5. Présentation des Langages Automates Programmables

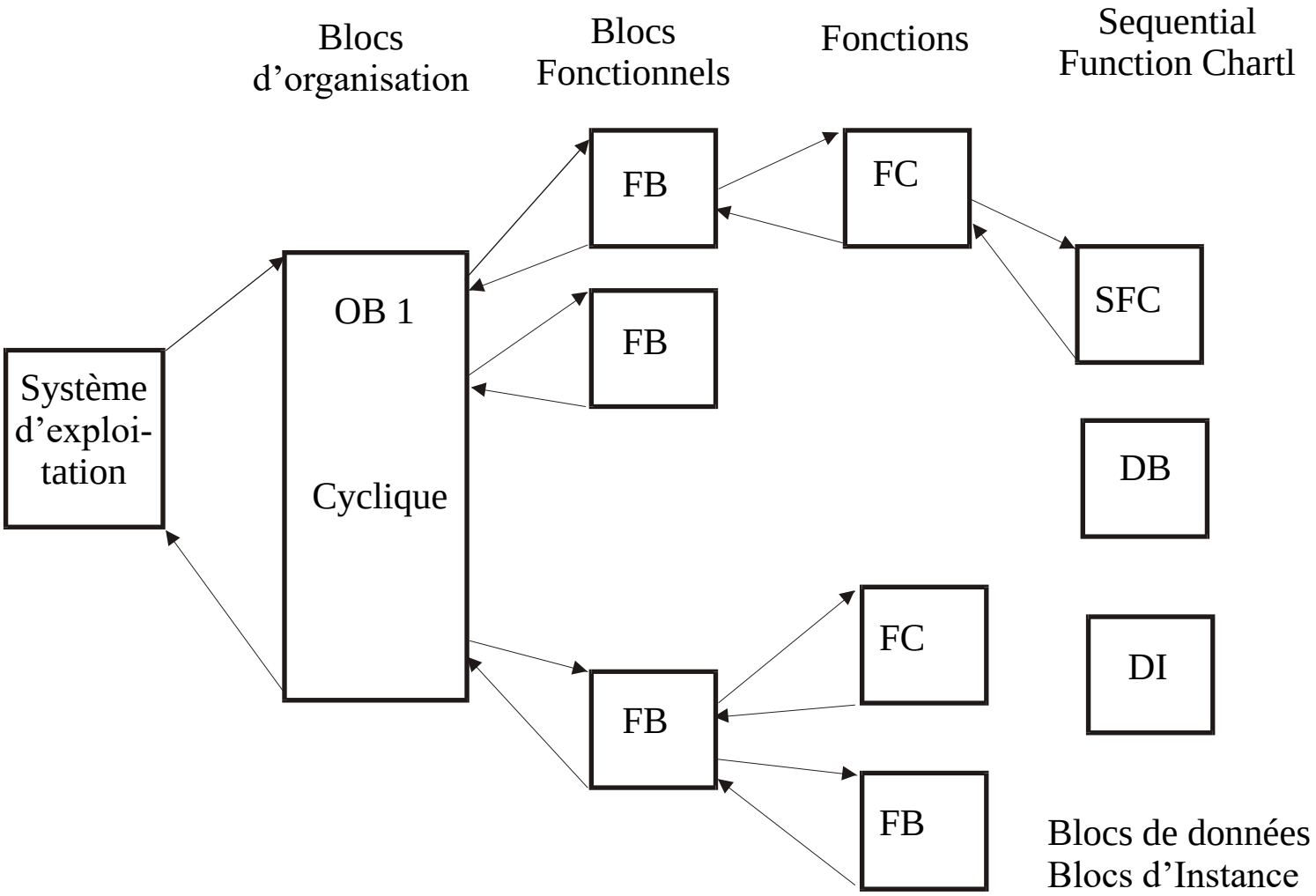
Pierre Duysinx
Université de Liège
Année académique 2021-2022

SIEMENS PLCs Série S7

CARACTÉRISTIQUES PRINCIPALES

- **Multiprocesseur**
  - bit processor
  - word processor
  - PID processor
  - communication processor
- **Multi langage**
  - Liste d'Instructions (IL) ou liste d'instructions
  - Blocs Fonctionnels (LOG) ou logigramme
  - Contact (CONT) ou langage à contact
- **Mode séquentiel**
  - Programmation GRAFCET
  - Mode d'exécution séquentiel

STRUCTURE DES PROGRAMMES ET DES DONNÉES



STRUCTURE DES PROGRAMMES ET DES DONNÉES

- **Blocs d'organisation OB**

  L'OB1 est examiné à chaque cycle d'automate. C'est à partir de ce bloc que l'on fera appels aux différents blocs de programmes.

  L'OB100 et l'OB101 sont appelés au démarrage à chaud et à froid uniquement. Ils servent à l'initialisation des données.

- **Fonctions FC**

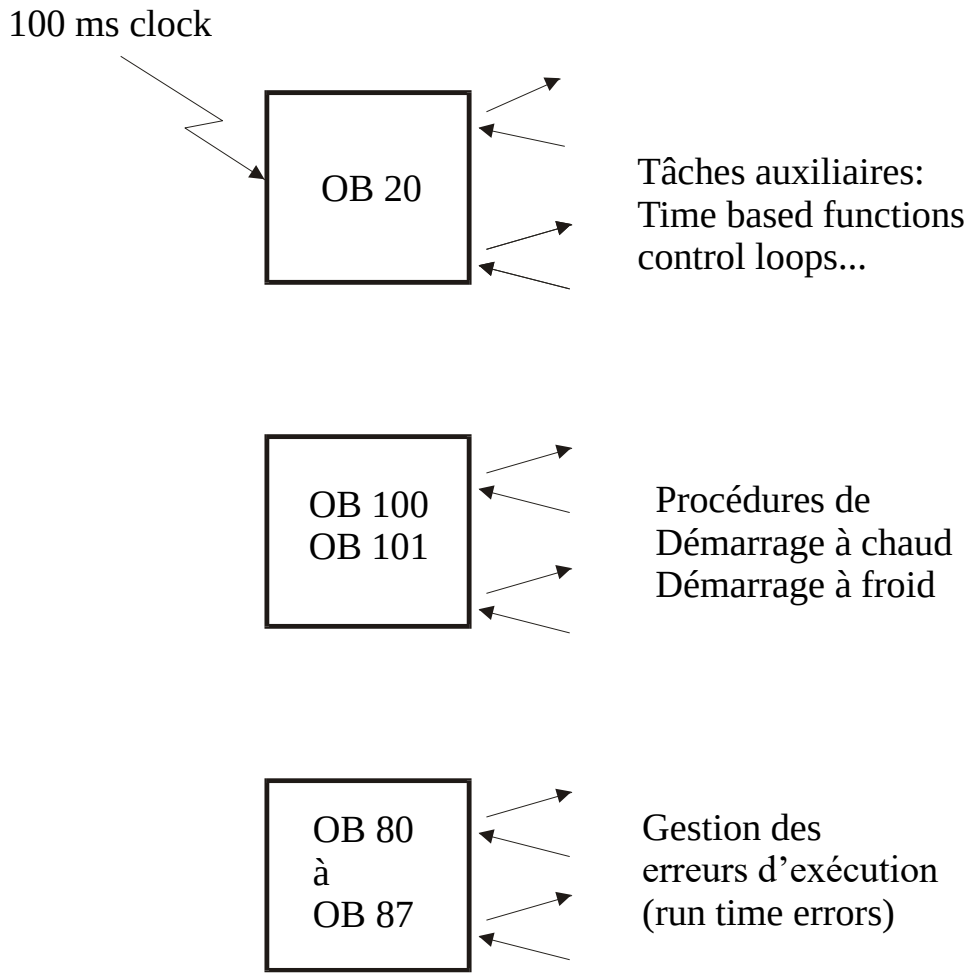
  C'est dans ces blocs que l'on va mettre les instructions à exécuter. La numérotation est libre (de 0 à 255). Ces blocs n'ont pas de mémoire.

- **Blocs de fonctions FB**

  Ces blocs sont des FC paramétrables. On peut passer des données en créant des DB d'instance associés à un seul FB pour le passage de paramètres. La numérotation est libre (de 0 à 255)

- **Fonctions systèmes SFC, blocs fonctionnels systèmes SFB, blocs fonctionnels de communication CFB**

STRUCTURE DES PROGRAMMES ET DES DONNÉES



STRUCTURE DES PROGRAMMES ET DES DONNÉES

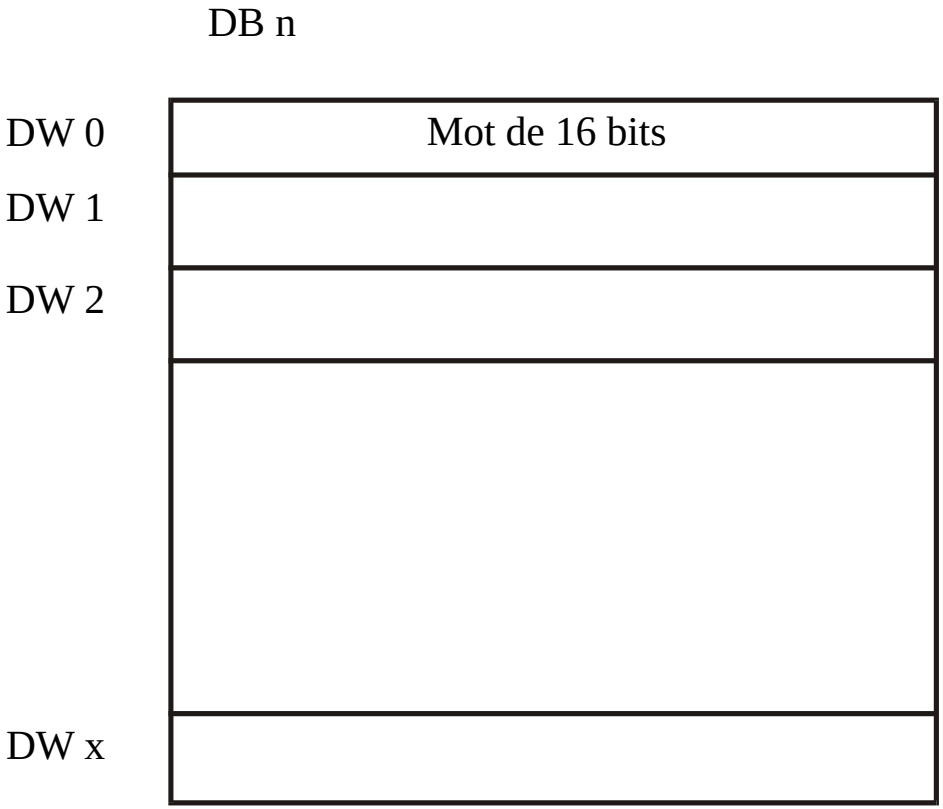
- LES BLOCS DE DONNEES

  Mots composés de 16 bits (DW) dans lesquels on peut lire et écrire des données

Un bloc de données ouvert reste valide jusqu'à ce qu'un autre bloc de données soit appelé ou qu'un bloc de code appelant soit terminé avec un BE ou un BEB

Dans un programme:

```
AUF  DBn
L    DW5
```



REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- BIT VARIABLES

Input	E	0.0 à 127.7	(#byte . #bit)
Output	A	0.0 à 127.7	
Flag (mémo interne)	M	0.0 à 255.7	
Data	D	0.0 à 255.7	
Timer Output	T	0 à 127	
Counter	Z	0 à 127	

- BYTE VARIABLES (=8 bits)

Input	EB	0 à 127	
Output	AB	0 à 127	
Flag	MB	0 à 255	
Data (left byte)		DL	0 à 255
(right byte)	DR	0 à 255	

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- WORD VARIABLES (= 16 bits)

Input	EW	0 à 126
Output	AW	0 à 126
Flag (mémo interne)	MW	0 à 254
Data	DW	0 à 255

- DOUBLE WORD VARIABLES (=32 bits)

Input	ED	0 à 126
Output	AD	0 à 126
Flag (mémo interne)	MD	0 à 254
Data	DD	0 à 254

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- CONSTANTES

Bit	TRUE/FALSE 1/0
Integer unsigned	B#(0,0) à B#(255,255)
Integer signed	-32768 à +32767
Real	-1.175494 E -38 à 3.402823 E 38
Hexadecimal (8 bits)	B#16#0 à B#16#FF
(16 bits)	W#16#0 à W#16#FFFF
Bit pattern	2#0 à 2#11111111 11111111
ASCII characters	‘ abcd ’
Time Constant (S5Time)	S5T#0ms, S5T2h46m30s
Time Constant (CPU TIME)	T#-24d20h31m23s746ms
	T#-65535ms à T#+65535ms
Counter	C#0 à C#999
Pointer	P#x.y

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- INSTRUCTIONS LOGIQUES (sur bit variables)

Instructions déjà disponibles en step 5

U	AND ou chargement de l'accumulateur du RLG si vide
UN	AND NOT
O	OR
ON	OR NOT
U(	AND sur expression entre parenthèses
O(	OR sur expression entre parenthèses
)	fin parenthèse
S	SET à ' 1 ' de l'opérande (permanente)
R	RESET à ' 0 ' de l'opérande (permanente)
=	assignation de l'opérande à la valeur du RLG (1 cycle)

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- INSTRUCTIONS LOGIQUES (sur bit variables)

Nouvelles instructions disponibles avec le step 7

CLR	mise à ‘ 0 ’ du RLG=ACCU
SET	mise à ‘ 1 ’ du RLG=ACCU
NOT	négation du registre logique
X	OU EXCLUSIF
XN	(OU NON) EXCLUSIF
FP	front montant d’une variable
FN	front descendant d’une variable

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- <u>INSTRUCTIONS DE BASE SUR BYTES, WORDS ET CONSTANTES</u>

  Travaille sur les accumulateurs arithmétiques (ACCU1, ACCU2)

Instructions de chargement

AUF	Ouverture DB	
L	Chargement:	ACCU1 $\rightarrow$ ACCU2,
		Opérande $\rightarrow$ ACCU1
T	Transfert	ACCU1 transféré dans opérande

Instructions arithmétiques

+I/+D/+R	Addition:	ACCU1 = ACCU2 + ACCU1
-I/-D/-R	Soustraction:	ACCU1 = ACCU2 - ACCU1
$\times$I/$\times$D/$\times$R	Multiplication:	ACCU1 = ACCU2 $\times$ ACCU1
$\div$I/$\div$D/$\div$R	Division:	ACCU1 = ACCU2 $\div$ ACCU1
	I si integer 16 bits, D si integer 32 bits, R si réels 32 bits	

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- INSTRUCTIONS DE COMPARAISON SUR ENTIERS ET REELS

	Bit accumulateur mis à ‘ 1 ’ si
==D/ ==I/ ==R	ACCU2 = ACCU1
><D/ ><I/ ><R	ACCU2 ≠ ACCU1
>D/ >I/ >R	ACCU2 > ACCU1
>=D/ >=I/ >=R	ACCU2 ≥ ACCU1
<D/ <I/ <R	ACCU2 < ACCU1
<=D/ <=I/ <=R	ACCU2 ≤ ACCU1

D pour entier 32 bits, I pour entiers 16 bits, R pour réels 32 bits

- OPERATIONS COMBINATOIRES SUR MOTS

OD/OW	OU sur MOT DOUBLE (32 bits) ou MOT (16 bits)
UD/UW	ET sur MOT DOUBLE (32 bits) ou MOT (16 bits)
XOD/XOW	OU EXCLUSIF sur MOT DOUBLE ou MOT

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- <u>OPERATION D'APPEL DES BLOCS</u>

UC	FC/FB	Saut inconditionnel, appel du bloc
CC	FC/FB	Saut conditionnel, appel du bloc si bit accumulateur RLG =1
CALL	FC/FB/SFC/SFB	saut avec transfert de paramètres

exemples: call FCn
call SFCn
call FBn1,DBn2
call SFBn1,DBn2

- <u>INSTRUCTIONS DE FIN DE BLOC</u>

BE	Fin de bloc (requis)
BEB	Saut conditionnel (si RLG=1) à la fin du bloc
BEA	Saut inconditionnel à la fin du bloc

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- TYPES DE TEMPOS (TIMERS)

SE: Temporisation sous forme de retard à la montée

Démarrage si RLG passe de 0 à 1

Particularité: arrêt de la temporisation si le signal de départ retombe à 0

SS: Temporisation sous forme de retard à la montée mémorisé

Particularité: la tempo continue même si le signal de départ retombe à 0

SA: Temporisation sous forme de retard à la retombé

Démarrage si RLG passe de 1 à 0

SI: Temporisation sous forme d'impulsion

La tempo monte à 1 après le départ et redescend après le temps imparti

SV: Temporisation sous forme d'impulsion prolongée

Particularité: la tempo continue même si le signal de départ retombe à 0

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- MODE D'ADRESSAGE DIRECT ET INDIRECT

Mode direct:

```
L     4
L     MW7
```

Mode indirect:

```
L     5
T     MW2
L     T[MW2]  équivalent à     L     T5

L     P#8.7
L     MD2
U     E[MD2]  équivalent à     U E8.7
```

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

- NOUVEAUX TYPES DE VARIABLES

Array: ARRAY[x1..x2,y1..y2,z1..z2]

Structure: STRUCT
:
:
END_STRUCT

<name-struct.name-var>

REVUE DES PRINCIPALES INSTRUCTIONS DU LANGAGE STEP 7 IL

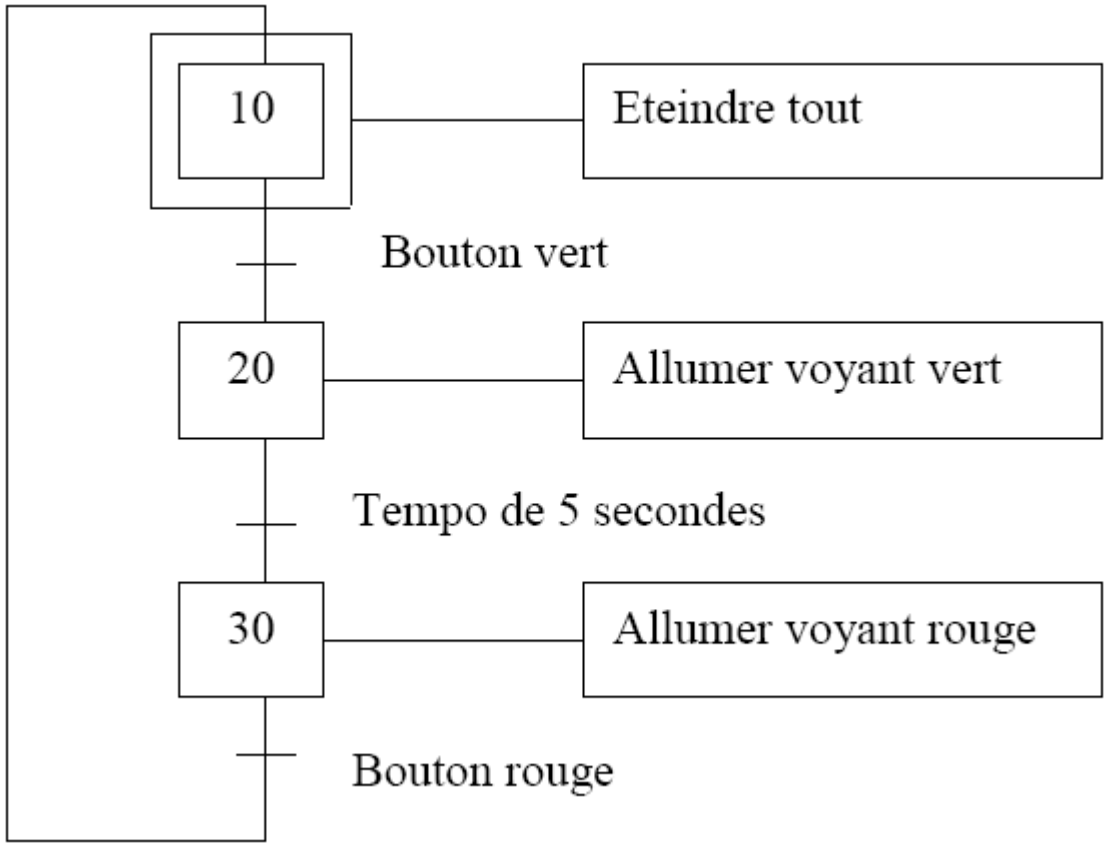
- PROGRAMME SOUS WINDOWS
- ARCHITECTURE ARBORESCENTE DU PROJET
  - *niveau 1, le projet:* la station connectée et le réseau (MPI)
  - *niveau 2, la définition de la station:* CPU et sa configuration matérielle
  - *niveau 3, les différents programmes (programmes S7) et la table de connexion pour le réseau*
  - *niveau 4:* sources externes, tables de mnémoniques
  - *niveau 5*: les blocs de programmes.
- PROGRAMMER ON LINE ou OFF LINE
- PLUSIEURS LANGAGES: IL, CONT, LOG
- VISUALISATION DYNAMIQUE ET FORCAGE DS VARIABLES
- DEFINITION DE LA CONFIGURATION MATERIELLE

TRANSPOSITION D'UN GRAFCET EN IL

- En partant d'une application séquentielle écrite en GRAFCET, on transpose de manière systématique en STEP 7 de la façon suivante. On divise le programme en trois parties:
- 1. *Calcul des réceptivités.* On associe un bit interne (mémento) à chaque réceptivité. Celui-ci est mis à 1 si la réceptivité est vraie.
- 2. *Évolution des étapes.* On utilise des bits d'étapes. A chaque étape est associé un bit interne qui est mis à 1 quand l'étape est active. On passe à l'étape suivante quand l'étape qui précède est active (validation de la transition) et que la réceptivité est vraie, ce qui correspond à mettre le bit d'étape suivante à 1 (set) et celui de l'étape précédente à 0 (reset).
- 3. *Actions associées aux étapes.* Il suffit d'imposer comme condition d'activation d'une action le bit correspondant à l'étape dans laquelle cette action doit être exécutée.

TRANSPOSITION D'UN GRAFCET EN IL

- Exemple



TRANSPOSITION D'UN GRAFCET EN IL

- **OB100**

  :CALL FC 1

  :BE

- **OB101**

  :CALL FC 1

  :BE

- **OB1 appel des sous routines**

  :CALL FC 2

  :CALL FC 3

  :CALL FC 4

  :BE

- **FC1 activation de l'étape initiale**

  :O M 0.0

  :ON M0.0

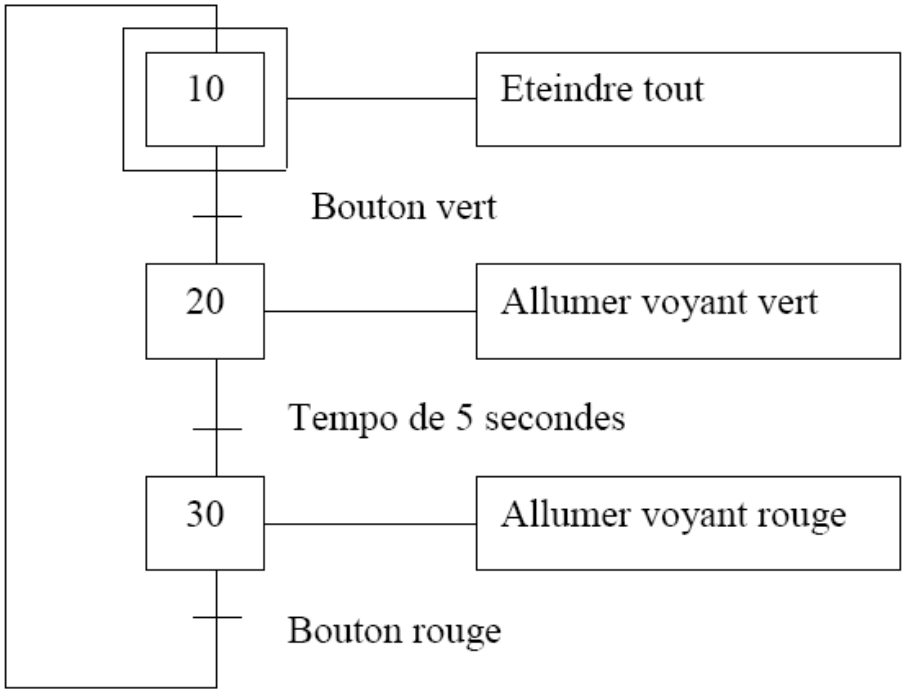
  :S M 10.0

  :BE

TRANSPOSITION D'UN GRAFCET EN IL

- **FC2 calcul des réceptivités**

```
:U E 0.4
:= M 10.1
:***
:UN T 1
:= M 20.1
:***
:UN E 0.5
:= M 30.1
:BE
```

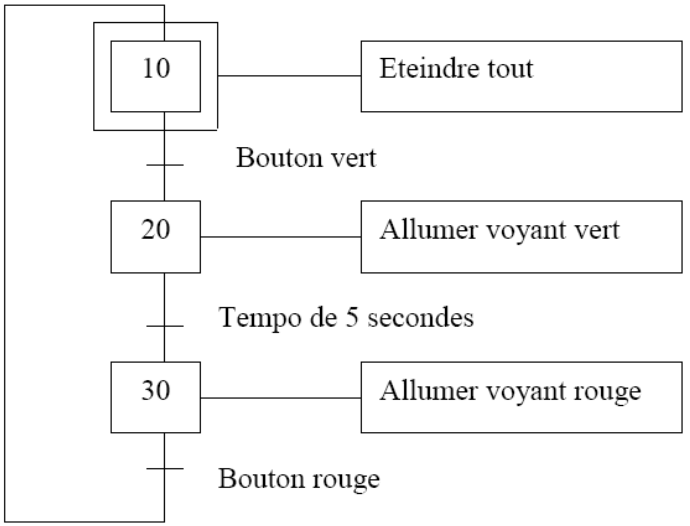


TRANSPOSITION D'UN GRAFCET EN IL

- **FC3 évolution du grafcet**

```
:U M 10.0
:U M 10.1
:S M 20.0
:R M 10.0
:BEB
:***
:U M 20.0
:U M 20.1
:S M 30.0
:R M 20.0
:BEB
:***
```

```
:U M 30.0
:U M 30.1
:S M 10.0
:R M 30.0
:BE
```



TRANSPOSITION D'UN GRAFCET EN IL

- **FC4 actions associées aux étapes**

```
:U M 20.0
:L S5T# 5s
:SV T1
:***
:U M 20.0
:= A0.2
:U M 30.0
:= A 0.6
:BE
```