3 Le fonctionnement d'un ordinateur

Peter Schlagheck

Université de Liège

3 Le fonctionnement d'un ordinateur

3.1 La machine de Turing

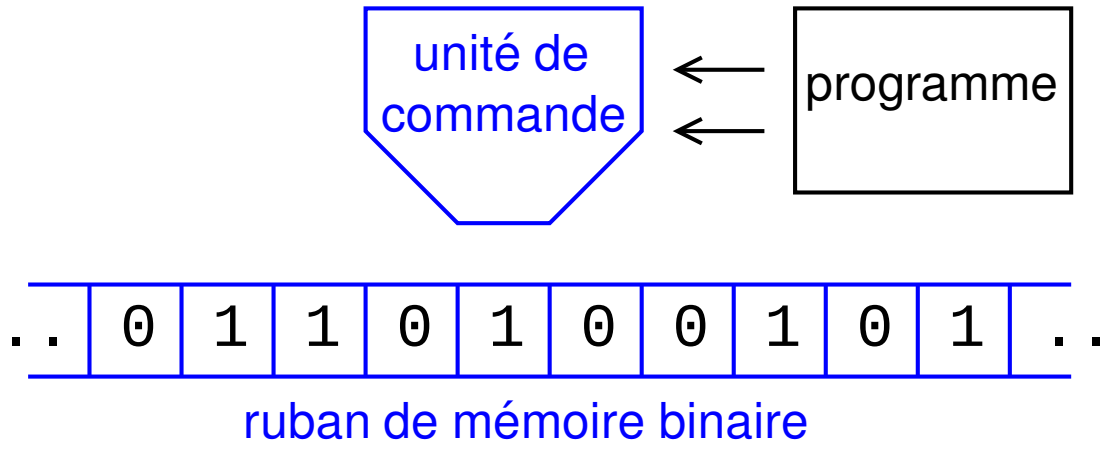
Alan Turing (1912 - 1954), mathématicien

$\longrightarrow$ grande contribution au déchiffrement des codes allemands "Enigma" pendant la deuxième guerre mondiale (ordinateur électromécanique "bombe", 1940)

$\longrightarrow$ conception de la machine de Turing (1936)
= une machine de calcul universelle qui sait effectuer tous les algorithmes numériques possibles

. . . afin de démontrer qu'il n'est pas possible de déterminer si de tels algorithmes s'arrêtent ou non

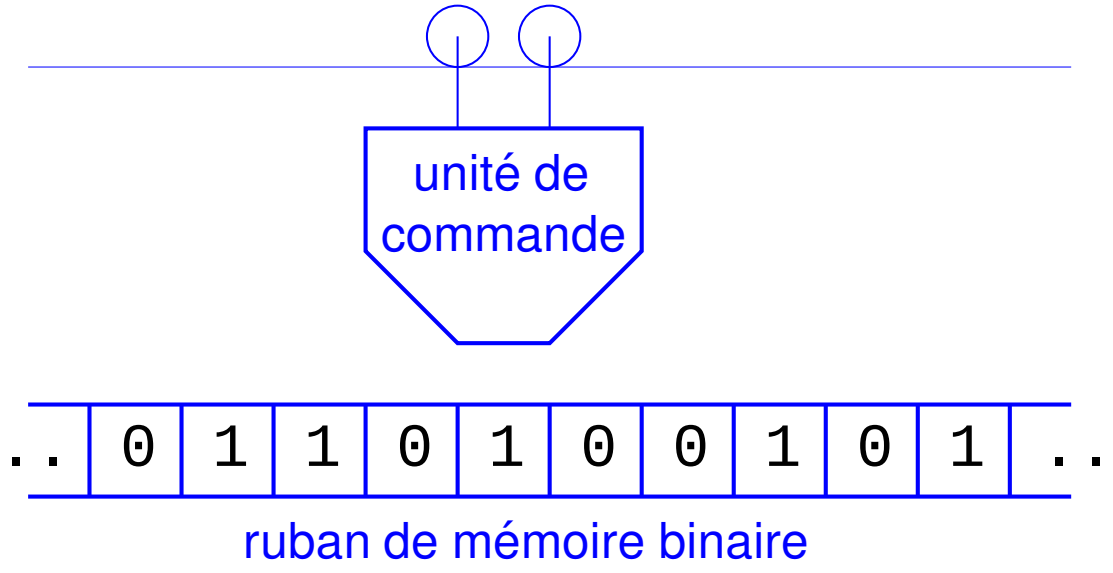
La machine de Turing



La machine de Turing est constitué

- d'un ruban de mémoire binaire infini et
- d'une unité de commande
  chargée d'un programme

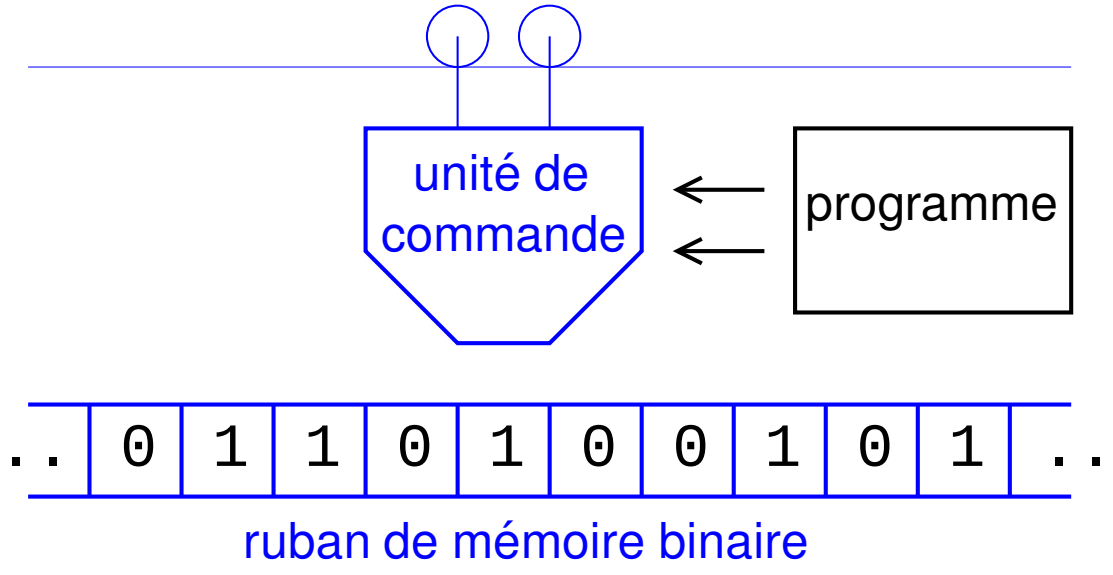
La machine de Turing



L'unité de commande peut

- lire le contenu (`0` ou `1`) de la cellule actuelle du ruban
- écrire `0` ou `1` dans la cellule actuelle du ruban
- se déplacer à gauche ou à droite par une cellule

La machine de Turing



Le programme definit plusieurs “états” de l’unité de commande. Chaque état se compose

- d’une lecture et écriture dans la cellule actuelle
- d’un déplacement à gauche ou à droite
- d’un lien/saut à un autre état à executer à la suite

. . . tout dépendant du contenu de la cellule actuelle

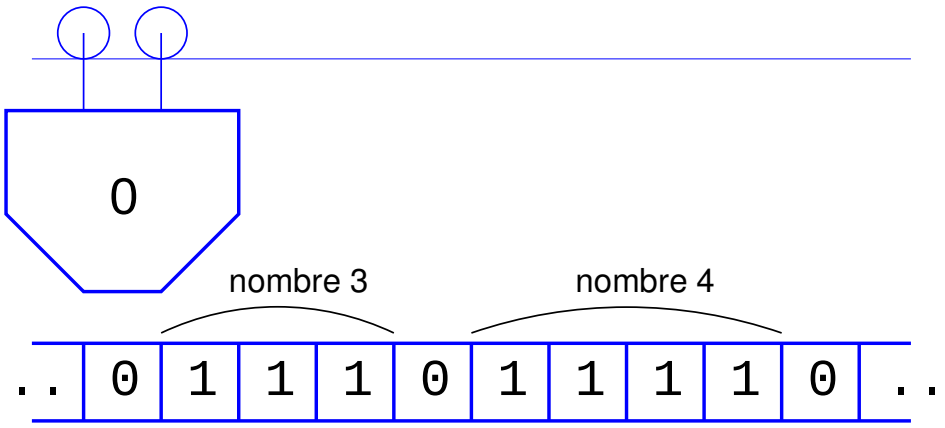
La machine de Turing

Exemple d'un programme (additions) pour la machine de Turing:

état actuel	cellule actuelle		étape prochaine	
	lecture	écriture	déplacement	état suivant
état 0	0	0	droite	état 0
	1	1	droite	état 1
état 1	0	1	droite	état 2
	1	1	droite	état 1
état 2	0	0	gauche	état 3
	1	1	droite	état 2
état 3	0	0	droite	STOP
	1	0	droite	STOP

La machine de Turing

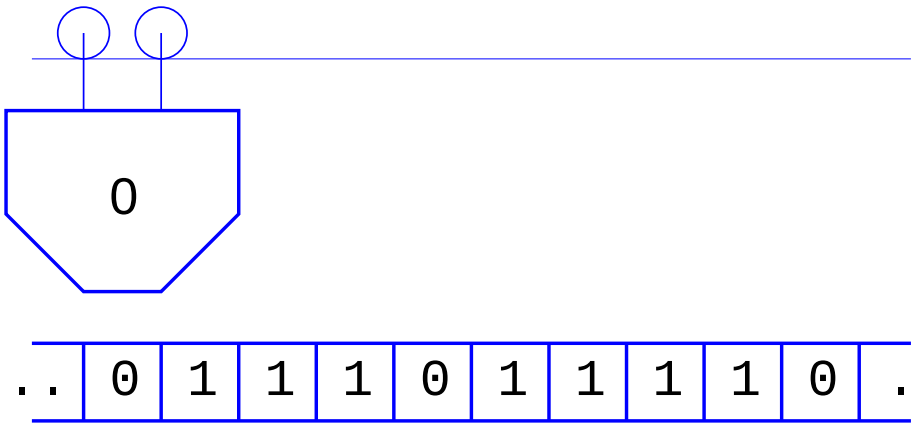
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

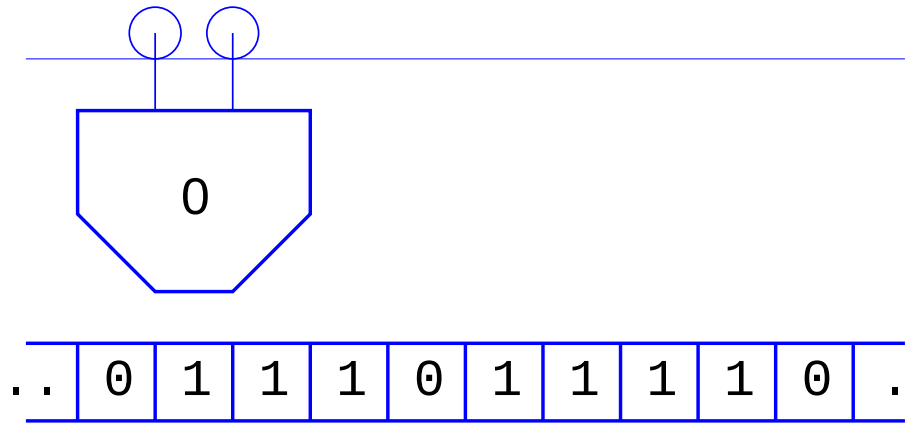
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

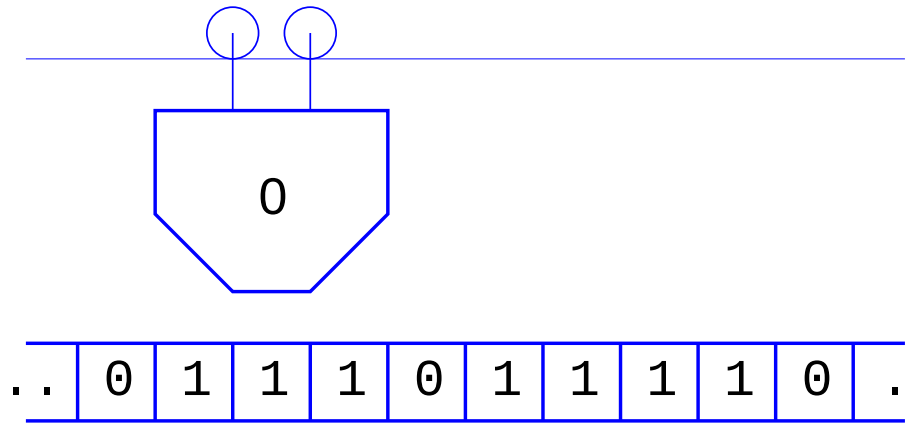
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

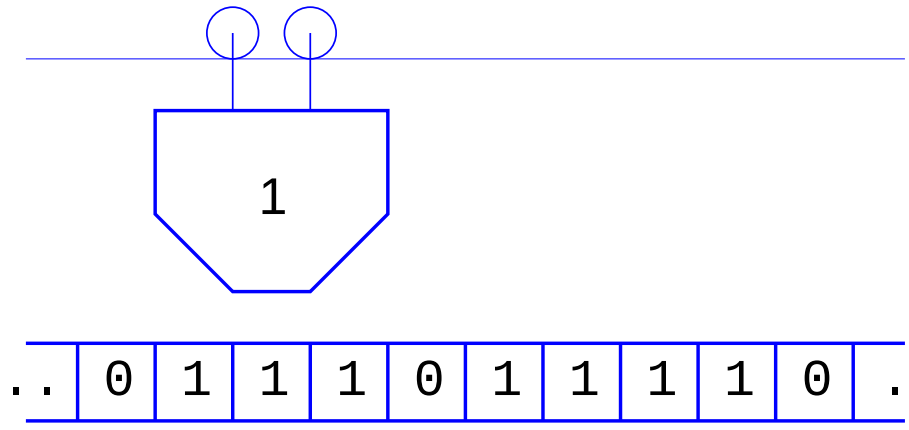
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

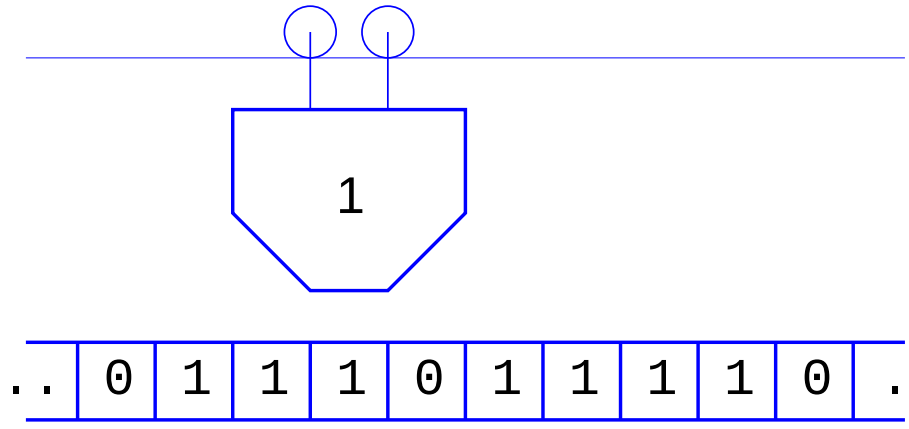
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

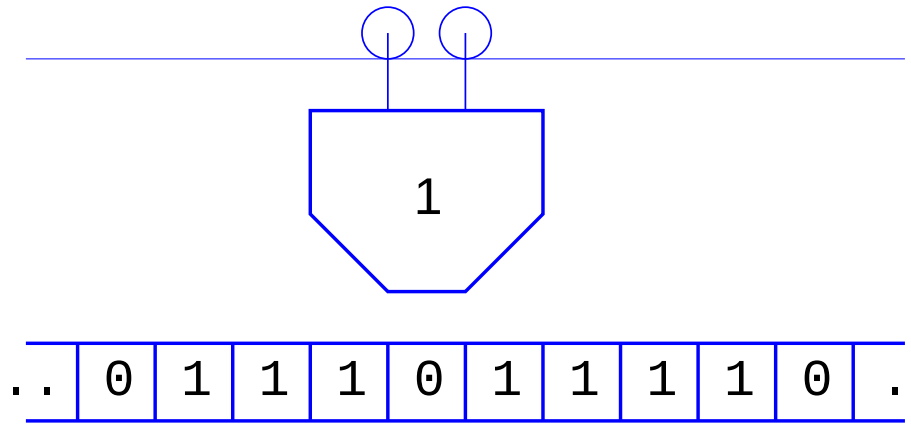
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

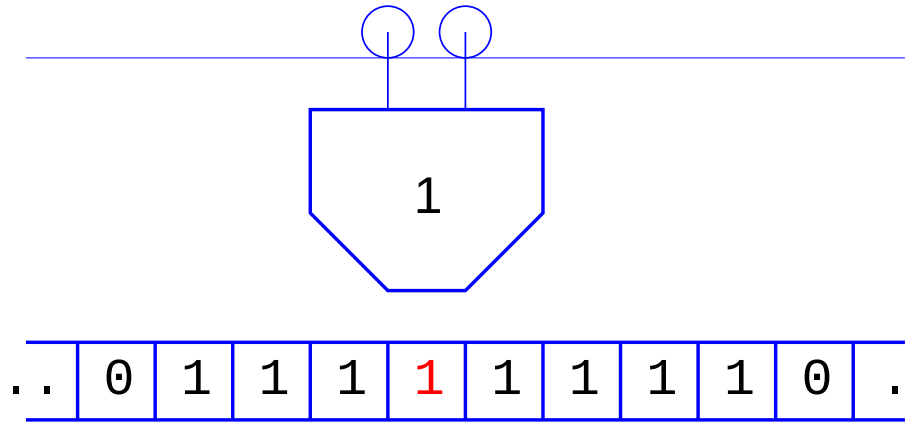
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

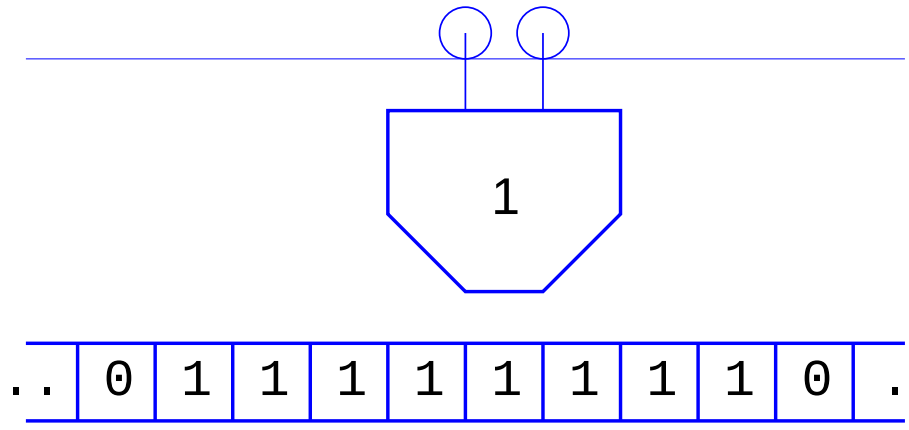
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

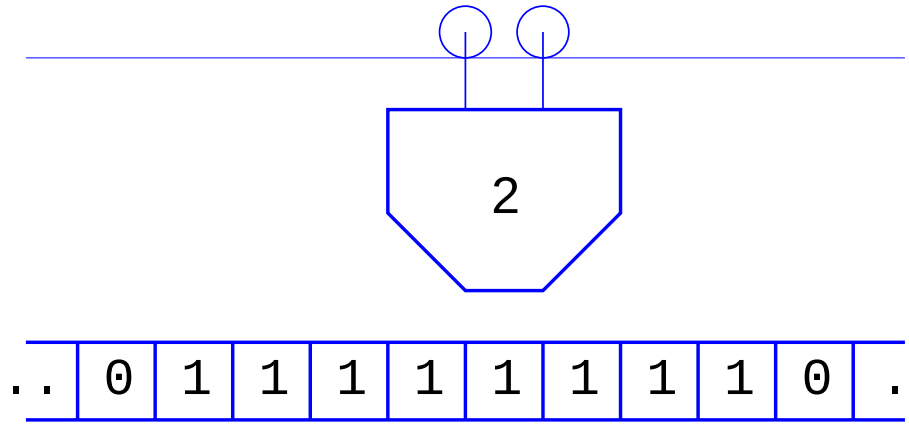
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

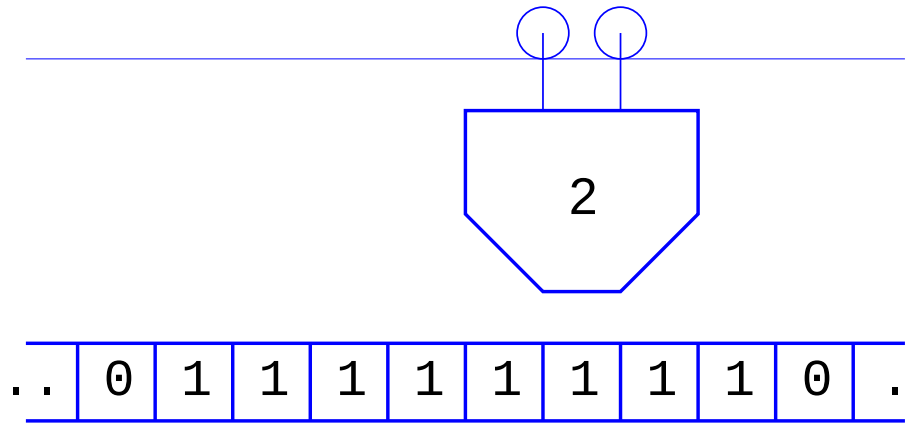
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

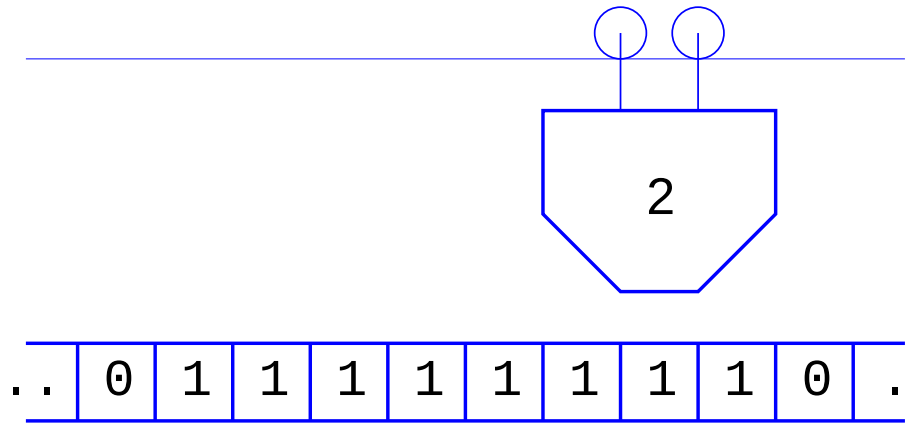
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

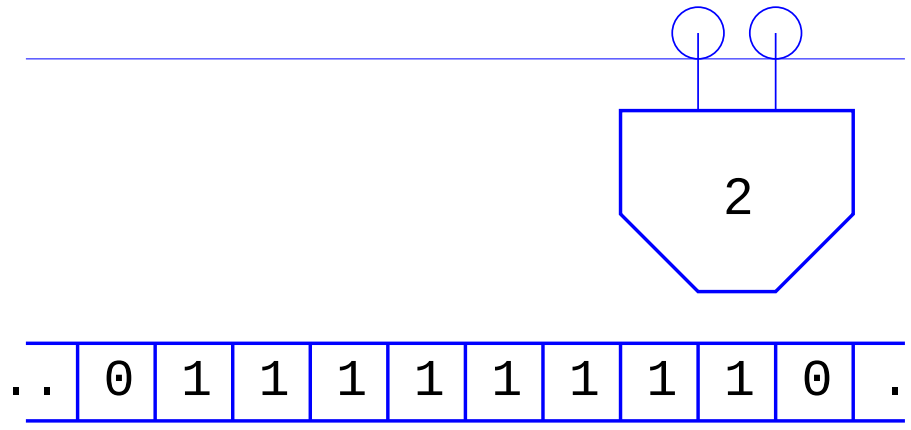
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

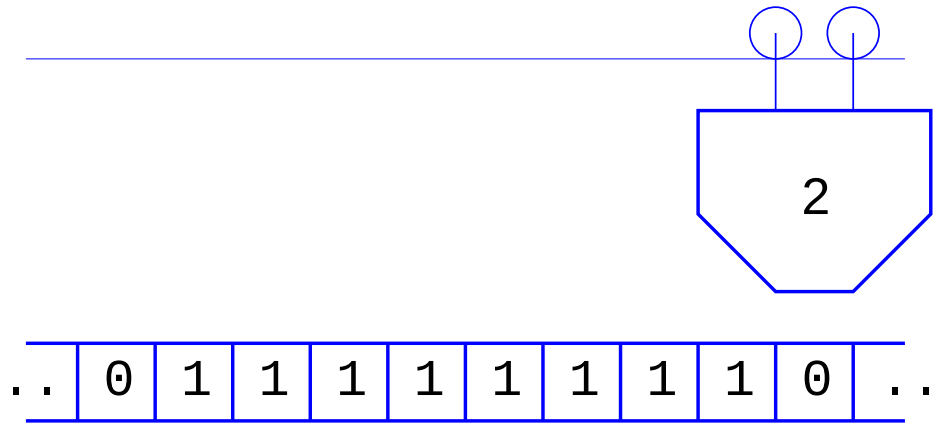
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

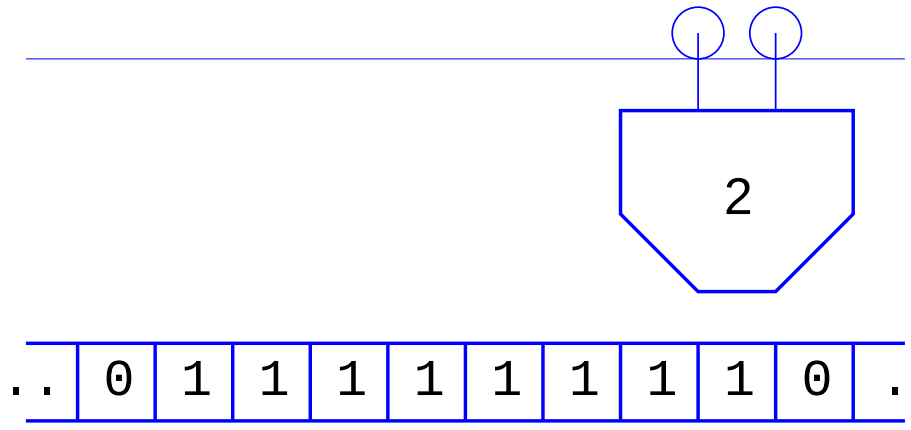
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

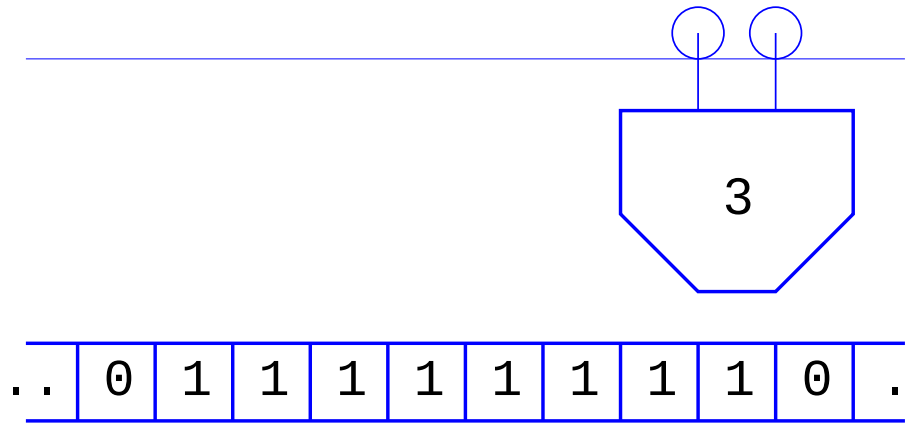
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

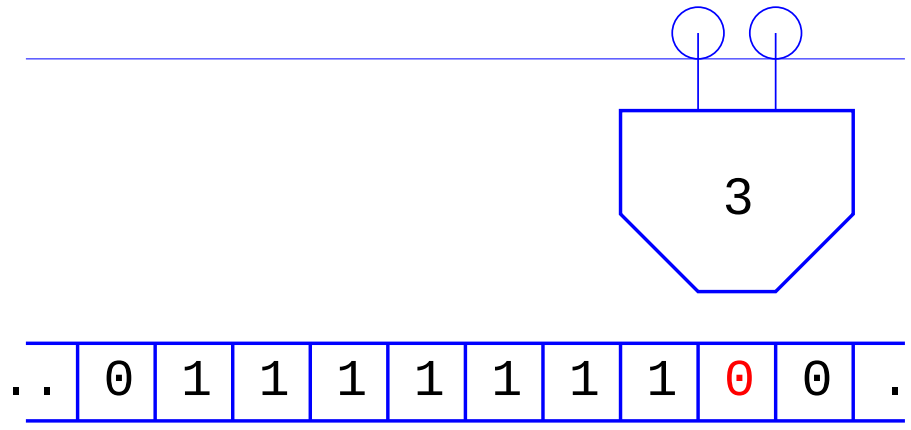
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

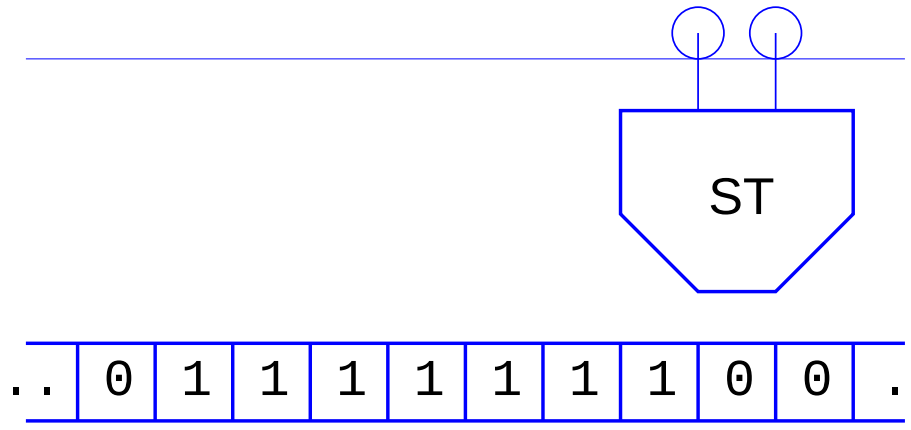
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

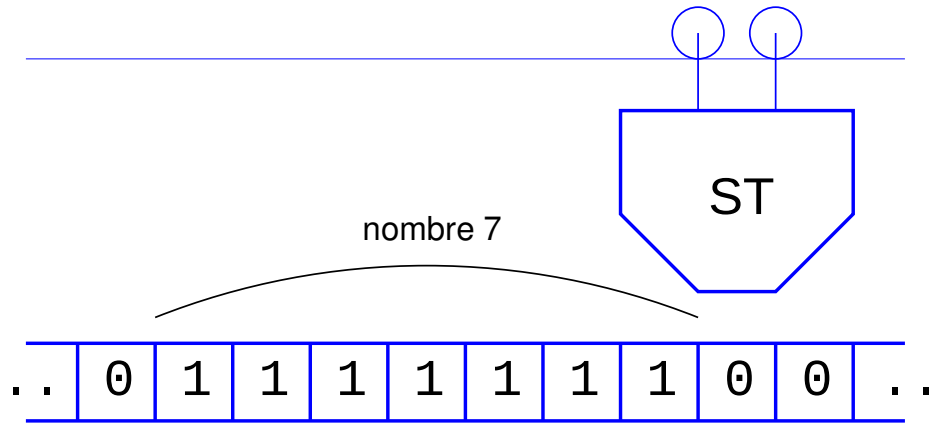
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

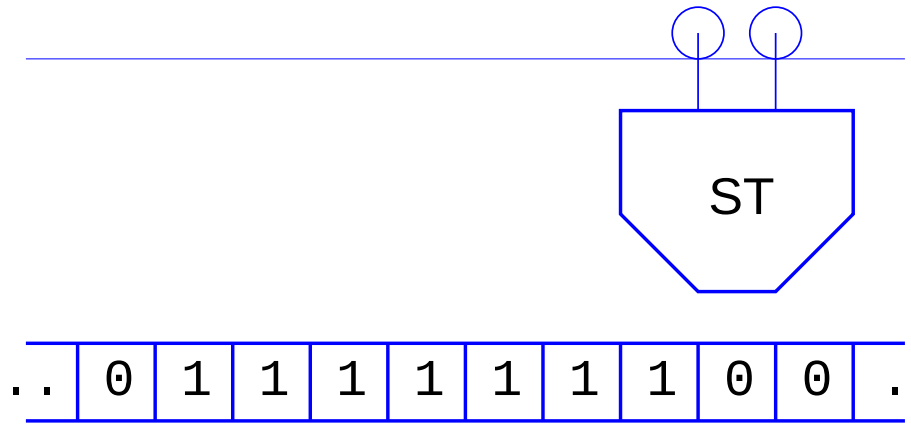
Additions avec la machine de Turing:



0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

La machine de Turing

Additions avec la machine de Turing:



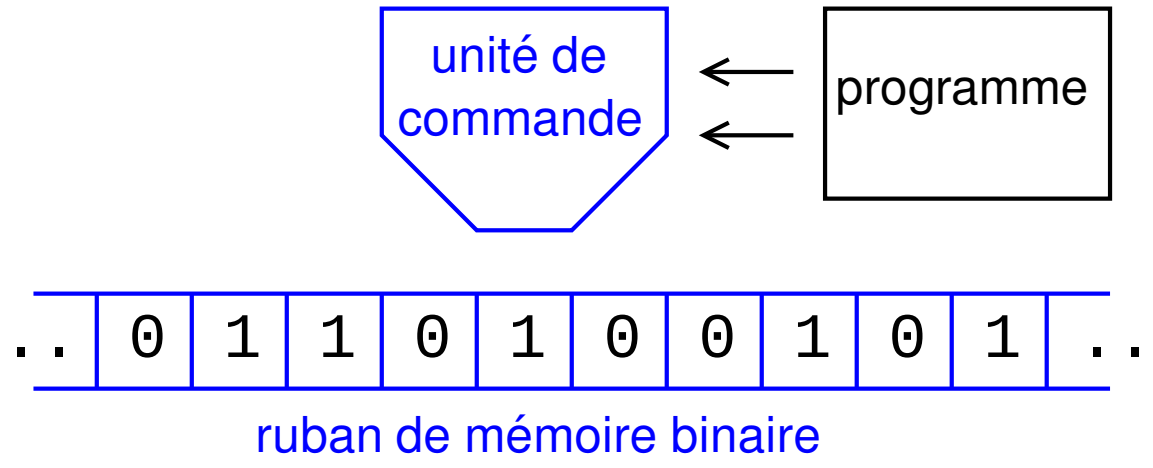
0	0	0	→	0
	1	1	→	1
1	0	1	→	2
	1	1	→	1
2	0	0	←	3
	1	1	→	2
3	0	0	→	ST
	1	0	→	ST

$\longrightarrow$ tous les algorithmes numériques peuvent être représentés sur une telle machine (thèse de Church-Turing)

(ce n'est pas forcément la meilleure façon de les exécuter . . .)

Comment est-ce qu'on pourrait faire mieux

. . . après tout ce qu'on a déjà appris ?



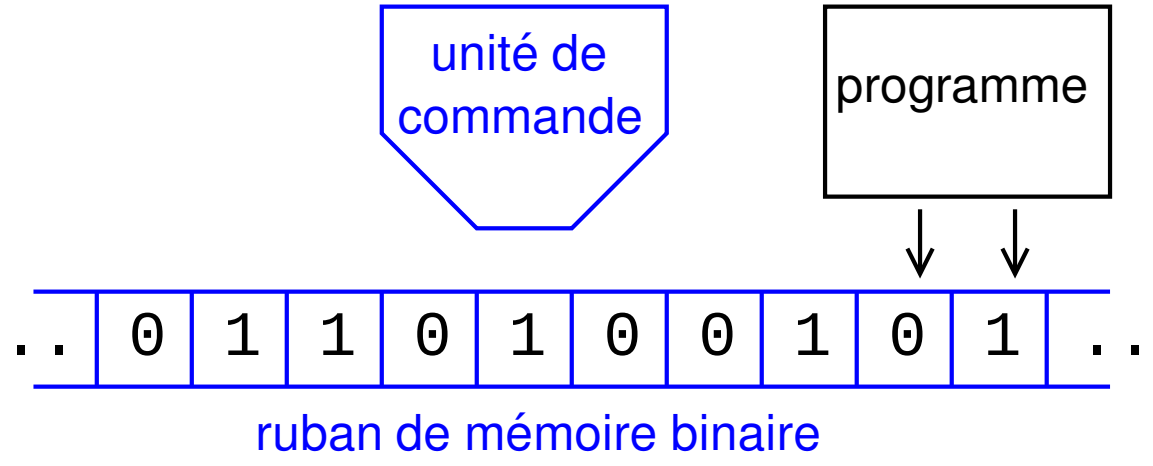
$\longrightarrow$ remplacer tous les bits par des octets

$\longrightarrow$ déplacement arbitraire de l'unité de commande

$\rightarrow$ nécessite l'adressage global des cellules du ruban

Comment est-ce qu'on pourrait faire mieux

. . . après tout ce qu'on a déjà appris ?



$\longrightarrow$ remplacer tous les bits par des octets

$\longrightarrow$ déplacement arbitraire de l'unité de commande

$\longrightarrow$ effectuer des calculs élementaires (additions des octets) à l'interieur de l'unité de commande

$\longrightarrow$ encoder le programme dans la mémoire

3.2 Le fonctionnement d'un processeur

John von Neumann (1903 - 1957), mathématicien

Contributions:

- théorie de logique
- physique quantique
- théorie des jeux
- Manhattan project
- informatique
  $\rightarrow$ "l'architecture de von Neumann"

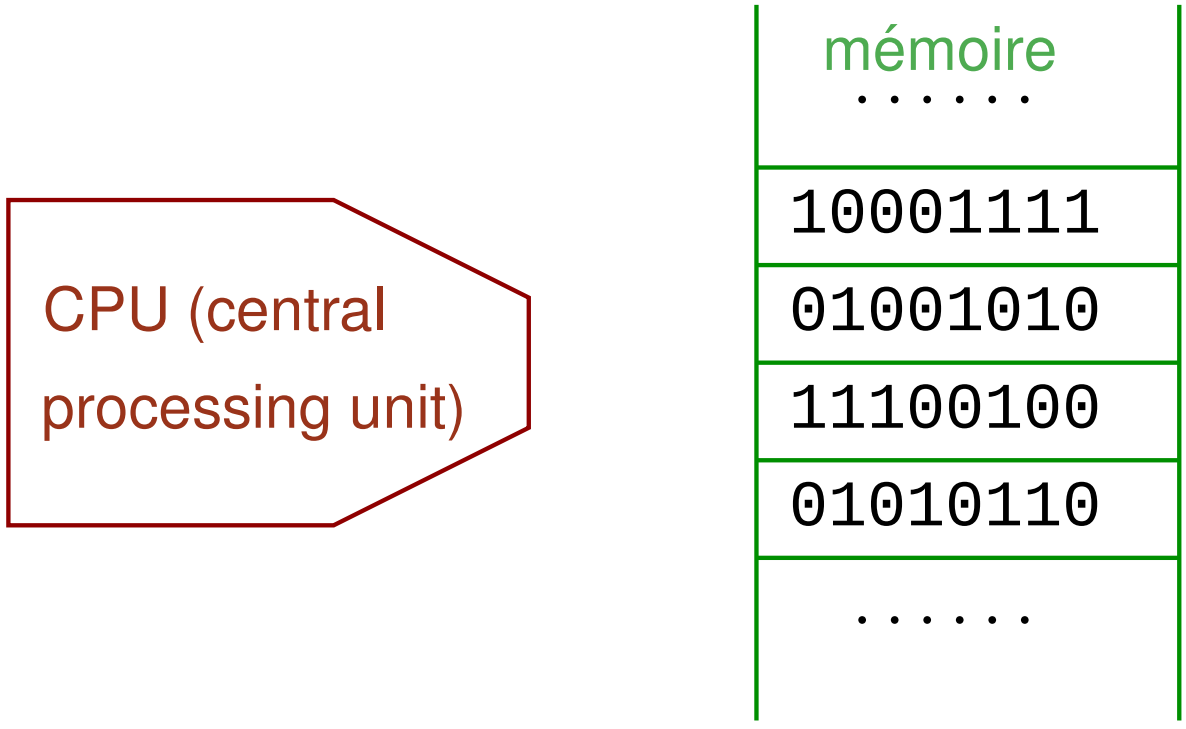
3.2 Le fonctionnement d'un processeur

Le principe de von Neumann:

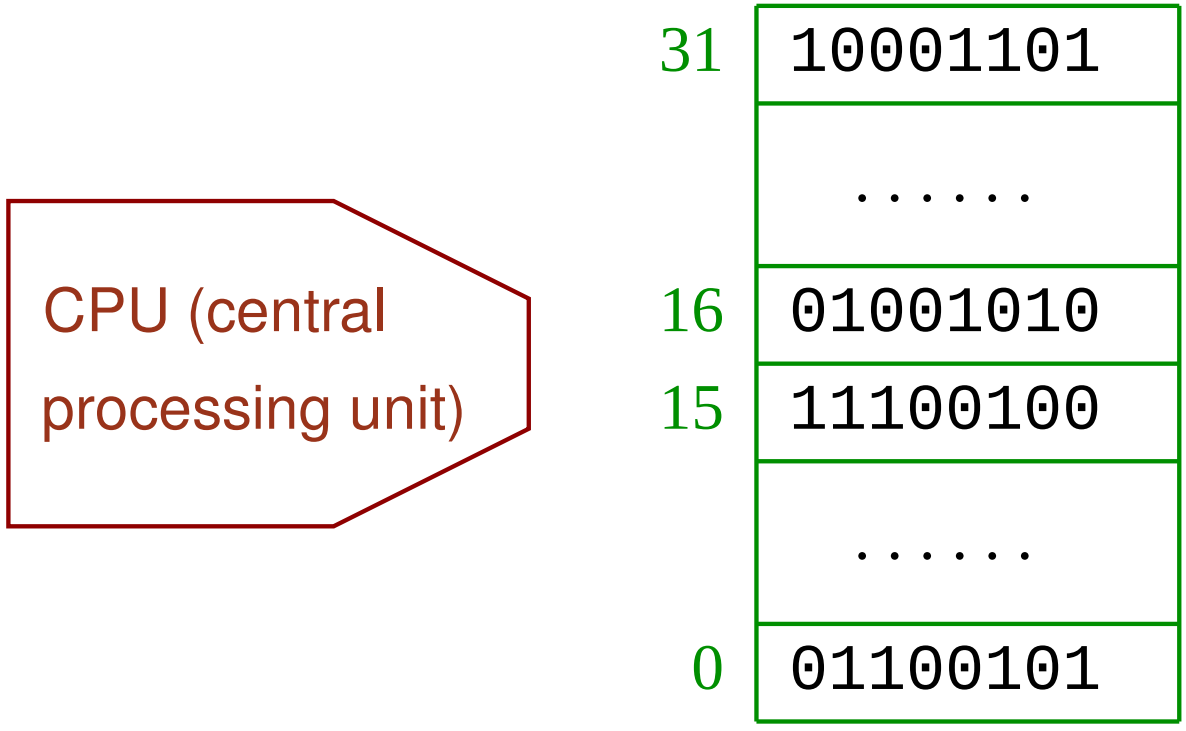
Un ordinateur est constitué

- d'une unité de commande,
- d'une unité arithmétique et logique,
- d'une mémoire centrale et
- des unités d'entrée et de sortie
  (clavier, souris, écran, imprimante, ...)

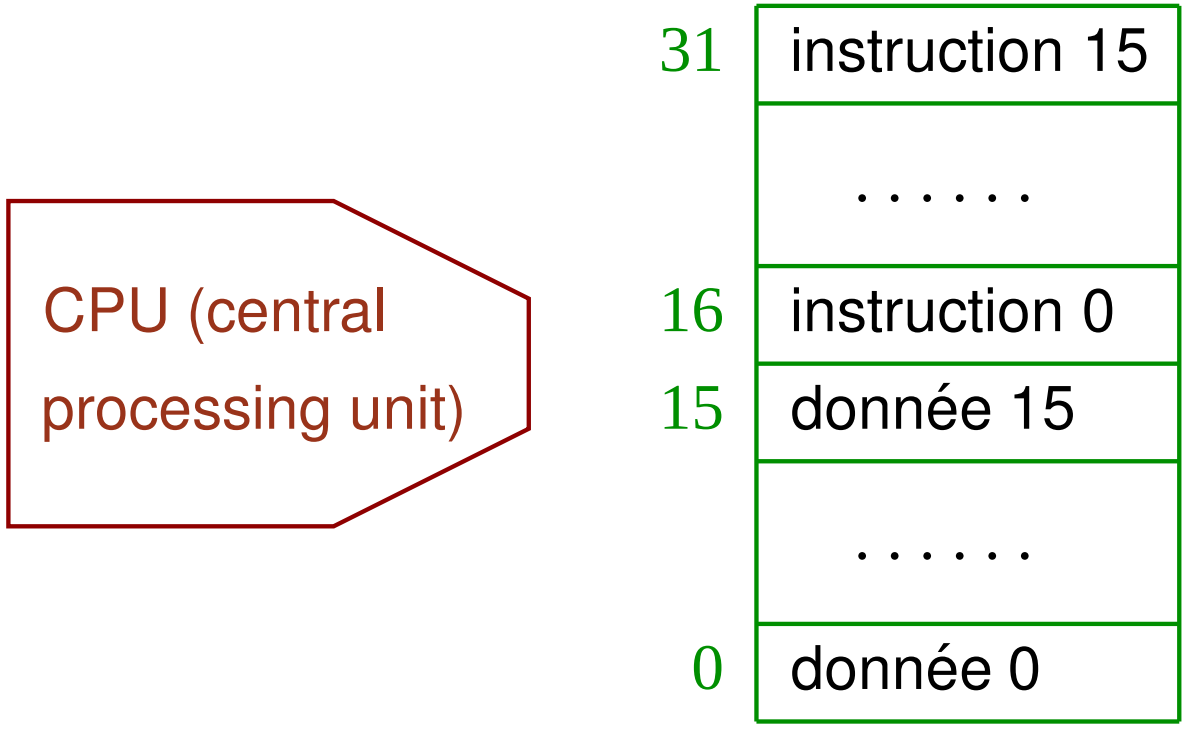
Le fonctionnement d'un processeur primitif



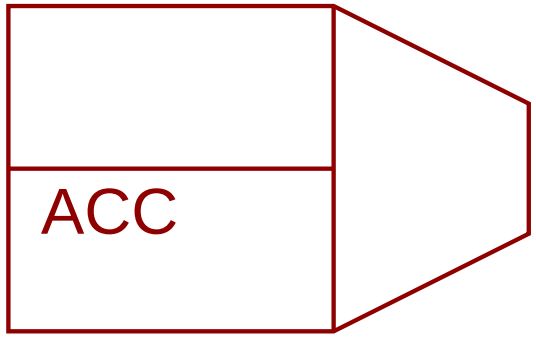
Le fonctionnement d'un processeur primitif

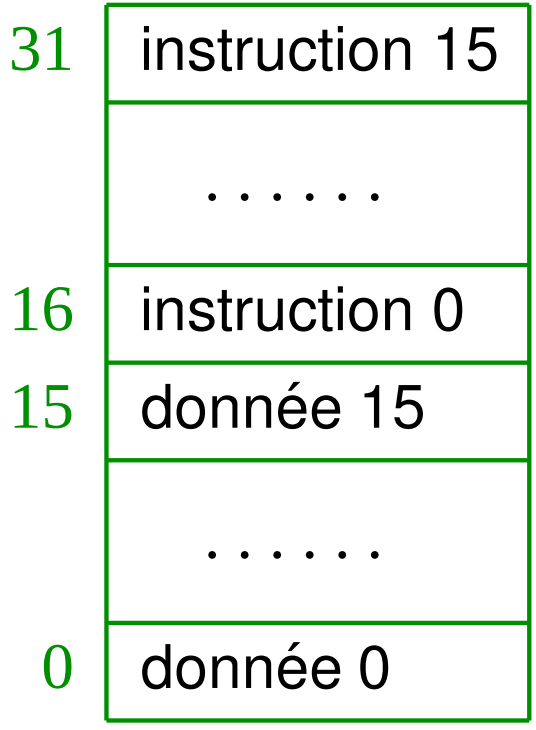


Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif

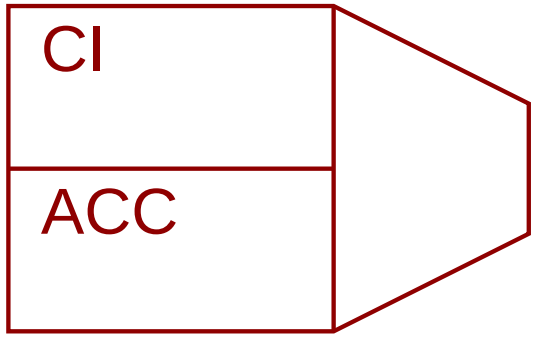


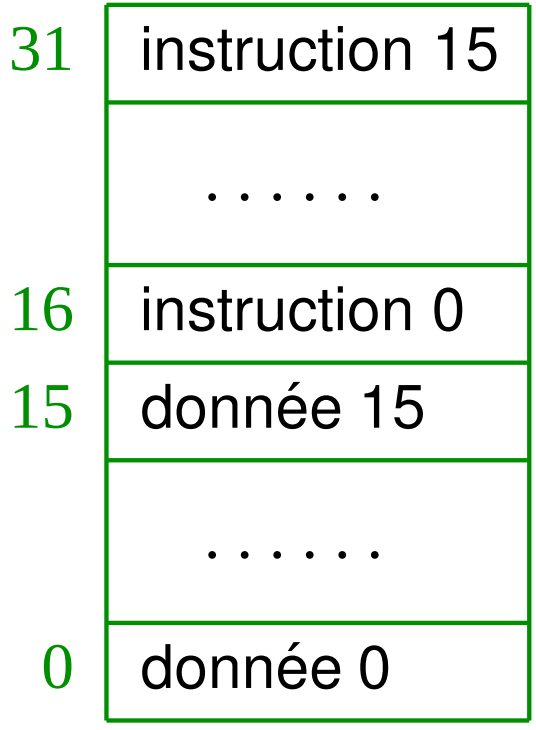


ACC: accumulateur (1 octet)

$\longrightarrow$ sauvegarde d'une donnée pour des opérations arithmétiques et logiques

Le fonctionnement d'un processeur primitif



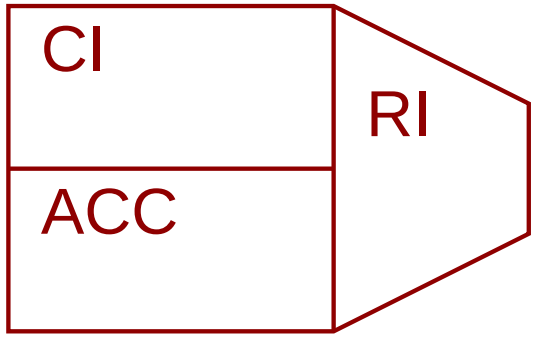


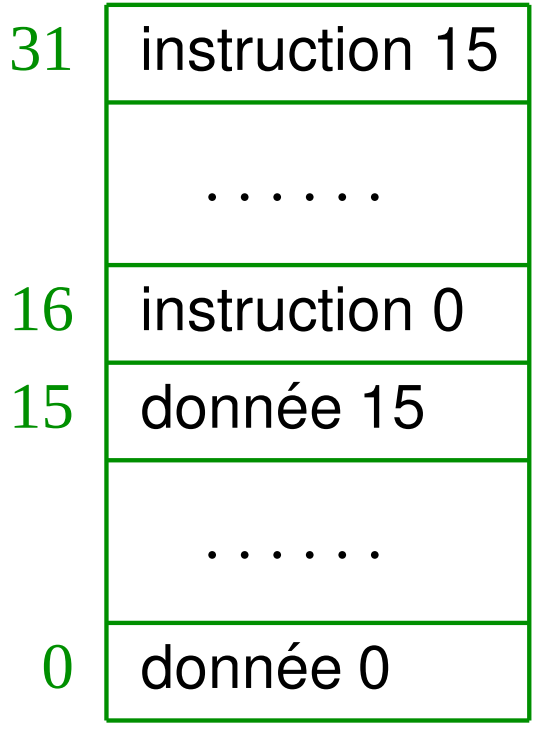
ACC: accumulateur (1 octet)

CI: compteur d'instructions (1 octet)

$\longrightarrow$ contient l'adresse de l'instruction actuelle

Le fonctionnement d'un processeur primitif



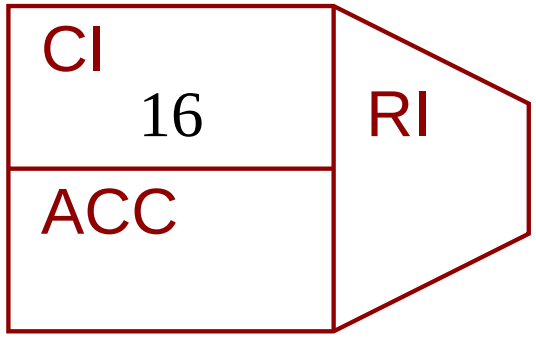


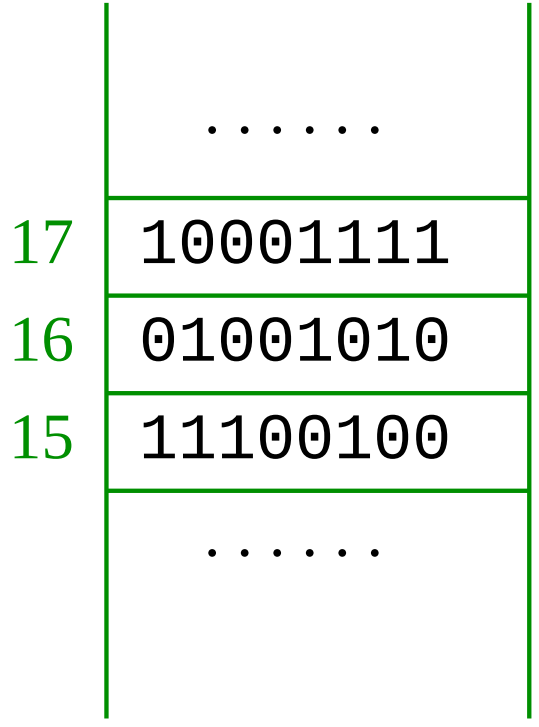
ACC: accumulateur (1 octet)

CI: compteur d'instructions (1 octet)

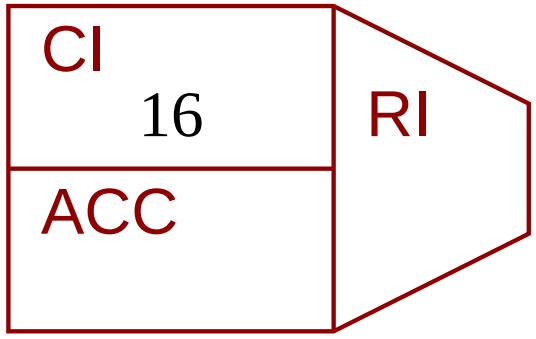
RI: registre d'instruction (1 octet)

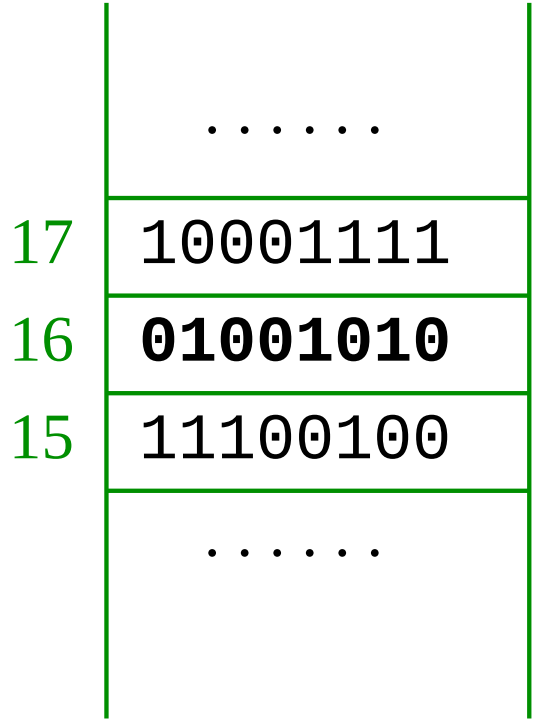
Le fonctionnement d'un processeur primitif



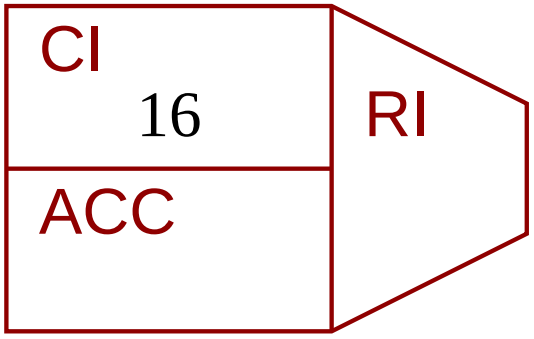


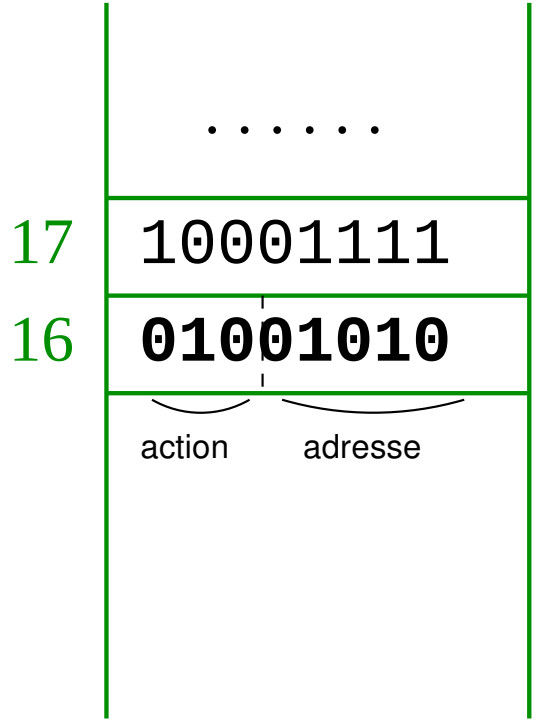
Le fonctionnement d'un processeur primitif



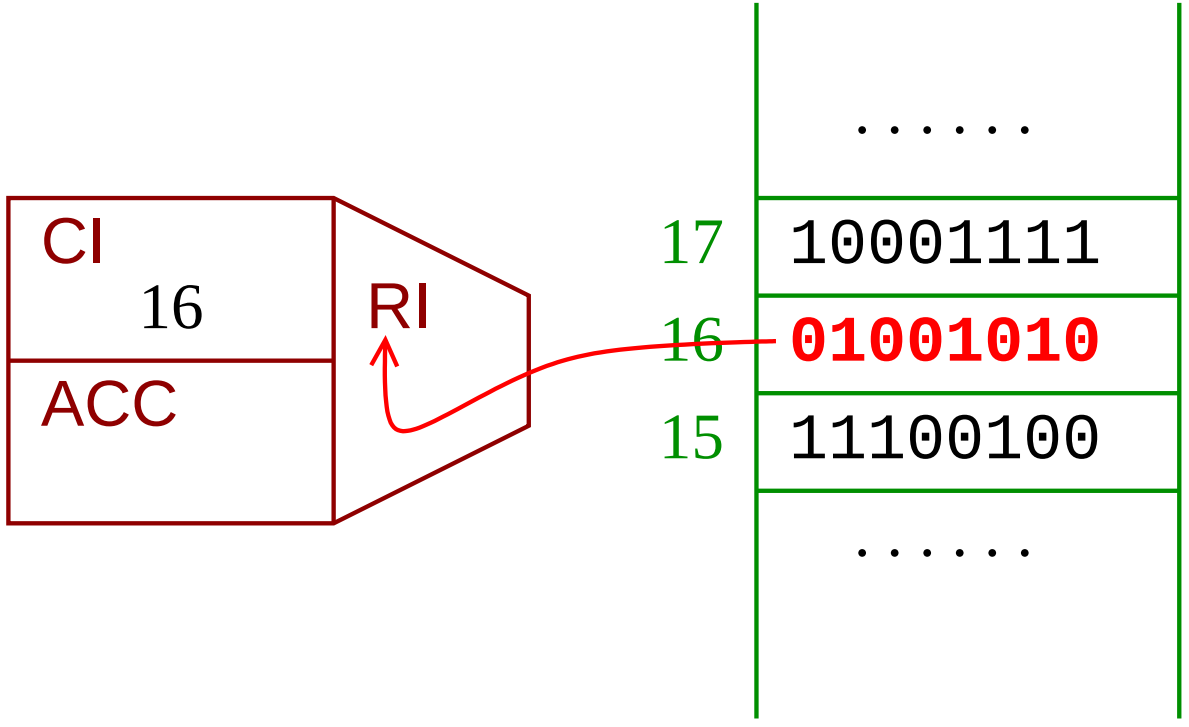


Le fonctionnement d'un processeur primitif

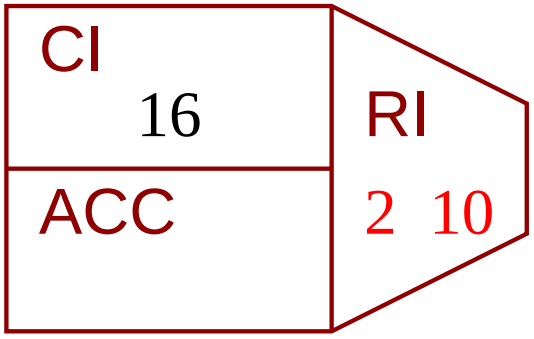


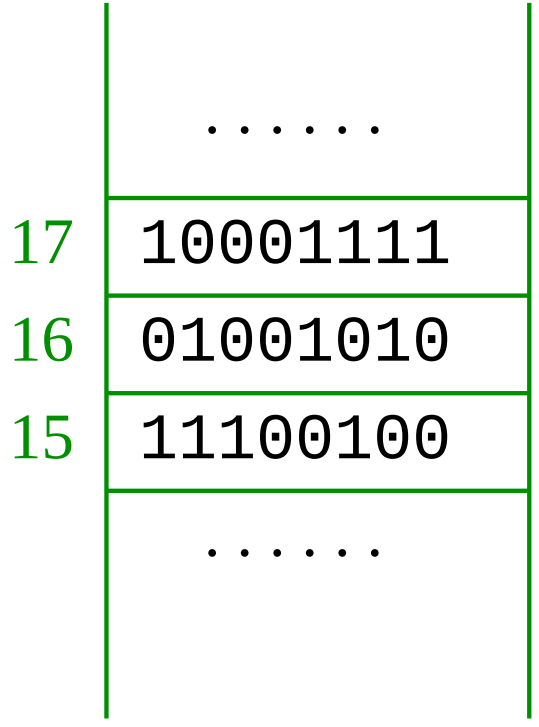


Le fonctionnement d'un processeur primitif

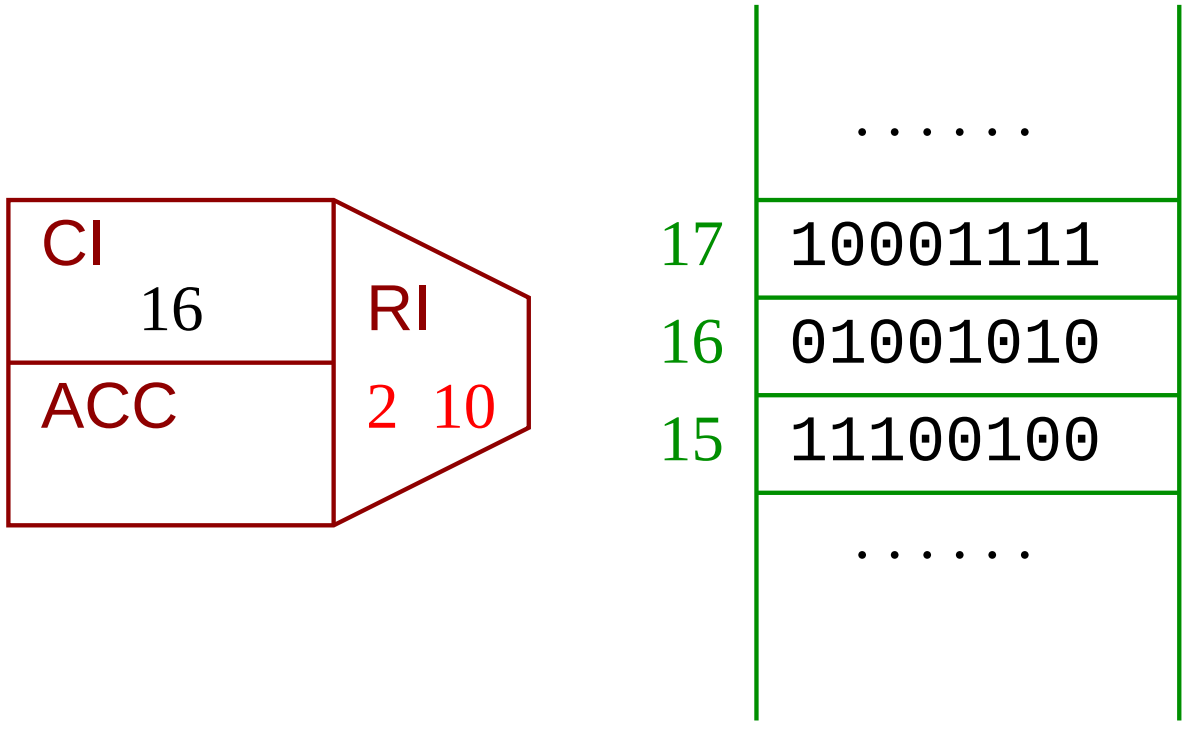


Le fonctionnement d'un processeur primitif





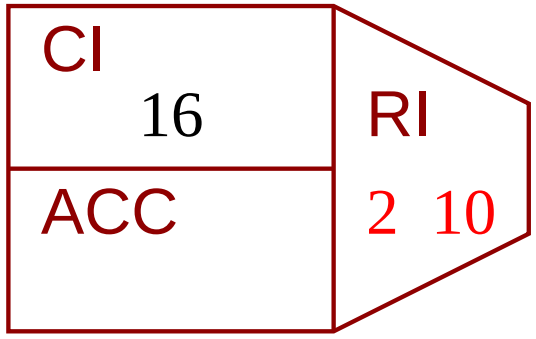
Le fonctionnement d'un processeur primitif

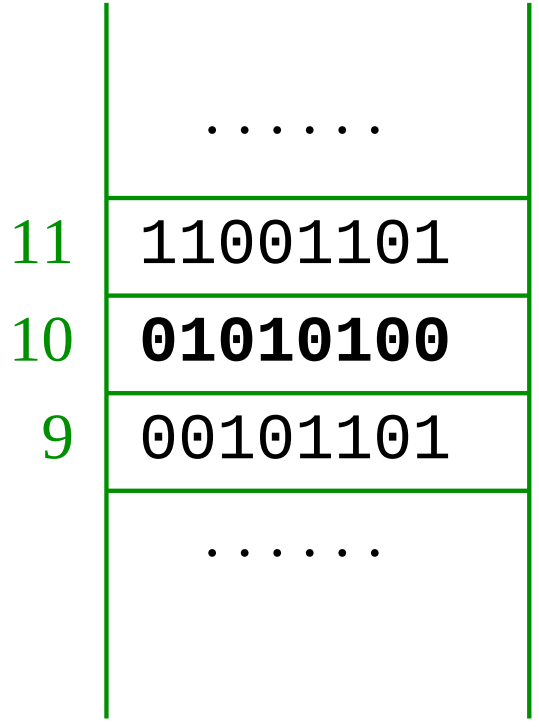


Opération 2:

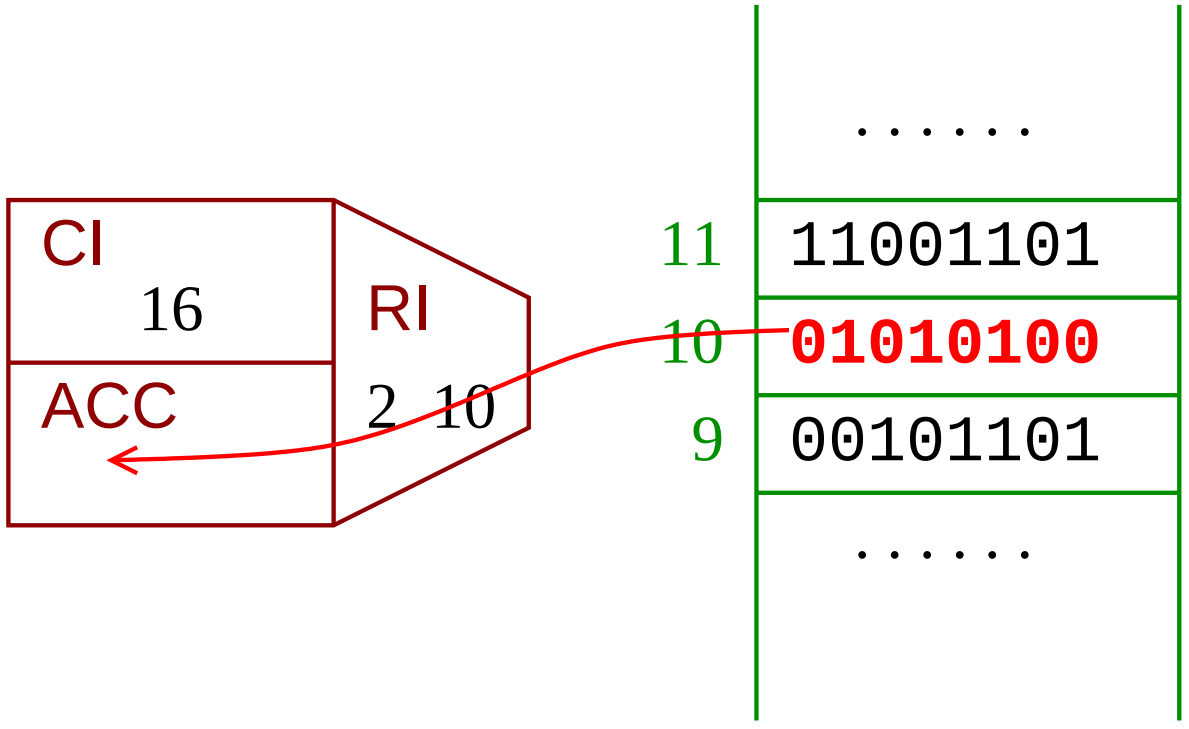
copier le contenu à l'adresse indiquée dans l'accumulateur

Le fonctionnement d'un processeur primitif

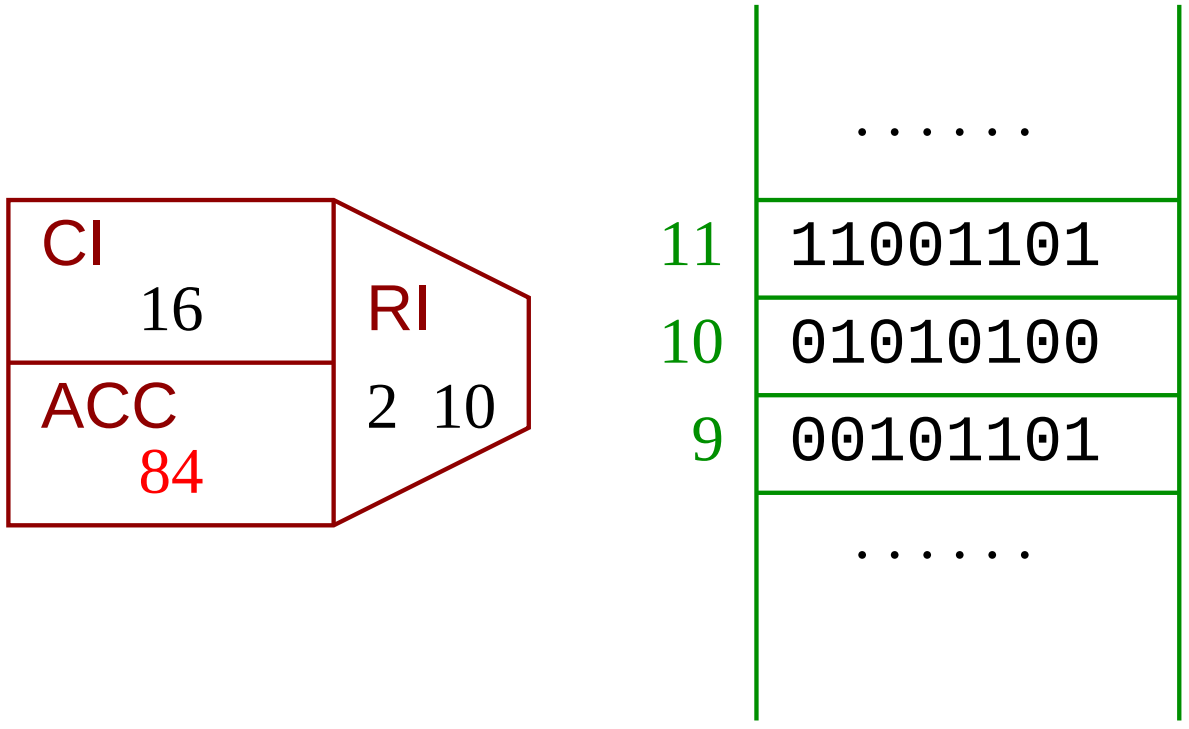




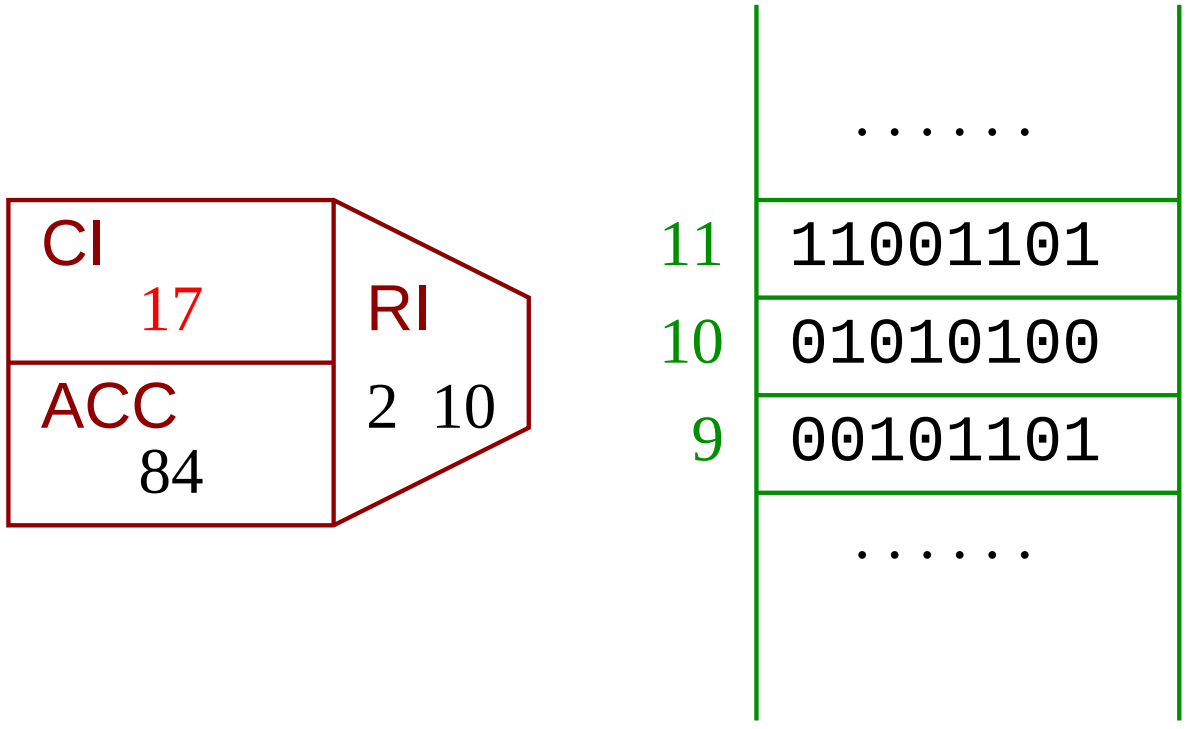
Le fonctionnement d'un processeur primitif



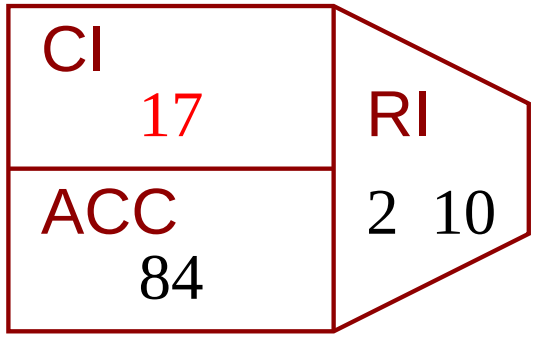
Le fonctionnement d'un processeur primitif

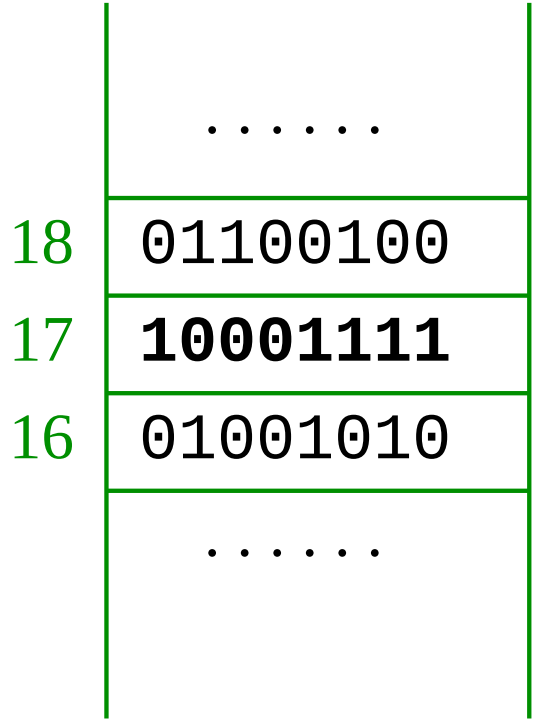


Le fonctionnement d'un processeur primitif

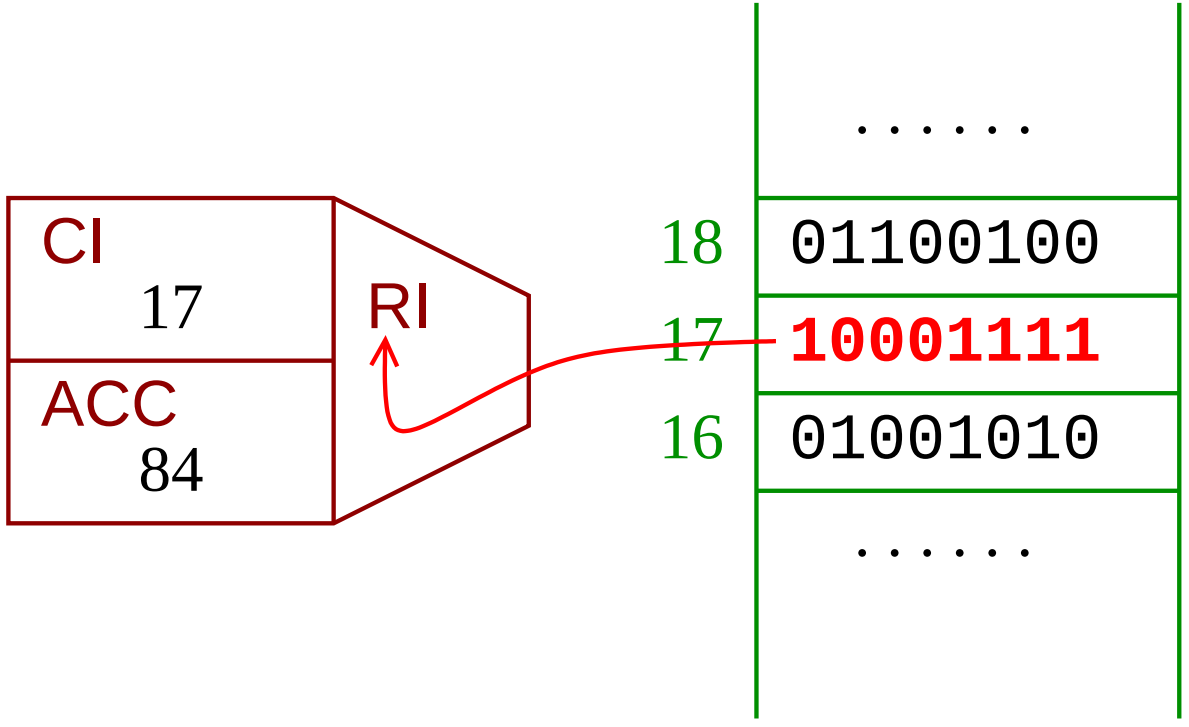


Le fonctionnement d'un processeur primitif

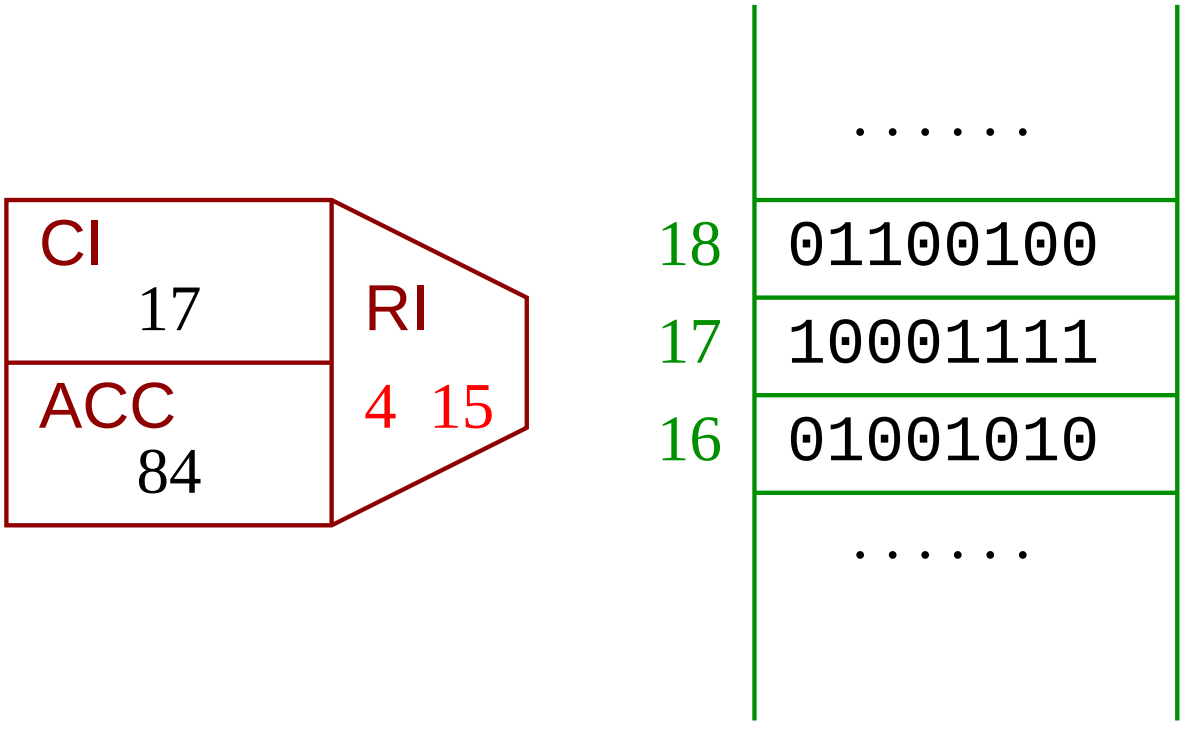




Le fonctionnement d'un processeur primitif



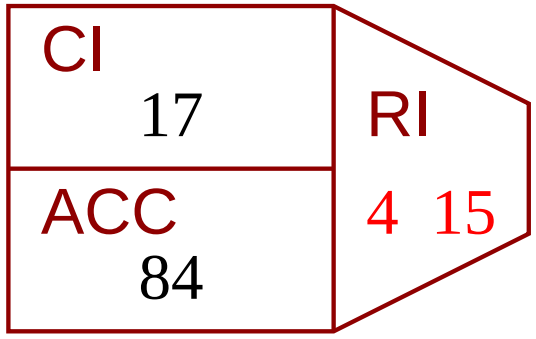
Le fonctionnement d'un processeur primitif

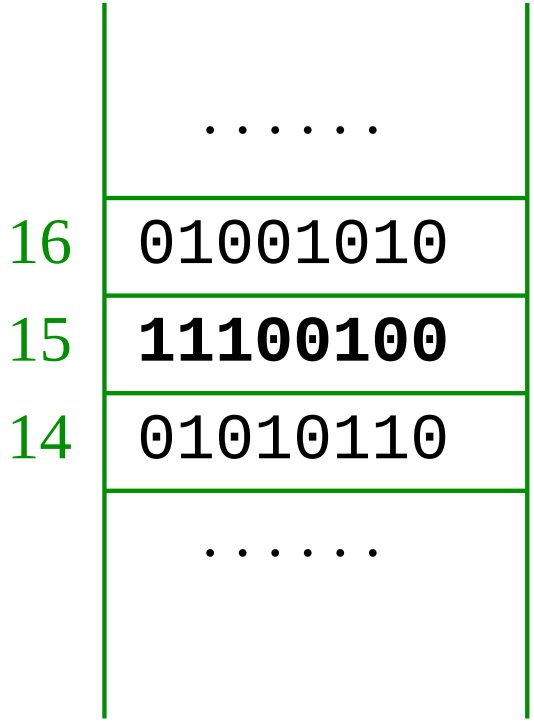


Opération 4:

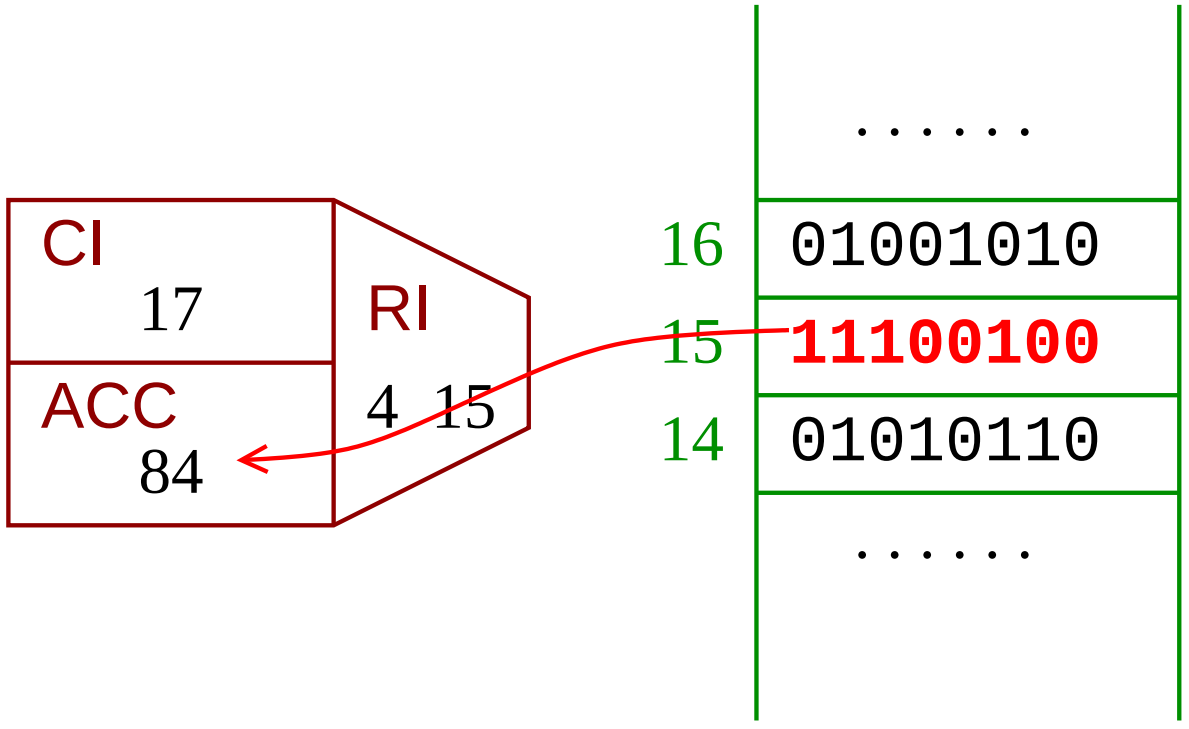
ajouter à l'accumulateur le contenu dans l'adresse indiquée

Le fonctionnement d'un processeur primitif

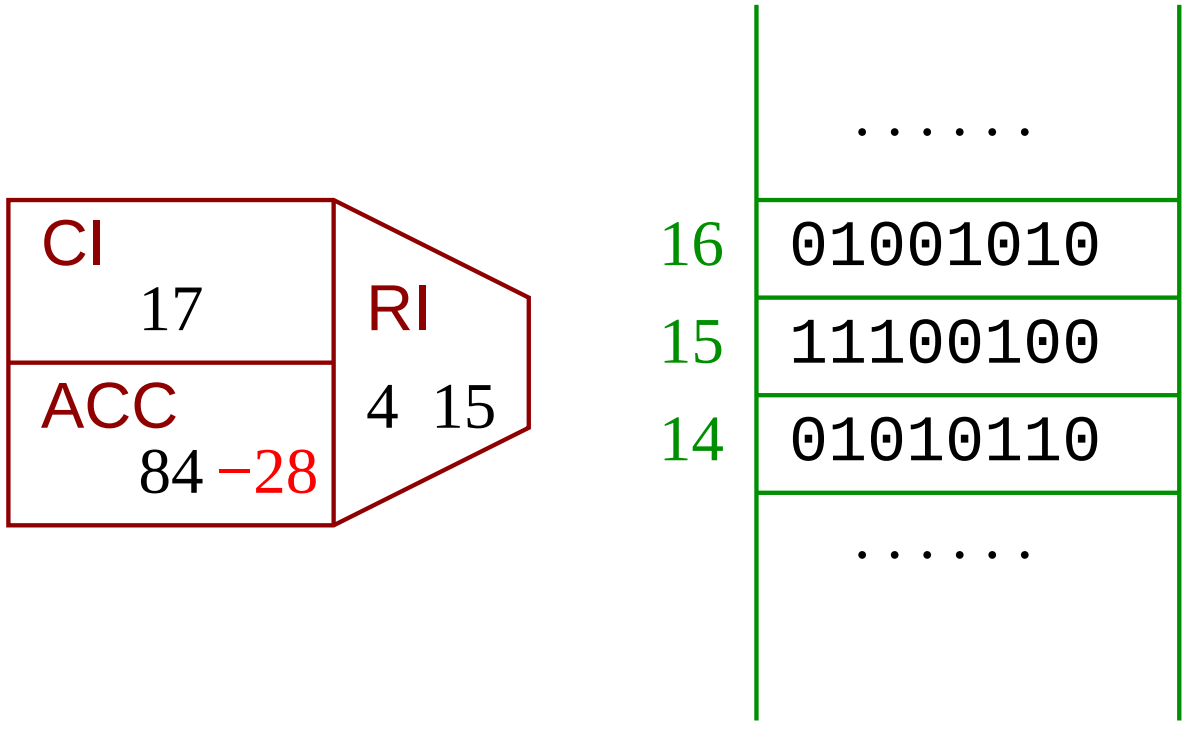




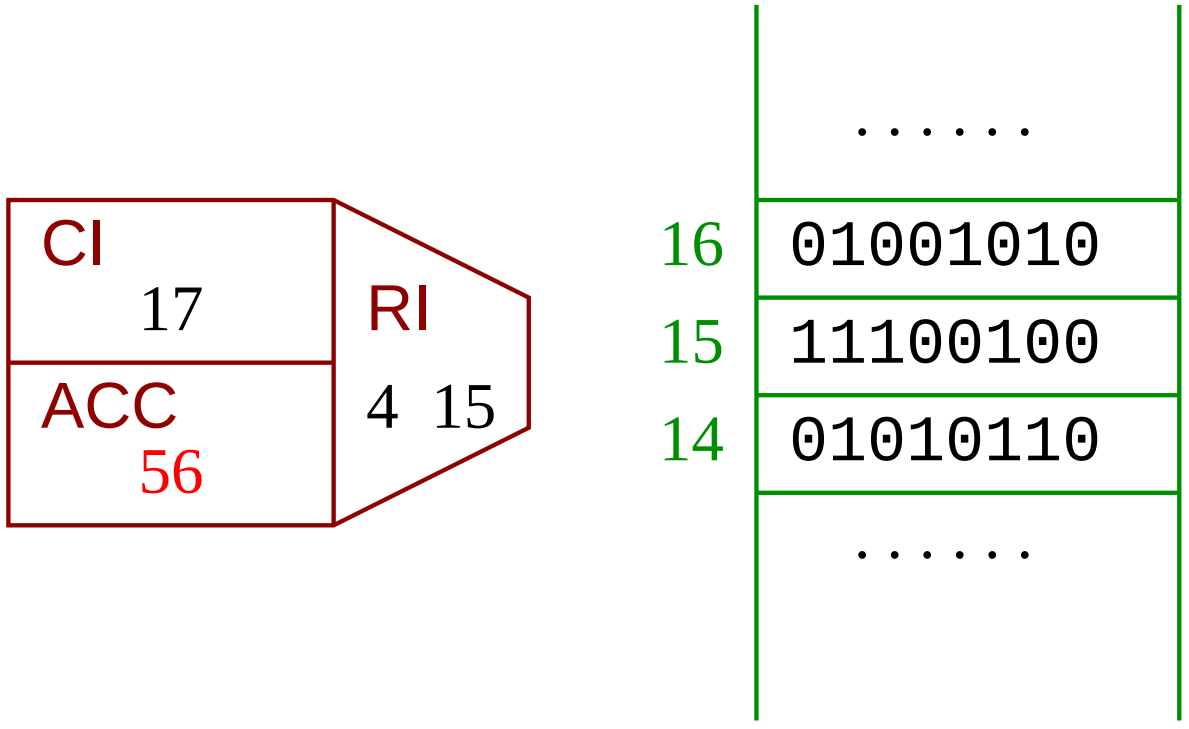
Le fonctionnement d'un processeur primitif



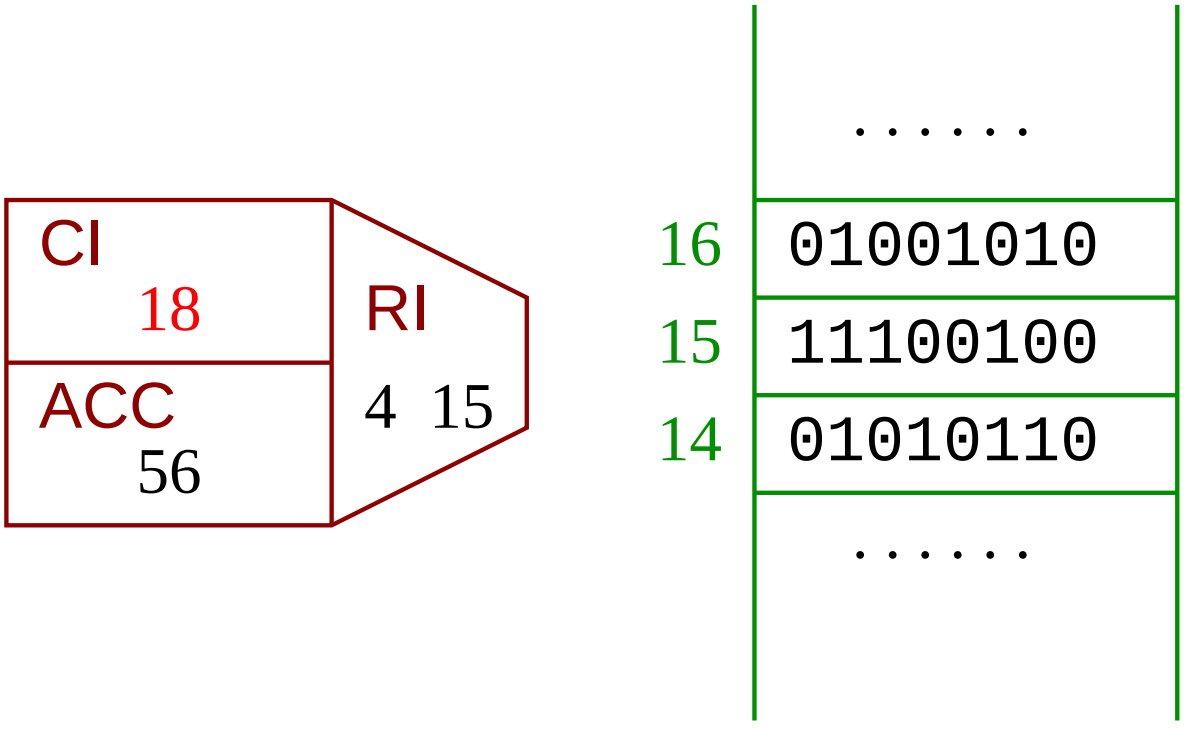
Le fonctionnement d'un processeur primitif



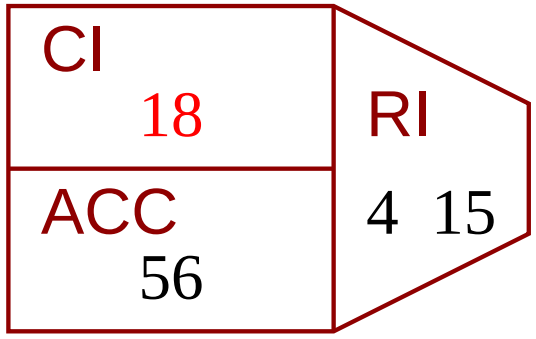
Le fonctionnement d'un processeur primitif

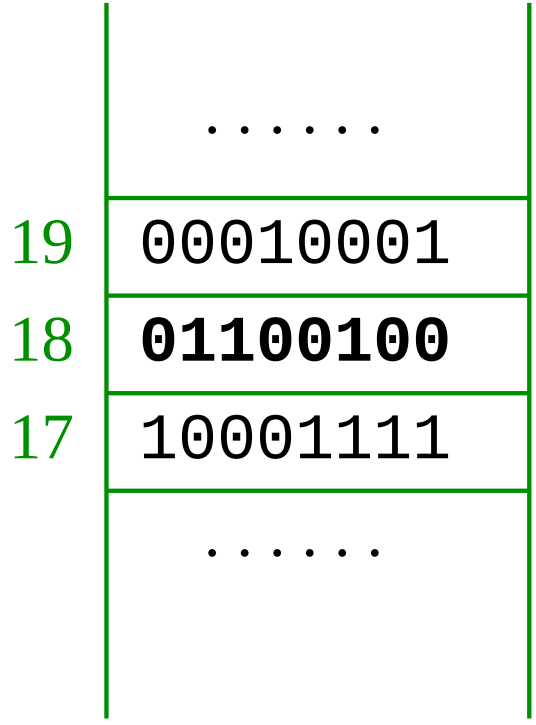


Le fonctionnement d'un processeur primitif

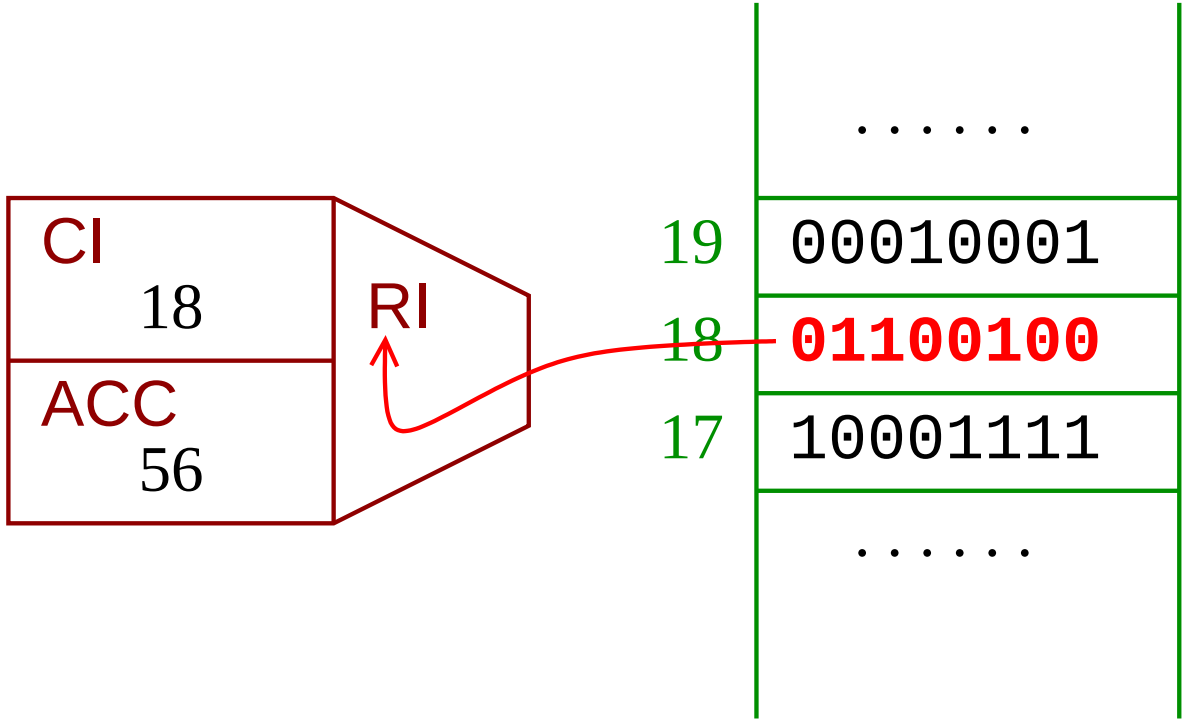


Le fonctionnement d'un processeur primitif

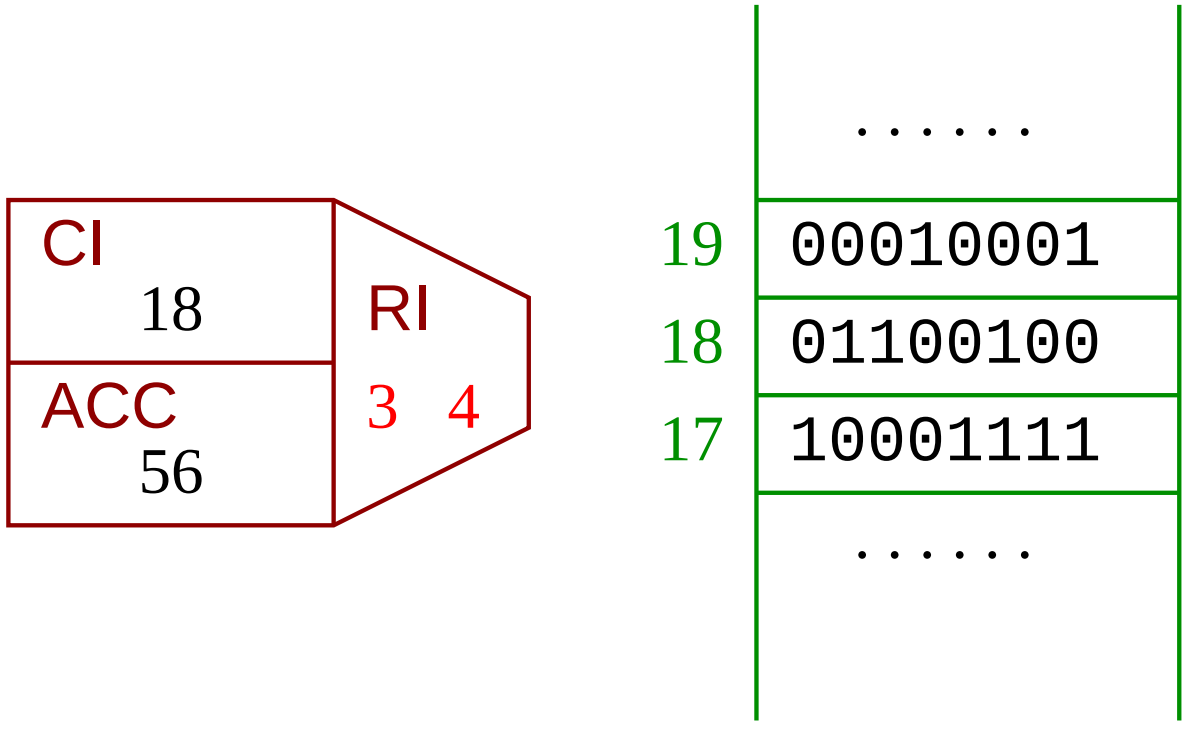




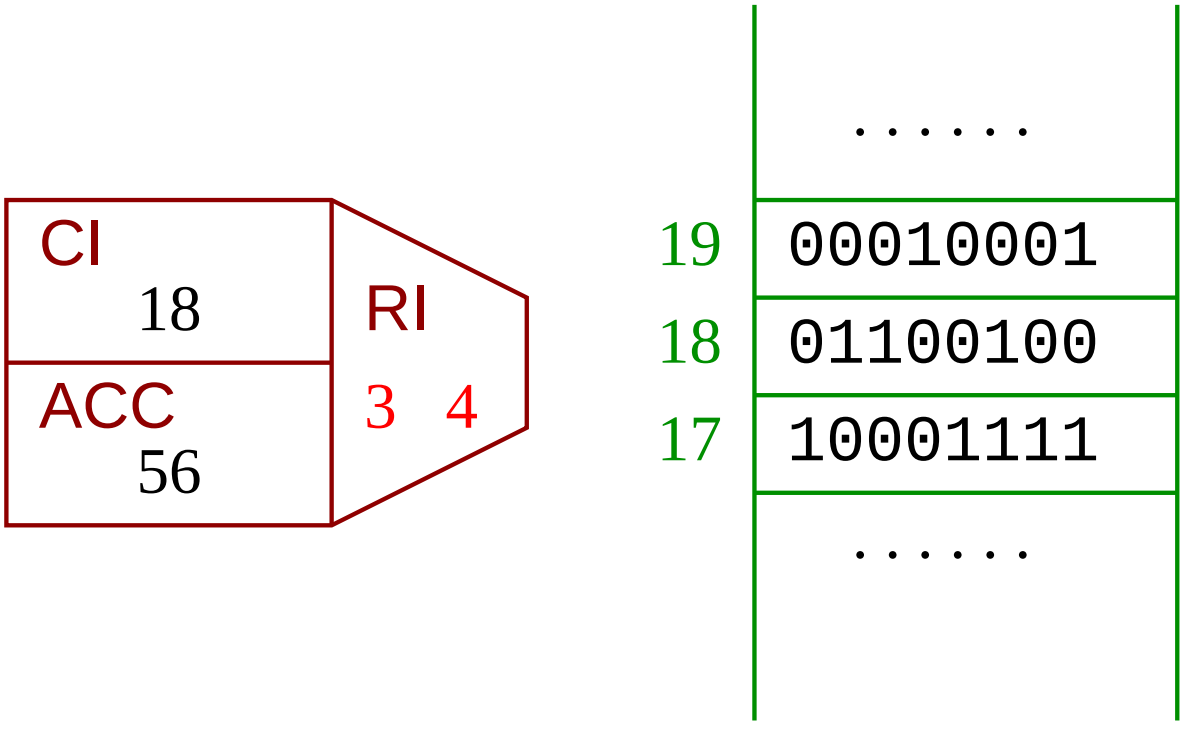
Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif



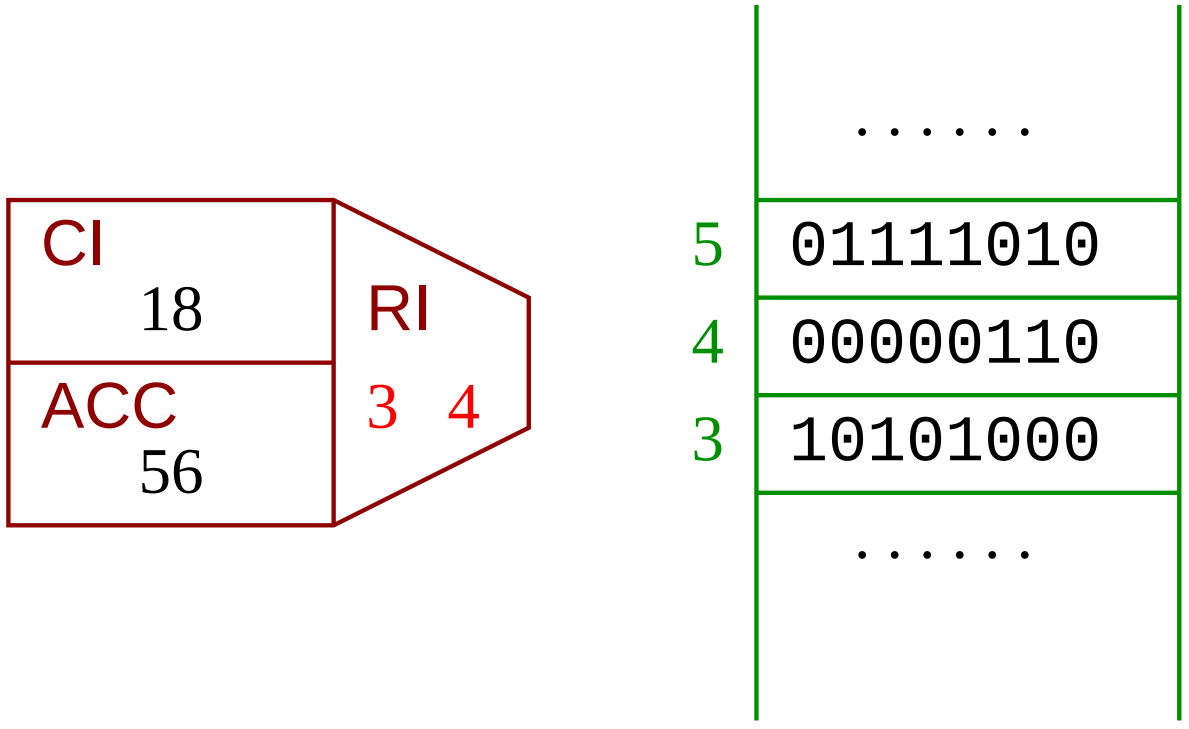
Le fonctionnement d'un processeur primitif



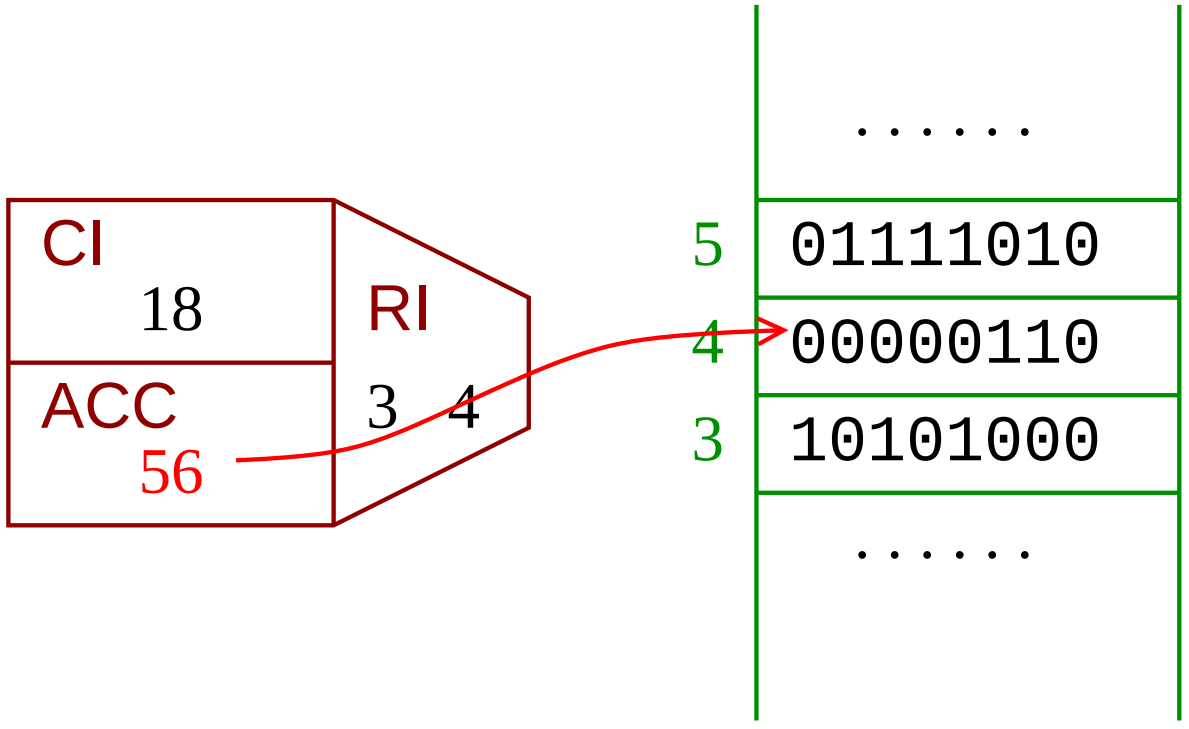
Opération 3:

écrire le contenu de l'accumulateur dans l'adresse indiquée

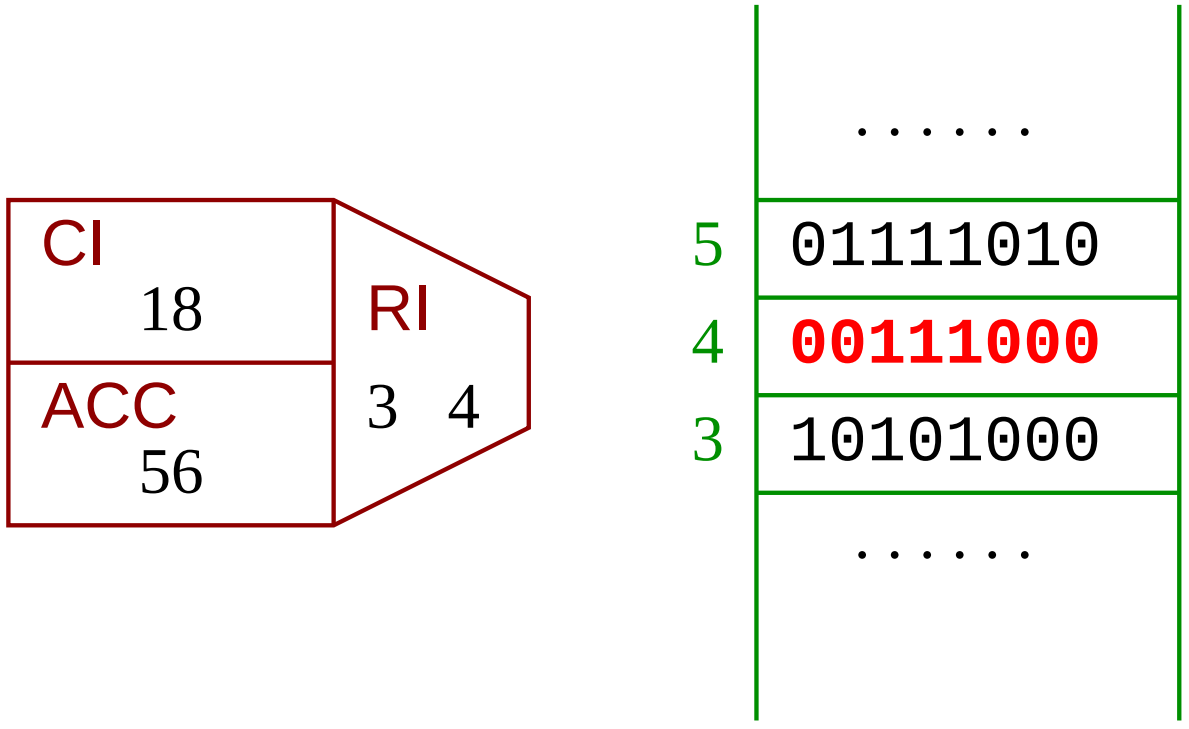
Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif

Les instructions élémentaires:

2	010	LOAD	copier la valeur de la mémoire dans l'accumulateur
3	011	STORE	copier la valeur de l'accumulateur dans la mémoire
4	100	ADD	ajouter la valeur de la mémoire à l'accumulateur
5	101	MPY	multiplier la valeur dans l'accumulateur avec la valeur de la mémoire

Le fonctionnement d'un processeur primitif

Les instructions élémentaires:

2	010	LOAD	copier la valeur de la mémoire dans l'accumulateur
3	011	STORE	copier la valeur de l'accumulateur dans la mémoire
4	100	ADD	ajouter la valeur de la mémoire à l'accumulateur
5	101	MPY	multiplier la valeur dans l'accumulateur avec la valeur de la mémoire
1	001	BR	copier l'adresse indiqueée dans le compteur d'instructions

$\longrightarrow$ interruption de la séquence,
continuation avec une autre partie du programme

Le fonctionnement d'un processeur primitif

Les instructions élémentaires:

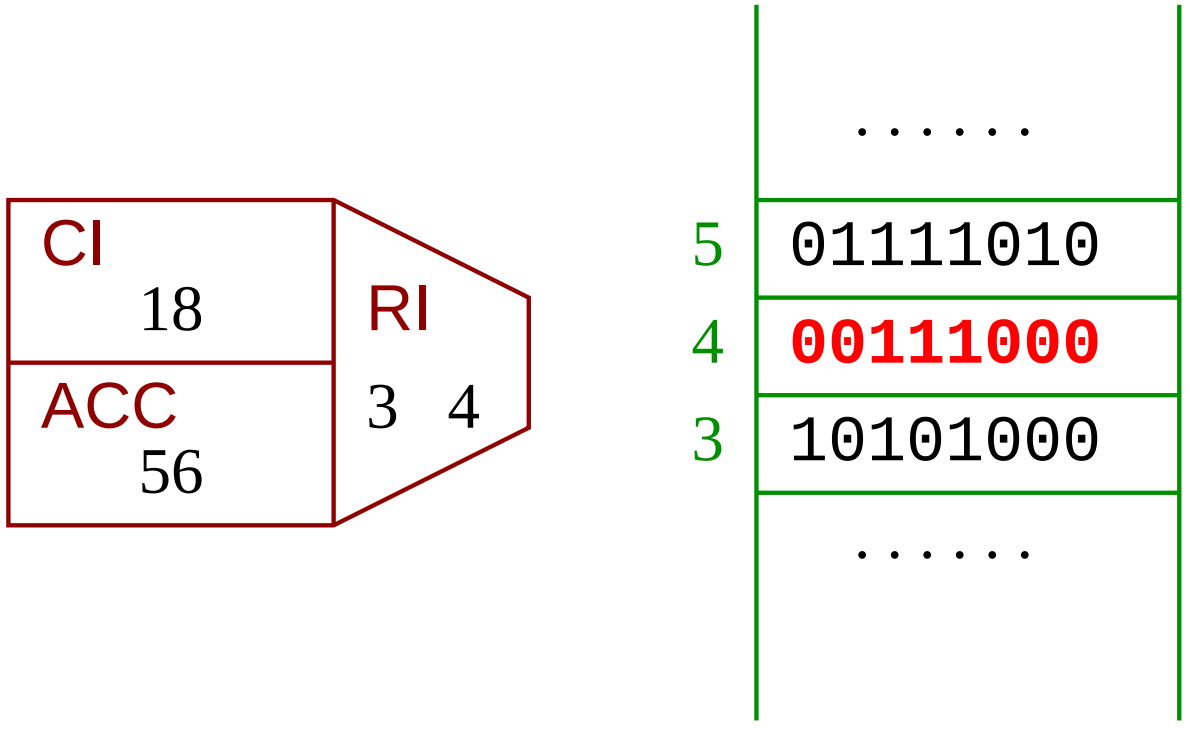
2	010	LOAD	copier la valeur de la mémoire dans l'accumulateur
3	011	STORE	copier la valeur de l'accumulateur dans la mémoire
4	100	ADD	ajouter la valeur de la mémoire à l'accumulateur
5	101	MPY	multiplier la valeur dans l'accumulateur avec la valeur de la mémoire
1	001	BR	copier l'adresse indiqueée dans le compteur d'instructions
0	000	BRG	copier l'adresse indiqueée dans le compteur d'instructions si l'accumulateur ne vaut pas zéro

Le fonctionnement d'un processeur primitif

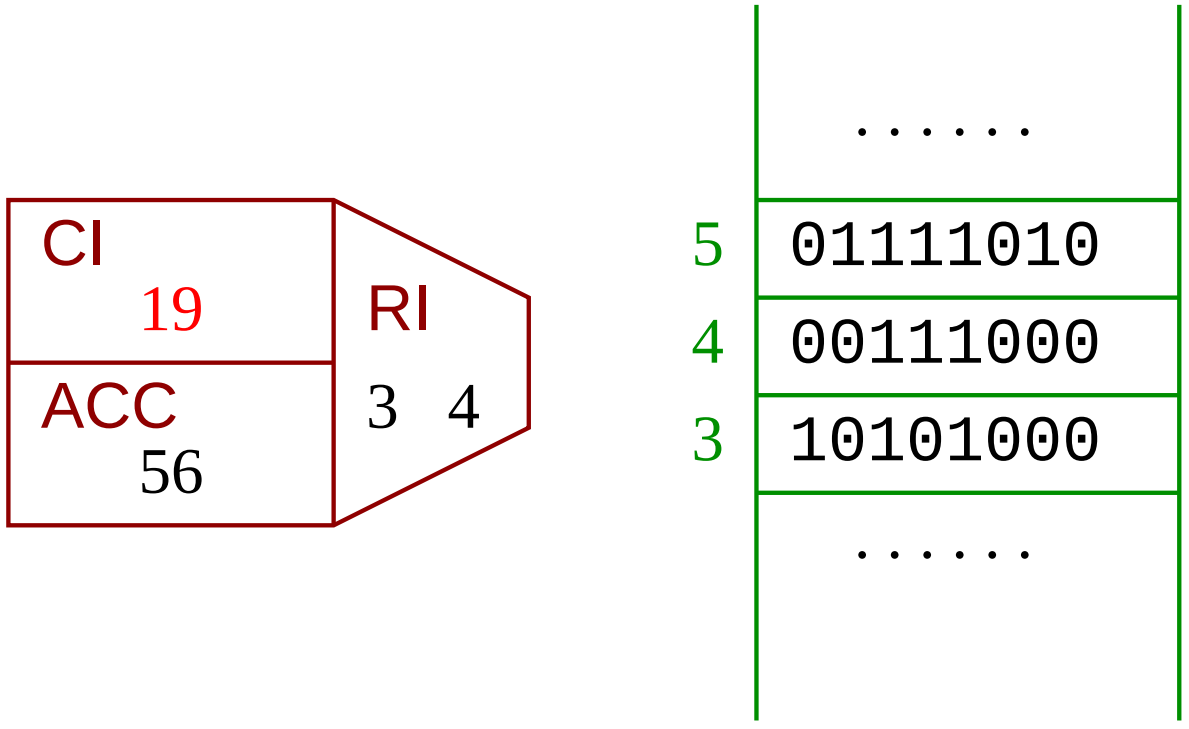
Les instructions élémentaires:

2	010	LOAD	copier la valeur de la mémoire dans l'accumulateur
3	011	STORE	copier la valeur de l'accumulateur dans la mémoire
4	100	ADD	ajouter la valeur de la mémoire à l'accumulateur
5	101	MPY	multiplier la valeur dans l'accumulateur avec la valeur de la mémoire
1	001	BR branchement	copier l'adresse indiqueée dans le compteur d'instructions
0	000	BRG branchement conditionnel	copier l'adresse indiqueée dans le compteur d'instructions si l'accumulateur ne vaut pas zéro

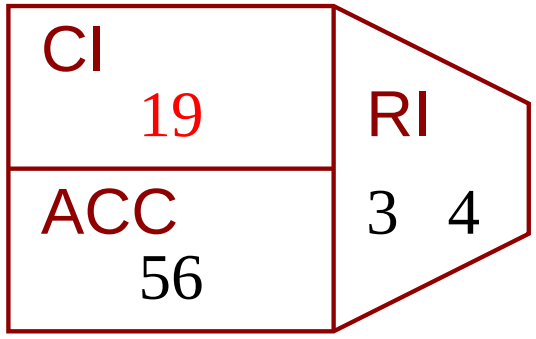
Le fonctionnement d'un processeur primitif

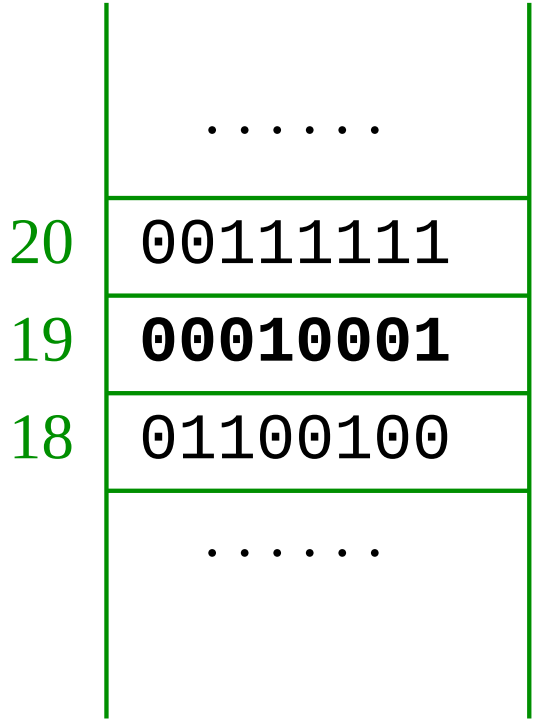


Le fonctionnement d'un processeur primitif

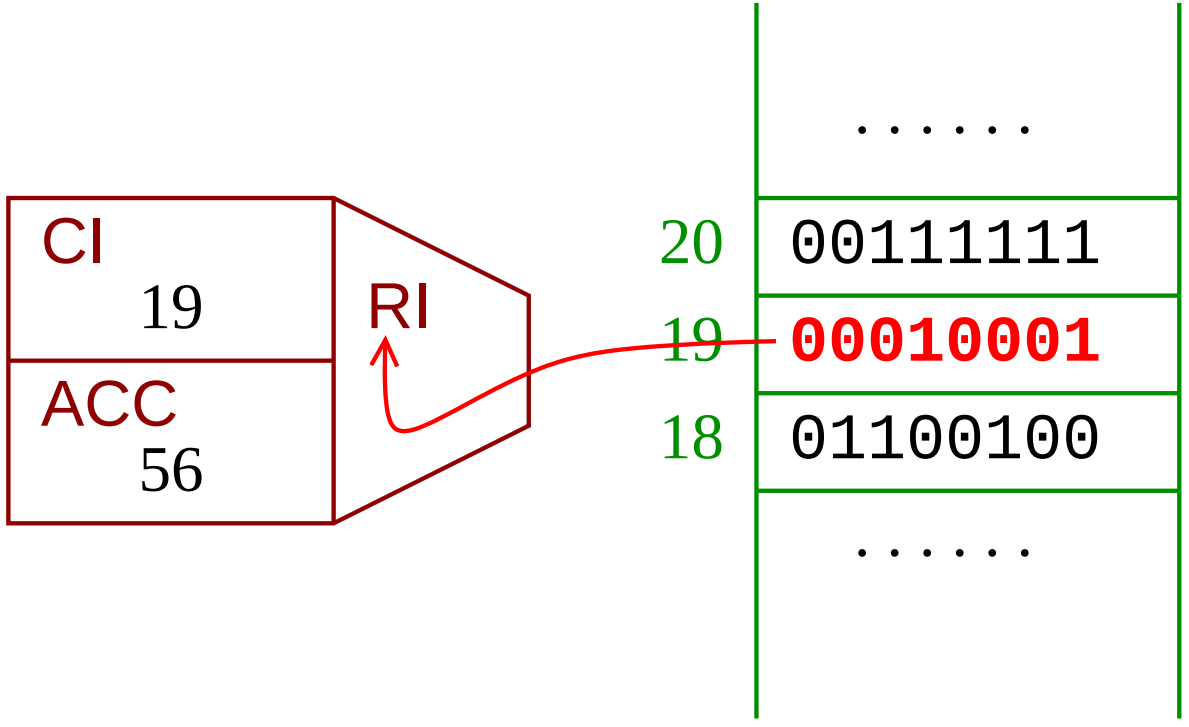


Le fonctionnement d'un processeur primitif

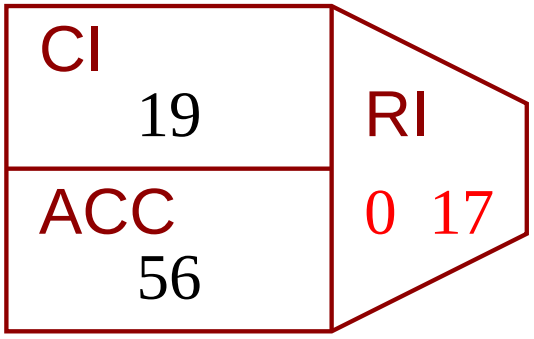


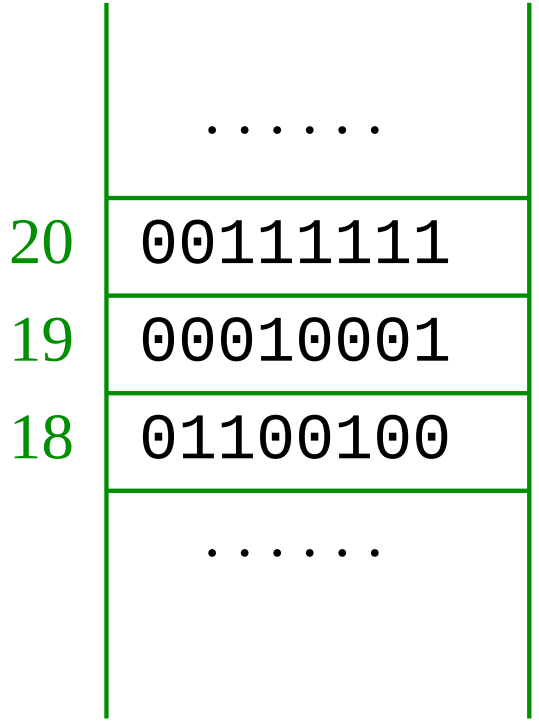


Le fonctionnement d'un processeur primitif

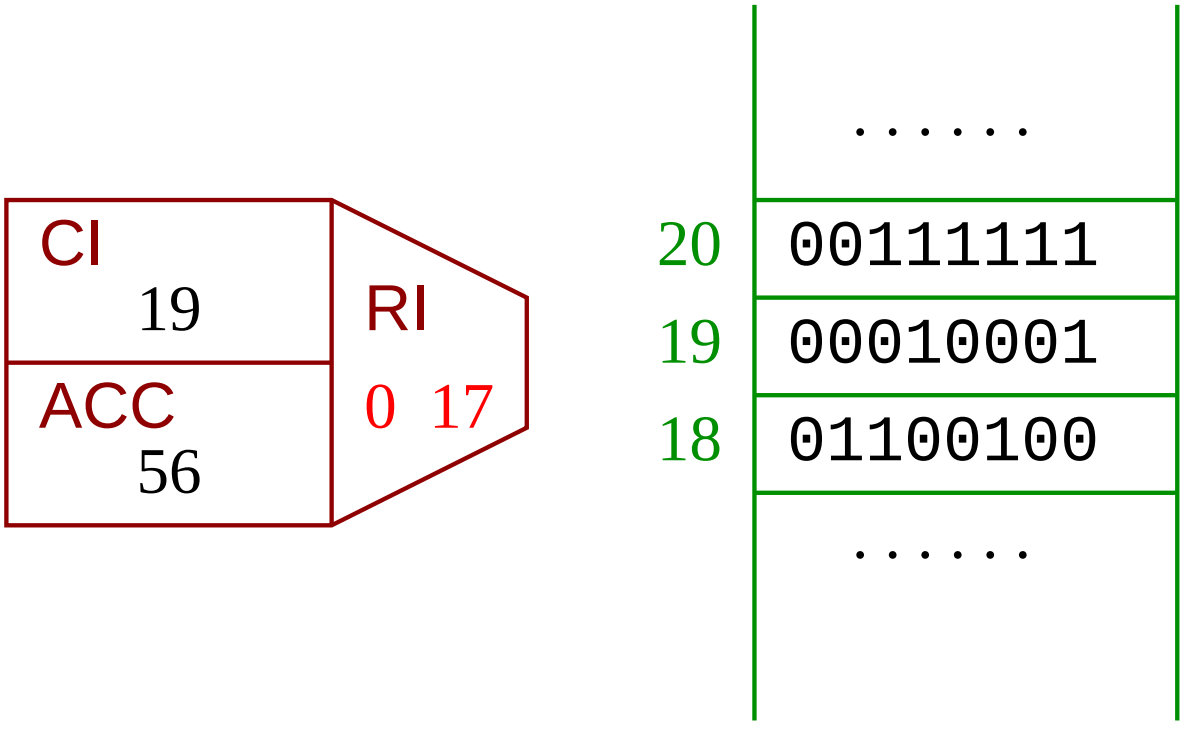


Le fonctionnement d'un processeur primitif





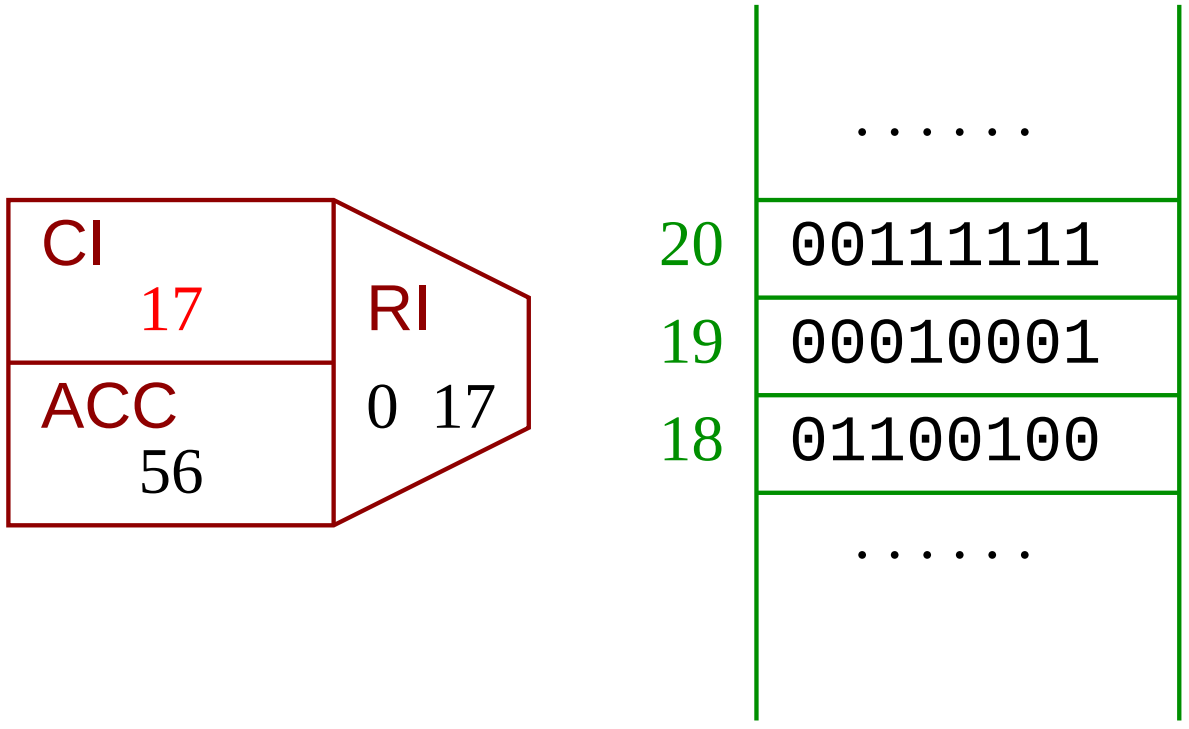
Le fonctionnement d'un processeur primitif



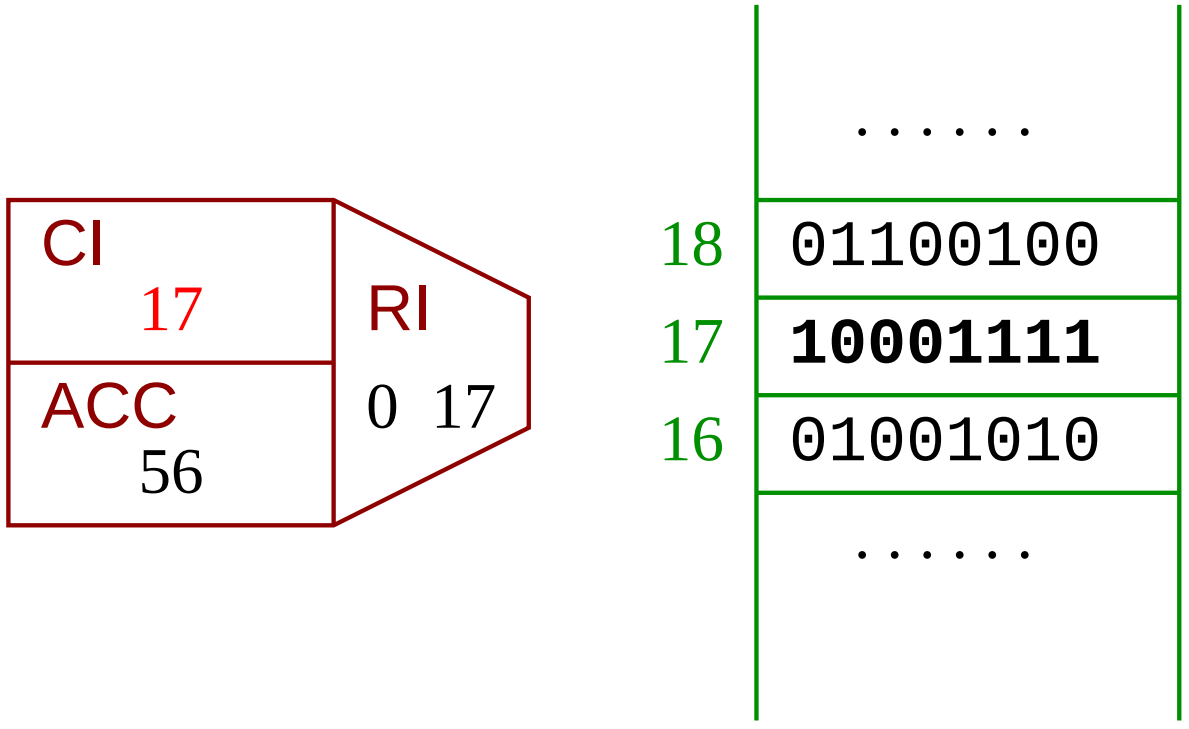
Opération 0:

continuer avec l'instruction dans l'adresse indiquée
si le contenu de l'accumulateur ne vaut pas zéro

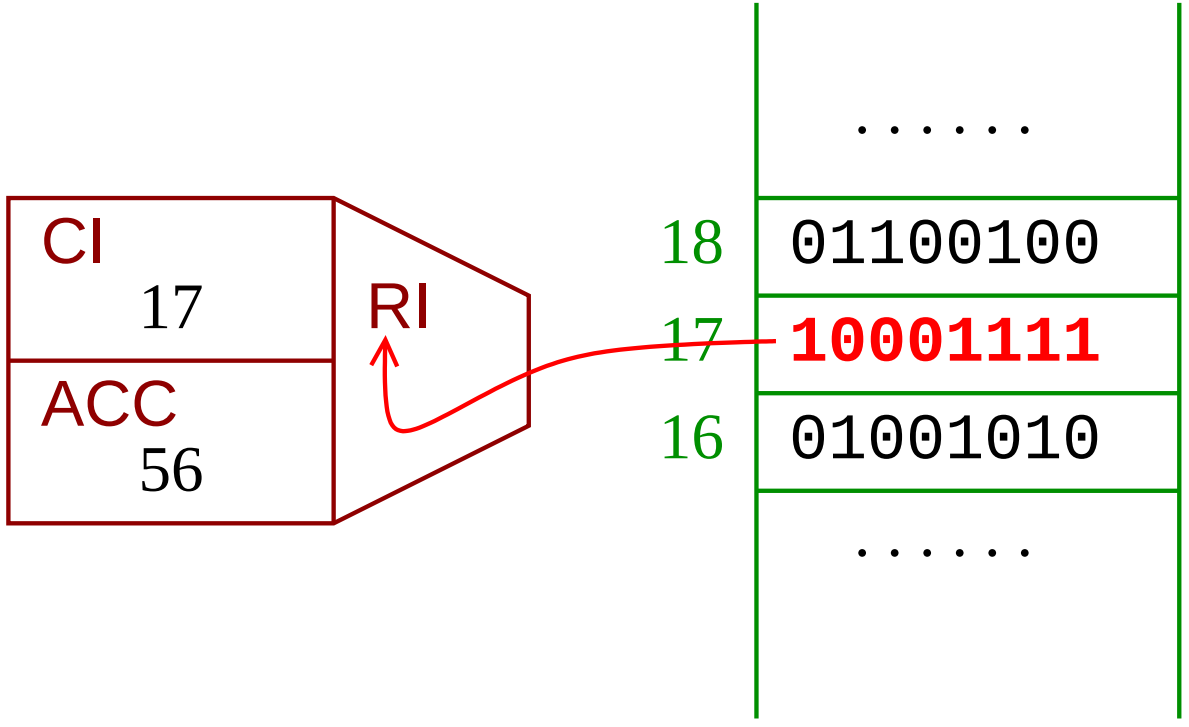
Le fonctionnement d'un processeur primitif



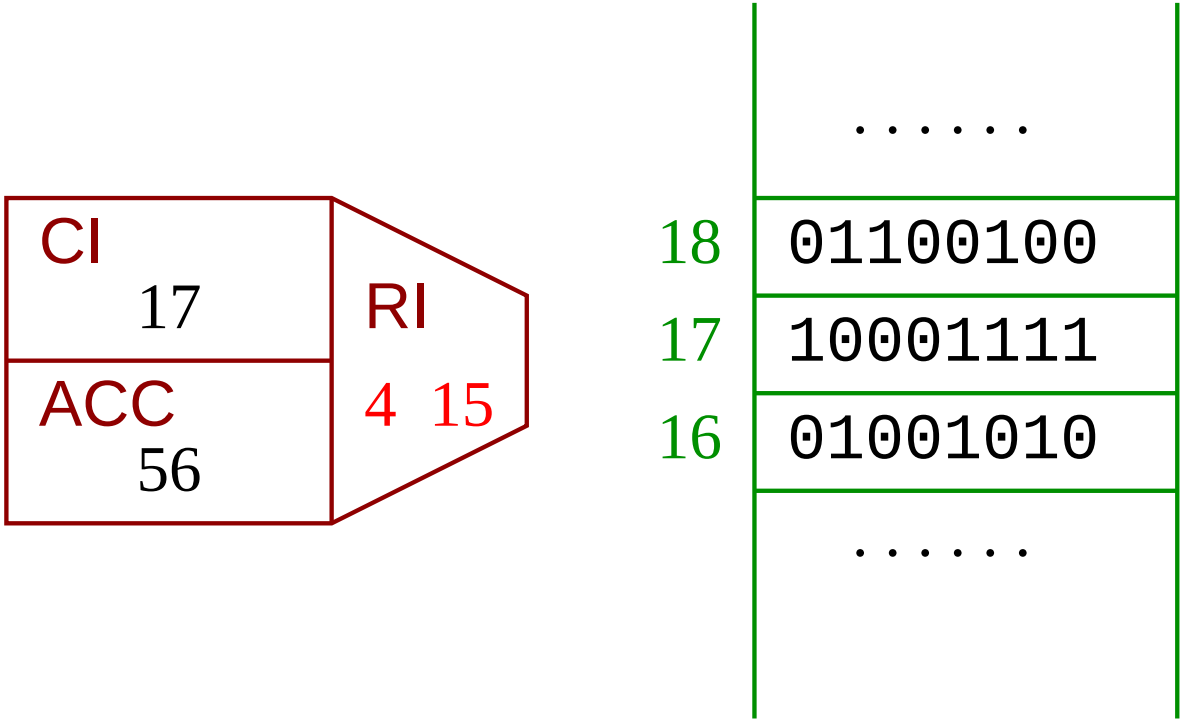
Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif



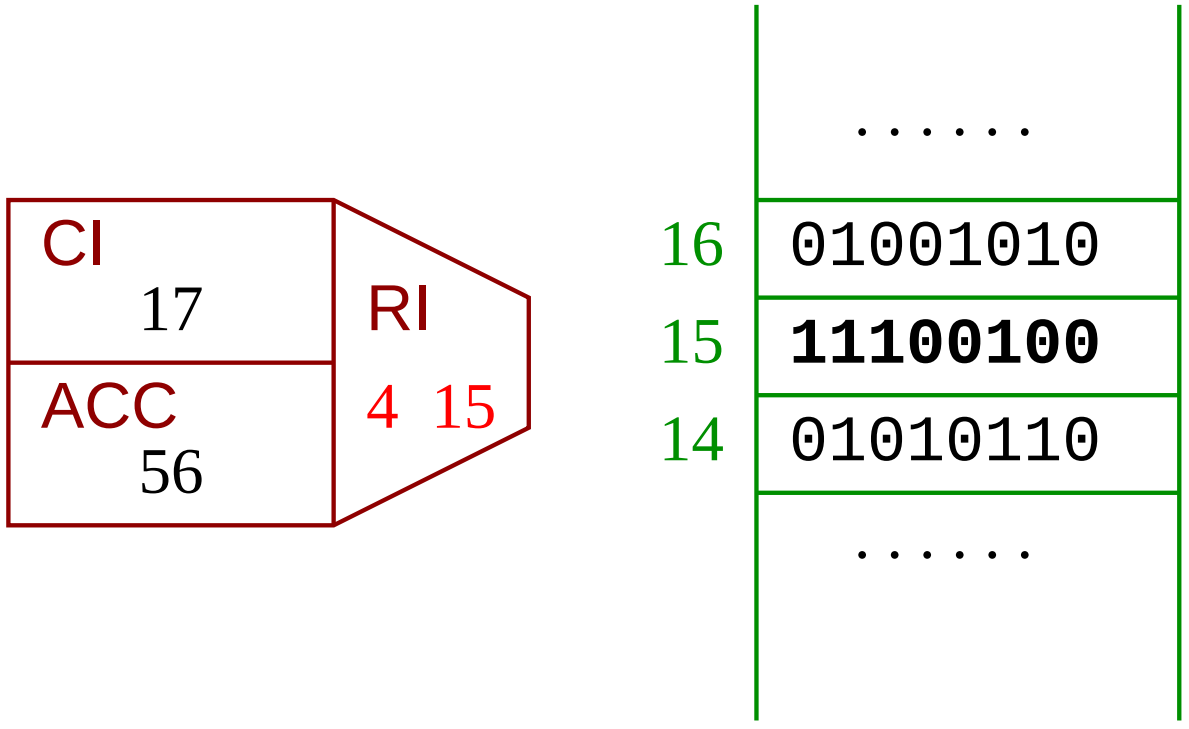
Le fonctionnement d'un processeur primitif



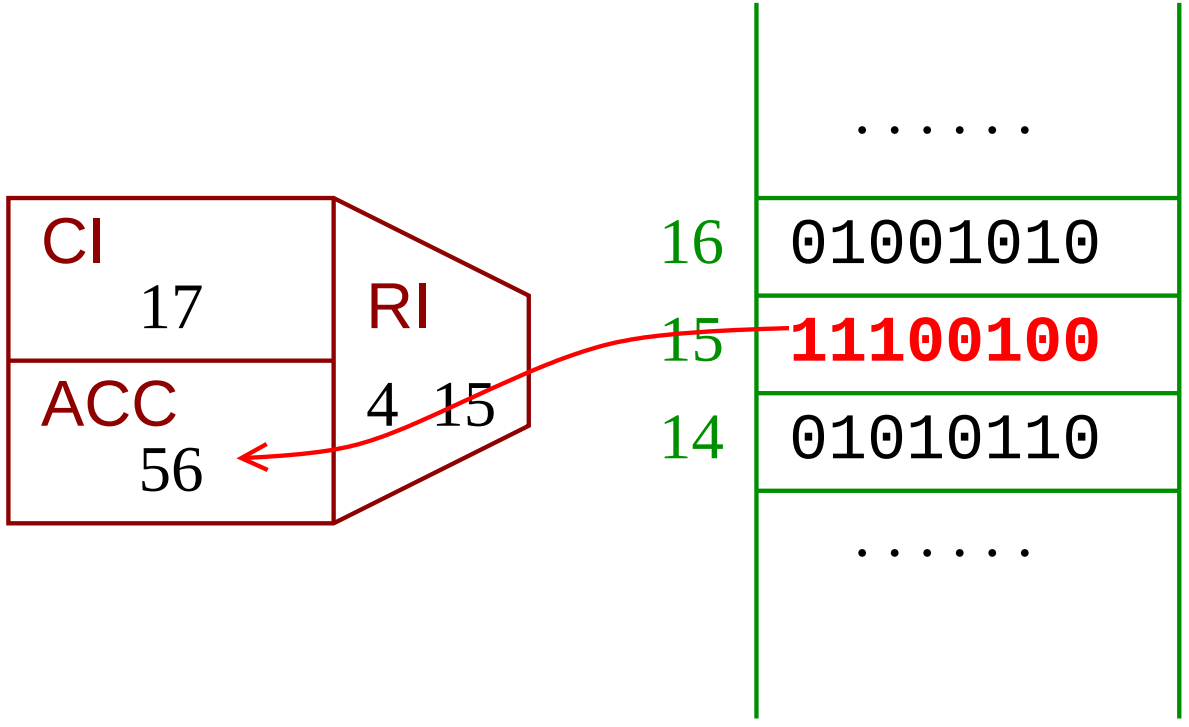
Opération 4:

ajouter à l'accumulateur le contenu dans l'adresse indiquée

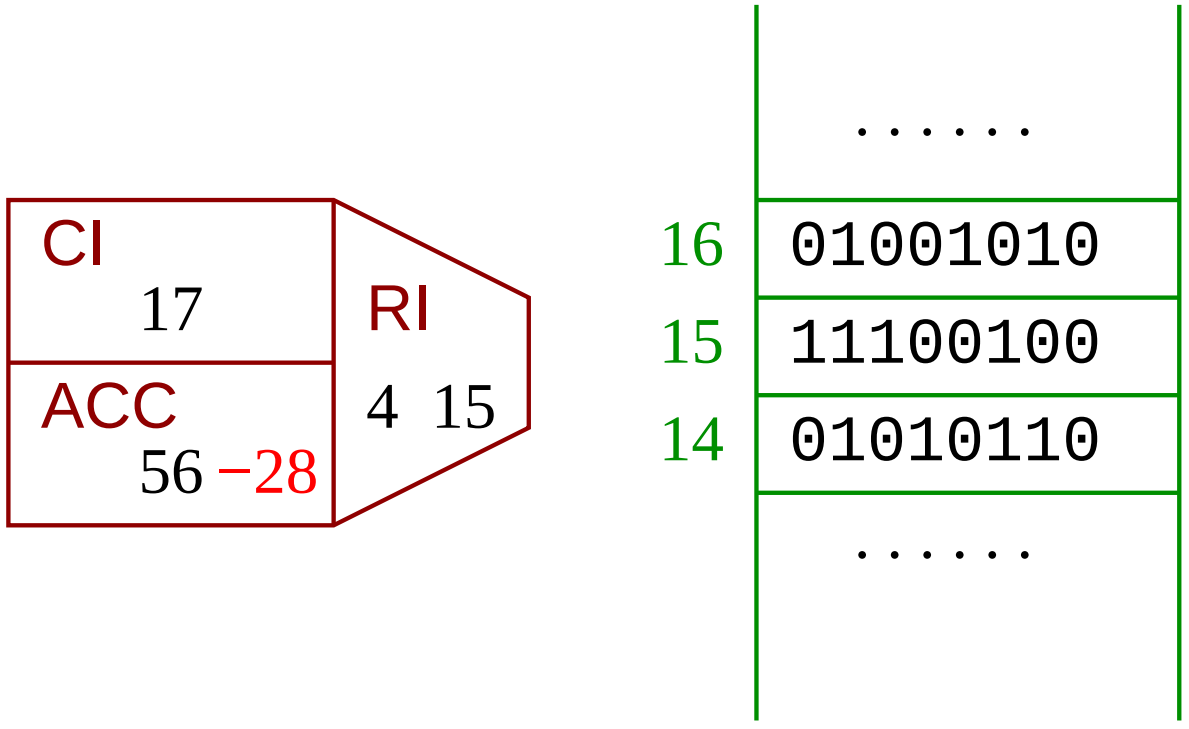
Le fonctionnement d'un processeur primitif



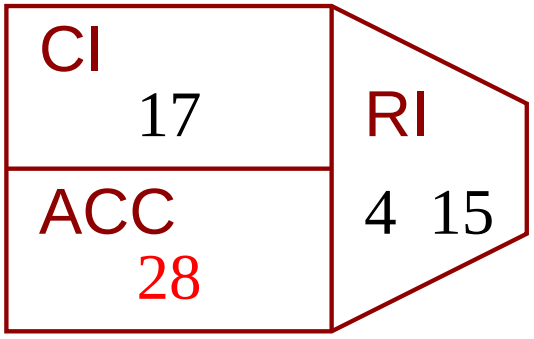
Le fonctionnement d'un processeur primitif

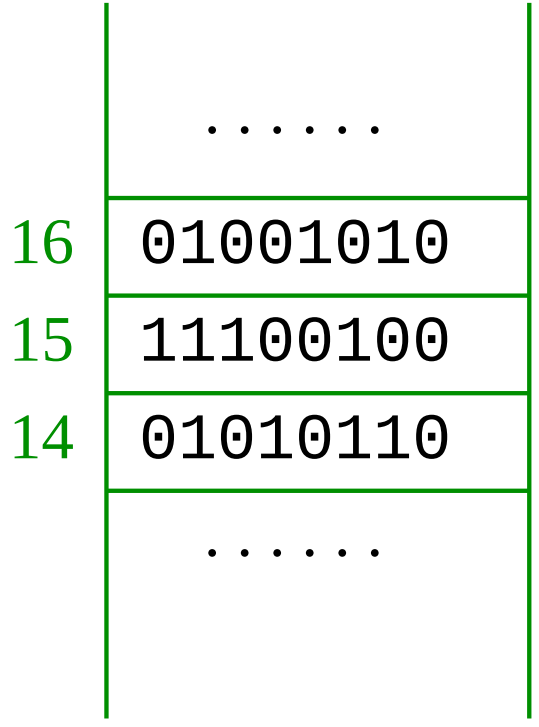


Le fonctionnement d'un processeur primitif

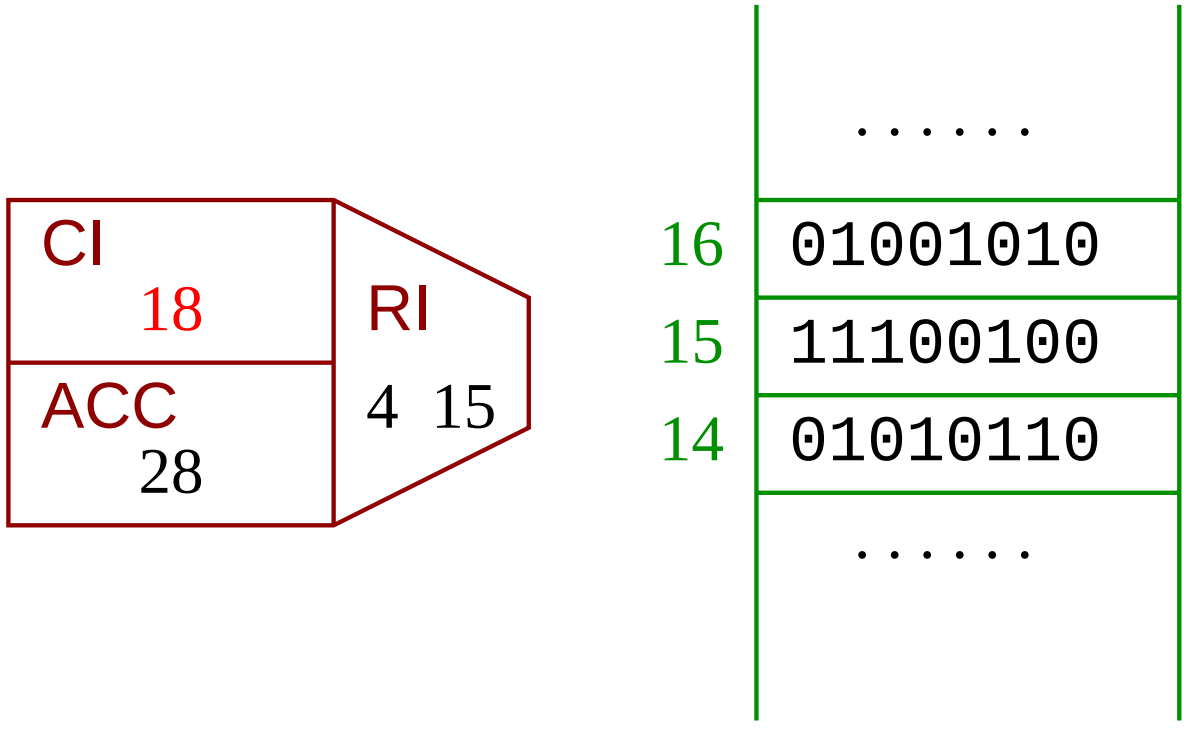


Le fonctionnement d'un processeur primitif

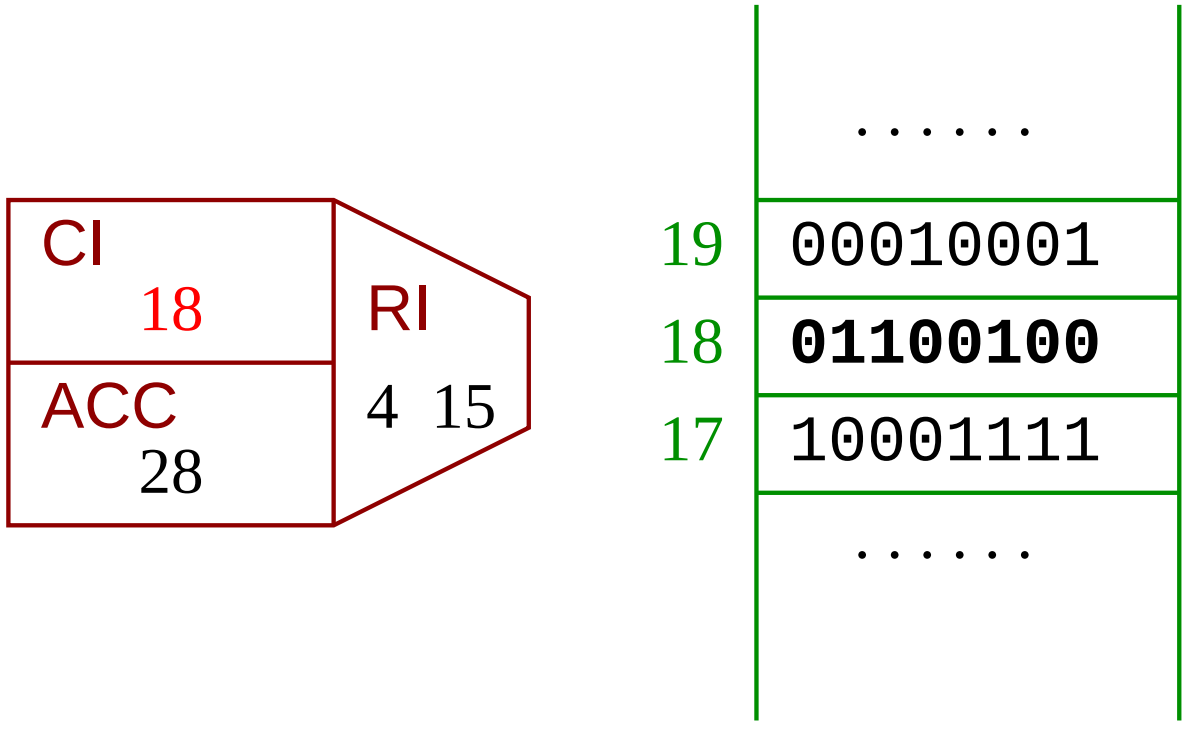




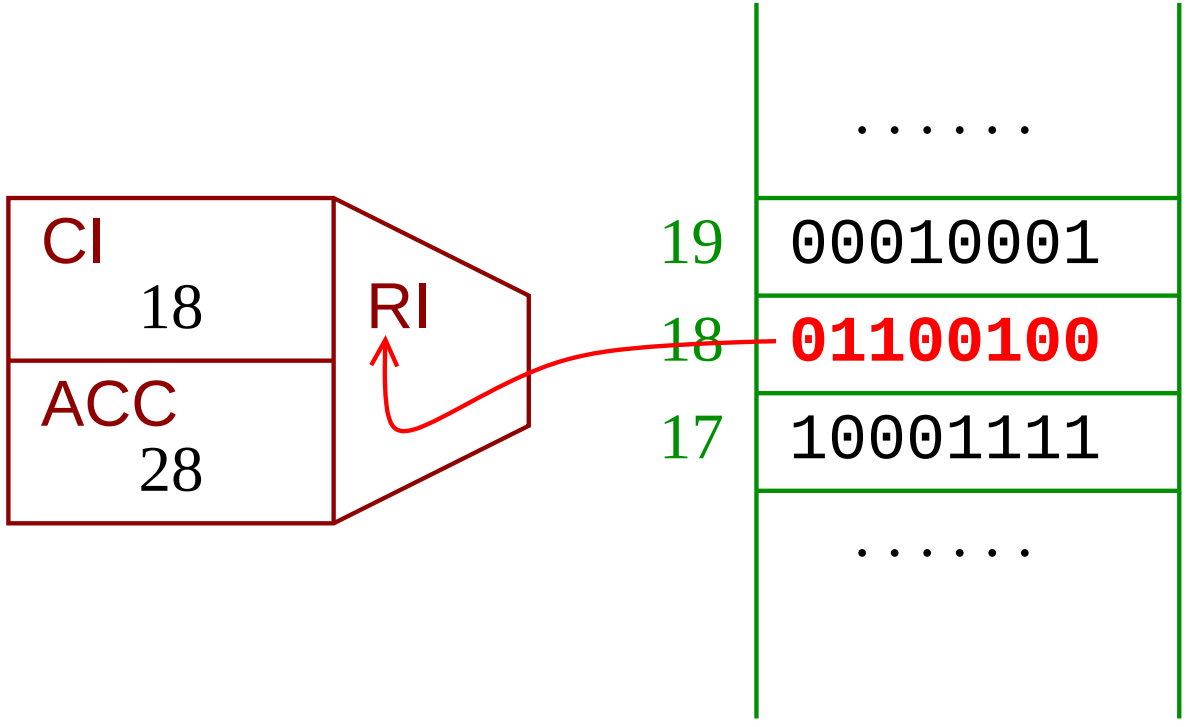
Le fonctionnement d'un processeur primitif



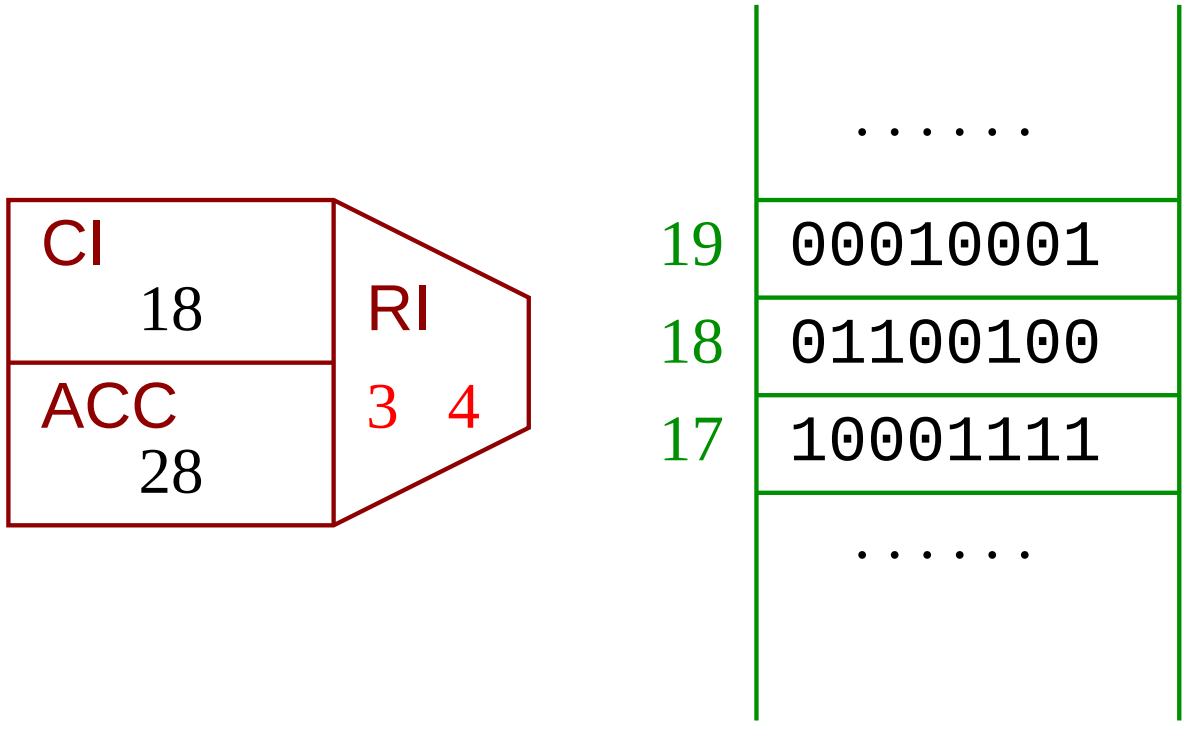
Le fonctionnement d'un processeur primitif



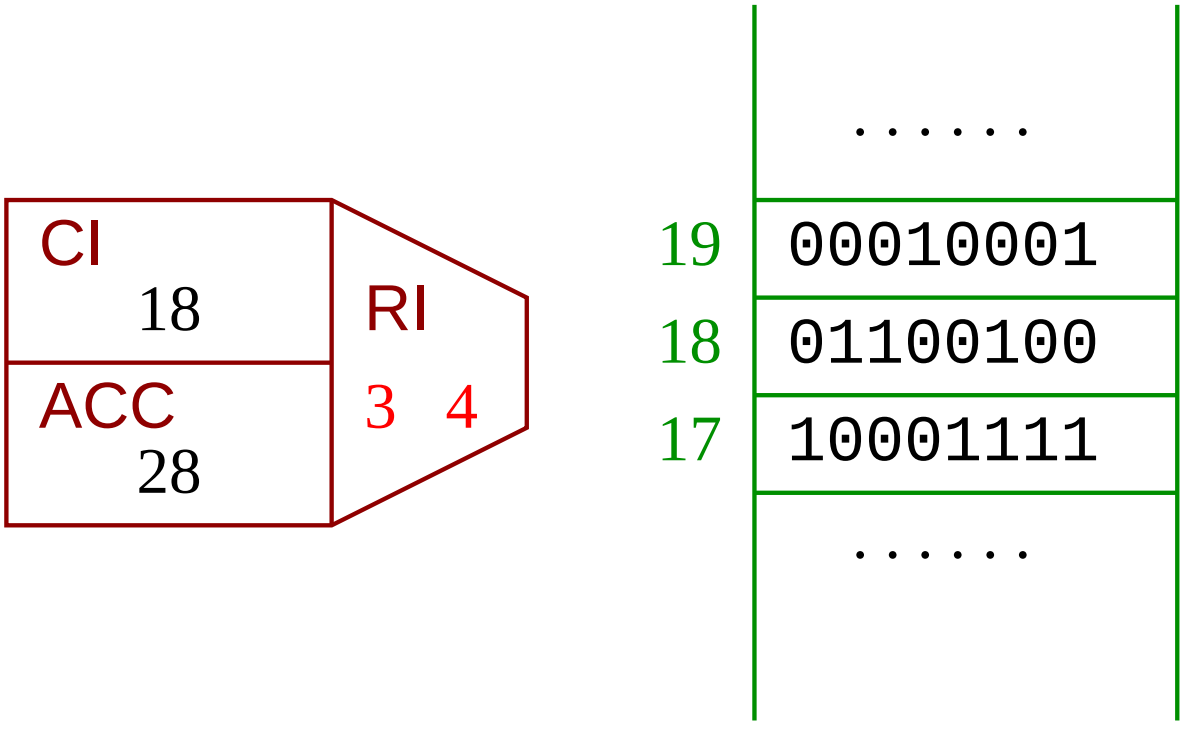
Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif



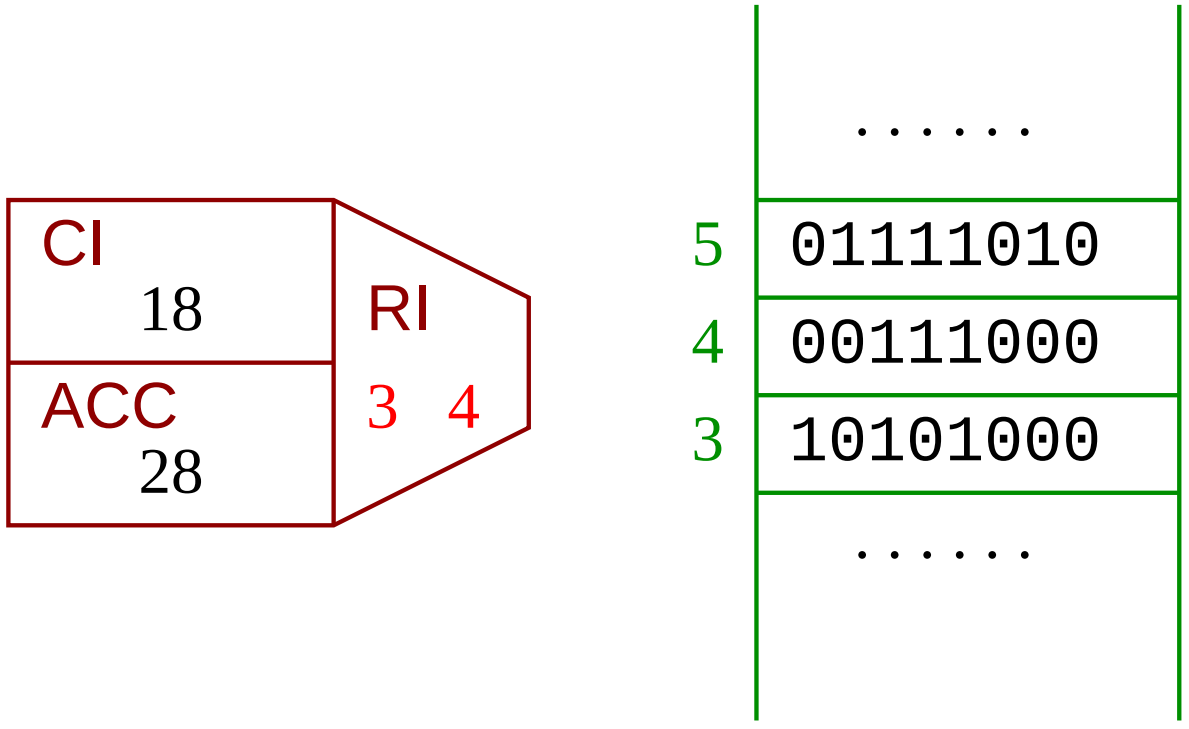
Le fonctionnement d'un processeur primitif



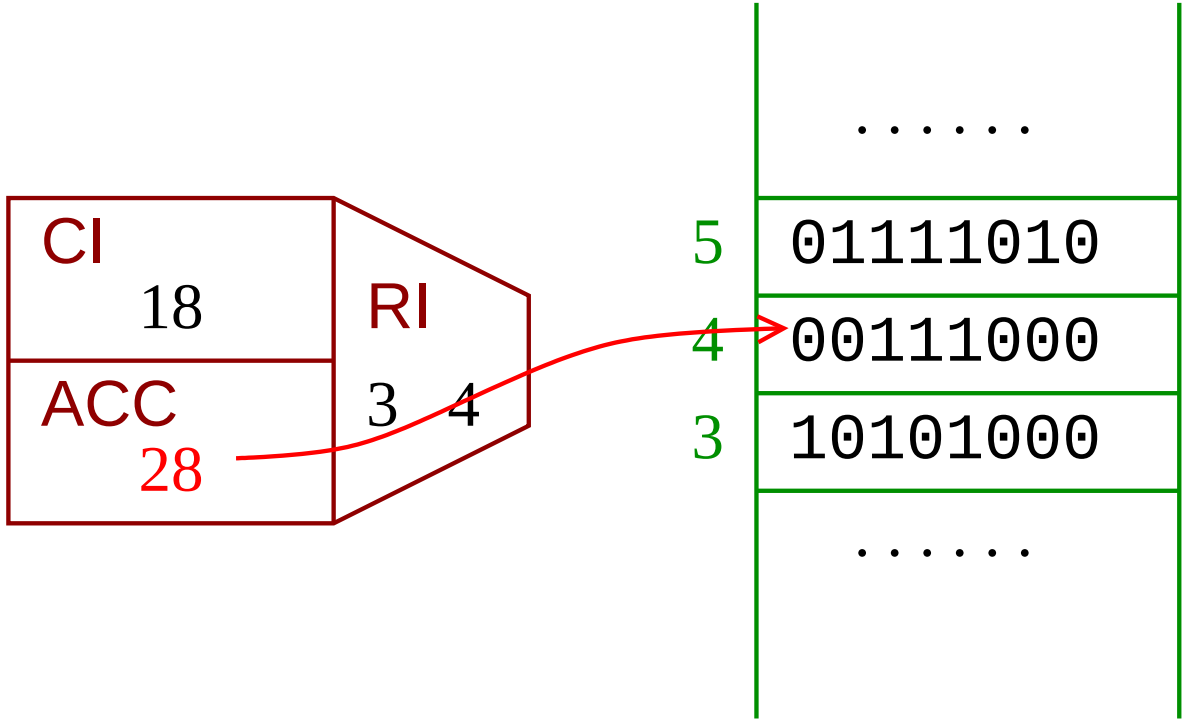
Opération 3:

écrire le contenu de l'accumulateur dans l'adresse indiquée

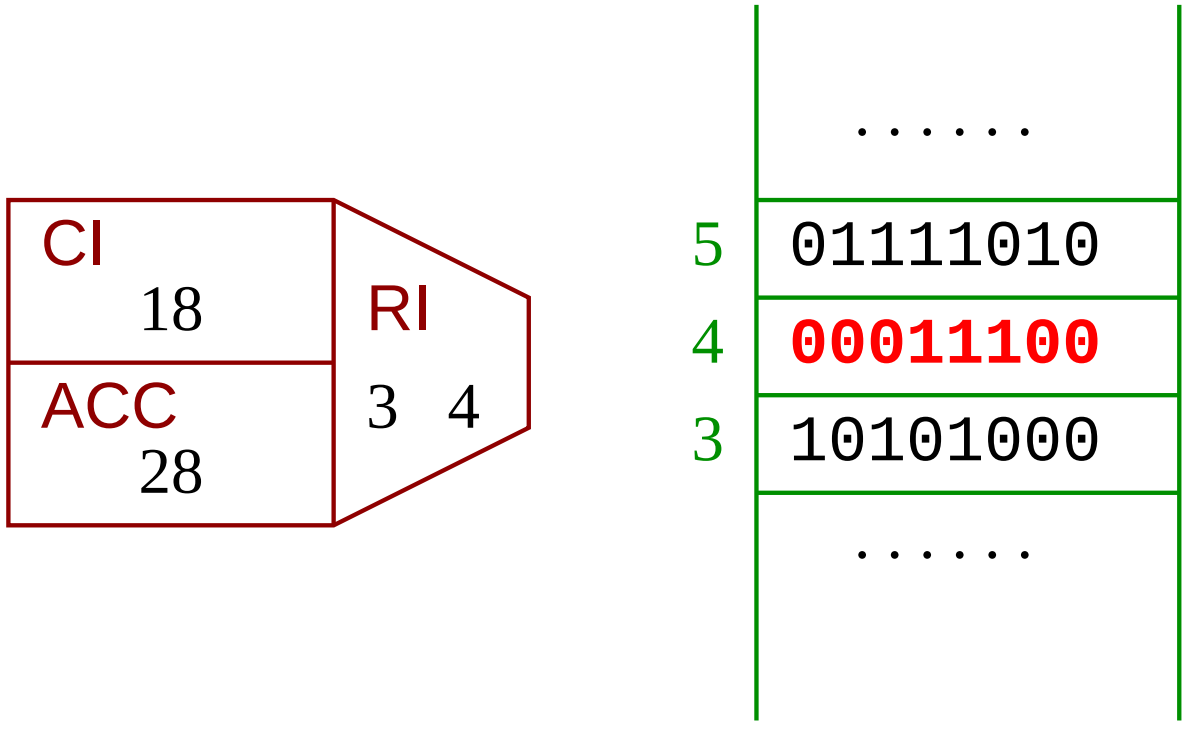
Le fonctionnement d'un processeur primitif



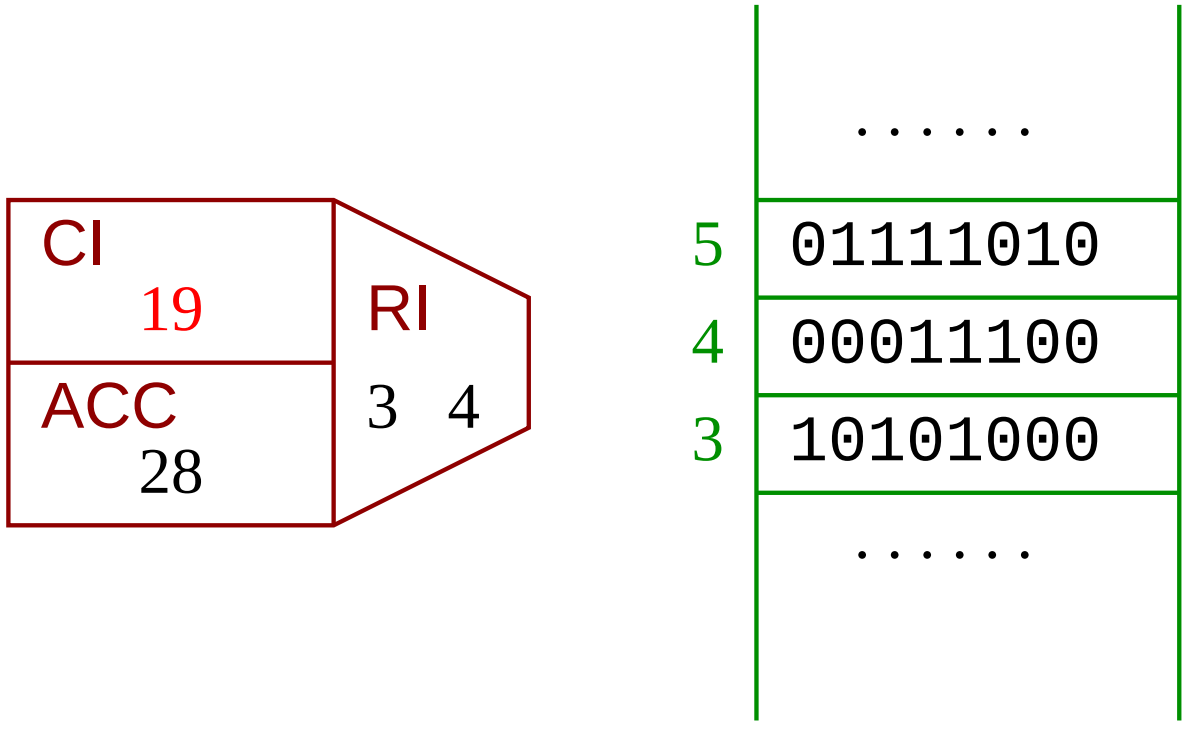
Le fonctionnement d'un processeur primitif



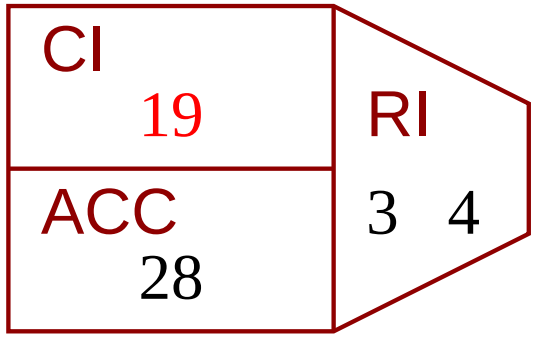
Le fonctionnement d'un processeur primitif

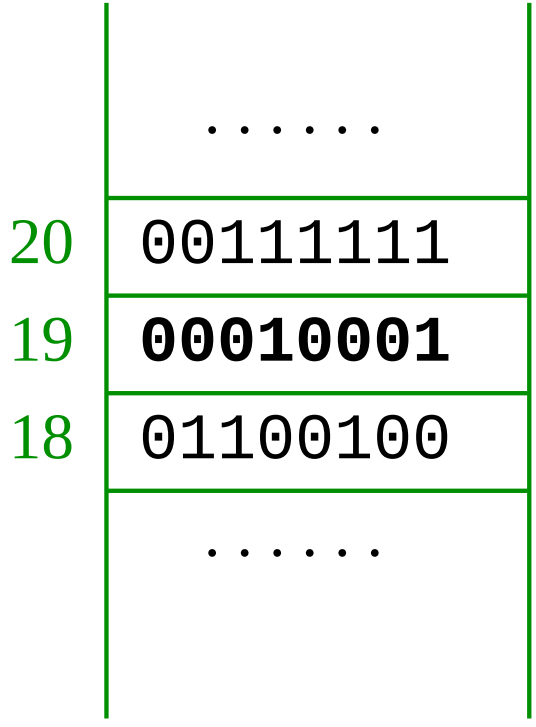


Le fonctionnement d'un processeur primitif

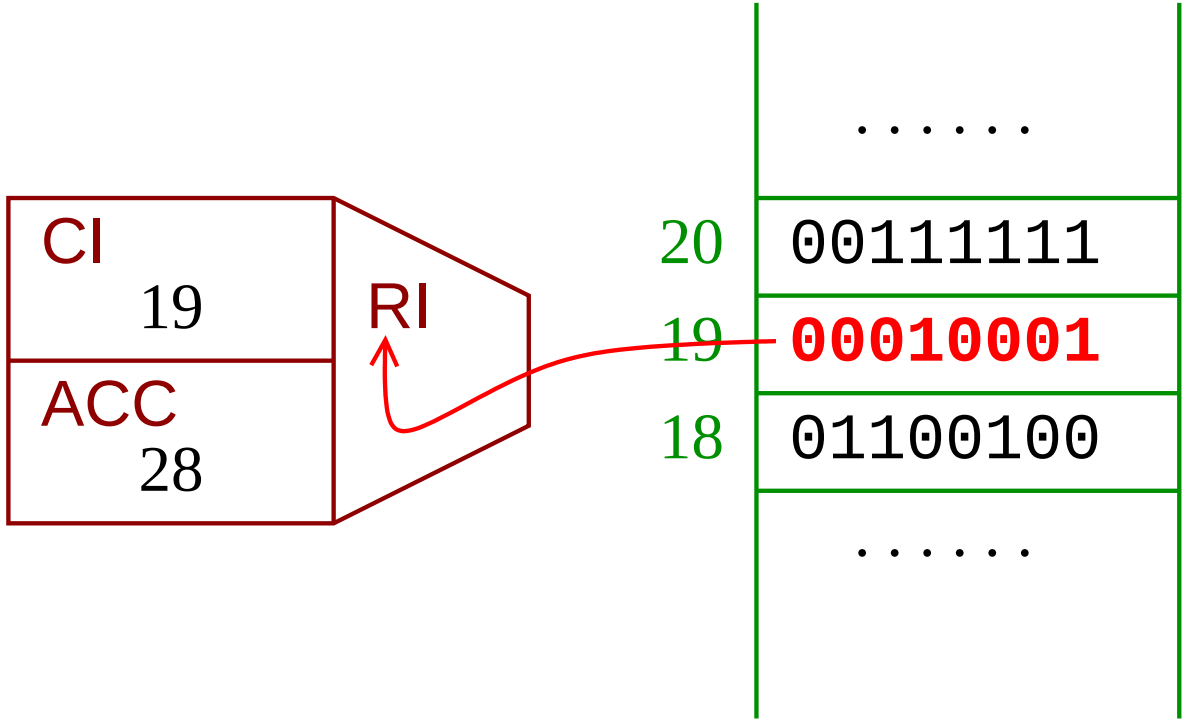


Le fonctionnement d'un processeur primitif

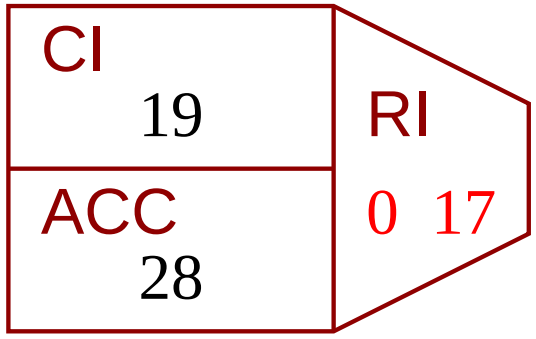


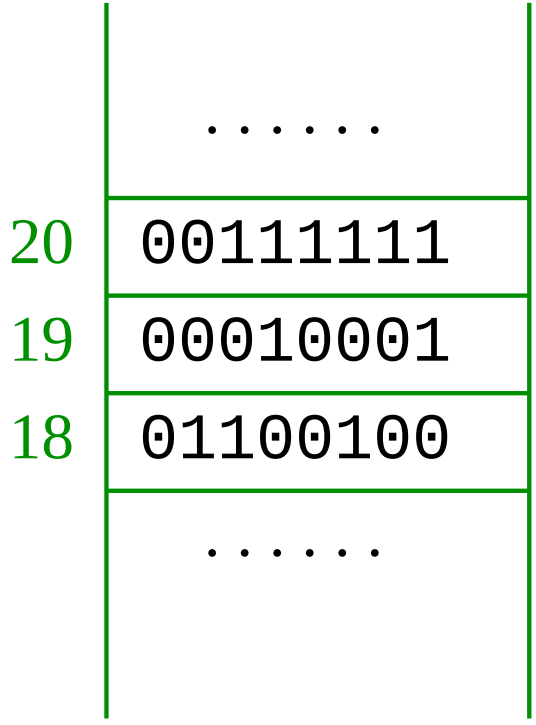


Le fonctionnement d'un processeur primitif

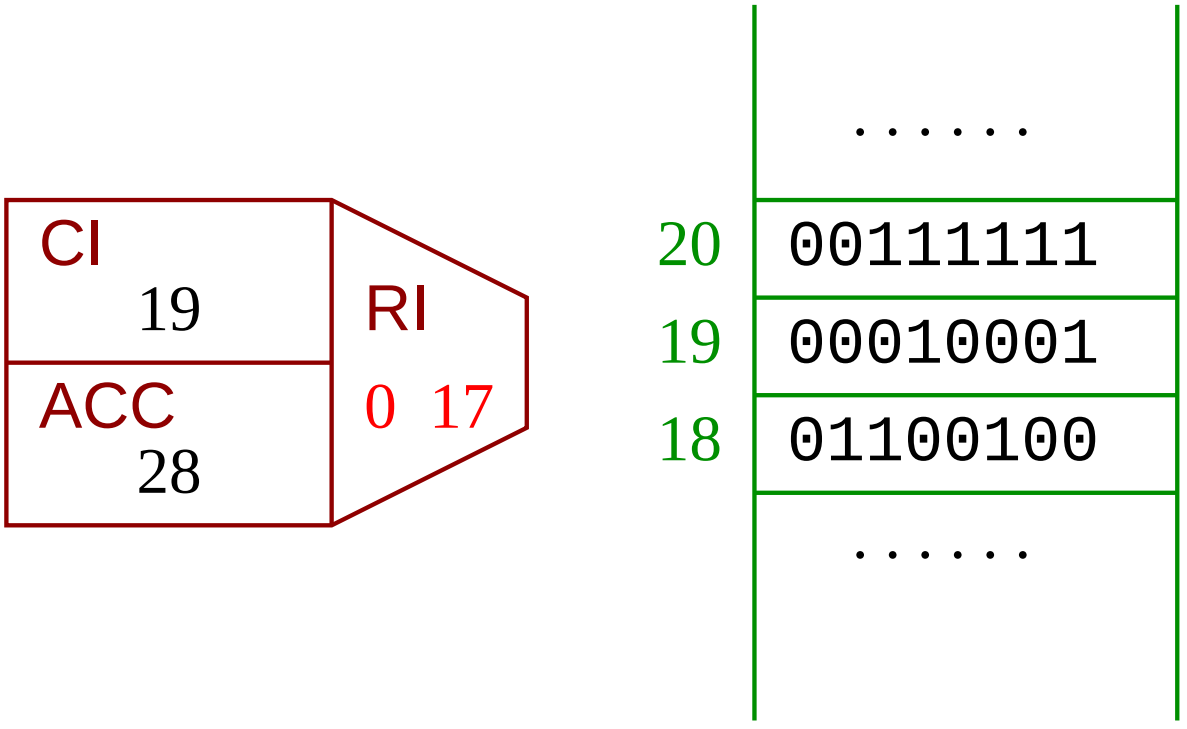


Le fonctionnement d'un processeur primitif





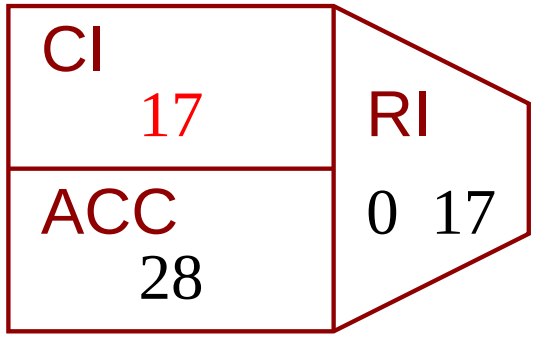
Le fonctionnement d'un processeur primitif

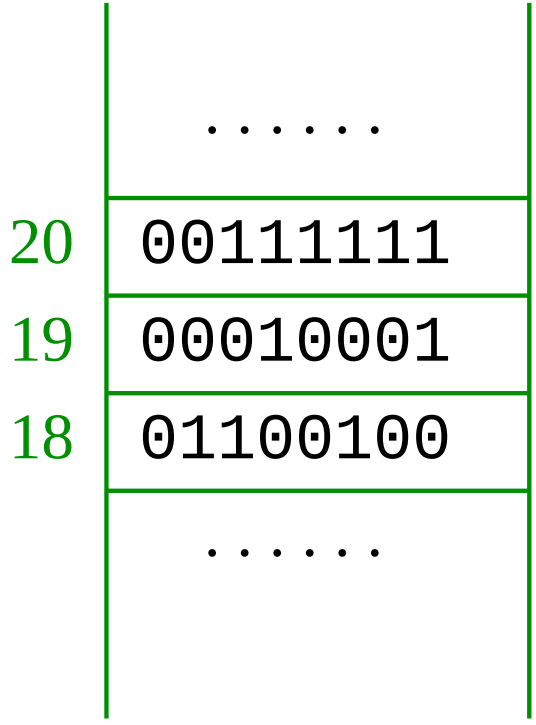


Opération 0:

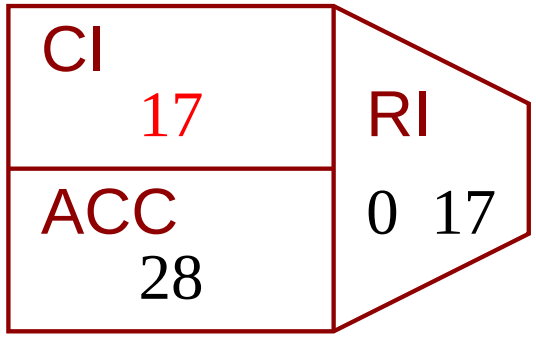
continuer avec l'instruction dans l'adresse indiquée
si le contenu de l'accumulateur ne vaut pas zéro

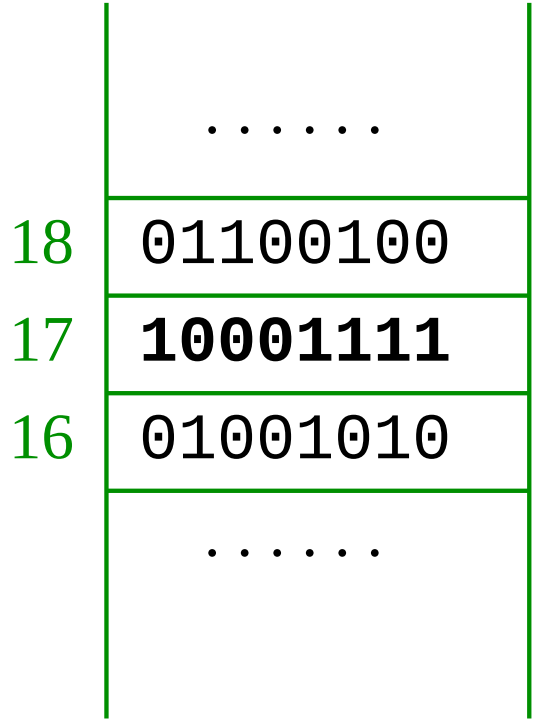
Le fonctionnement d'un processeur primitif



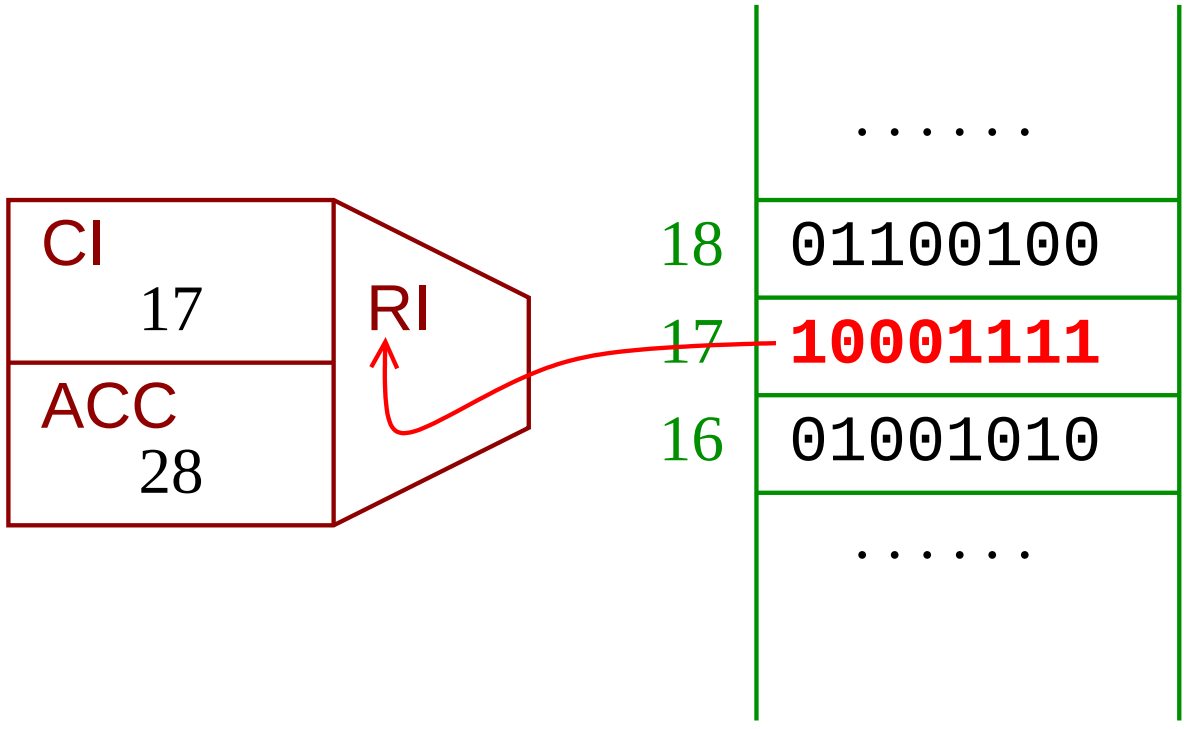


Le fonctionnement d'un processeur primitif

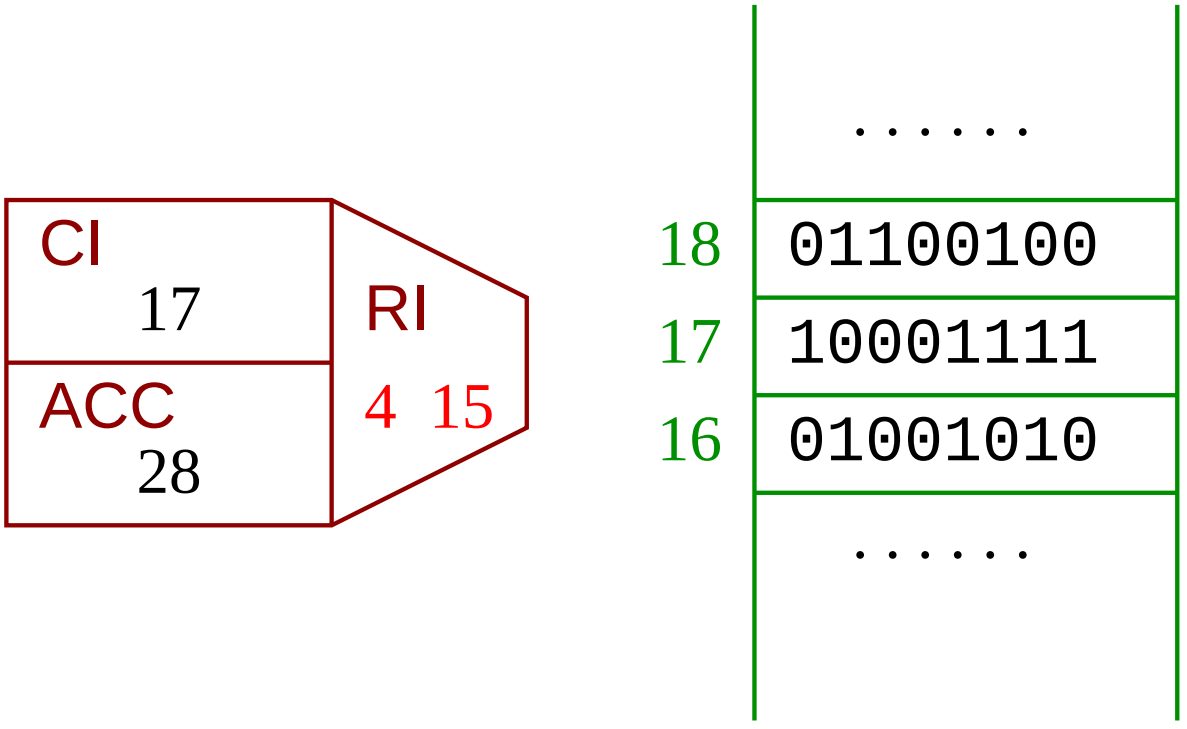




Le fonctionnement d'un processeur primitif



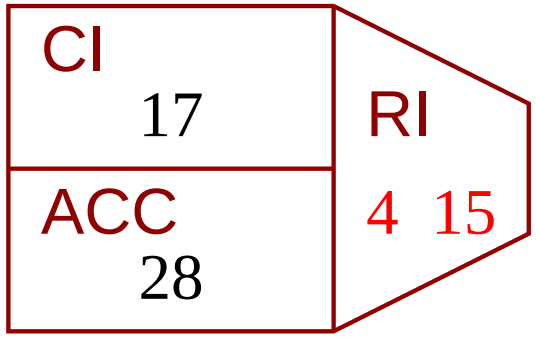
Le fonctionnement d'un processeur primitif

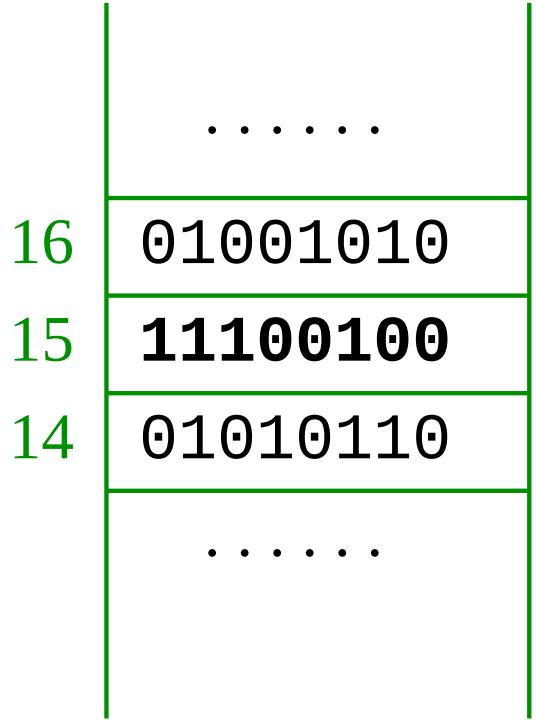


Opération 4:

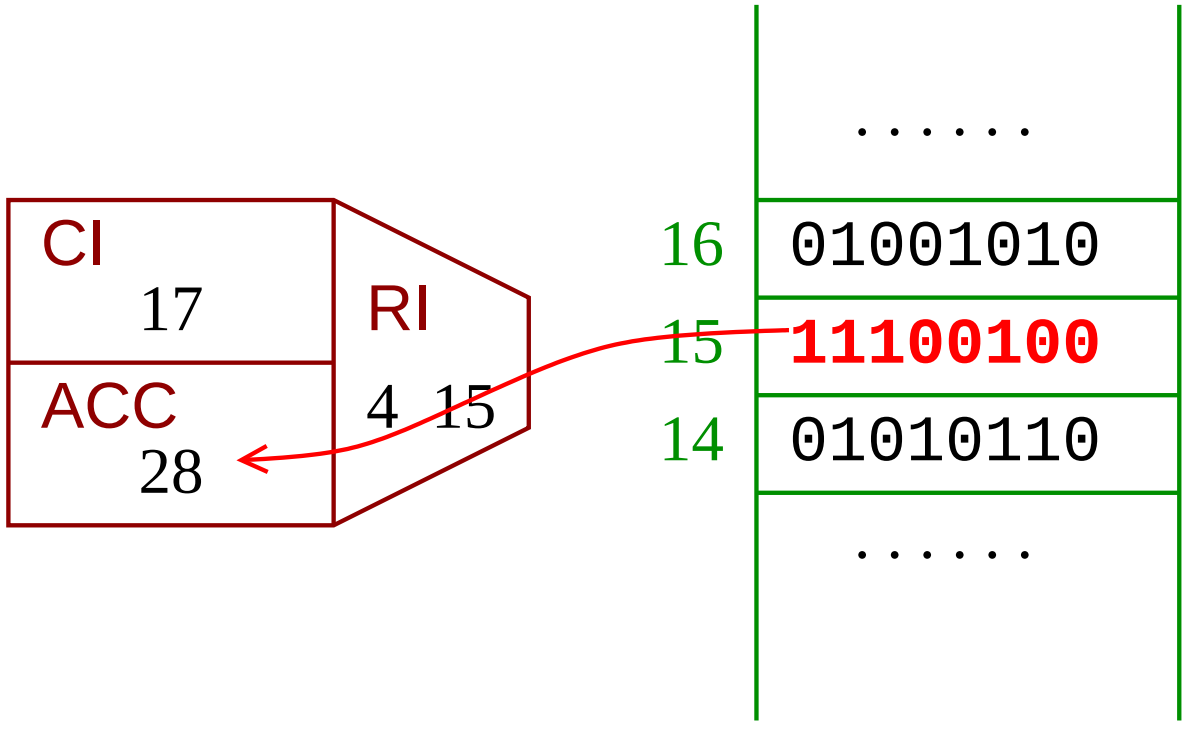
ajouter à l'accumulateur le contenu dans l'adresse indiquée

Le fonctionnement d'un processeur primitif

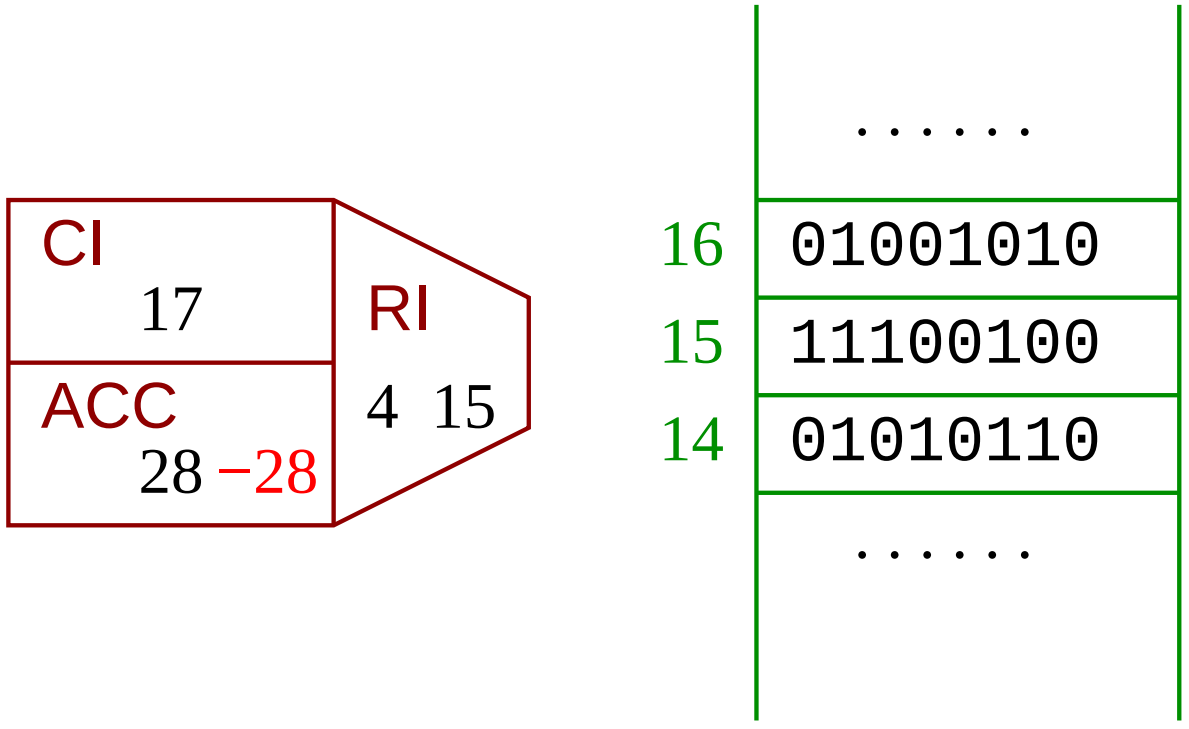




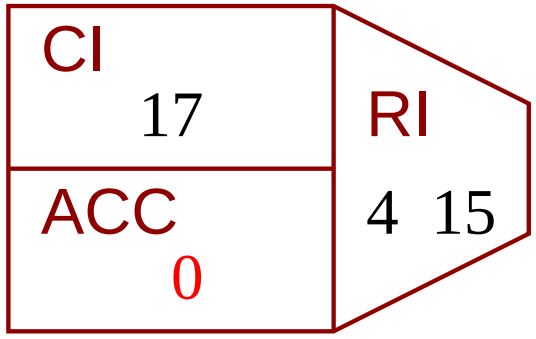
Le fonctionnement d'un processeur primitif

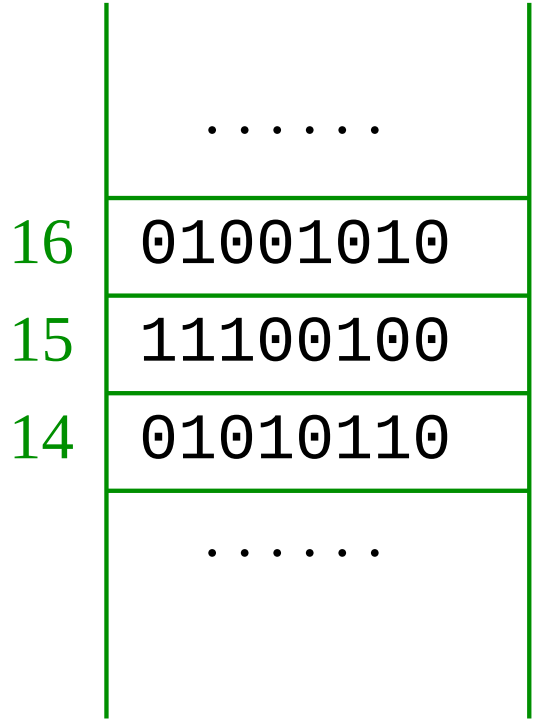


Le fonctionnement d'un processeur primitif

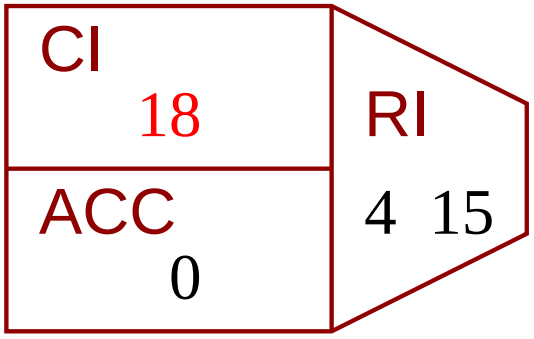


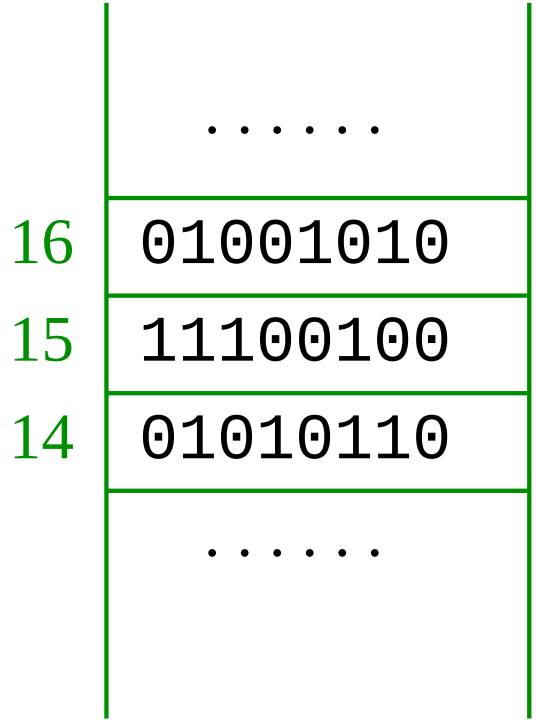
Le fonctionnement d'un processeur primitif



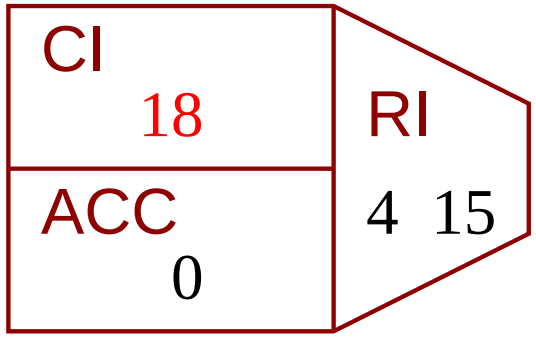


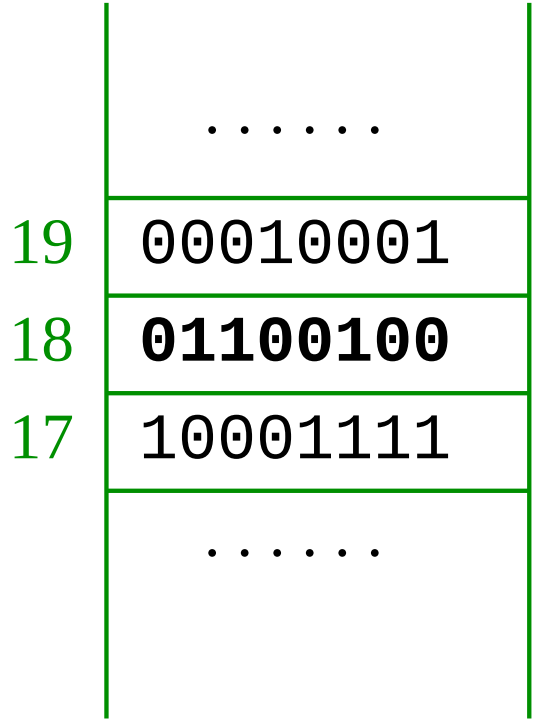
Le fonctionnement d'un processeur primitif



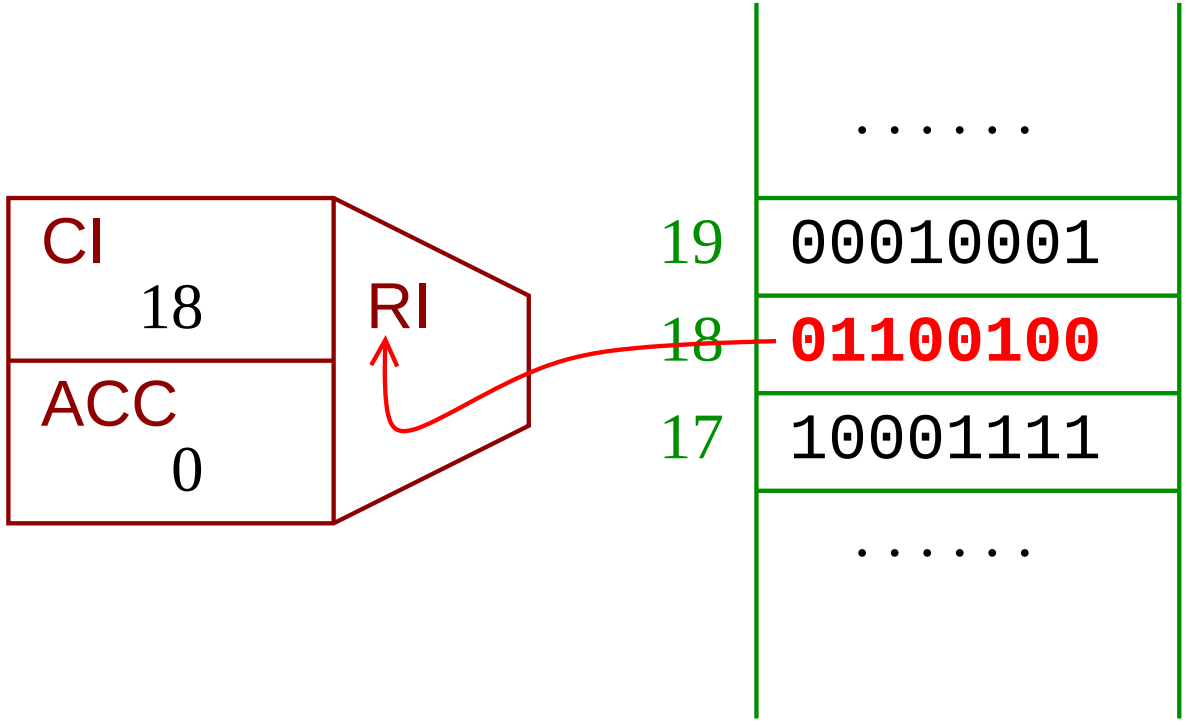


Le fonctionnement d'un processeur primitif

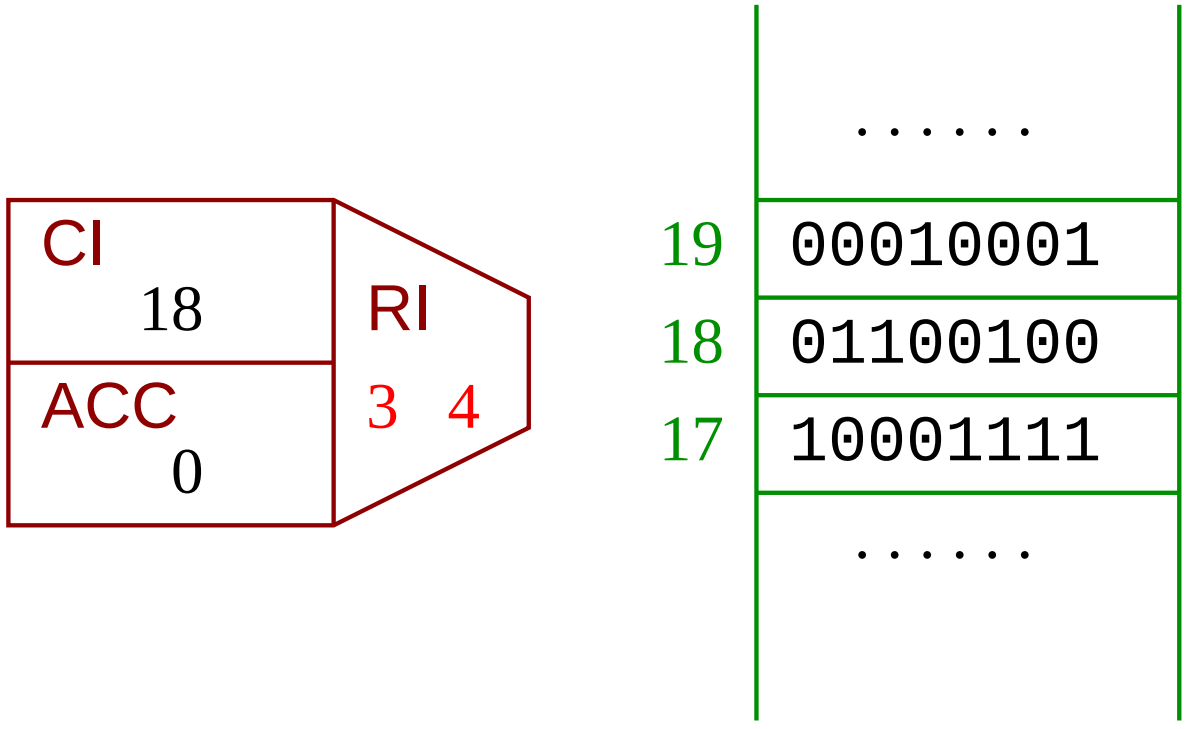




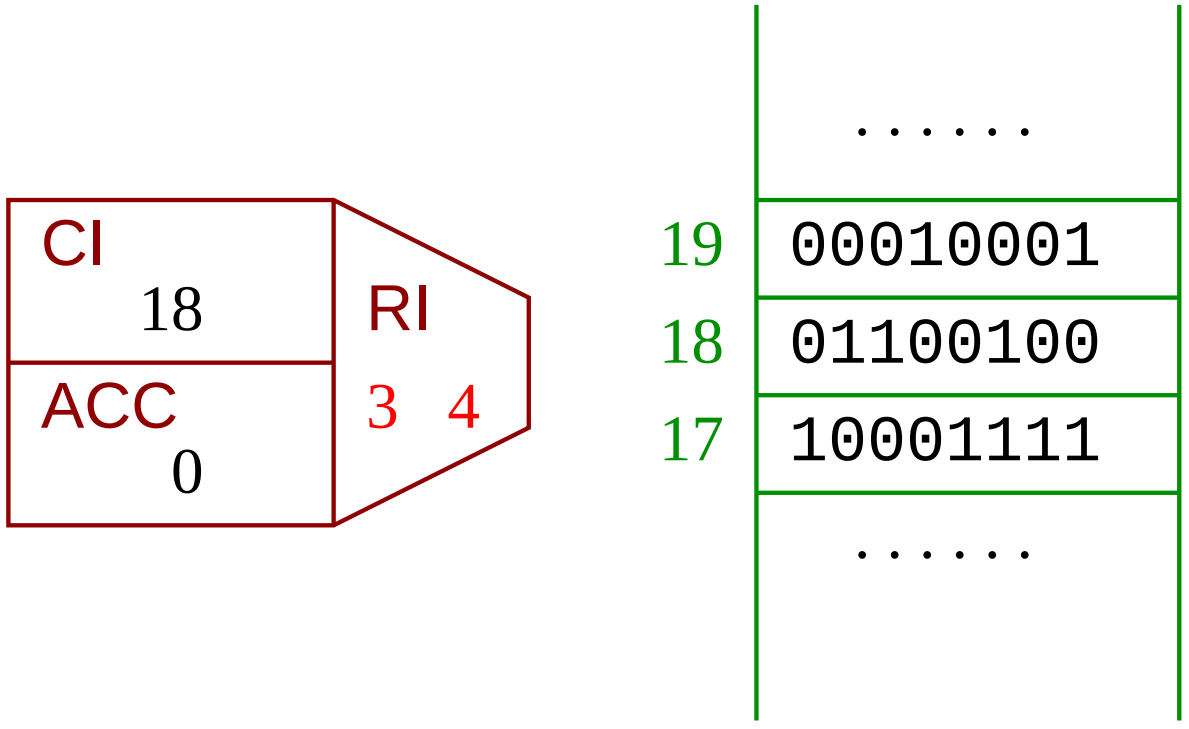
Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif



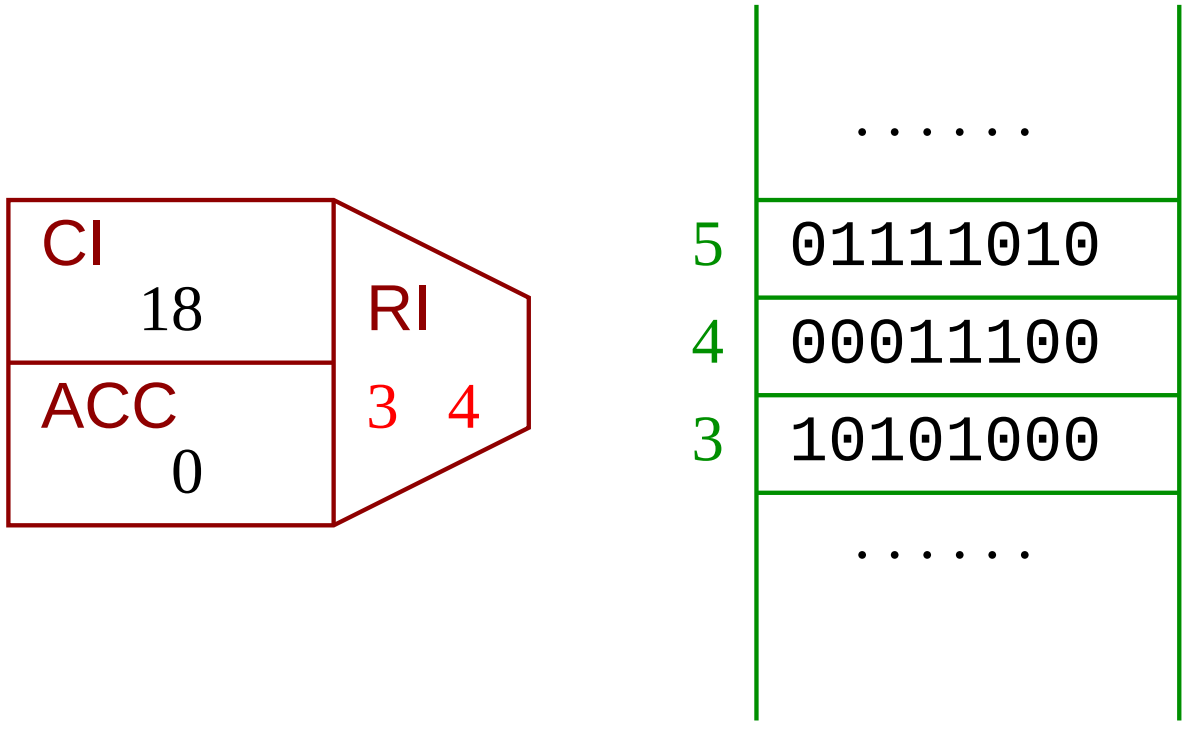
Le fonctionnement d'un processeur primitif



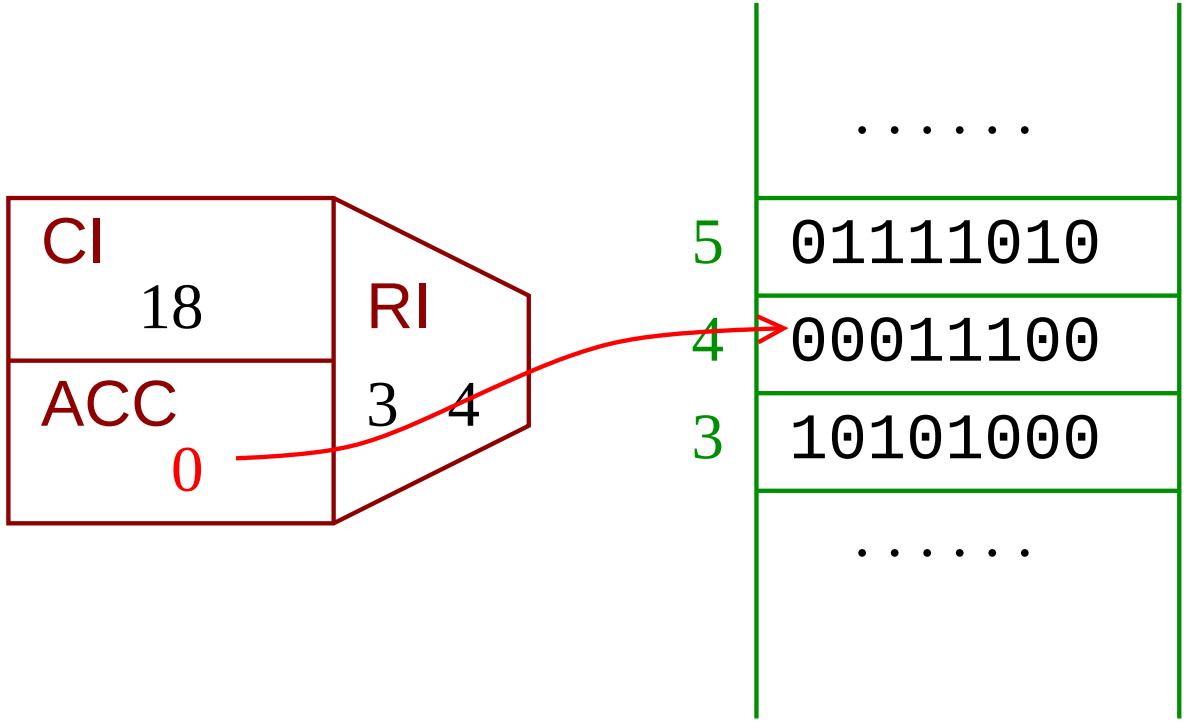
Opération 3:

écrire le contenu de l'accumulateur dans l'adresse indiquée

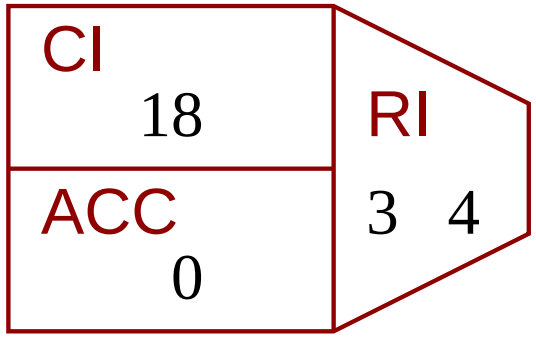
Le fonctionnement d'un processeur primitif

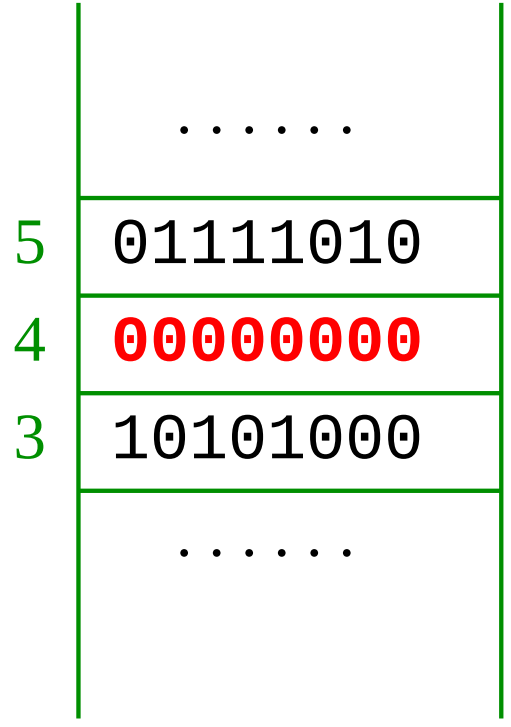


Le fonctionnement d'un processeur primitif

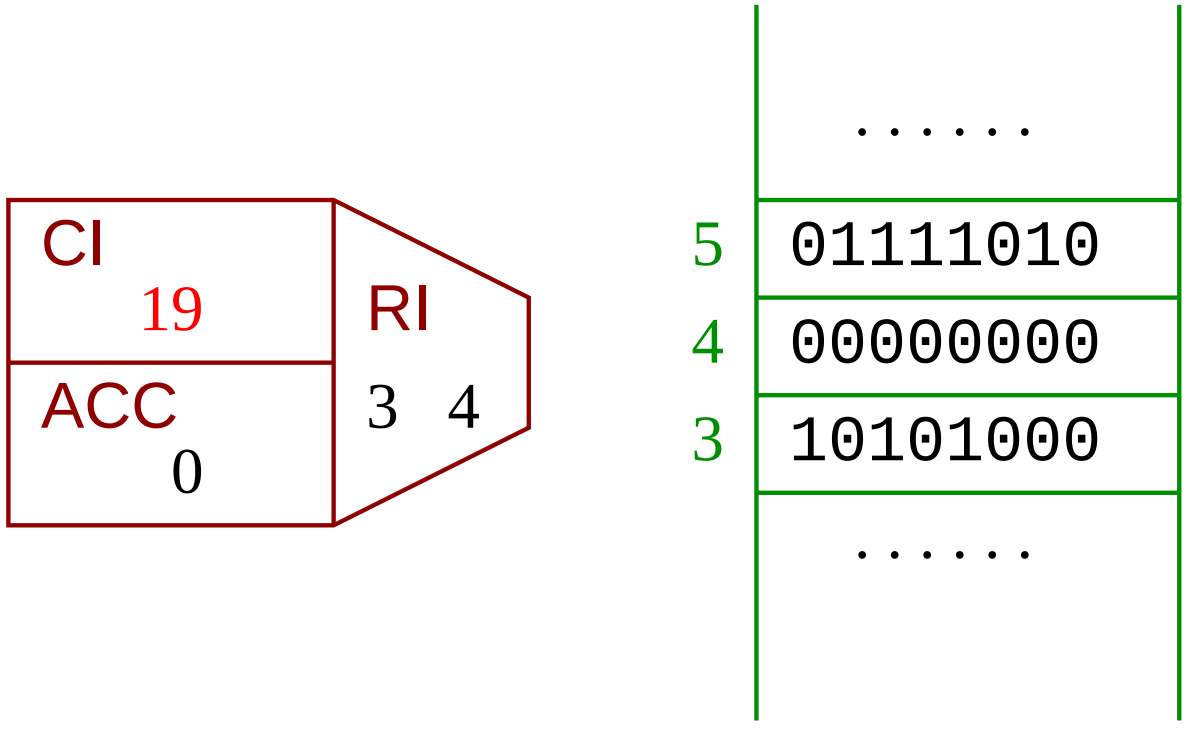


Le fonctionnement d'un processeur primitif

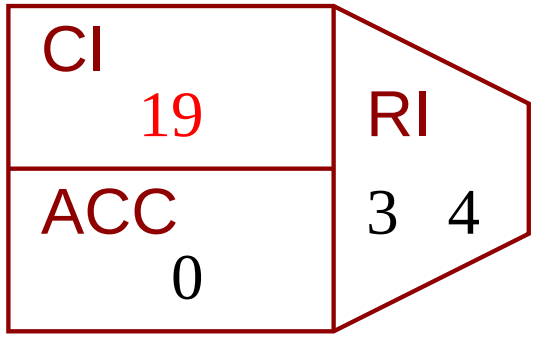


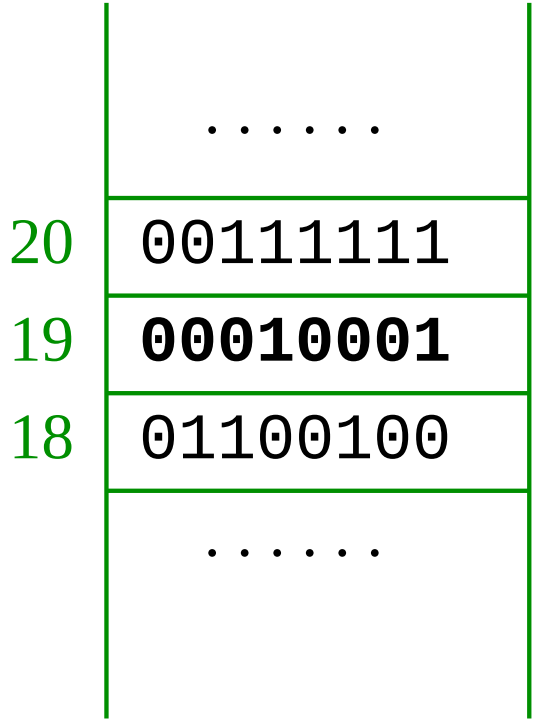


Le fonctionnement d'un processeur primitif

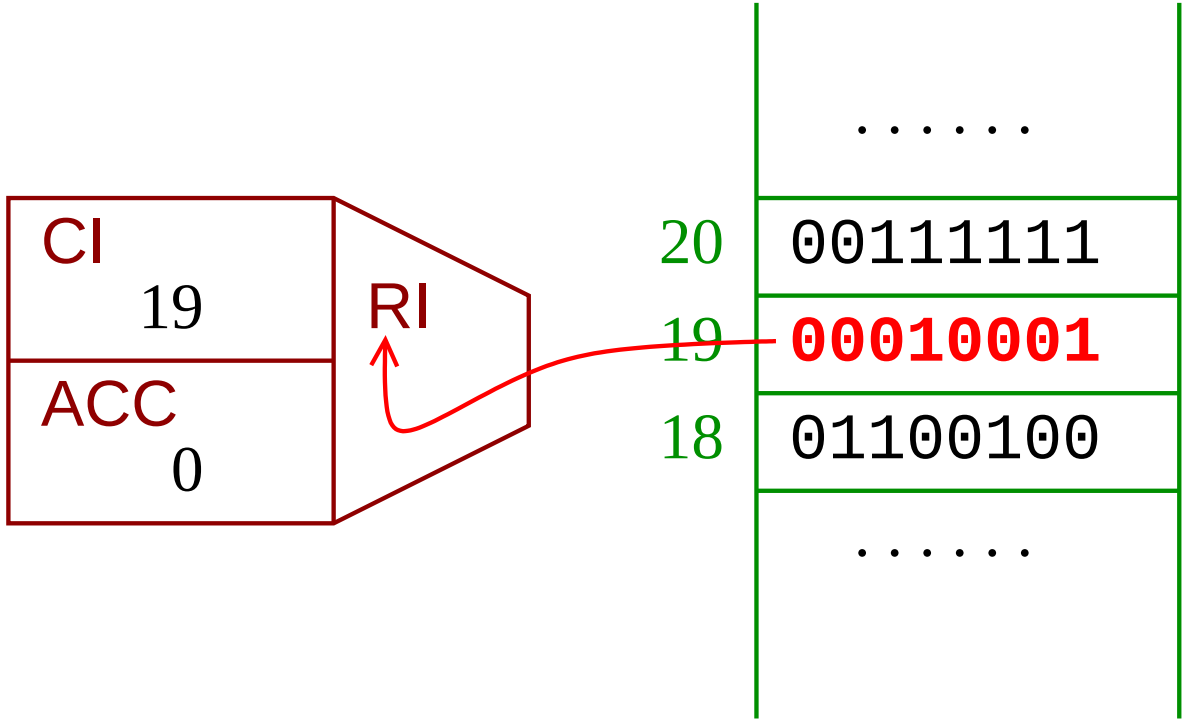


Le fonctionnement d'un processeur primitif

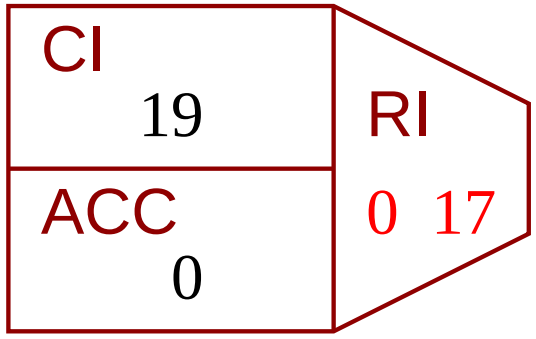


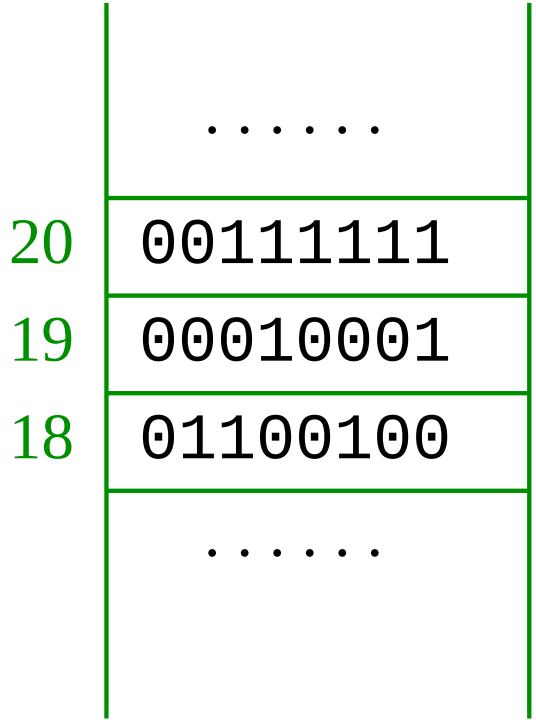


Le fonctionnement d'un processeur primitif

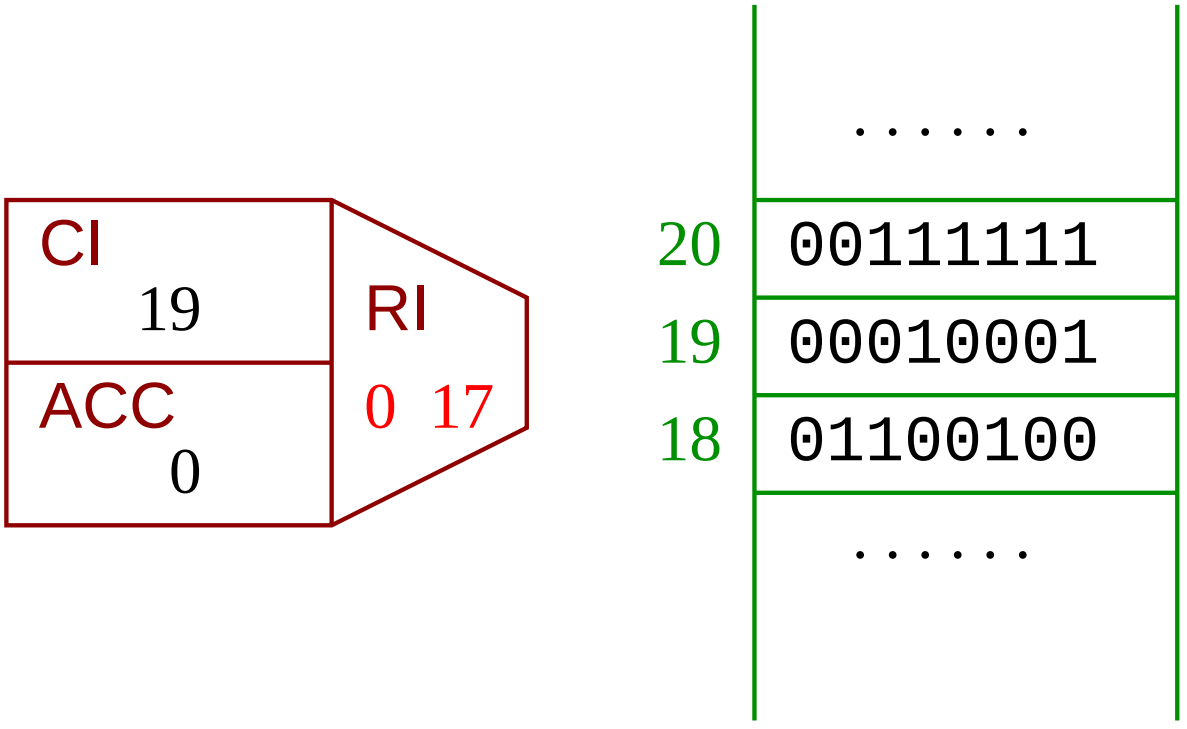


Le fonctionnement d'un processeur primitif





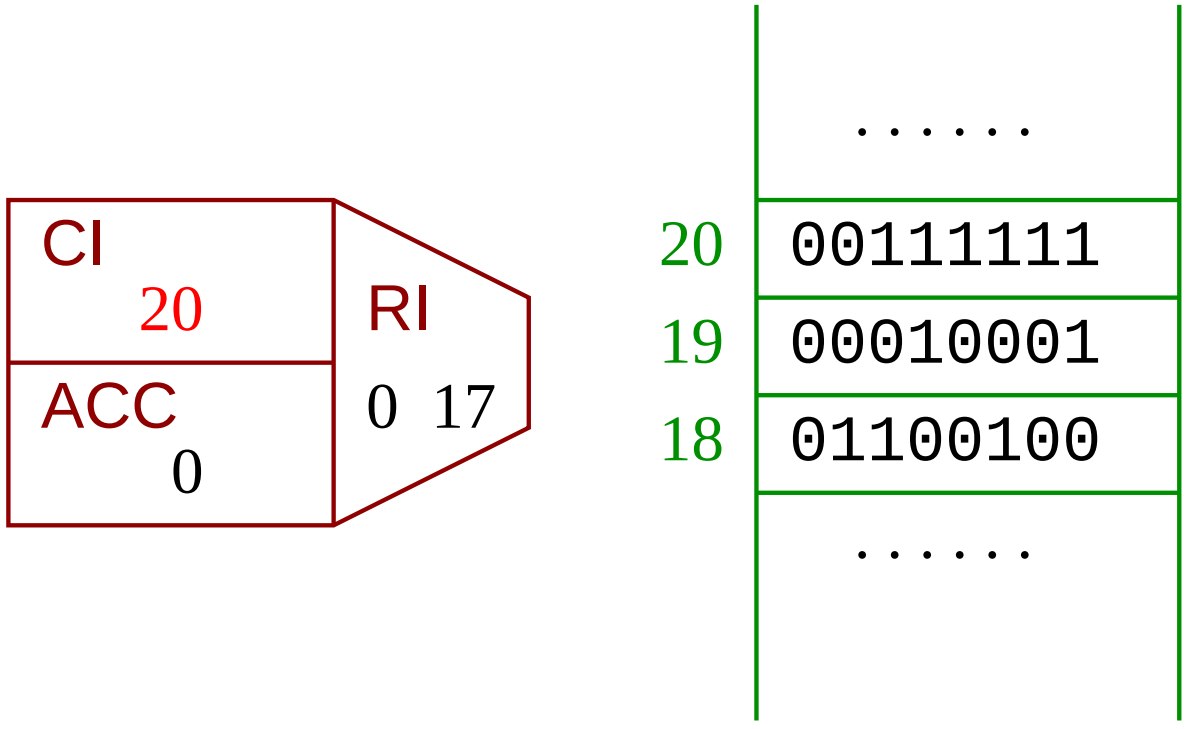
Le fonctionnement d'un processeur primitif



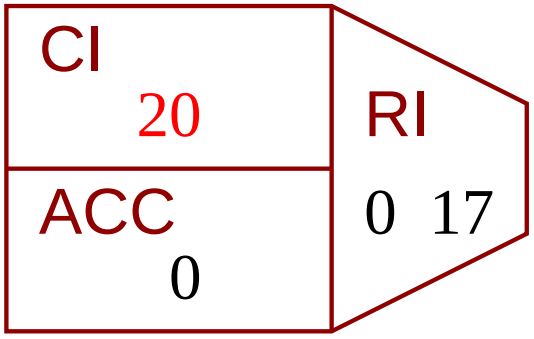
Opération 0:

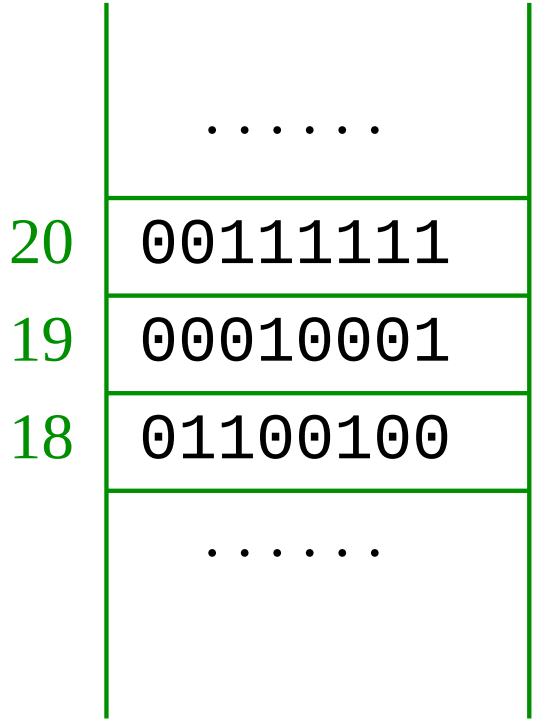
continuer avec l'instruction dans l'adresse indiquée
si le contenu de l'accumulateur ne vaut pas zéro

Le fonctionnement d'un processeur primitif



Le fonctionnement d'un processeur primitif





$\longrightarrow$ réalisation d'une boucle

Le fonctionnement d'un processeur primitif

Le programme exprimé dans un langage symbolique (assembleur):

16: LOAD *10

17: ADD *15

18: STORE *4

19: IF (ACC NOT 0) GOTO 17

20: ...

*10 = la valeur encodée à l'adresse 10

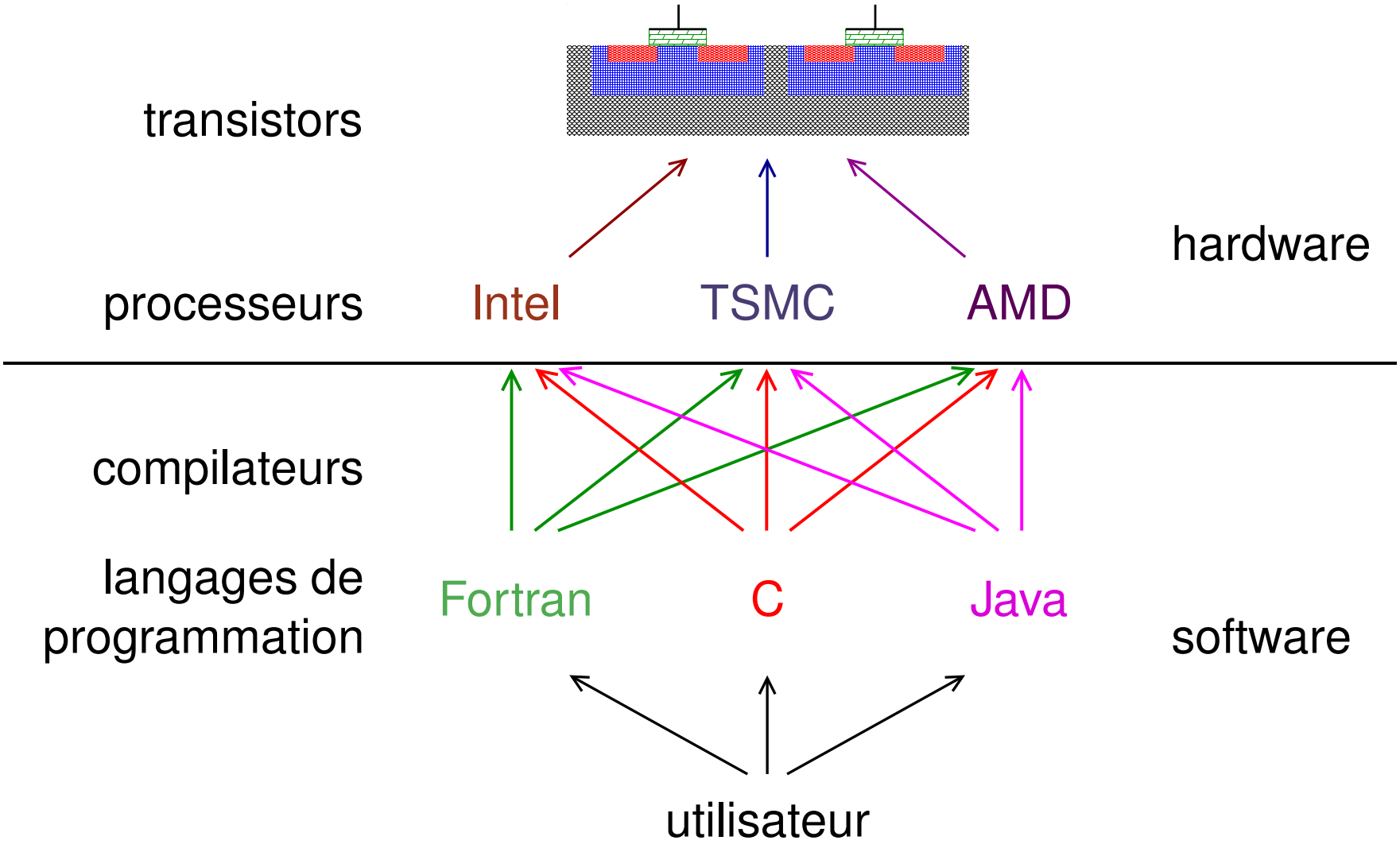
Le fonctionnement d'un processeur primitif

Le programme exprimé dans un langage symbolique (assembleur):

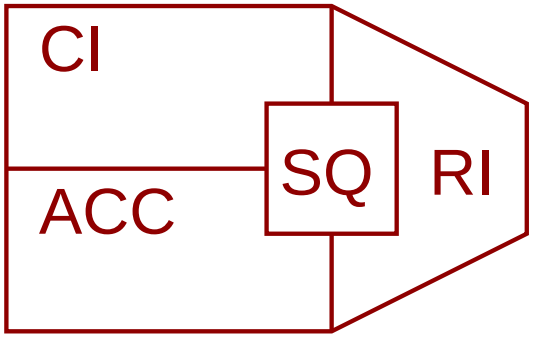
```
16:   LOAD *10
17:   ADD *15
18:   STORE *4
19:   IF (ACC NOT 0) GOTO 17
20:   ...
```

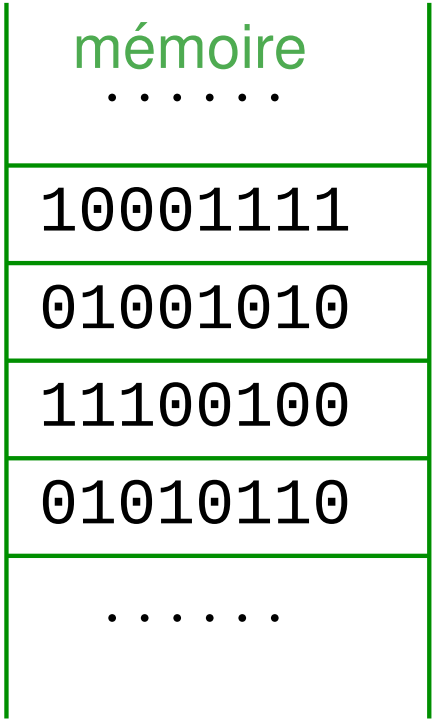
$\longrightarrow$ toute complexité de la fonctionnalité des ordinateurs peut être introduite par la programmation (software)

“hardware” et “software”



3.3 Réalisation en pratique





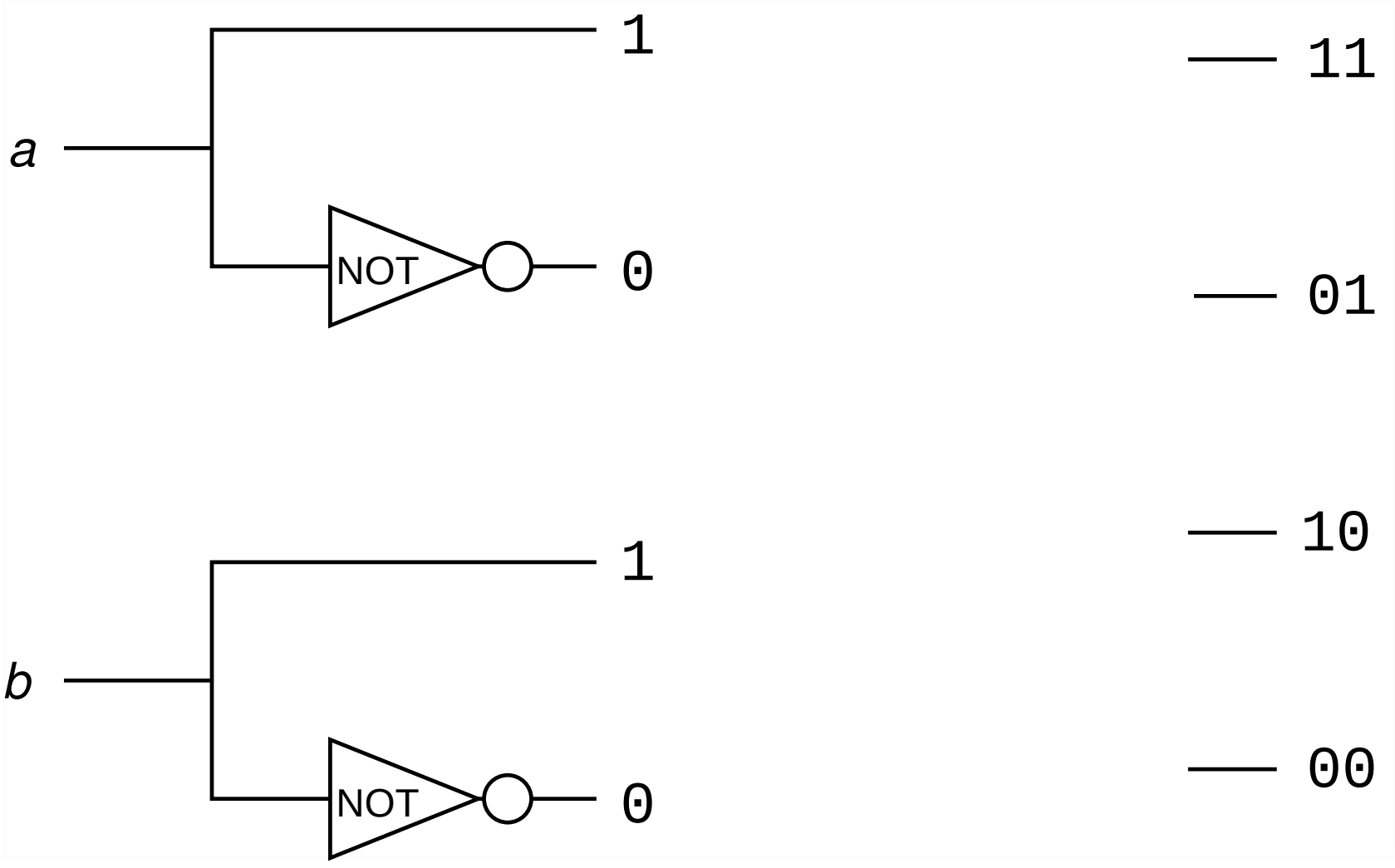
Installations supplémentaires:

$\longrightarrow$ séquenceur (cablé ou microprogrammé)

$\rightarrow$ contrôle le déroulement de toutes les actions

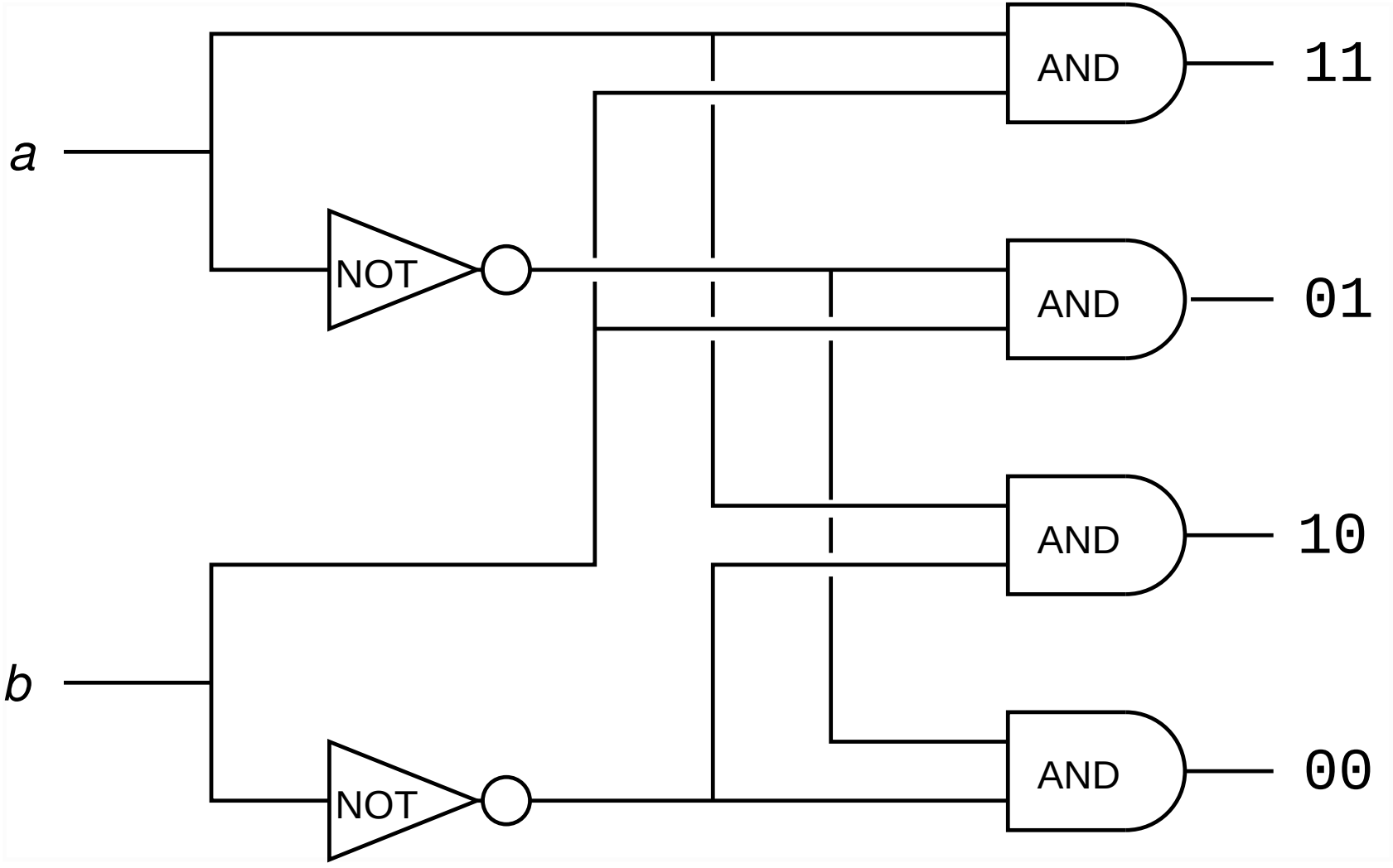
Comment décoder des codes d'instructions ?

. . . pour deux bits: *a b*

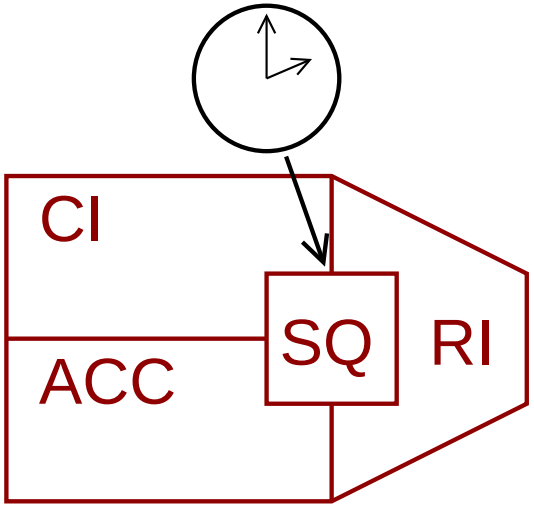


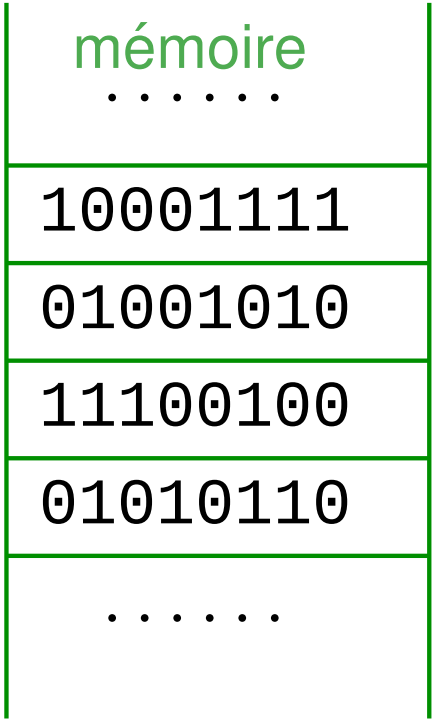
Comment décoder des codes d'instructions ?

. . . pour deux bits: *a b*



3.3 Réalisation en pratique





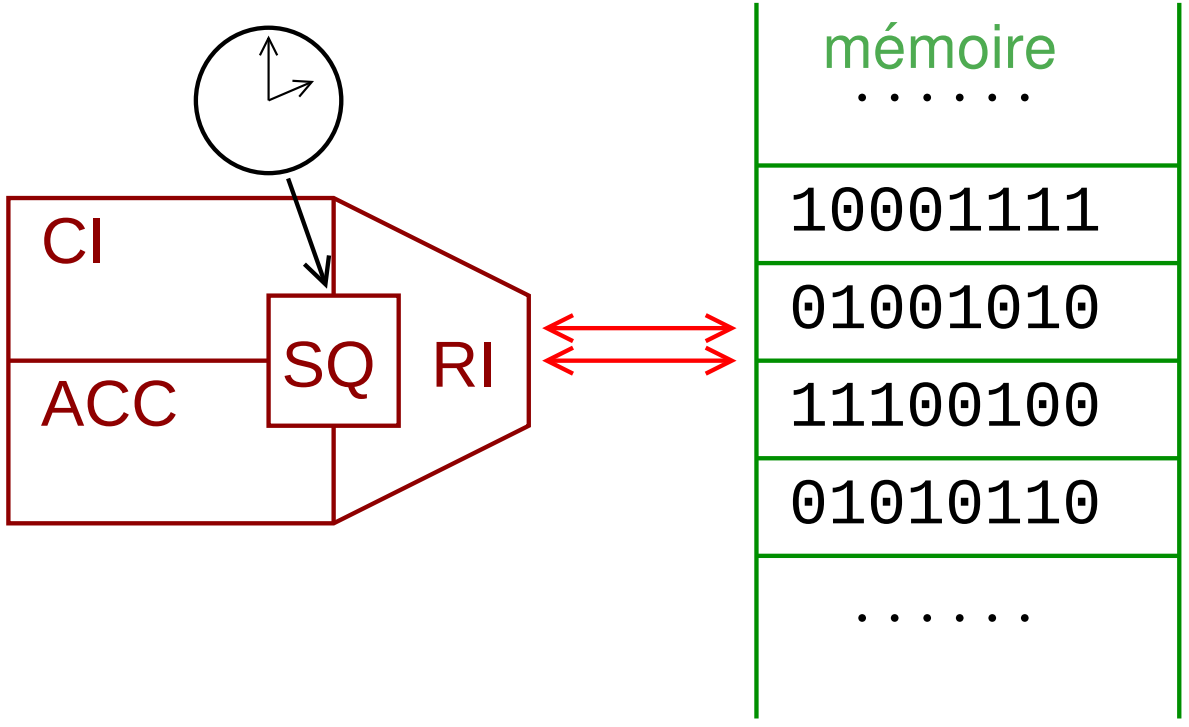
Installations supplémentaires:

$\longrightarrow$ séquenceur

$\longrightarrow$ horloge (oscillateur de quartz)

$\rightarrow$ donne des signaux de cadence réguliers (GHz) pour lancer des actions

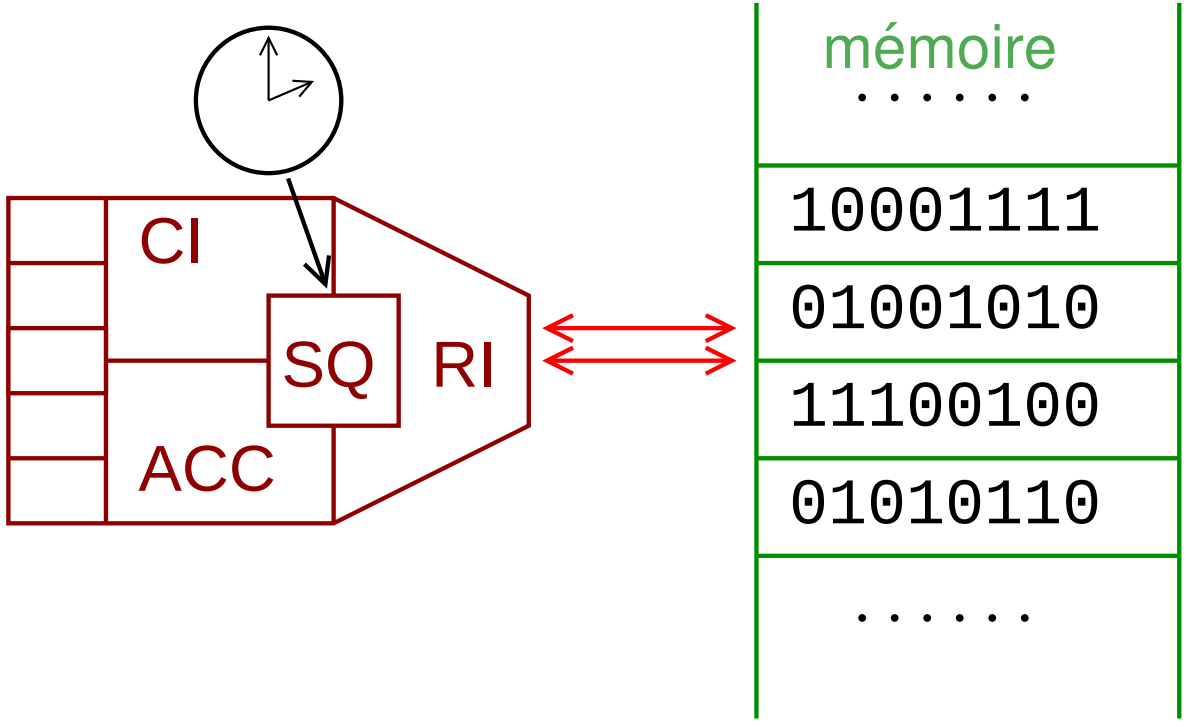
3.3 Réalisation en pratique



Installations supplémentaires:

$\longrightarrow$ séquenceur
$\longrightarrow$ horloge
$\longrightarrow$ “bus” de transfert entre mémoire et processeur
. . . pour des adresses et des données

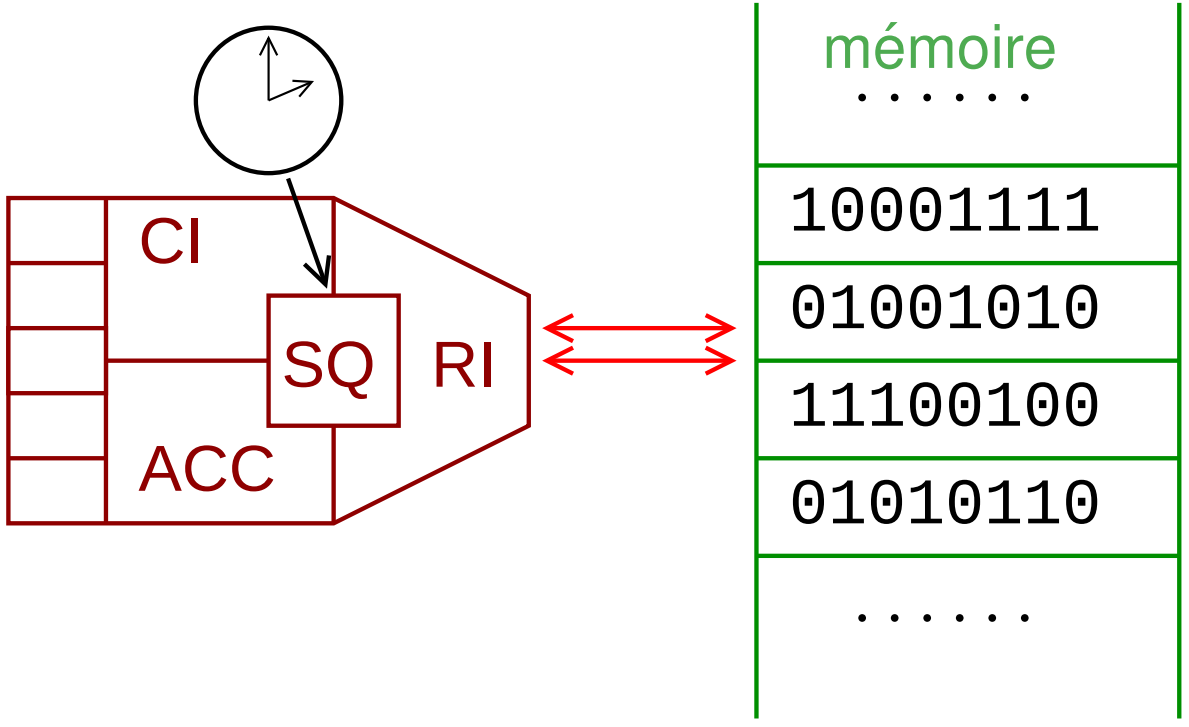
3.3 Réalisation en pratique



Installations supplémentaires:

$\longrightarrow$ séquenceur
$\longrightarrow$ horloge
$\longrightarrow$ “bus” de transfert
$\longrightarrow$ registres supplémentaires (SRAM)

3.3 Réalisation en pratique

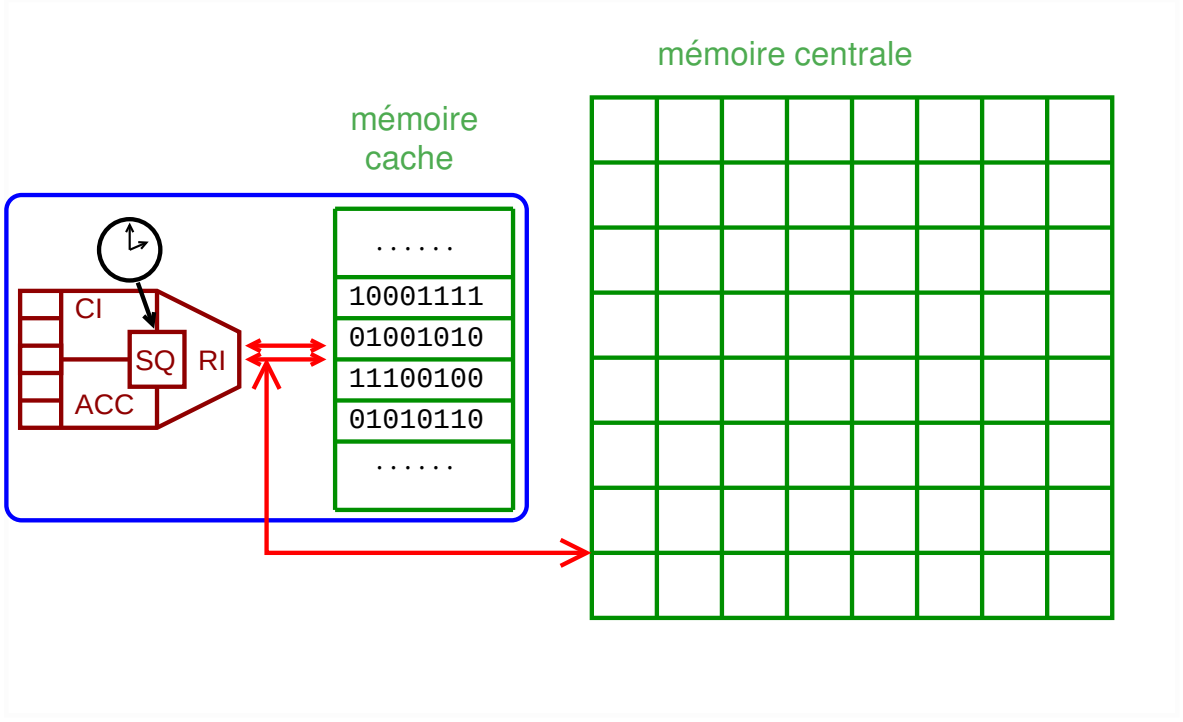


Optimisations:

$\longrightarrow$ execution parallèle des opérations indépendantes

(p.ex. copie des données et incrément du CI)

3.3 Réalisation en pratique



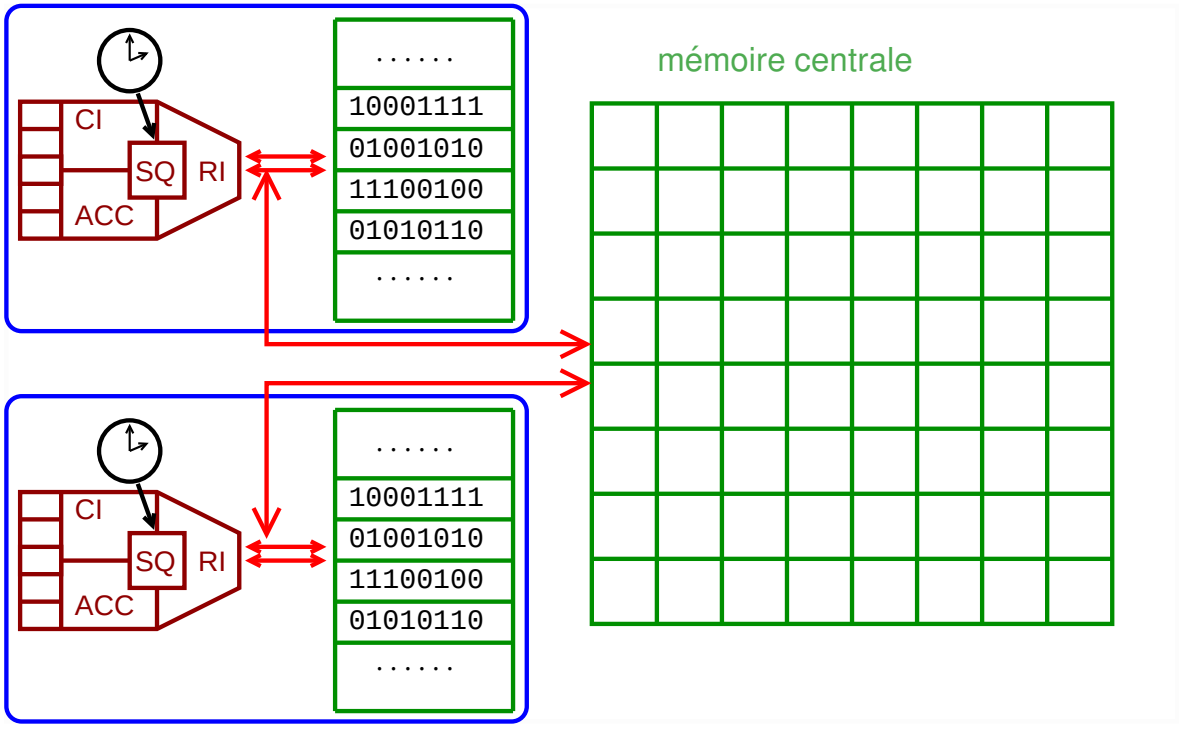
Optimisations:

$\longrightarrow$ execution parallèle des opérations indépendantes

$\longrightarrow$ mémoire “cache” et mémoire centrale

(réalisation avec SRAM et DRAM)

3.3 Réalisation en pratique



Optimisations:

$\longrightarrow$ execution parallèle des opérations indépendantes
$\longrightarrow$ mémoire “cache” et mémoire centrale
$\longrightarrow$ opération parallèle de plusieurs processeurs
$\Rightarrow$ architecture “massivement parallèle” (p.ex. 256 processeurs)