AMELIORATION DES PERFORMANCES DU PROCESSEUR A CHEMIN DE DONNEES A CYCLE UNIQUE PAR LA TECHNIQUE DU PIPELINE

INTRODUCTION (1/6)

- ***Le pipelining* : technique permettant à plusieurs instructions de se chevaucher pendant l'exécution.**
- **Un *pipeline d'instructions* est analogue à une *chaîne de montage de véhicule* : à chaque *étage (poste)* du *pipeline (chaine)* est achevée une partie de *l'exécution d'une instruction (construction d'un véhicule)*.**
- **Les *unités (ouvriers)* effectuent de petites tâches, telle que *lecture de l'instruction (installation des sièges)*.**
- **La puissance du *pipeline (chaîne)* réside dans le fait qu'un *grand nombre de petites tâches sont effectuées collectivement* pour produire finalement un grand nombre *d'instructions/seconde (véhicules/jour)*.**
- **Un *pipeline (chaîne)* est bien équilibré(e), si une nouvelle *instruction (véhicule)* sort du *pipeline (chaîne)* dans le temps qu'il faut pour effectuer l'une des nombreuses *petites tâches*.**
- **Le *pipeline (chaîne)* *ne réduit pas le temps nécessaire pour exécuter (construire)* chaque *instruction (véhicule)* ; *Il (elle) accroît le nombre d'instructions exécutées (véhicules construits) simultanément* ➔ accroissement de la *cadence*.**
- **Le travail pour exécuter une instruction donnée est découpé en petits morceaux. Chaque morceau prend une fraction du temps nécessaire pour effectuer l'instruction complète. *Pipeline asynchrone ou synchrone ?***
- **Les étages du pipeline sont juxtaposées dans l'ordre d'exécution : les instructions entrent d'un côté du pipeline suivent les différentes étages et ressortent à l'autre extrémité *en évitant les retours dans les étages (feedback)*.**

La technique du pipeline améliore le débit des instructions plutôt que le temps d'exécution de chaque instruction.

INTRODUCTION (2/6)

- **Le débit d'un pipeline d'instructions est donné par la fréquence avec laquelle une instruction sort du pipeline.**
- **Dans le cas idéal, le temps nécessaire à une instruction pour avancer d'un étage est de un cycle d'horloge.**
- **La durée du cycle d'horloge est définie par le temps pris par l'étage le plus lent : tous les étages doivent fonctionner à la même cadence.**

Lors de la conception d'un pipeline il faut chercher à équilibrer la durée des différents étages sinon, du temps serait perdu au cours de chacun d'eux.

Le temps qui sépare 2 instructions dans un pipeline parfaitement équilibrée (cas idéal) est égal à :

$$Temps\_entre\_instructions\_pipelinées = \frac{Temps\_entre\_instructions\_non\_pipelinées}{Nombre\_d'étages\_du\_pipeline}$$

- ***Dans le cas idéal :*** ***l'accélération (speed up) est égale au nombre d'étages du pipeline ;***
- ***Dans le cas réel :***
  - ***les étages ne sont généralement pas parfaitement équilibrés.***
  - ***De plus, la technique du pipeline engendre des délais supplémentaires.***

INTRODUCTION (3/6)

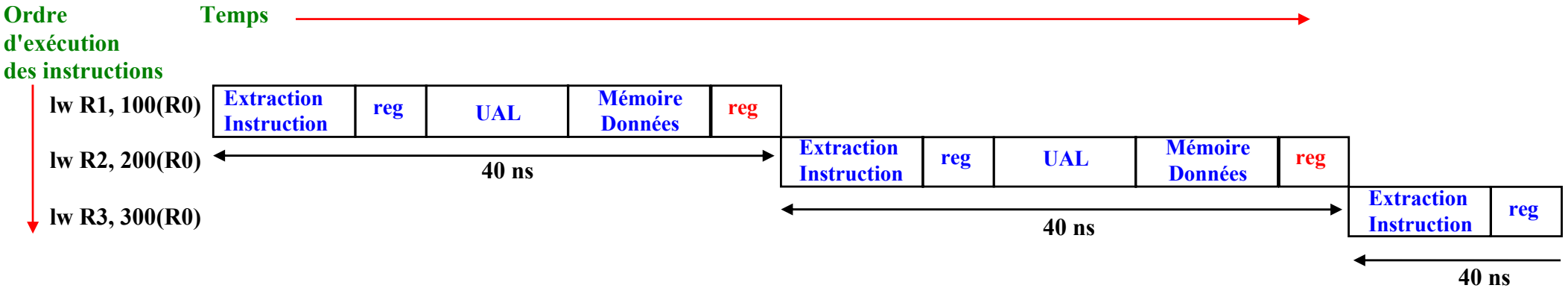
Construisons un pipeline à partir du même noyau d'instructions : (*lw, sw*), (*add, sub, and, et or*), (*beq*) et *(j.)*

Exemple : **supposons que le temps d'opération pour les principales unités fonctionnelles vaut :**

- **Unités mémoire : 10 ns,**
- **UAL et additionneurs : 10 ns,**
- **Banque de registres (lecture ou écriture) : 5 ns.**
- **MUX, unité de contrôle, unité d'extension de signe, accès au CP, et les bus ne génèrent pas de délais,**

Type ou classe d'instructions	Mémoire d'instructions	Lecture de registre	Opération UAL	Mémoire de données	Ecriture de registre	Temps total
Chargement d'un mot (Iw)	10 ns	5 ns	10 ns	10 ns	5 ns	40 ns
Rangement d'un mot (sw)	10 ns	5 ns	10 ns	10 ns		35 ns
Format R (add, sub, and, or…)	10 ns	5 ns	10 ns		5 ns	30 ns
Branchement (beq)	10 ns	5 ns	10 ns			25 ns

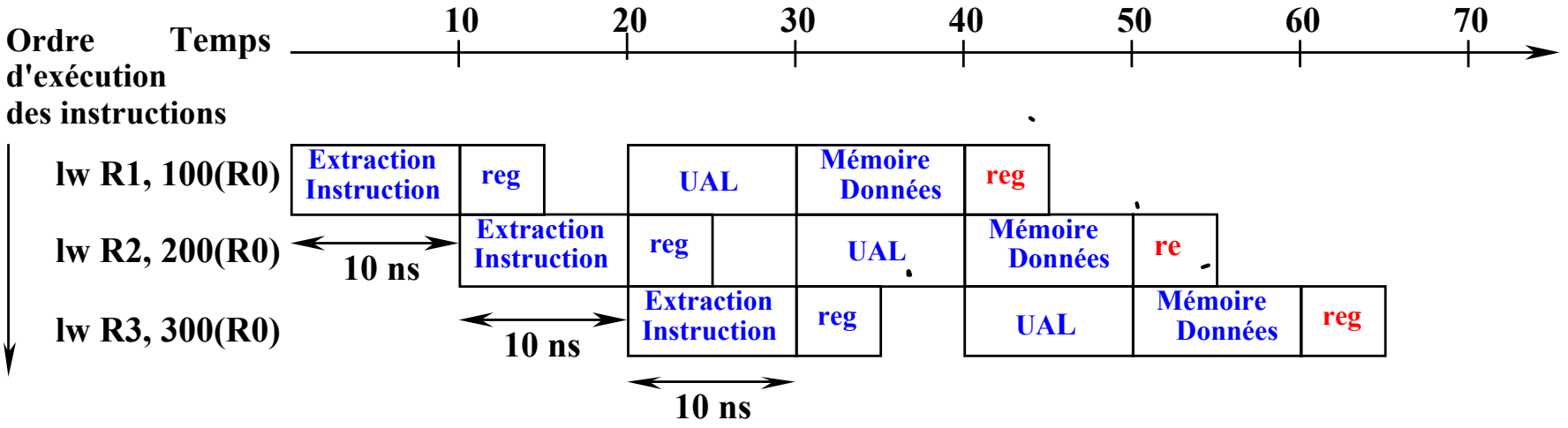
Dans le modèle à cycle unique, le temps nécessaire pour toute instruction est de 40 ns.



Le temps qui s'écoule entre la 1ère et la 4ème instruction est de 120 ns.

INTRODUCTION (3/6)

- Les instructions nécessitent 5 étapes ➔ un pipeline à 5 étages est un bon point de départ.
- 5 étages ➔ 5 instructions simultanément, une dans chaque étage du pipeline.
- Chaque étage prend un cycle d'horloge, celui-ci doit être choisi pour traiter l'étape la plus lente.
- la séquence d'instructions de chargement dans la mise en œuvre pipelinée serait la suivante :



Ce pipeline à 5 étages devrait conduire a une accélération d'un facteur de 5 ➔ un cycle d'horloge = 8 ns

Or l'étage le plus lent nécessite 10 ns ➔ l'accélération réelle n'est que d'un facteur 4, la différence est due au déséquilibre entre les étages du pipeline.

INTRODUCTION (4/4)

- **Bien que le pipeline ait une accélération d'un facteur 4, le gain en terme de temps total d'exécution pour les 3 instructions est plus modeste : 70 ns au lieu de 120 ns.**

- **Etendons la séquence à 1003 instructions en rajoutant 1000 instructions à gauche du pipeline. le temps total d'exécution = 1000 x 10 ns + 70 ns = 10070 ns.**

- **Dans l'exemple non pipeliné, le temps total d'exécution = 1000 x 40 ns + 120 ns = 40120 ns.**

- **Le rapport des temps d'exécution pour des programmes réels entre machines non pipelinées et pipelinées serait proche du rapport des temps entre instructions :**

$$\frac{40120ns}{10070ns} = 3{,}98 \approx \frac{40ns}{10ns}$$

- **La technique du pipeline améliore les performances en *augmentant le débit de sortie des instructions, plutôt qu'en diminuant le temps d'exécution d'une instruction individuelle*.**

- **Le *débit* des instructions est la *métrique* qu'il faut considérer, car les programmes réels exécutent des milliards d'instructions.**

Le pipelining exploite le parallélisme entre instructions d'un flot séquentiel d'instructions. Il présente l'avantage substantiel d'être, contrairement à d'autres techniques d'accélération, transparent au programmeur.

CHEMIN DE DONNEES PIPELINE (1/5)

- **La décomposition d'une instruction en 5 étapes conduit à un pipeline à 5 étages.**

- **Le chemin de données à cycle unique sera découpé en 5 étages, qui sont baptisés en fonction de l'étape de l'exécution de l'instruction :**

1. ***EI :** Extraction d'instruction (lecture de l'instruction) ;*
2. ***DI :** Décodage d'instruction et extraction des registres ;*
3. ***EX** : Exécution et calcul d'adresse effective ;*
4. ***MEM :** Accès mémoire ;*
5. ***ER** : Ecriture du Résultat.*

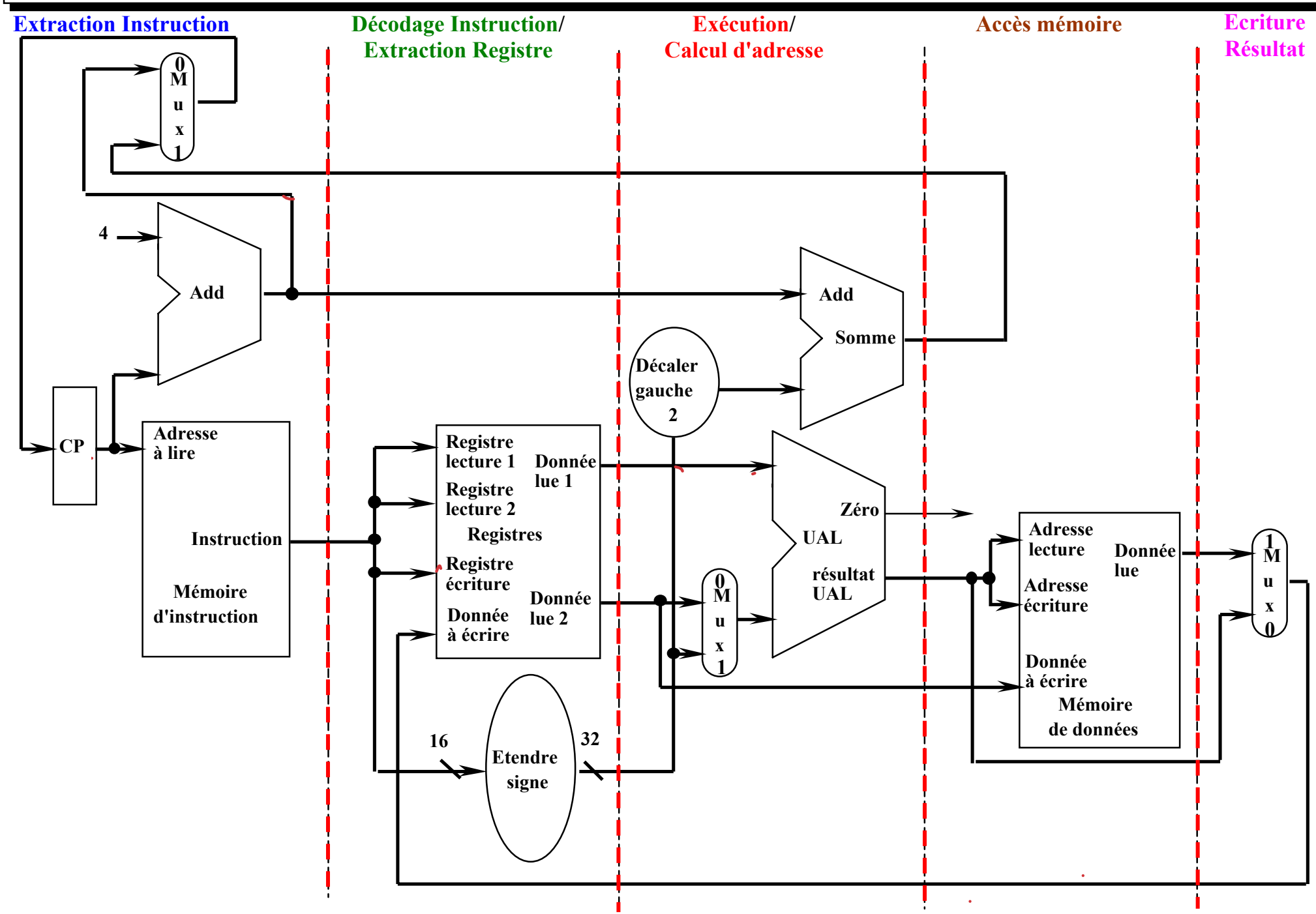
- **Les instructions et données traversent généralement de gauche à droite les 5 étages durant leur exécution.**

- **Il y a 2 exceptions à ce flot de gauche à droite :**

  - **L'étape Ecriture du Résultat, place le résultat dans le banc de registres situé au milieu du chemin de données.**
  - **La prochaine valeur du CP, est choisie entre le CP incrémenté et l'adresse de branchement à l'étape MEM.**

Le mouvement de données *de droite à gauche n'influe pas sur l'instruction courante* par contre *les instructions ultérieures du pipeline peuvent être influencées*.

CHEMIN DE DONNEES PIPELINE (2/5)

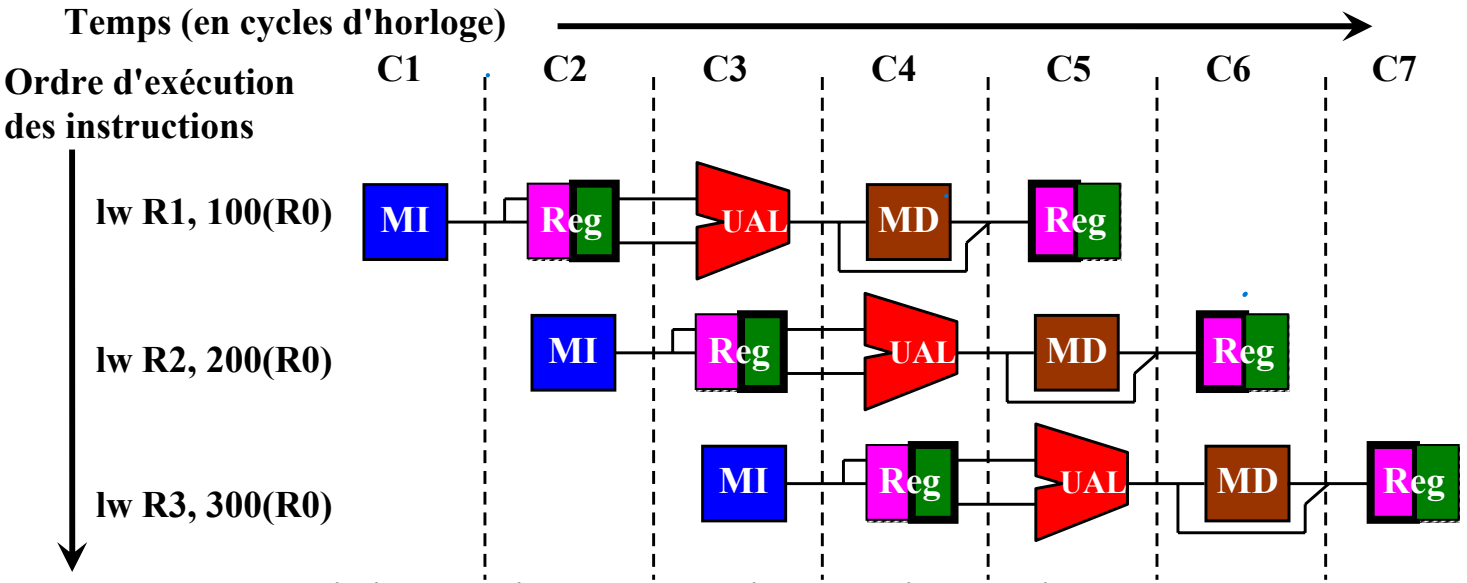


CHEMIN DE DONNEES PIPELINE (3/5)

Comment représenter une exécution pipelinée ?

- √ simuler un chemin de données particulier pour chaque instruction,
- √ placer ces chemins sur une échelle de temps pour montrer leurs relations temporelles/dépendances.

- L'exécution d'un flot d'instructions en représentant les chemins individuels sur une échelle de temps commune est la suivante :



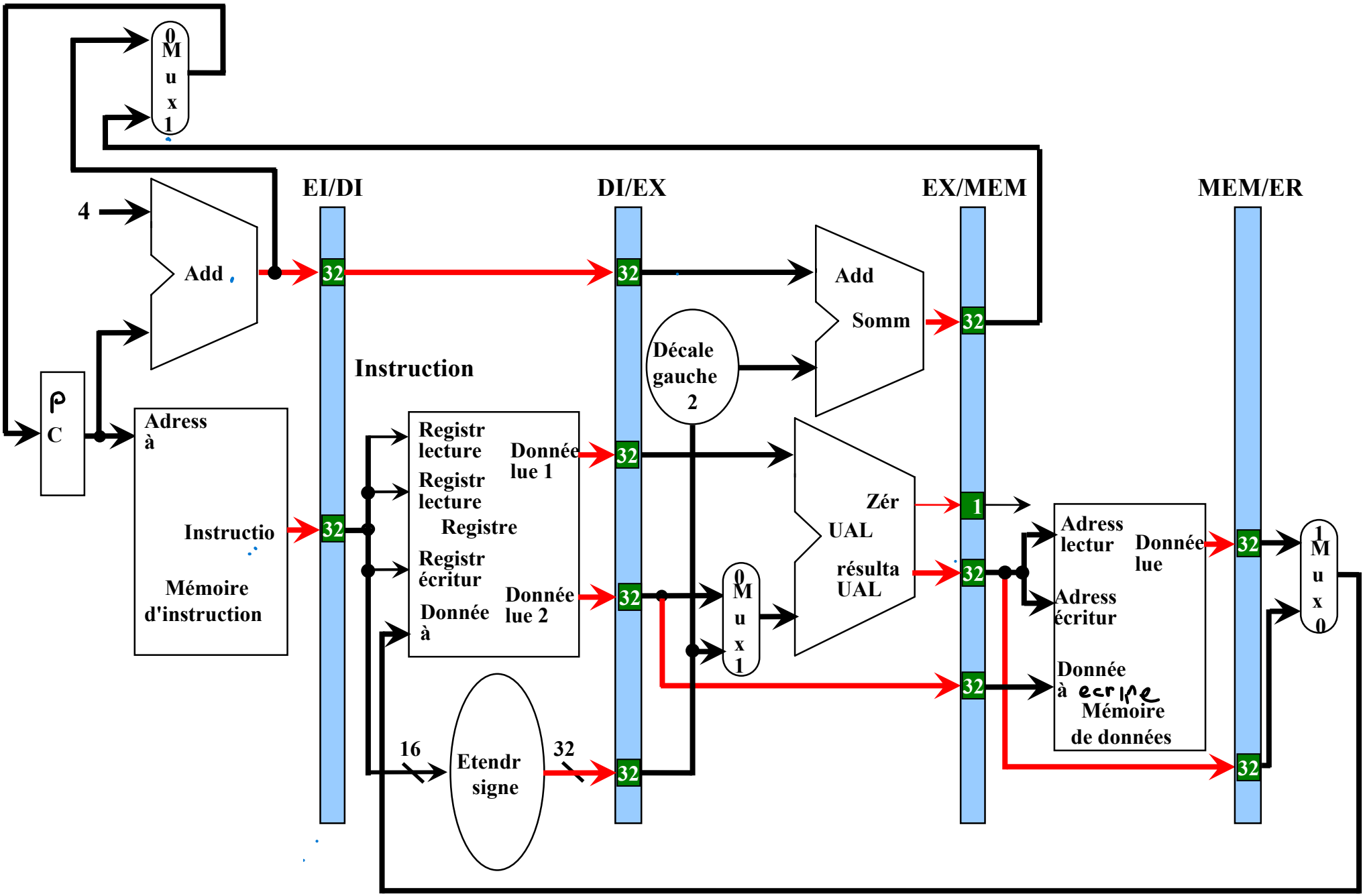
- Ceci semble suggérer que trois instructions ont besoin de trois chemins de données.

- Dans le chemin à seul cycle, une instruction utilise toutes les unités durant toute la durée de son exécution.

- Dans le chemin pipeliné, une instruction utilise une unité durant le 1/5ème de la durée totale de son exécution.

- Pour que l'instruction continue son exécution dans les autres unités (4 étages), sa valeur lue en mémoire d'instructions doit être stockée dans un registre.

CHEMIN DE DONNEES PIPELINE (4/5)



CHEMIN DE DONNEES PIPELINE (5/5)

- **Toute instruction avance à chaque cycle d'horloge d'un registre pipeline au suivant. Elle peut subir ou non un traitement dans l'étage suivant.**

- **La séparation entre 2 étages doit être réalisée par un registre pipeline qui sert d'élément d'état.**

- ***Exception :*** **aucun registre pipeline ne sépare l'étage d'Ecriture du Résultat (ER) de celui de l'Extraction (la prochaine) Instruction (EI).**

  - **L'étage Ecriture Résultat (ER) met à jour l'état de la machine en écrivant dans la banque de registres qui est un élément d'état ➔ ajouter un registre pipeline séparé pour stocker l'état serait redondant.**

  - **Une instruction de chargement par exemple, placera son résultat dans l'un des 32 registres, et toute instruction ultérieure ayant besoin de cette donnée lira simplement le registre approprié.**

- **Nous allons ignorer pour l'instant ce qui se passe lorsqu'il existe des dépendances entre instructions dans une exécution pipelinée !!!**

FONCTIONNEMENT DU CHEMIN DE DONNEES PIPELINE (1/14)

Expliquer le fonctionnement du pipeline revient à :

- **Observer l'acheminement d'une instruction, lors de son exécution, à travers les différents étages.**
- **Mettre en valeur les éléments actifs du chemin de données lors du passage par chaque étage.**

Nous considérons une instruction de chargement parce qu'elle est active durant les cinq étages. Nous mettons en valeur la *moitié droite* des registres ou de la mémoire lorsqu'ils sont *lus* et la *moitié gauche* lorsqu'ils sont *écrits*.

1. Extraction d'Instruction :

- **A partir de l'adresse dans CP :**

Début du cycle en parallèle :

√ **Lecture de l'instruction en mémoire d'instruction.**

√ **Calcul de la nouvelle adresse CP + 4, puis *chargement dans CP* pour être prêt pour le prochain cycle.**

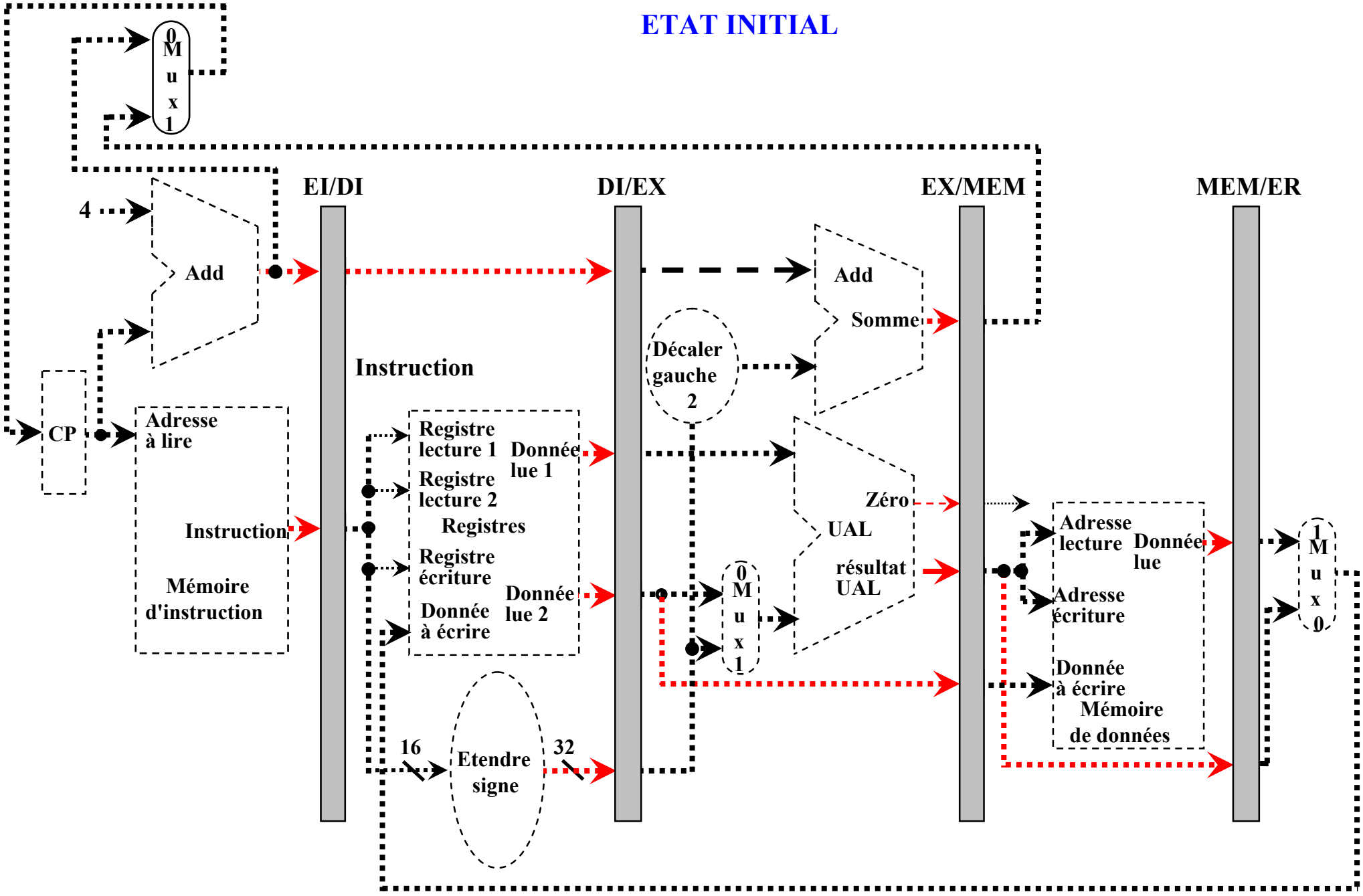
- **A la fin du cycle stockage dans le registre pipeline EI/DI.**

√ **de l'instruction Ins[31-0] (EI/DI est semblable au registre Instruction).**

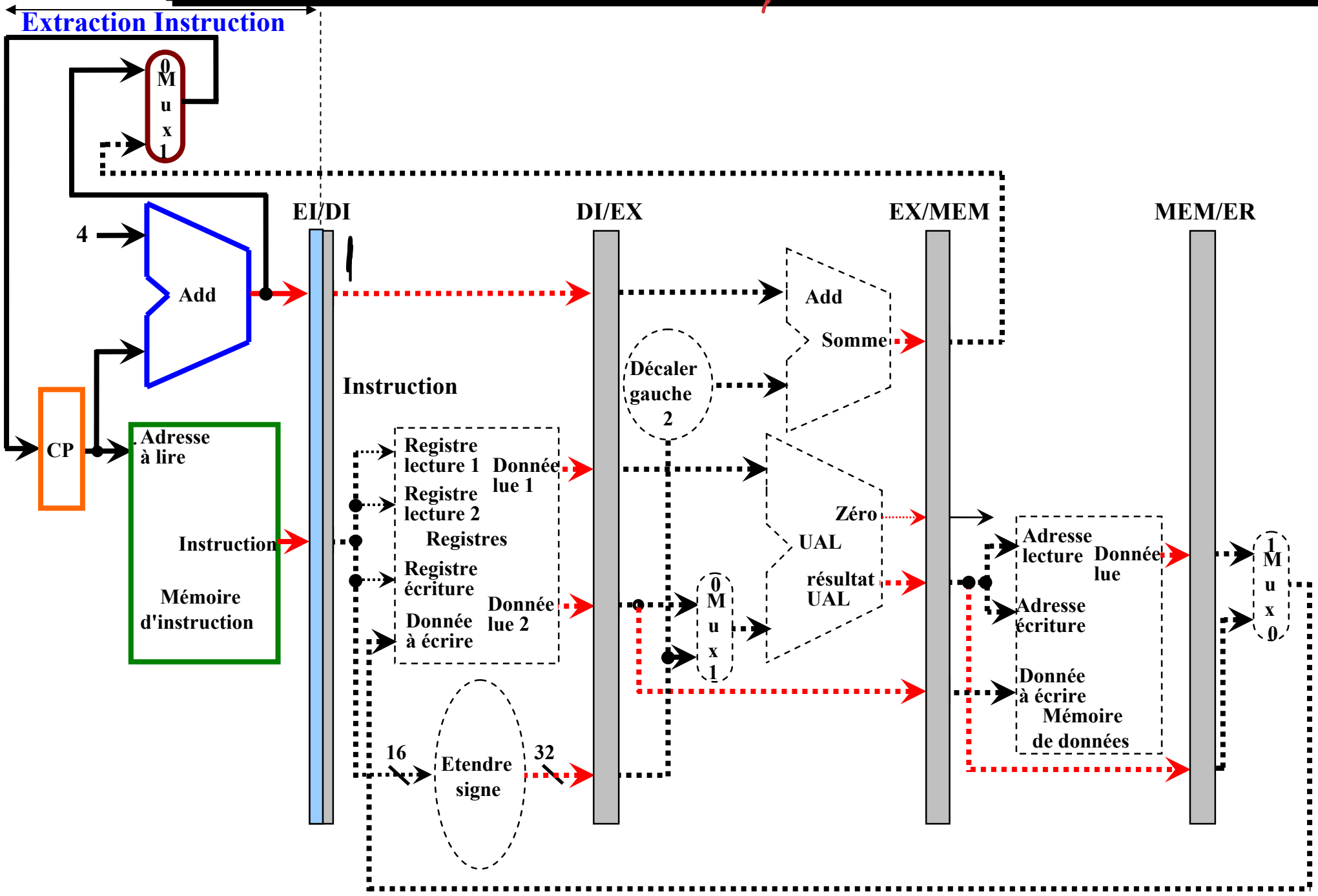
√ **de la nouvelle adresse au cas où elle serait utile à une instruction ultérieure, telle que beq.**

A ce stade on ne sait pas quel est le type de l'instruction extraite, il faut donc anticiper certaines opérations pour réduire le délai d'exécution et donc être prêt à traiter toute instruction.

FONCTIONNEMENT DU CHEMEIN DE DONNEES PIPELINE (2/15)



FONCTIONNEMENT DU CHEMIN DE DONNEES PIPELINE (3/15)



FONCTIONNEMENT DU CHEMIN DE DONNEES PIPELINE (4/15)

***2. Décodage d'Instruction et Lecture de Registre* :**

- **A partir du champ instruction Ins[31-0] en sortie du registre pipeline EI/DI :**

  Début du cycle en parallèle :

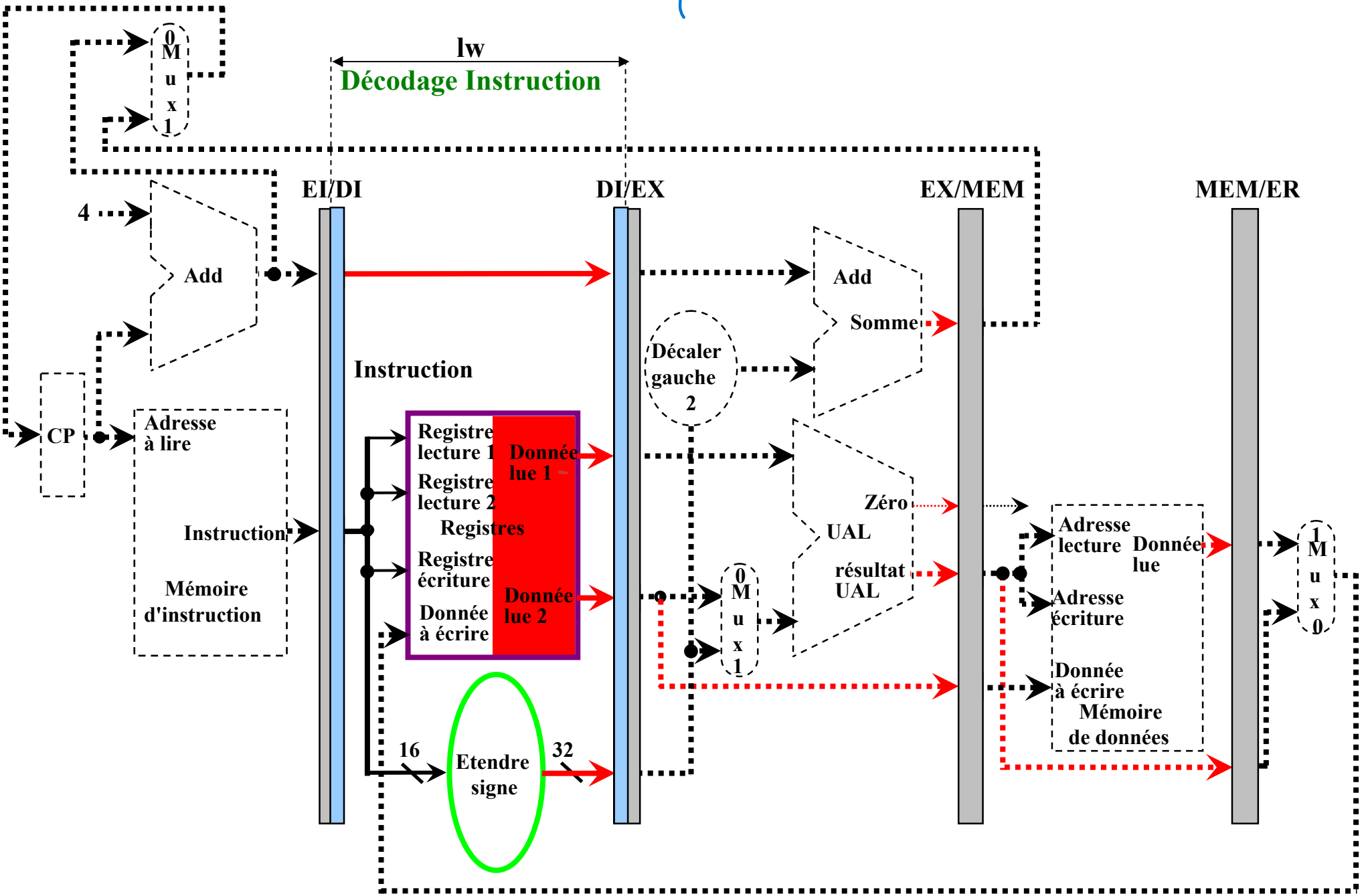
  √ Extension du signe à partir du champ immédiat (Ins[15-0]) ;
  √ Lecture registre 1 à partir du champ rs (Ins[25-21]) ; (c'est le registre de base pour lw)
  √ Lecture registre 2 à partir du champ rt (Ins[20-16]).

- **A la fin du cycle stockage dans le registre pipeline DI/EX :**
  √ de la valeur du registre1
  √ de la valeur du déplacement étendu
  √ de la valeur CP+4 calculée dans l'étage EI

Durant ce cycle d'horloge l'identité de l'instruction est déterminée ➔ on peut ne stocker que ce qui sera utile dans des cycles ultérieurs.

Mais tout sauvegarder ne coûte pas plus cher et simplifie le contrôle ➔ Nous transférons donc à nouveau tout ce qui pourrait être utile à une instruction pendant un cycle ultérieur.

FONCTIONNEMENT DU CHEMIN DE DONNEES PIPELINE (5/15)



FONCTIONNEMENT DU CHEMIN DE DONNEES PIPELINE (6/15)

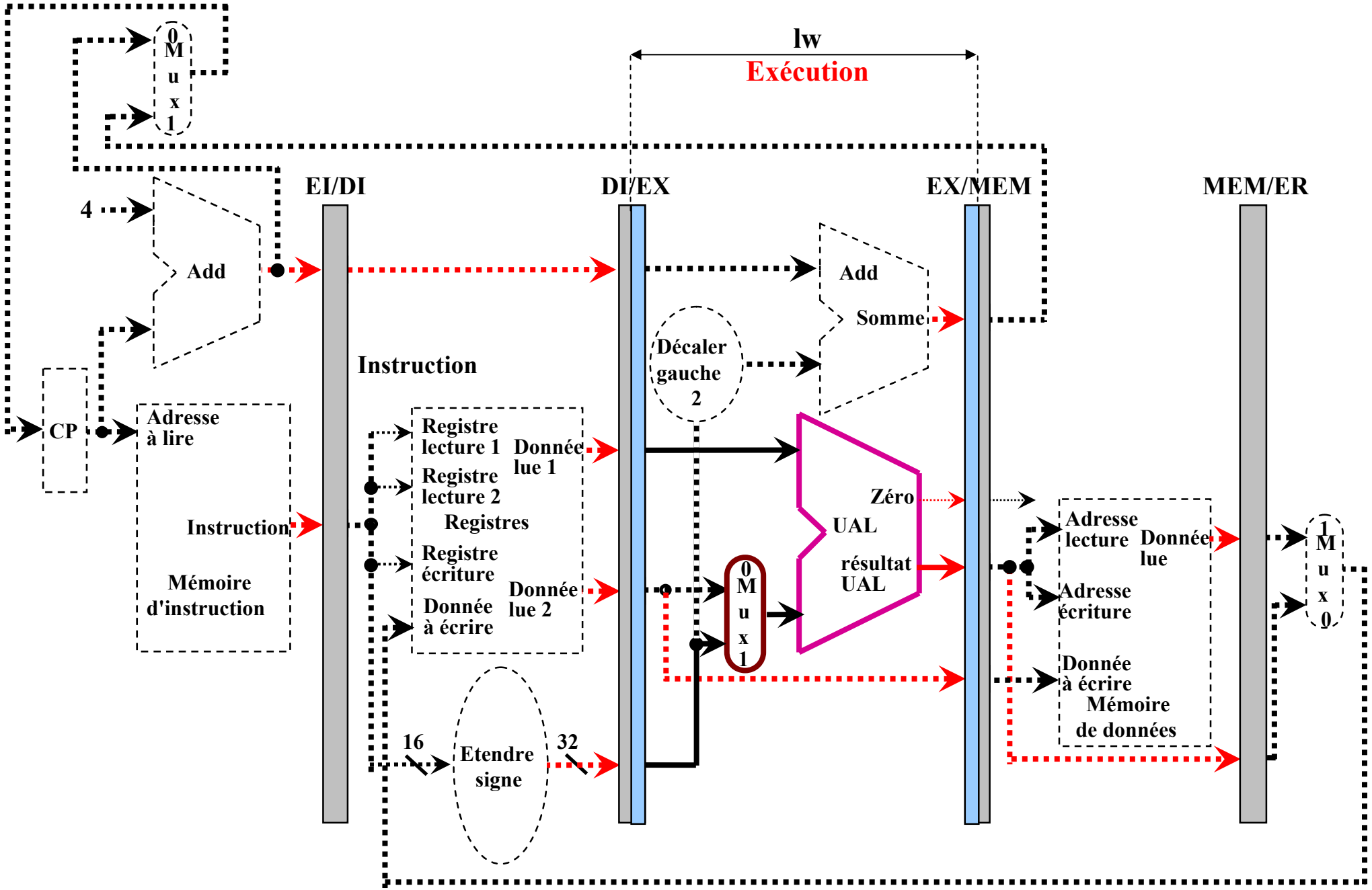
3. *Exécution et Calcul d'Adresse Effective* :

- **la valeur du registre 1 sur la 1ère entrée de l'UAL (entrée haute sur le schéma)**
- **la valeur du déplacement étendu sur la 2ème entrée de l'UAL (entrée basse sur le schéma) .**

- **A partir de ces valeurs en sortie du registre pipeline DI/EX (valeurs stockées au cycle précédent) :**
  Début du cycle :
  √ addition de ces 2 valeurs en vue de l'obtention de l'adresse effective

- **Stockage à la fin du cycle de l'adresse effective dans le registre pipeline EX/MEM.**

FONCTIONNEMENT DU CHEMEIN DE DONNEES PIPELINE (7/15)



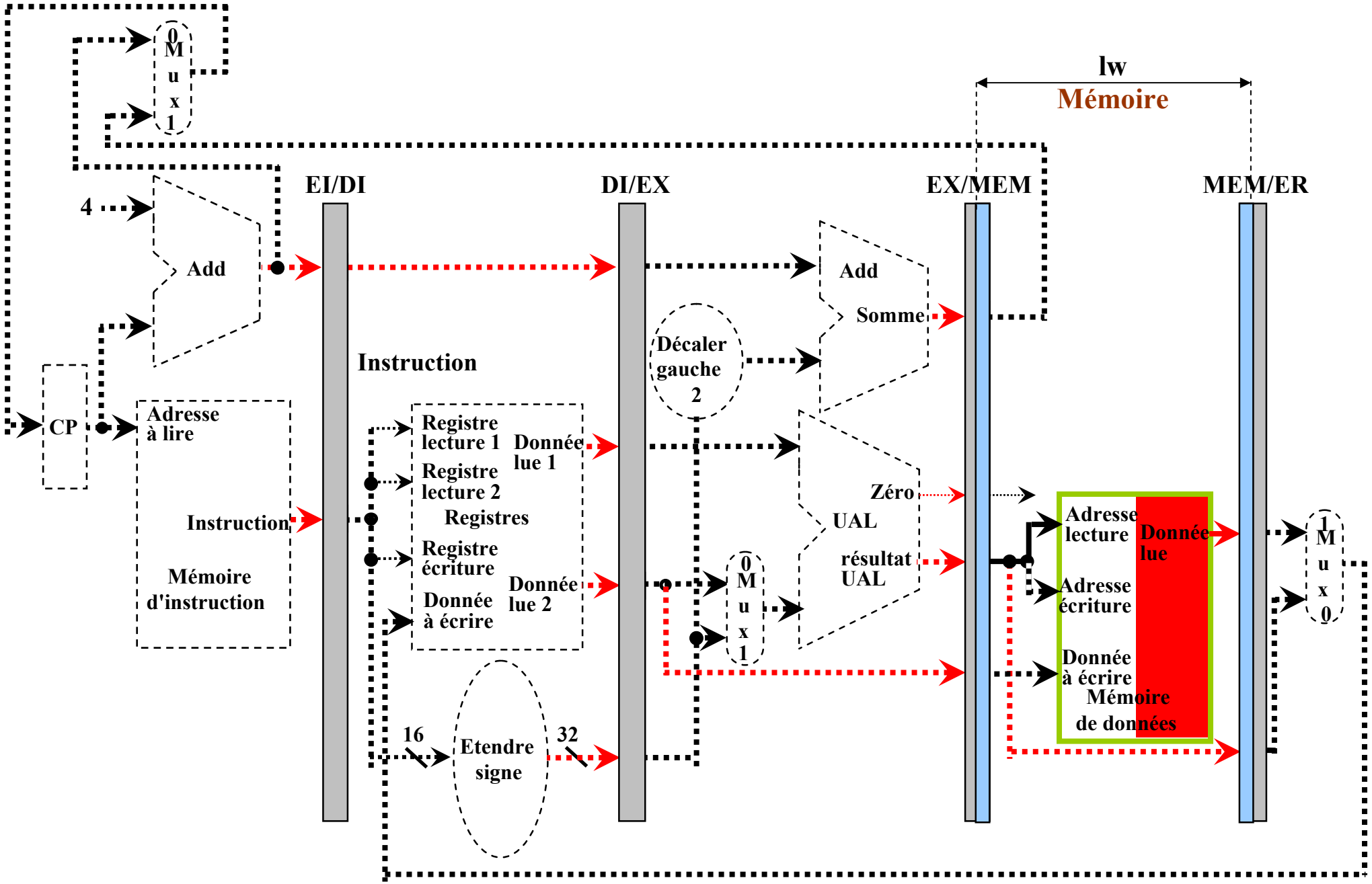
FONCTIONNEMENT DU CHEMIN DE DONNEES PIPELINE (8/15)

4. *Mémoire* :

- **A partir de l'adresse effective en sortie du registre pipeline EX/MEM (stockée au cycle précédent) :**
  Début du cycle :
  √ **Lecture de la mémoire de données**

- **A la fin du cycle la donnée fournie par la mémoire est stockée dans le registre pipeline MEM/ER.**

FONCTIONNEMENT DU CHEMEIN DE DONNEES PIPELINE (9/15)



FONCTIONNEMENT DU CHEMIN DE DONNEES PIPELINE (10/15)

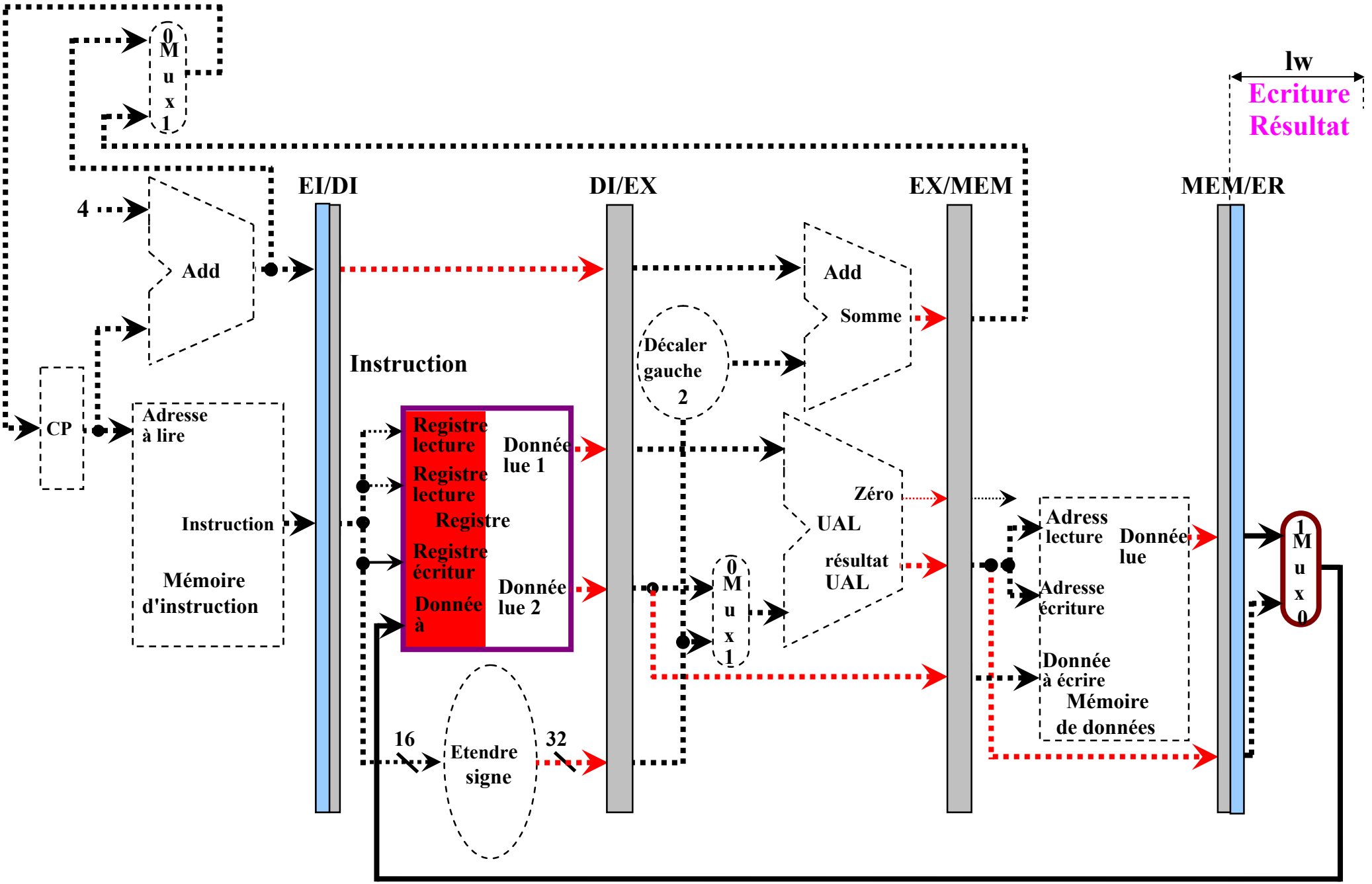
5. *Ecriture du Résultat* :

- **Le registre pipeline MEM/ER fournit la donnée stockée au cycle précédent :**
  - **√ Au banc de registres (2ème étage) en vue d'une écriture**

- **A la fin du cycle la donnée est stockée dans le registre de destination.**

Ce parcours des instructions de chargement montre que toute information utile dans l'étage suivant du pipeline doit y être transmise via un registre pipeline.

FONCTIONNEMENT DU CHEMEIN DE DONNEES PIPELINE (11/15)



FONCTIONNEMENT DU CHEMIN DE DONNEES PIPELINE (12/15)

Le parcours d'une instruction de rangement montre une similitude d'exécution des instructions. Il souligne de plus le besoin de conserver dans les registres pipeline les informations qui seront utilisées plus tard dans l'exécution de l'instruction. Le parcours des 5 étages de pipeline pour les instructions de rangement est le suivant :

1. Extraction d'Instruction :

- **A partir de l'adresse dans CP :**

  Début du cycle en parallèle :

  - √ **Lecture de l'instruction en mémoire d'instruction.**
  - √ **Calcul de la nouvelle adresse CP + 4, puis *chargement dans CP* pour être prêt pour le prochain cycle.**
- **A la fin du cycle stockage dans le registre pipeline EI/DI.**
  - √ **de l'instruction Ins[31-0] (EI/DI est semblable au registre Instruction).**
  - √ **de la nouvelle adresse au cas où elle serait utile à une instruction ultérieure, telle que beq.**

***2. Décodage Instruction et Lecture Registres* :**

- **A partir du champ instruction Ins[31-0] en sortie du registre pipeline EI/DI :**

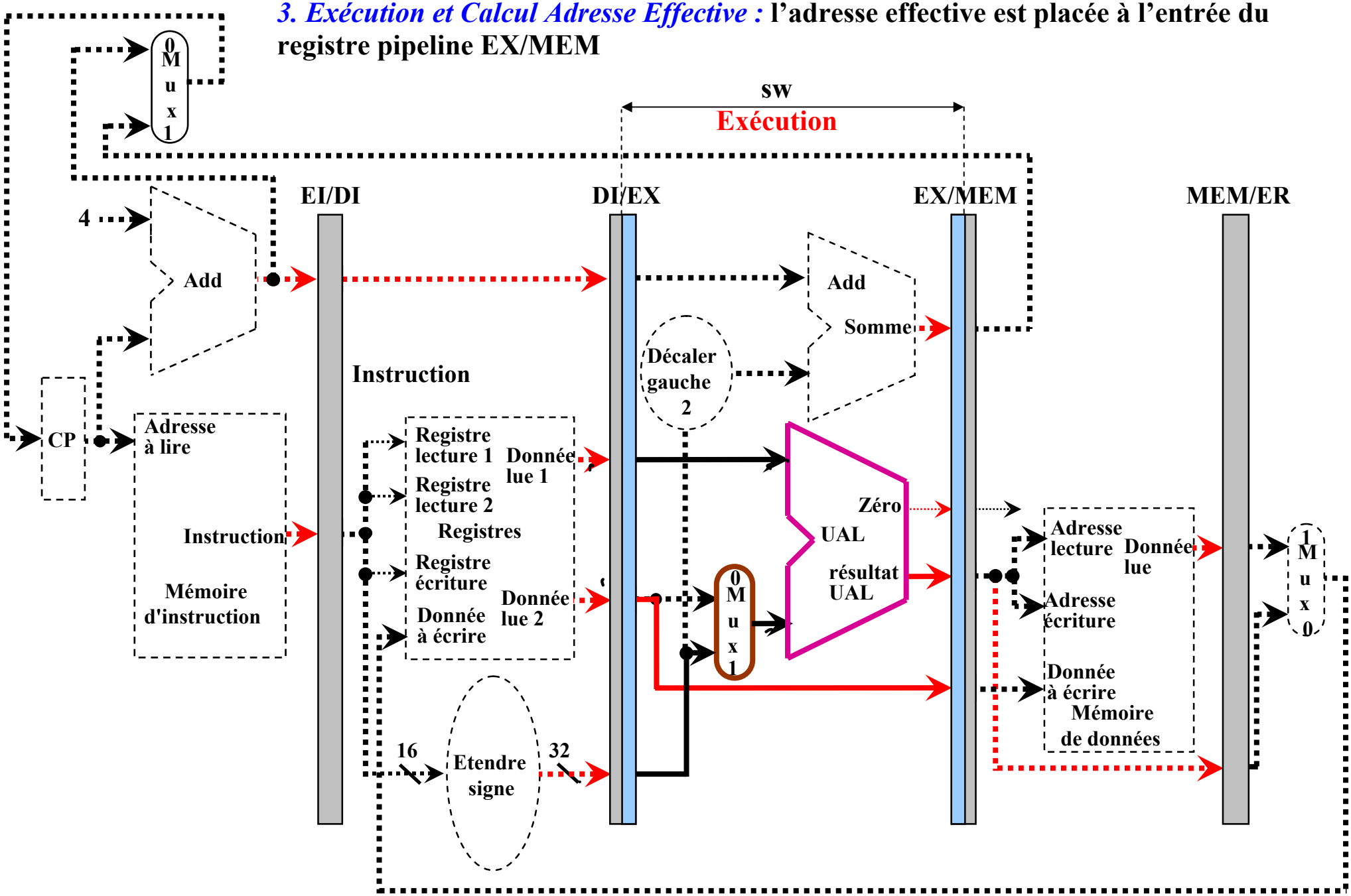
  Début du cycle en parallèle :

  - √ **Extension du signe à partir du champ immédiat (Ins[15-0]) ;**
  - √ **Lecture registre 1 à partir du champ rs (Ins[25-21]) ; (c'est le registre de base pour sw)**
  - √ **Lecture registre 2 à partir du champ rt (Ins[20-16]).**
- **A la fin du cycle stockage dans le registre pipeline DI/EX :**
  - √ **de la valeur des 2 registres lus**
  - √ **de la valeur du déplacement étendu**
  - √ **de la valeur CP+4 calculée dans l'étage EI**

Ces 2 premiers étages sont utilisés par toutes les instructions ➔ trop tôt pour identifier l'instruction.

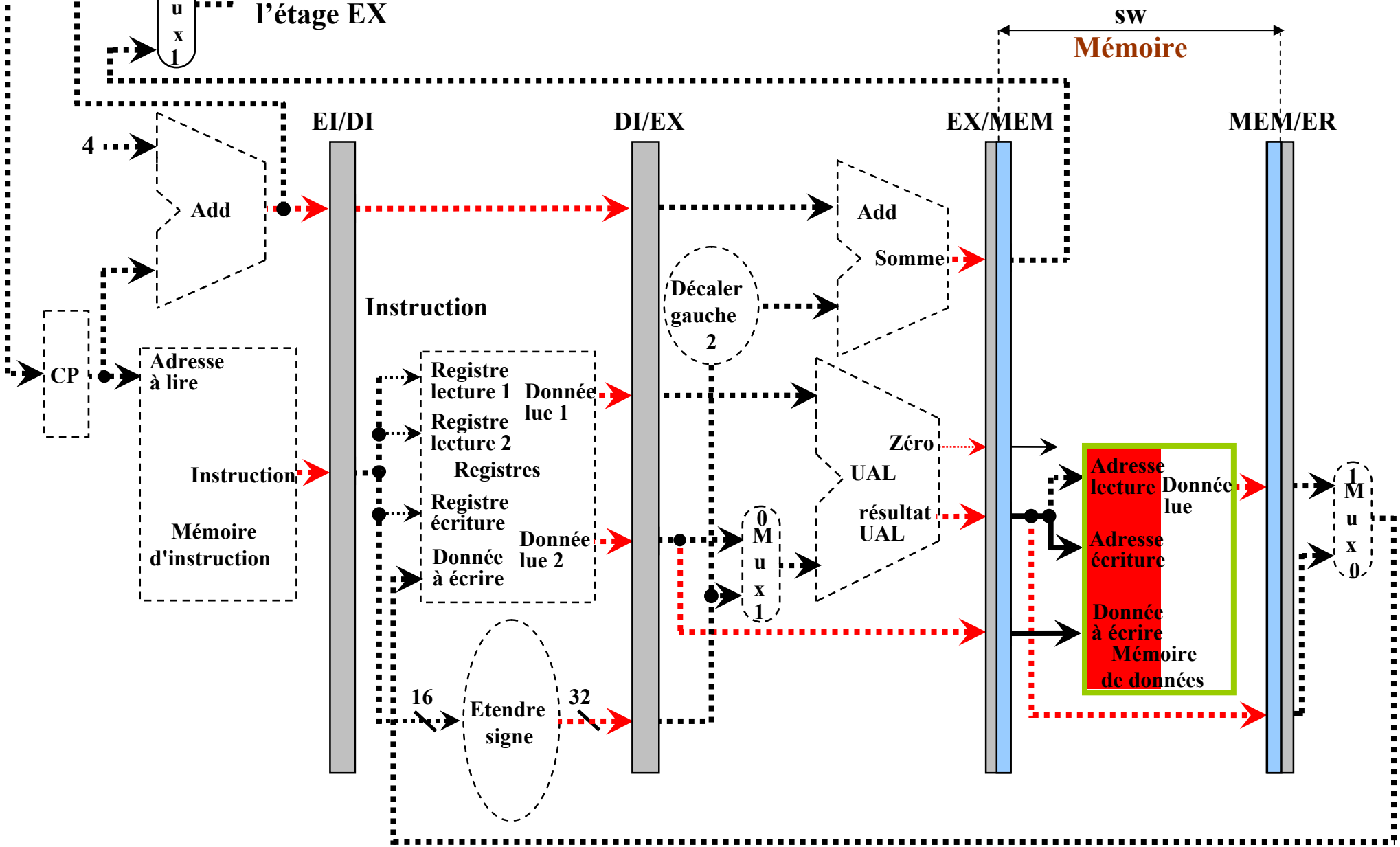
FONCTIONNEMENT DU CHEMEIN DE DONNEES PIPELINE (13/15)

3. Exécution et Calcul Adresse Effective : **l'adresse effective est placée à l'entrée du registre pipeline EX/MEM**



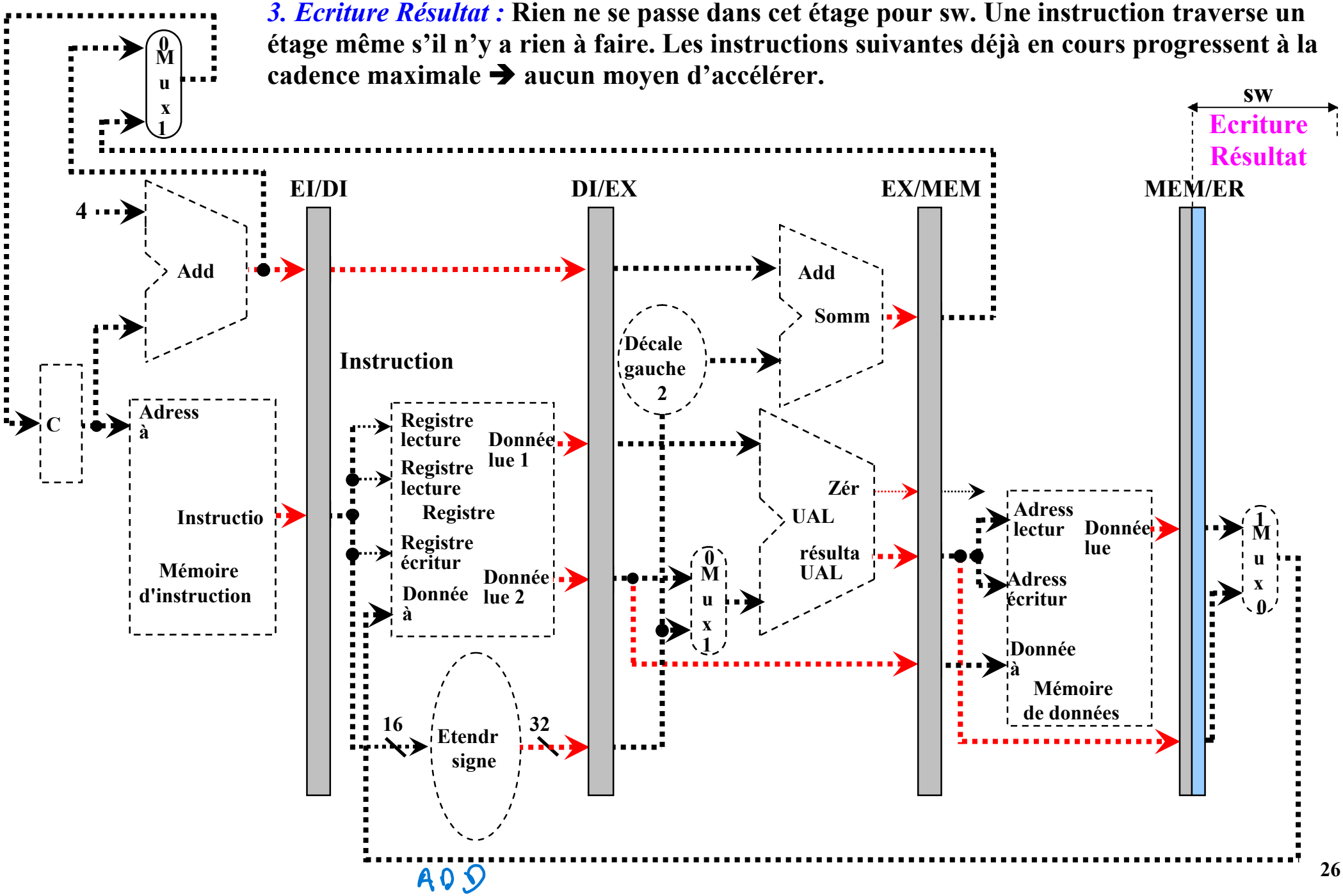
FONCTIONNEMENT DU CHEMEIN DE DONNEES PIPELINE (14/15)

4. Mémoire : **la valeur du registre à écrire a été lue à l'étage DI et stockée dans DI/EX. Le seul moyen de fournir cette valeur à l'étage MEM est de la placer dans EX/MEM durant l'étage EX**



FONCTIONNEMENT DU CHEMEIN DE DONNEES PIPELINE (15/15)

3. Ecriture Résultat : **Rien ne se passe dans cet étage pour sw. Une instruction traverse un étage même s'il n'y a rien à faire. Les instructions suivantes déjà en cours progressent à la cadence maximale ➔ aucun moyen d'accélérer.**



REMARQUES SUR LE FONCTIONNEMENT DU CHEMIN PIPELINE

- **Le rangement illustre que le transfert d'une information d'un étage antérieur à un étage ultérieur nécessite son placement dans un registre pipeline. sinon elle sera perdue lorsque l'instruction suivante entre dans cet étage.**

- **Le rangement a besoin de transférer l'un des 2 registres lus à l'étage DI vers l'étage MEM, où sa valeur sera stockée en mémoire. La donnée est d'abord placée dans le registre DI/EX puis transférée au registre EX/MEM.**

- **Chargement et rangement montrent *un autre point clé* :**

  √ **Chaque composante du chemin de données est utilisée à l'intérieur d'un seul étage du pipeline.**
  √ **Une composante et son contrôle peuvent donc être associés à un seul étage de pipeline.**

- ***A partir de ses remarques :***
  √ ***Une question se pose : Quel est le registre mis à jour par le chargement à l'étage ER ?***
  √ ***En d'autres termes : Quelle instruction fournit le numéro de registre à écrire ?***

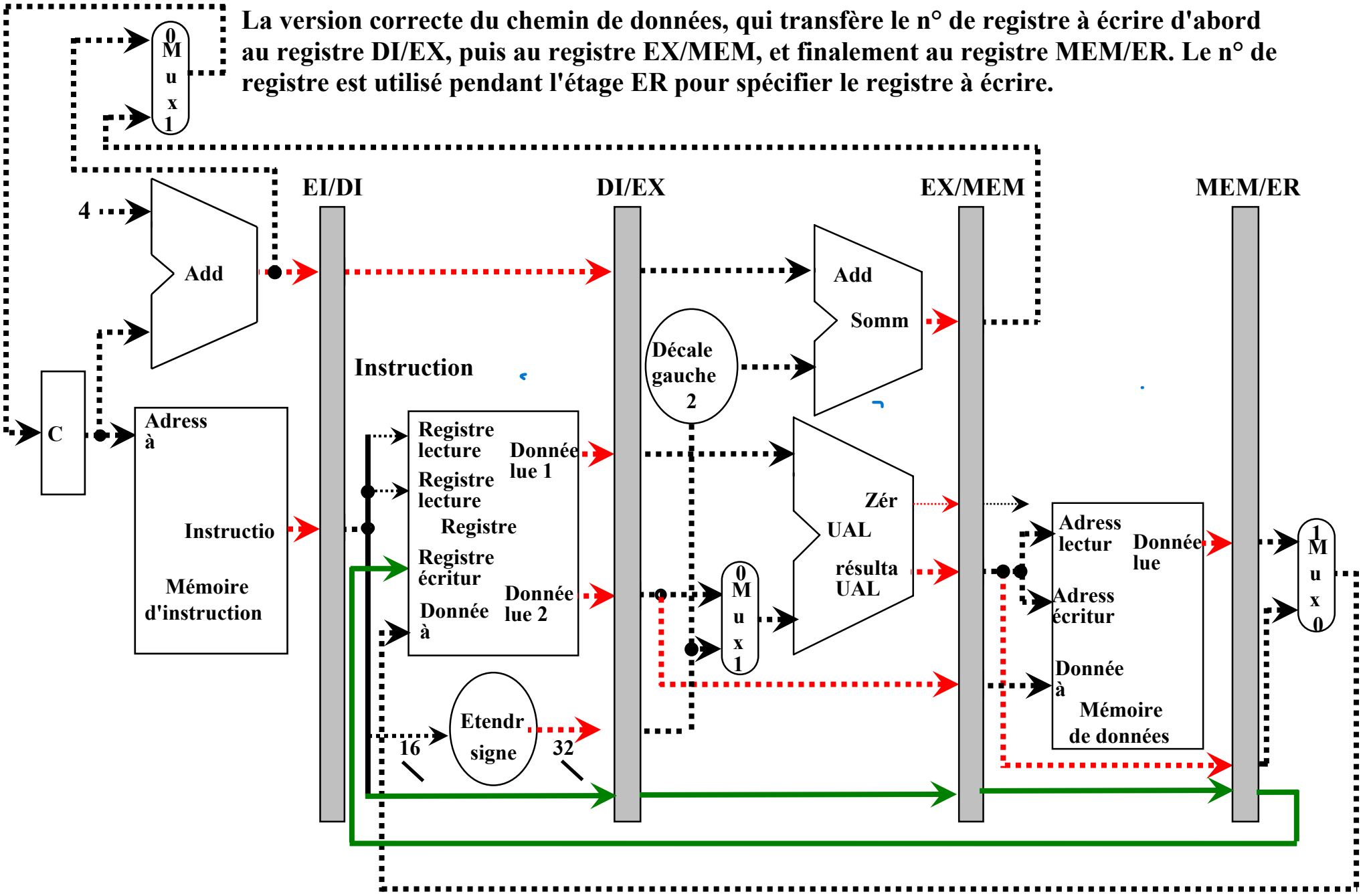
- ***La réponse :***

  √ ***C'est l'instruction dans le registre EI/DI qui fournit le numéro à la place du chargement à l'étage ER !!!***

- ***Solution :***
  √ ***Comme pour le rangement, le chargement doit transférer le numéro de registre à écrire entre les registres DI/EX et EX/MEM puis MEM/ER pour pouvoir l'utiliser à l'étage ER.***

CHEMIN DE DONNEES PIPELINE CORRIGE

La version correcte du chemin de données, qui transfère le n° de registre à écrire d'abord au registre DI/EX, puis au registre EX/MEM, et finalement au registre MEM/ER. Le n° de registre est utilisé pendant l'étage ER pour spécifier le registre à écrire.



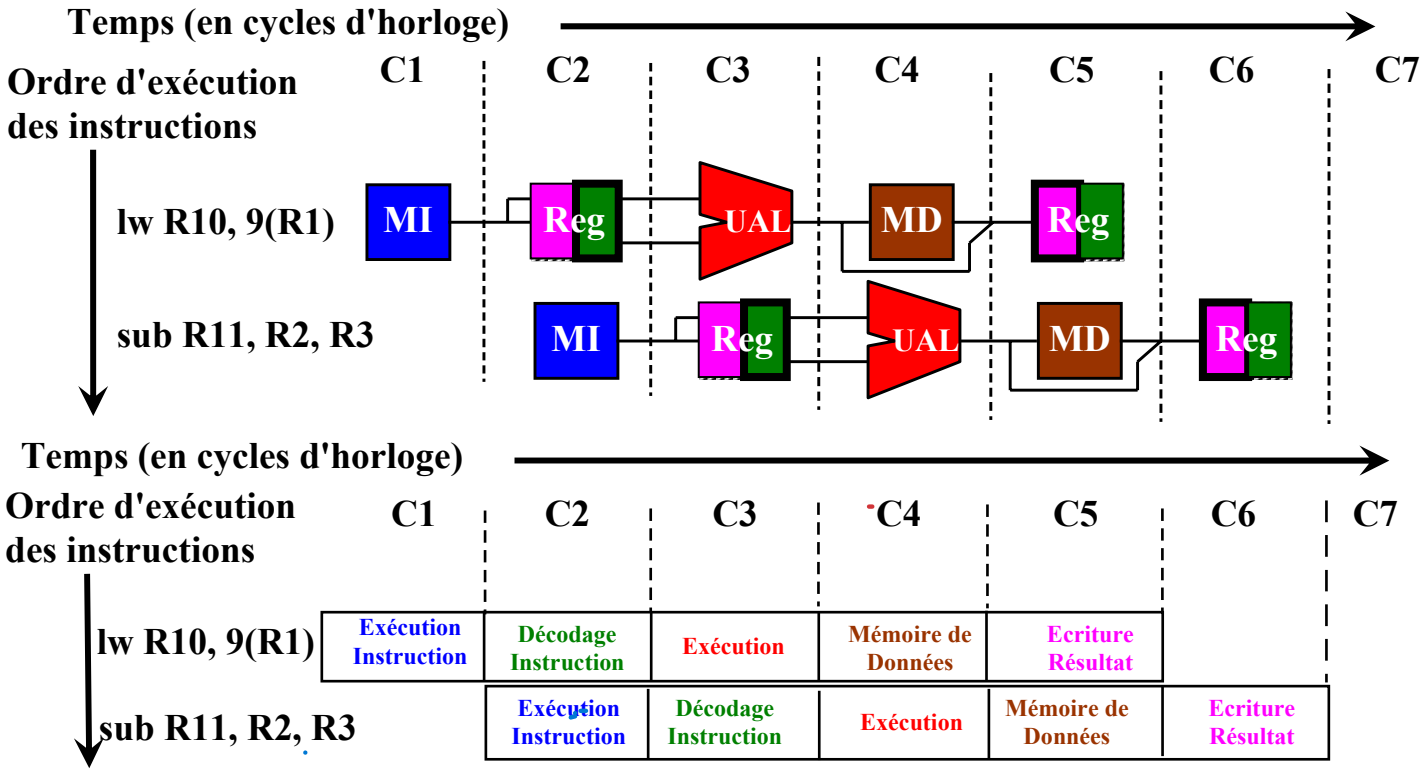
DIAGRAMMES DE REPRESENTATION DES PIPELINES (1/7)

- La technique du pipeline peut être difficile à comprendre : un grand nombre d'instructions s'exécutent simultanément dans le même chemin de données à chaque cycle d'horloge.

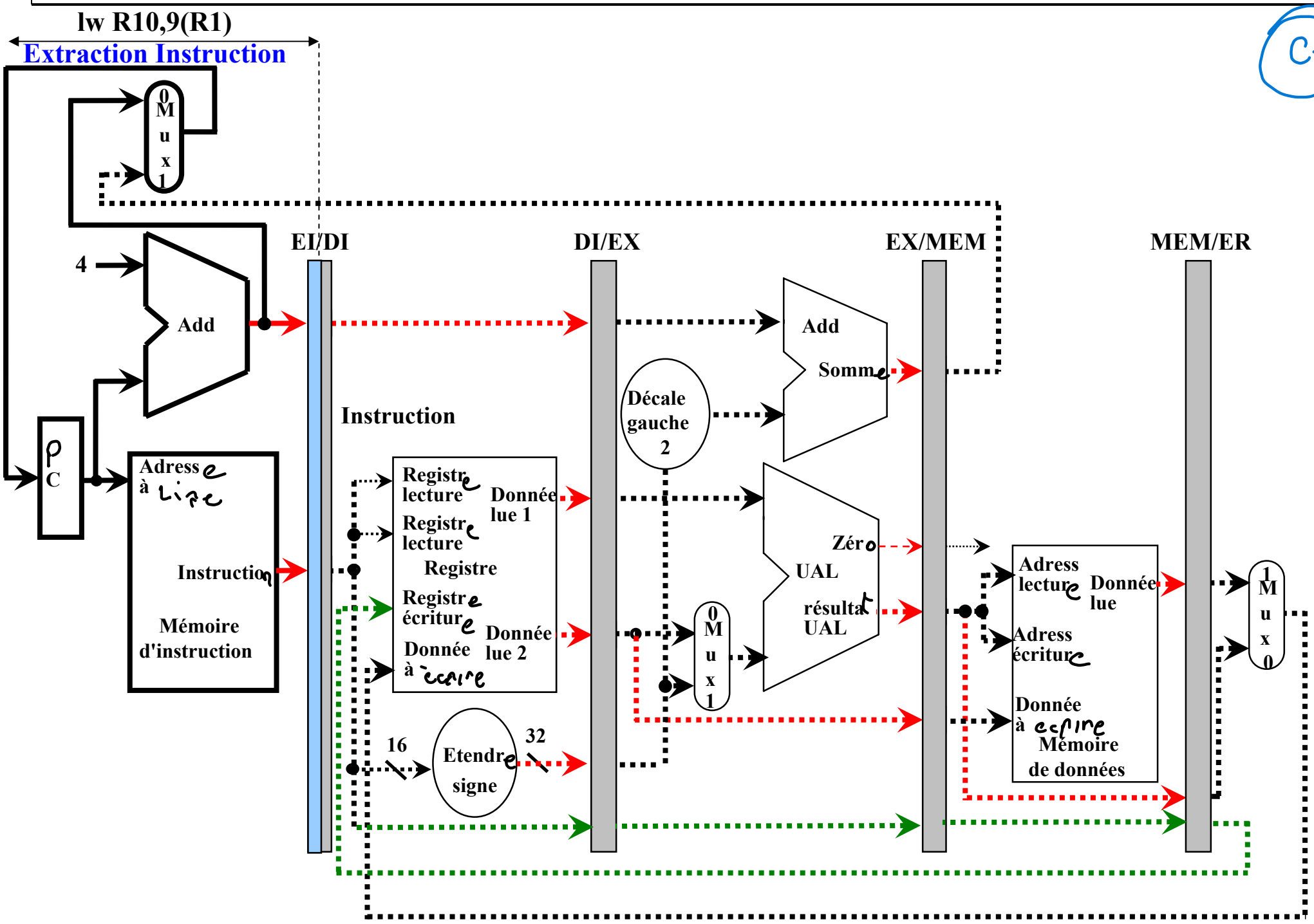
- Pour faciliter la compréhension, il existe deux 2 diagrammes de représentation :
  - √ Diagrammes à plusieurs cycles d'horloge,
  - √ diagrammes à un cycle d'horloge.

- Considérons une séquence d'instructions suivante : lw R10, 9(R1)
  sub R11, R2, R3

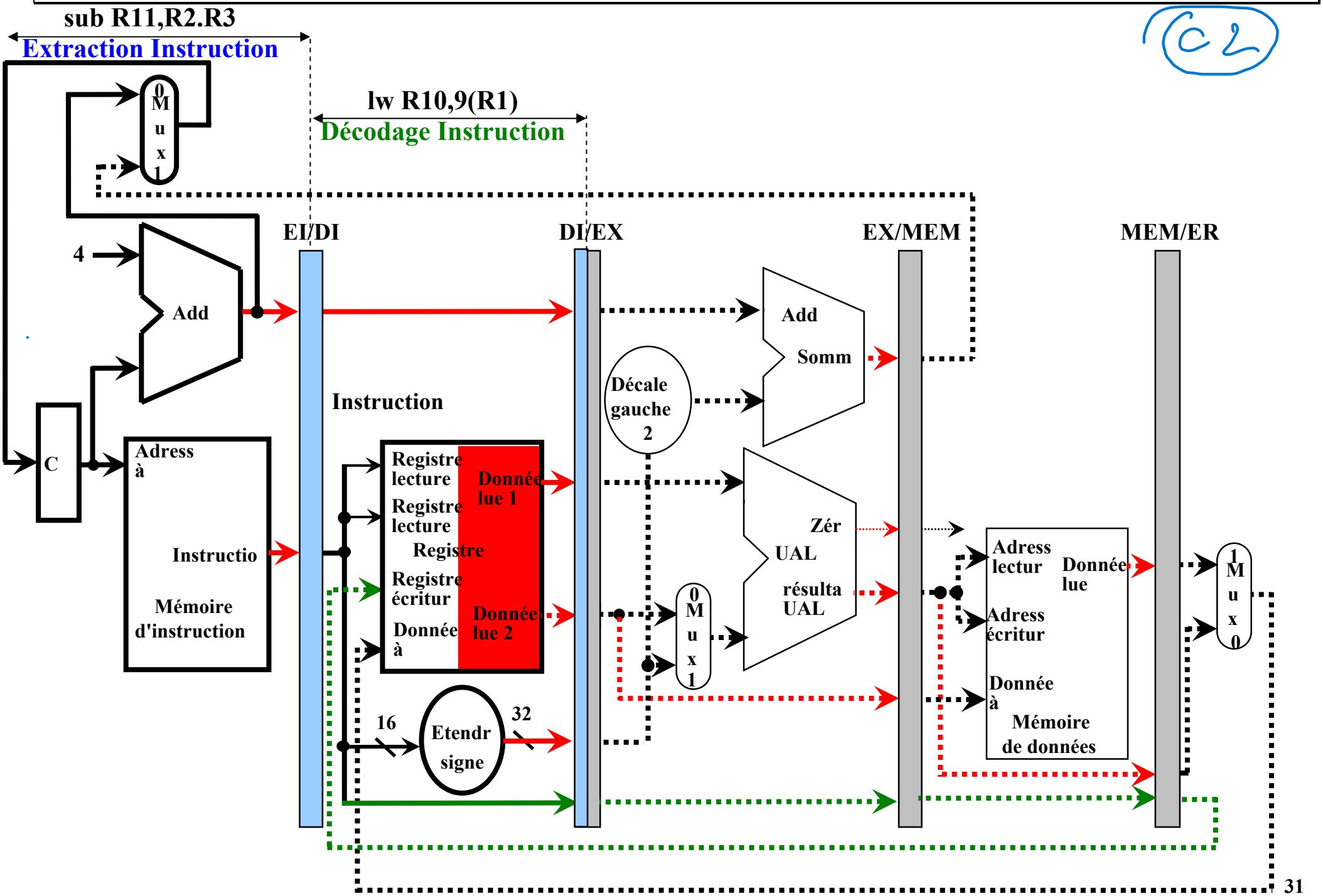
  √ Diagramme à plusieurs cycles d'horloge : diagramme espace-temps



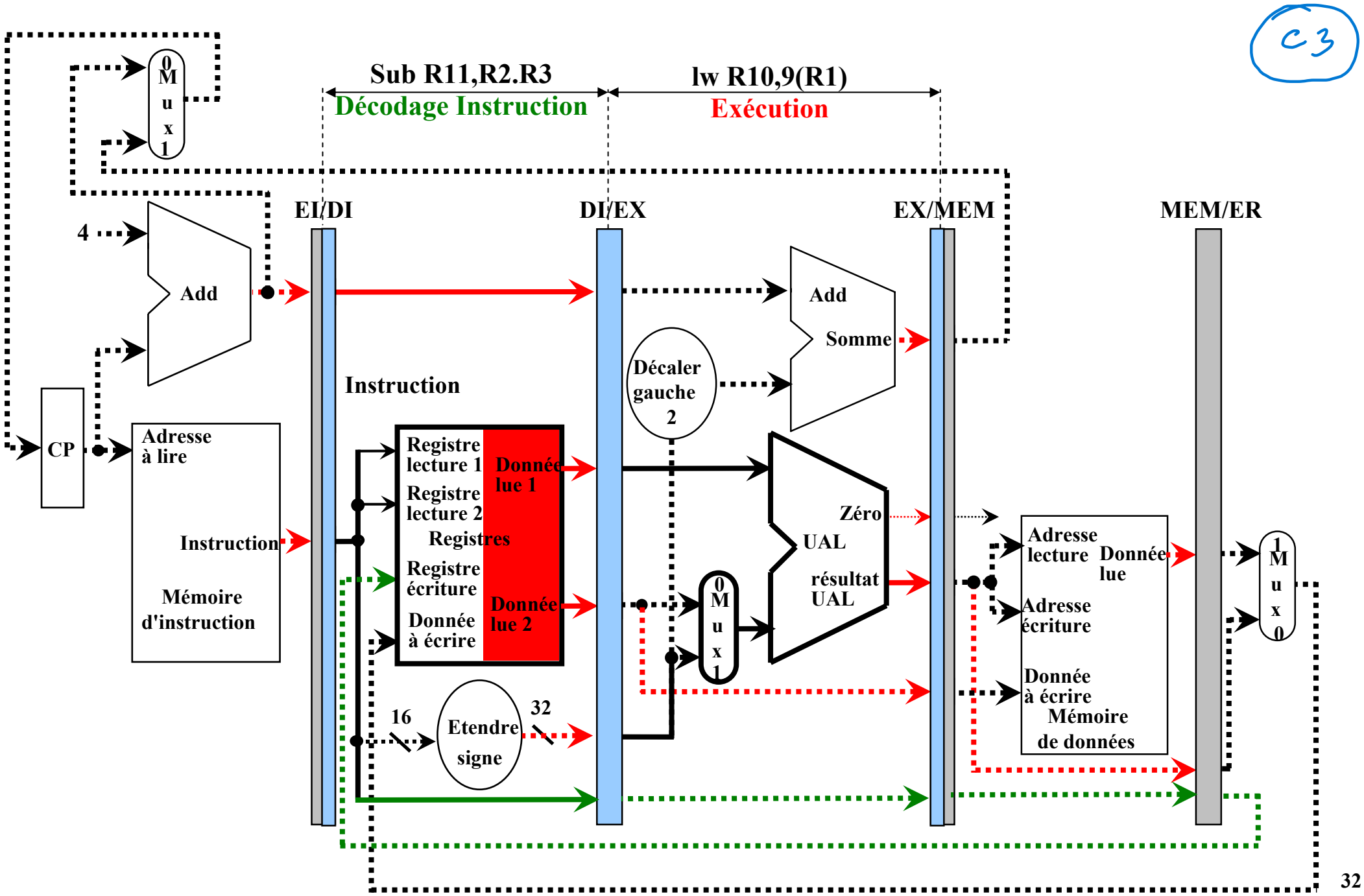
DIAGRAMMES DE REPRESENTATION DES PIPELINES (2/7)



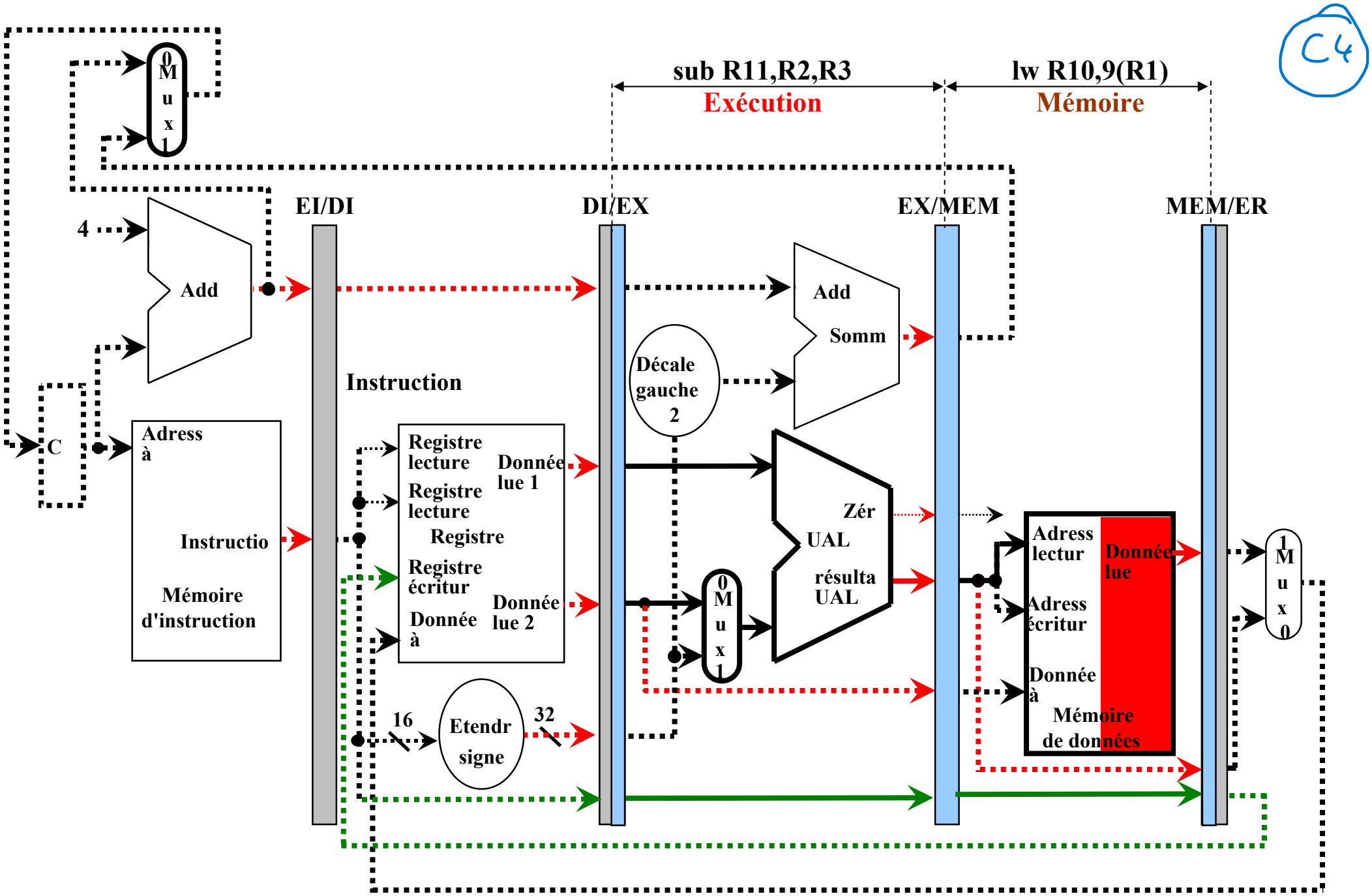
DIAGRAMMES DE REPRESENTATION DES PIPELINES (3/7)



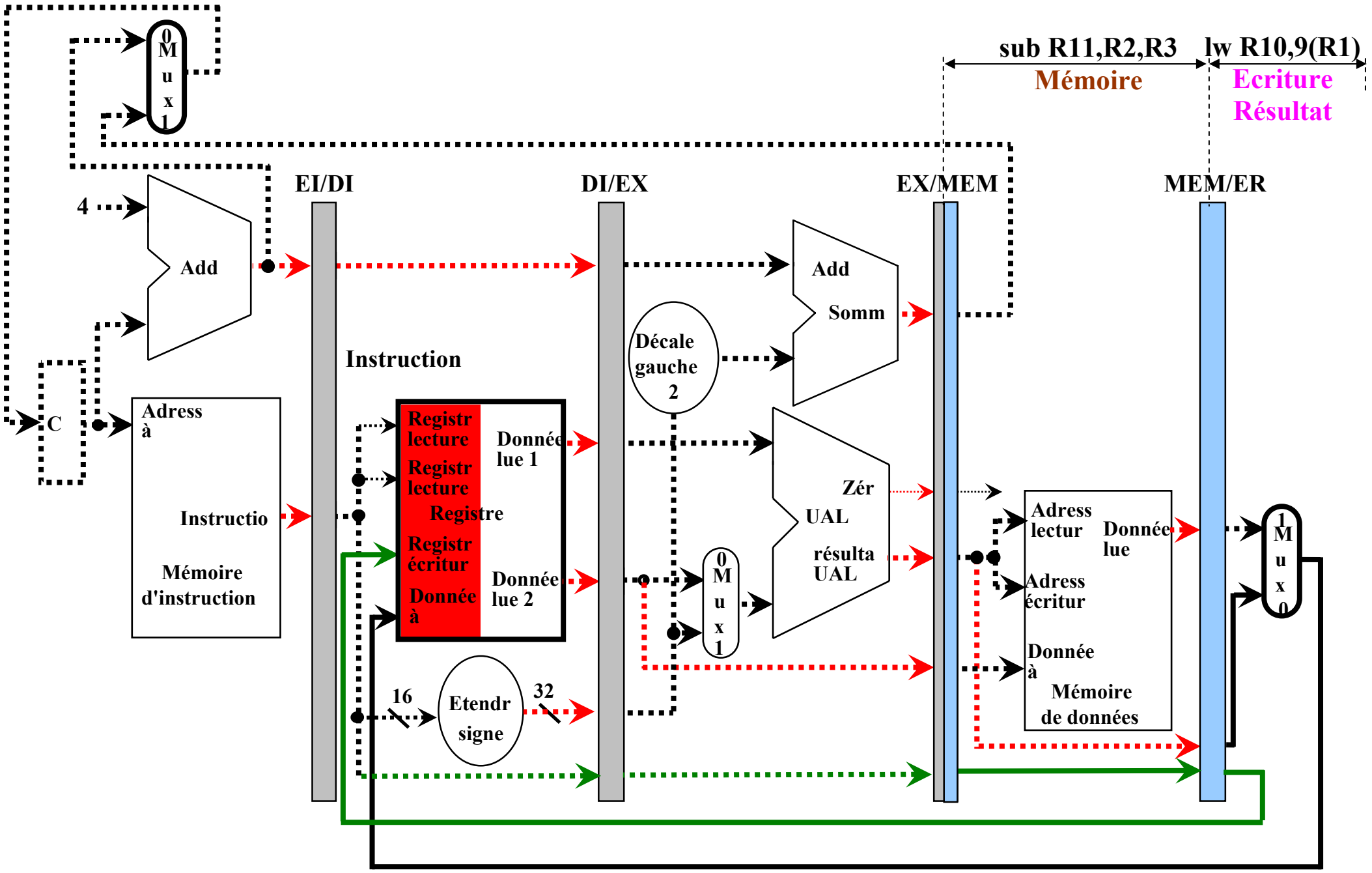
DIAGRAMMES DE REPRESENTATION DES PIPELINES (4/7)



DIAGRAMMES DE REPRESENTATION DES PIPELINES (5/7)

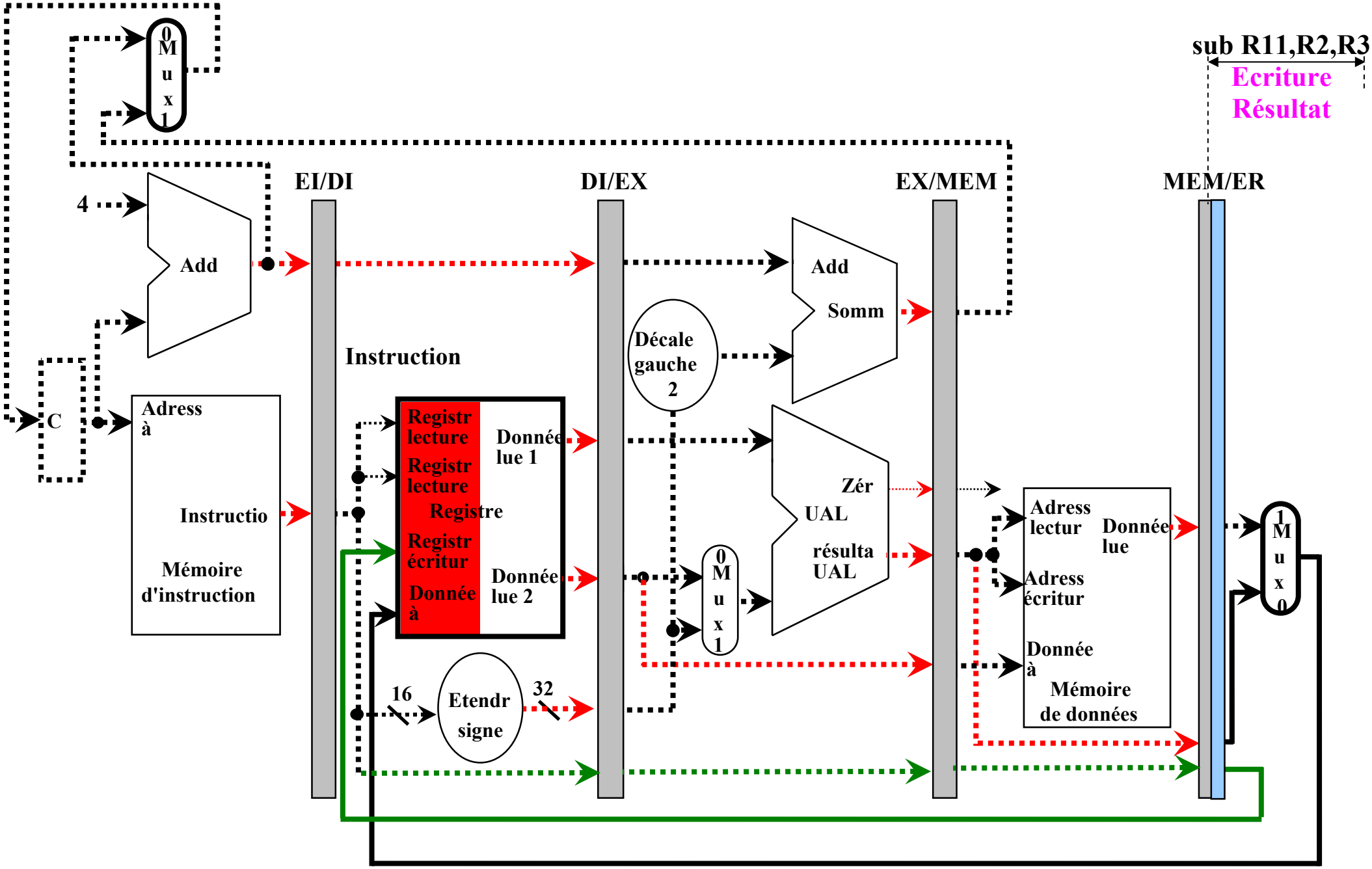


DIAGRAMMES DE REPRESENTATION DES PIPELINES (6/7)



DIAGRAMMES DE REPRESENTATION DES PIPELINES (7/7)

C6



REMARQUES SUR LES DIAGRAMMES DE REPRESENTATION DES PIPELINES

- *Dans un diagramme à plusieurs cycles d'horloge, le temps avance de la gauche vers la droite, et les instructions progressent du haut vers le bas.*

- *On utilise les diagrammes à plusieurs cycles d'horloge pour montrer les relations temporelles et les dépendances.*

- *Les diagrammes à un cycle d'horloge montrent l'état du chemin de données complet durant un seul cycle d'horloge, et chaque instruction du pipeline est identifiée par une étiquette au dessus de son étage respectif.*

- *On utilise les diagrammes à un cycle pour montrer en détail ce qui se passe à l'intérieur du pipeline pendant chaque cycle d'horloge.*

- *Les schémas sont généralement donnés en groupes pour montrer le fonctionnement du pipeline sur une séquence de cycles d'horloge pour plusieurs instructions.*

- *Extraire une tranche verticale de largeur un cycle d'un diagramme à plusieurs cycles d'horloge donne l'état du pipeline dans un diagramme à un cycle.*

- *L'ordre des instructions dans les deux diagrammes peut prêter à confusion : l'instruction la plus récente est en bas du diagramme à plusieurs cycles, et elle est à gauche dans le diagramme à un cycle.*

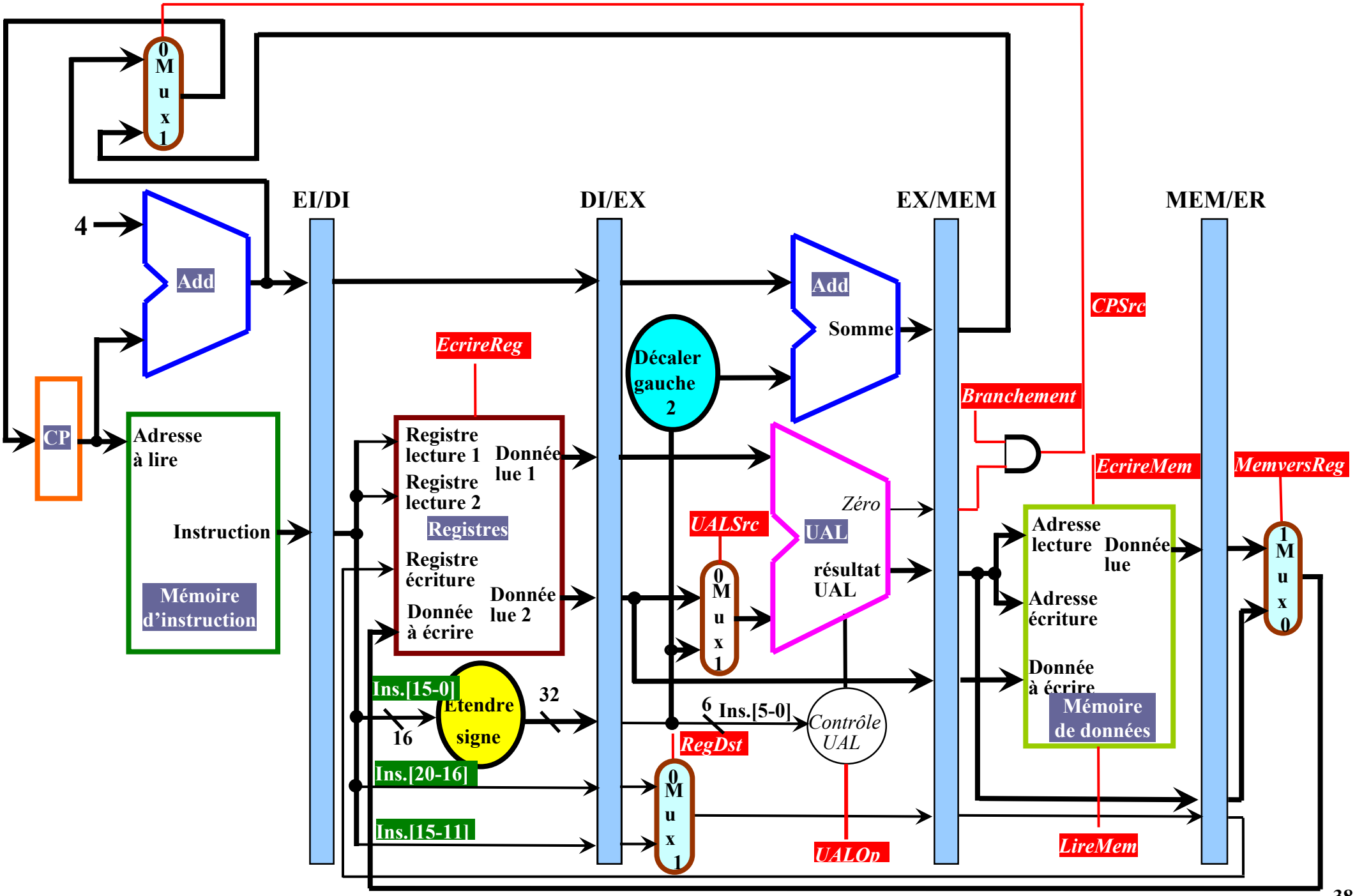
LE CONTROLE PIPELINE (1/6)

- On continue toujours à ignorer les aléas liés aux dépendances de données et de contrôle.
- Définir le contrôle pour le chemin de données pipeliné.

1. même étiquetage des lignes de contrôle sur le chemin de données.
2. même logique de contrôle de l'UAL, que le chemin de données à cycle unique simple
3. même logique de branchement,
4. même multiplexeur pour la sélection des numéros de registres
5. mêmes lignes de contrôle.

- Le CP est écrit à chaque cycle, comme dans le chemin à cycle unique simple, donc pas de signal d'écriture explicite.
- Les registres pipeline EI/DI, DI/EX, EX/MEM, et MEM/ER, sont mis à jour à chaque cycle ➔ pas de signaux spécifiques.

LE CONTROLE PIPELINE (2/6)



LE CONTROLE PIPELINE (3/6)

- **Entrées de contrôle de L'UAL et les fonctions qu'elle doit effectuer selon le type de l'instruction.**

Entrée de contrôle de l'UAL	Fonction
000	Et
001	Ou
010	Addition
110	Soustraction
111	Positionner si inférieur

- ***Positionnement des bits de contrôle de l'UAL en fonction des bits de contrôle UALOp et les codes fonctions pour les instructions de type R:***

Code-op instruction	UALOp classe	Opération de l'instruction	Code fonction	Actions de l'UAL désirées	Entrée de contrôle de l'UAL
LW	00	Chargement mot	xxxxxx	addition	010
SW	00	Rangement mot	xxxxxx	addition	010
BEQ	01	Branchement si =	xxxxxx	soustraction	110
type R	10	Addition	100000	addition	010
type R	10	Soustraction	100010	soustraction	110
type R	10	ET	100100	et	000
type R	10	OU	100101	ou	001
type R	10	Positionner si <	101010	soustraction	111

LE CONTROLE PIPELINE (4/6)

Description du fonctionnement des signaux de contrôle : 7 lignes de contrôle à 1 bit + les 2 lignes UALOp.

Nom du signal	Effet lorsque signal est inactif	Effet lorsque le signal est actif
LireMem	Aucun	Lecture mémoire : contenu placé sur «Donnée lue»
EcrireMem	Aucun	Ecriture mémoire : donnée placée sur «Donnée à écrire»
UALSrc	2nd opde UAL ← 2ème sortie de la banque de registres	2nd opde l'UAL ← unité d'extension de signe
RegDst	le n° de registre à écrire ← rt.	le n° de registre à écrire ← rd.
ÉcrireReg	Aucun	Ecriture d'un registre par valeur placée sur «Donnée à écrire».
CPSrc	Le CP ← CP + 4.	Le CP ← destination du branchement.
MemversReg	Donnée à écrire dans un registre ← UAL.	Donnée à écrire dans un registre ← Mémoire de données.

- L'unité de contrôle positionne tous les signaux de contrôle en fonction du code-op, sauf CPSrc.
- CPSrc, doit être positionnée si l'instruction est un branchement (beq) → décision que peut prendre l'unité de contrôle en fournissant un signal appelons le ***Branchement***, et si la sortie Zéro de l'UAL est vraie.

CPSrc = *Branchement* . Zéro

LE CONTROLE PIPELINE (5/6)

Chaque ligne de contrôle est associée à un élément qui n'est actif que pendant un seul étage ➔ donc on peut subdiviser les lignes de contrôle en cinq groupes, selon l'étage du pipeline :

- ***Extraction Instruction :*** **les signaux de contrôle pour lire la mémoire d'instruction et pour écrire le CP sont toujours actifs, et il n'y a donc rien de spécial à contrôler dans cet étage du pipeline.**

- ***Décodage Instruction et Extraction Registre :*** **la même chose se produit à chaque cycle d'horloge, il n'y a donc pas de lignes de contrôle optionnelles à positionner.**

- ***Exécution :*** **les signaux à positionner sont *RegDst*, *(UALOp1, UALOp0)*, et *UALSrc*. Ces signaux contrôlent respectivement : *la sélection du n° du registre à écrire*, *l'opération UAL*, et *la sélection du second opérande UAL***

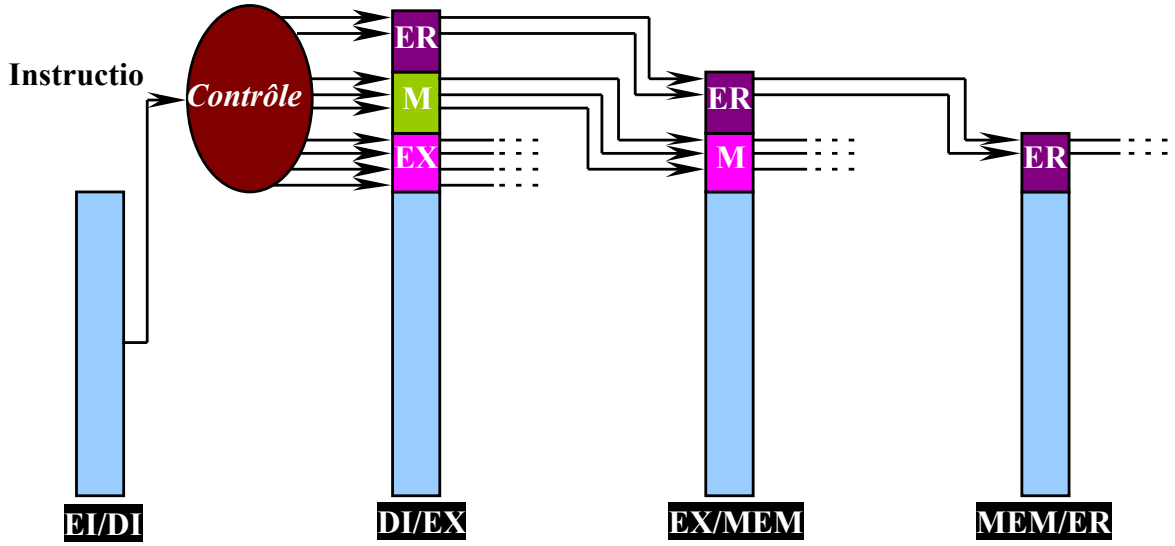
- ***Mémoire :*** **les lignes de contrôle à positionner sont : *Branchement*, *LireMem*, et *EcrireMem*. Ces signaux contrôlent respectivement : *la sélection de la source de la valeur du CP*, *l'opération mémoire (lecture ou écriture)*.**

- ***Ecriture du Résultat :*** **2 lignes de contrôle sont à positionner : *MemversReg* et *EcrireReg*. Ils contrôlent respectivement : *la sélection de la source de la valeur à écrire dans le banc* et *la validation de cette écriture.***

	Lignes de contrôle étage EX				Lignes de contrôle étage MEM			Lignes de contrôle étage ER	
Instruction	**RegDst**	**UALOp1**	**UALOp0**	**UALSrc**	**Branchement**	**LireMem**	**EcrireMem**	**EcrireReg**	**MemversReg**
Format R	1	1	0	0	0	0	0	1	0
lw	0	0	0	1	0	1	0	1	1
sw	X	0	0	1	0	0	1	0	X
beq	X	0	1	0	1	0	0	0	X

LE CONTROLE PIPELINE (6/6)

- **Pipeliner le chemin de données ne modifie pas le rôle des lignes de contrôle, seulement ils sont groupées par étage donc on peut utiliser les mêmes valeurs que précédemment pour chaque instruction.**
- **Mettre en œuvre le contrôle implique de positionner les 9 lignes de contrôle aux valeurs indiquées pour chaque étage et pour chaque instruction lors de son cheminement : *l'état du contrôle doit transiter avec l'instruction*.**
- **Le moyen le plus simple pour y parvenir est d'étendre les registres pipeline pour qu'ils contiennent les informations de contrôle.**
- **Les lignes de contrôle débutent à l'étage EX, ils sont donc créées et formées en groupes à l'étage de décodage**
- **Chaque groupe de signaux de contrôle est ensuite utilisé à l'étage adéquat au fur et à mesure que l'instruction avance dans le pipeline.**



CHEMIN DE DONNEES ET DE CONTROLE COMPLET