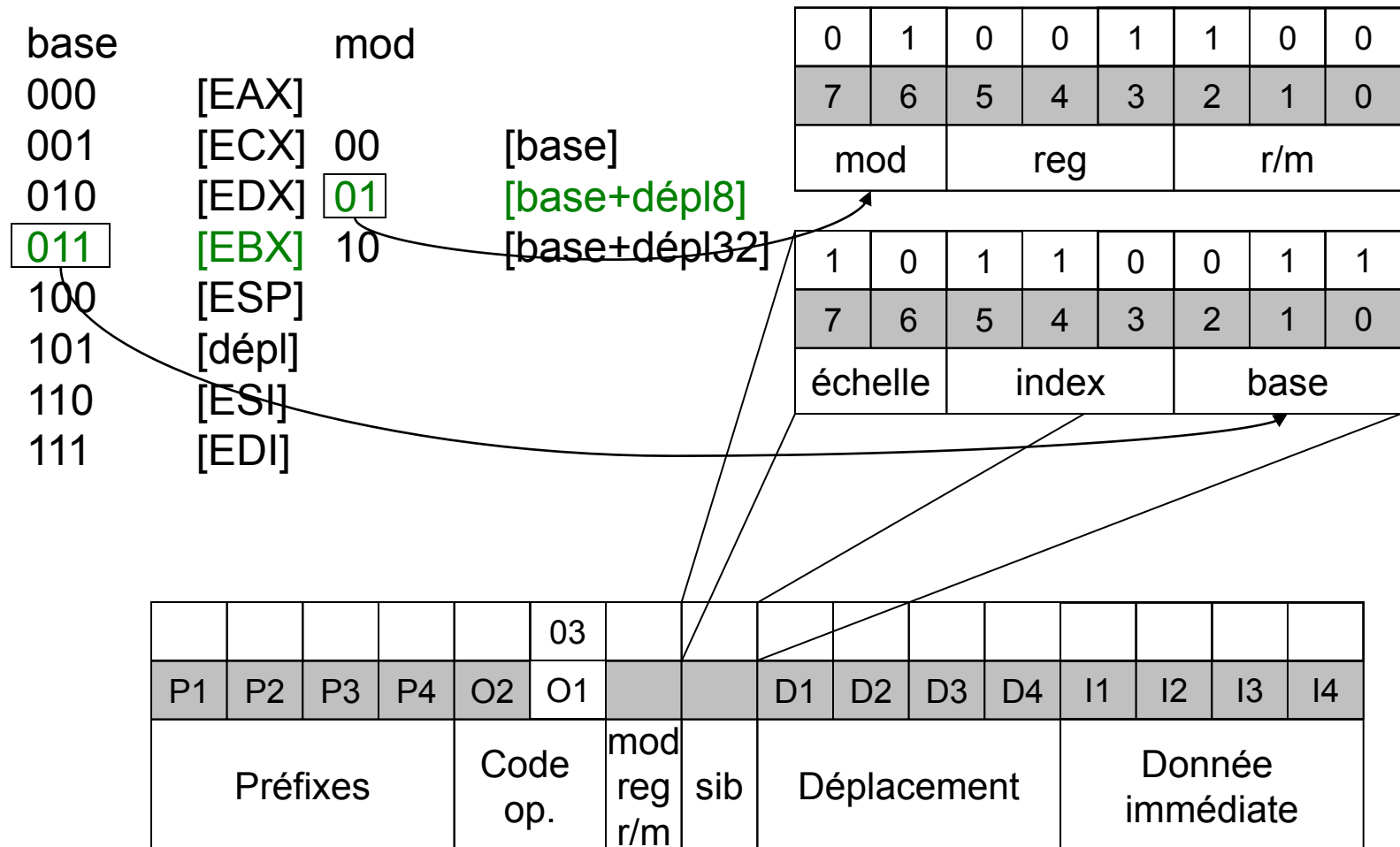
# 7.1. Encodage des instructions

- Opérande source :  $[EBX + ESI * 4 + 12]$



# 7.1. Encodage des instructions

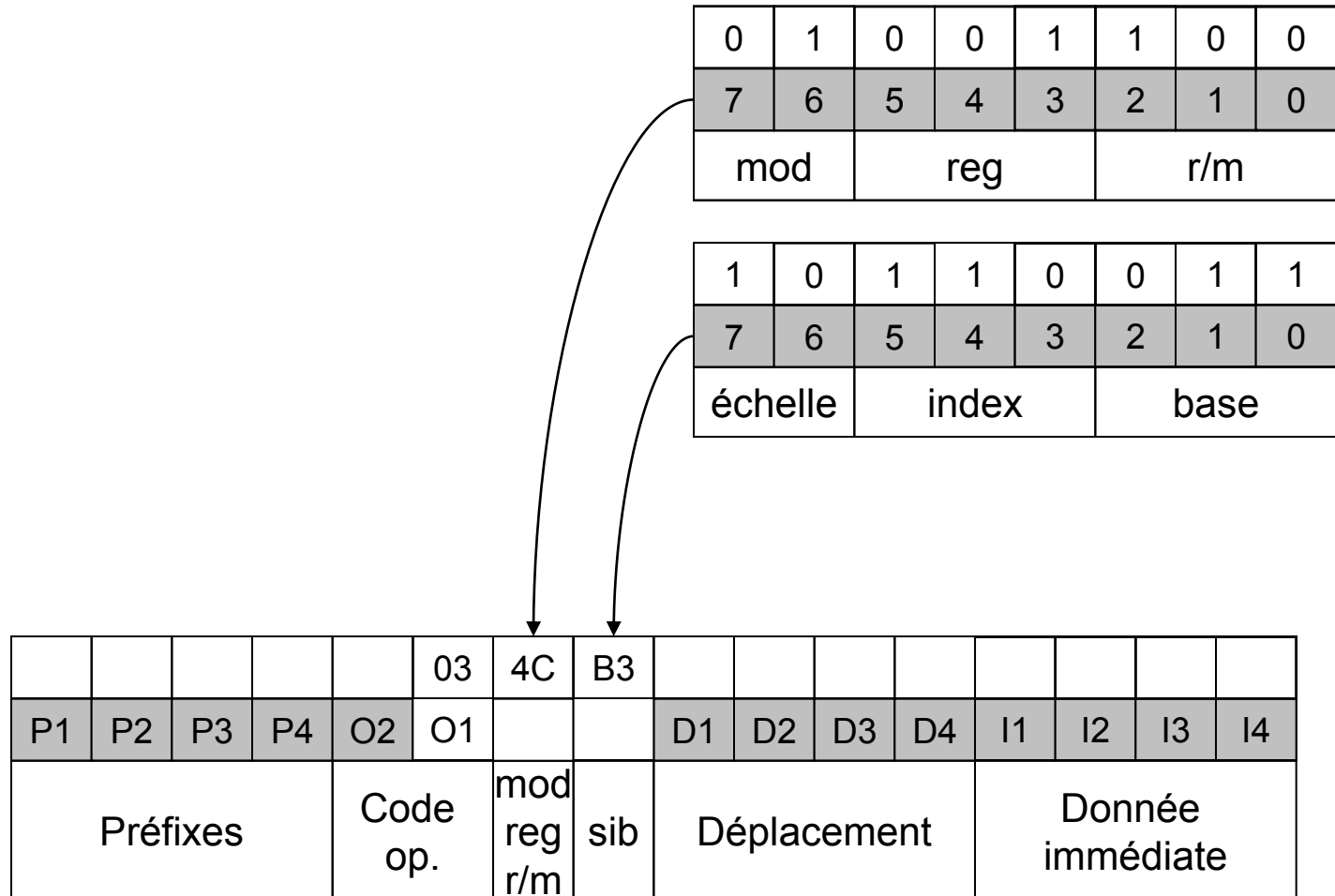
- Opérande source :  $[EBX + ESI * 4 + 12]$





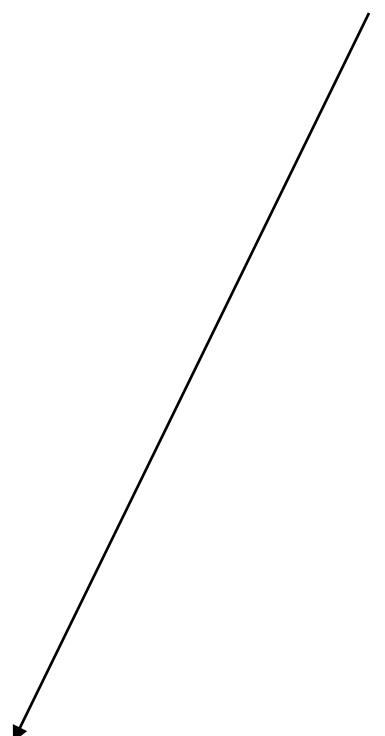
# 7.1. Encodage des instructions

- Opérande source :  $[EBX + ESI * 4 + 12]$



# 7.1. Encodage des instructions


- Opérande source :  $[EBX + ESI * 4 + 12]$
- Encodage final :  
03 4C B3 0C



					03	4C	B3	0C							
P1	P2	P3	P4	O2	O1			D1	D2	D3	D4	I1	I2	I3	I4
Préfixes				Code op.		mod reg r/m	sib	Déplacement				Donnée immédiate			

# 7.1. Encodage des instructions

- Les préfixes permettent d'encoder dans l'instruction des informations supplémentaires :
  - Taille d'opérande  
ADD CX, [EBX + ESI \* 4 + 12]
  - Substitution de segment  
ADD CX, ES:[EBX + ESI \* 4 + 12]...qui est donc encodée 66 26 03 4C B3 0C



		66	26		03	4C	B3	0C							
P1	P2	P3	P4	O2	O1			D1	D2	D3	D4	I1	I2	I3	I4
Préfixes				Code op.		mod reg r/m	sib	Déplacement				Donnée immédiate			