

Formation Automatique et Informatique Industrielle

Master 1 S2

Matière : Systèmes Embarqués et Systèmes
Temps Réel SE-STR

Par : ATOUI Hamza

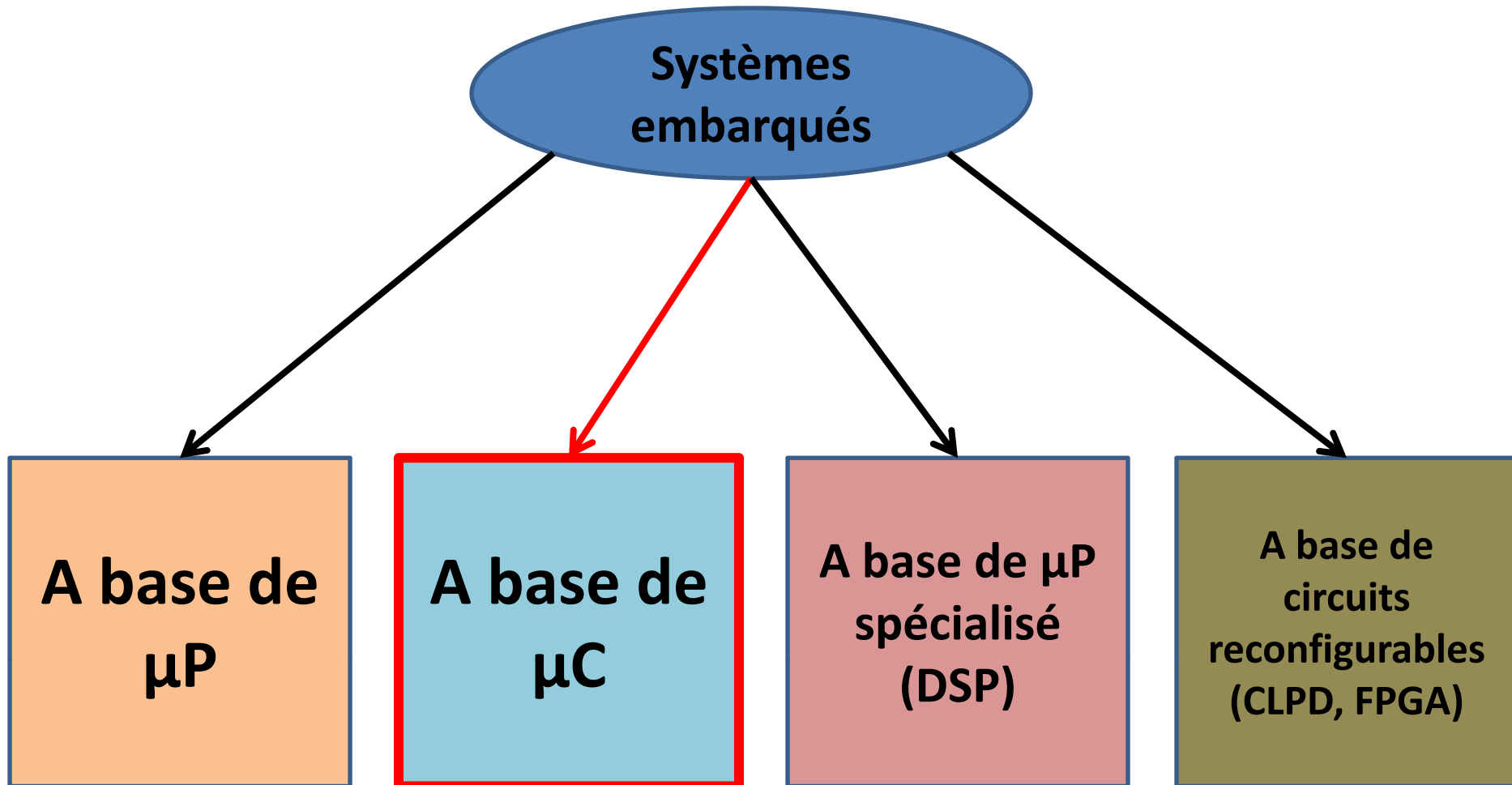
Plan du cours

- Systèmes Embarqués ?
- Types des systèmes embarqués selon l'architecture.
- Les principales contraintes liées au développement des systèmes embarqués.
- Du système à microprocesseur (μP) au Microcontrôleur (μC).
- Les principales caractéristiques des μC .
- Positionnement du problème.
- De la logique combinatoire vers La logique programmée.
 - Circuit de majorité.

Systèmes Embarqués?

- Un système embarqué est le résultat de l'union entre l'électronique et l'informatique pour réaliser un système de fonction ou tâche(s) bien précise comme :
 - Machine à café.
 - Système ABS
 - Télécommande
 - Téléphone portable
 - Imprimante
 - Jeux & gadgets
 - Etc.

Types des systèmes embarqués selon l'architecture

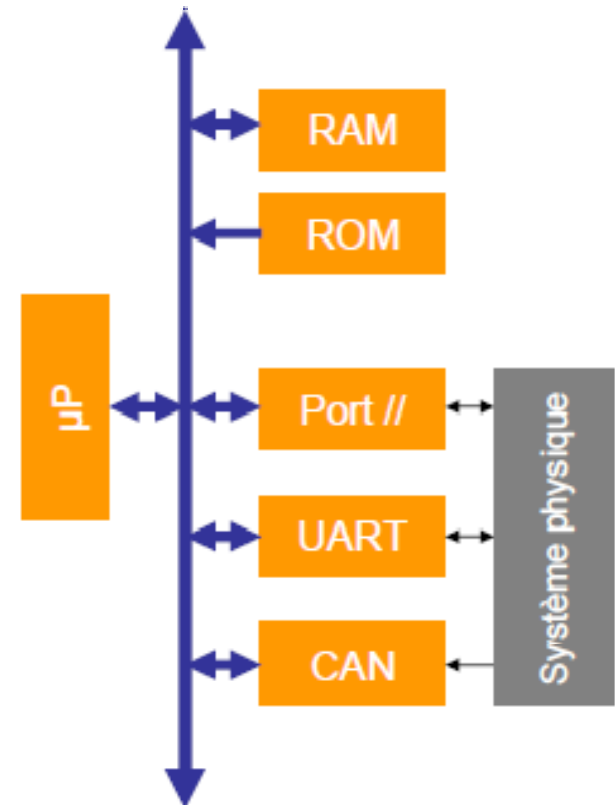


Les principales contraintes liées au développement des systèmes embarqués

- Puissance de calcul (MIPS).
- Taille de mémoire (CODE/DATA).
- Taille de bus (4, 8, 16,... bits).
- Consommation d'énergie/autonomie (mW/MIPS).
- Coût de développement (Soft/Hard).
- La durée de vie du système.
- Sûreté de fonctionnement (condition climatiques, position géographique).

Du système à microprocesseur (μP) au Microcontrôleur (μC)

- Un système à μP est composé de
 - Un μP (CPU).
 - Une mémoire CODE et DATA (ROM/RAM).
 - Quelques périphériques entrées-sorties.
 - Quelques périphériques de fonction précise (UART, USB, TIMER, ADC/DAC, CAN...).

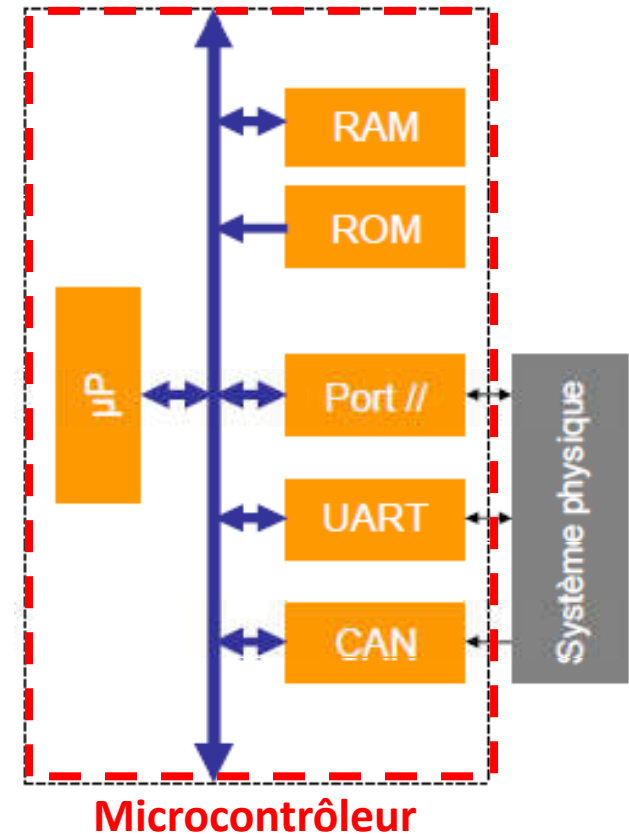


Du système à microprocesseur (μP) au Microcontrôleur (μC)

- En général, les systèmes à μP sont inventés pour résoudre des problèmes de grande complexité, mais comment faire avec les problèmes de moyenne et petite complexité ???

Du système à microprocesseur (μP) au Microcontrôleur (μC)

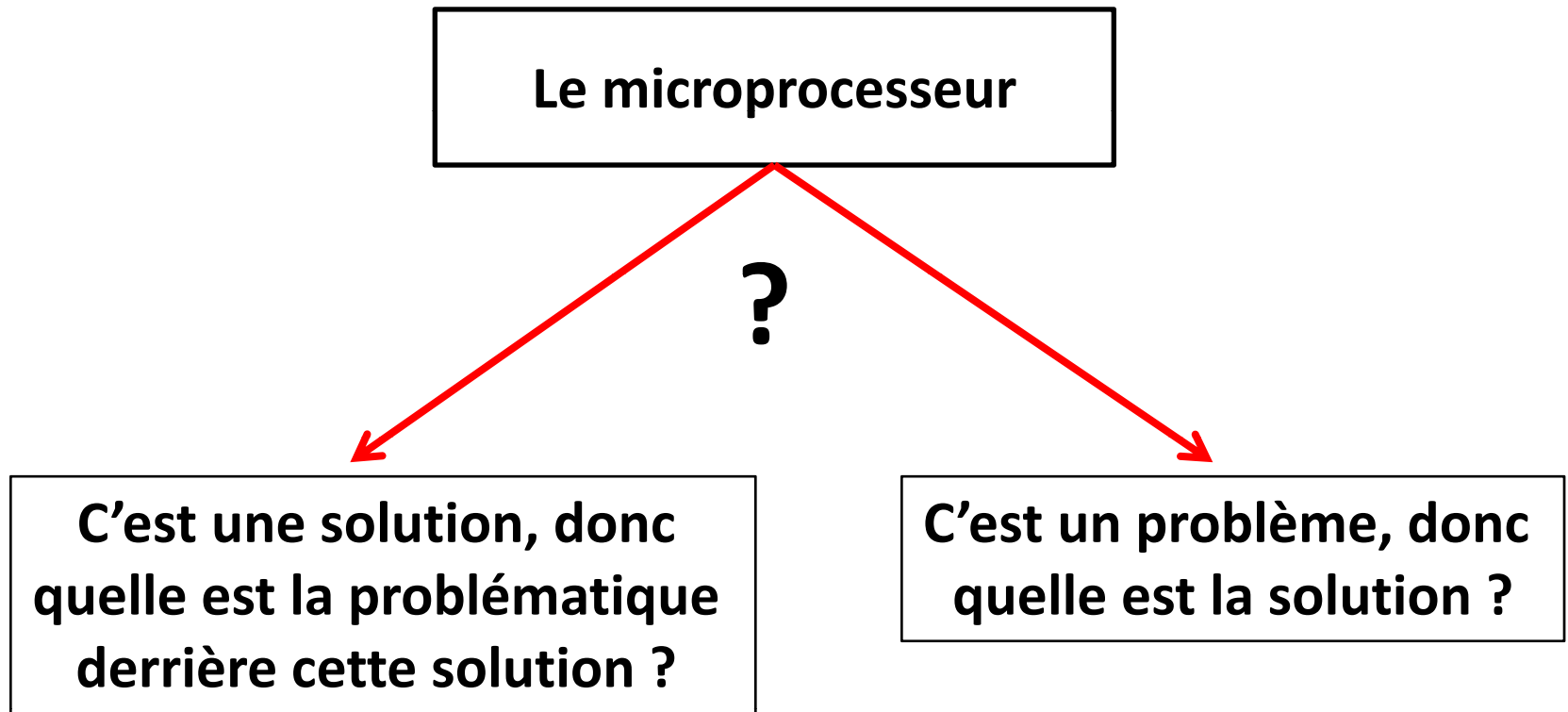
- La solution de notre problème est de réaliser des systèmes à μP de petite taille intégré dans un seul boîtier, qui s'appellent les **mono-chip**.
- Ce mono-chip est appelé dans l'industrie embarquée par le nom **Microcontrôleur** (μC).
- Les μC sont des versions **compactées** des micro-ordinateurs. Ils sont très utiles dans notre vie.
- Les plus célèbres μC sont les **PIC** chez **MICROCHIP**, le **8051/52** chez **Intel**, **68HC11/12** chez **Motorola**, le **AVR** chez **MICROCHIP/ATMEL**, et le **MSP430** chez **TI**.



Les principales caractéristiques des μ C

- Quelques ports I/O.
- Plusieurs périphériques de fonction spéciale (ADC, TIMER, USB, UART...).
- Une mémoire CODE de faible taille en général de type FLASH (512-256Ko).
- Une mémoire DATA de faible taille en général de type SRAM (32-32Ko).
- Une mémoire de type EEPROM pour sauvegarder des données en cas de coupure du courant (32-256 octet).
- Un processeur de 8-16 bits de faible puissance en calcul, ne dépasse pas quelques dizaine de MIPS.
- Une faible consommation électrique (standby/en marche).

Positionnement du problème



De la logique combinatoire vers La logique programmée

- Exemple : réalisation de circuit combinatoire qui fait le calcul de la fonction majorité de 3 bits (3 entrées).
- Pour résoudre ce genre de problème, il faut passer par la table de vérité.

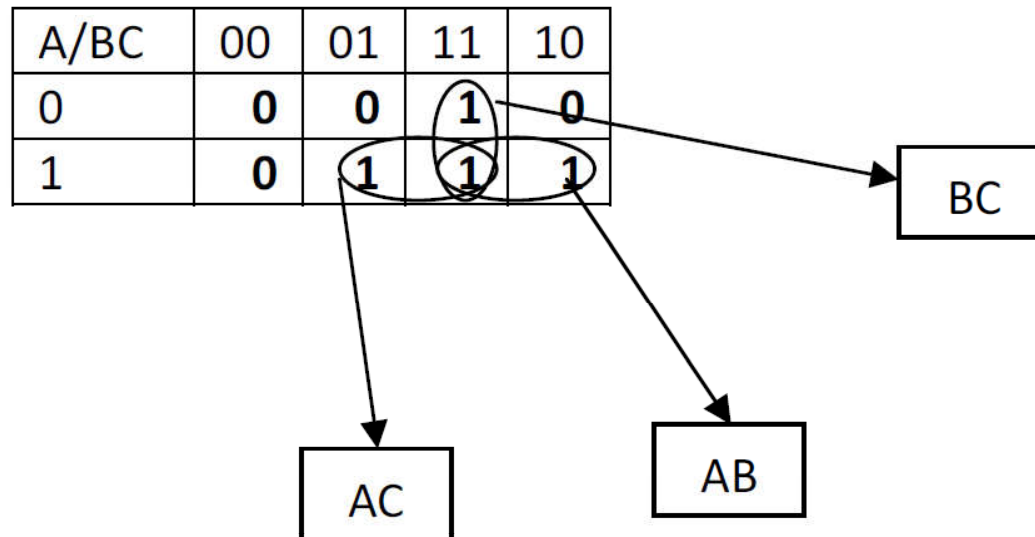
De la logique combinatoire vers La logique programmée

- Table de vérité de la fonction majorité :

A	B	C	Majorité
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

De la logique combinatoire vers La logique programmée

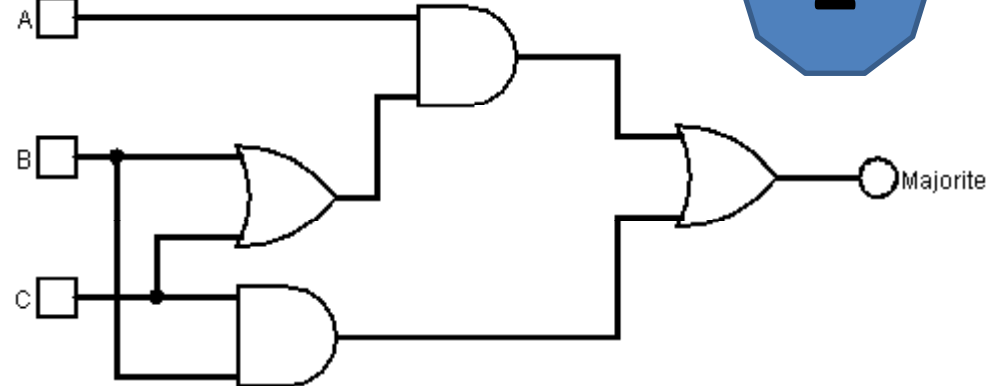
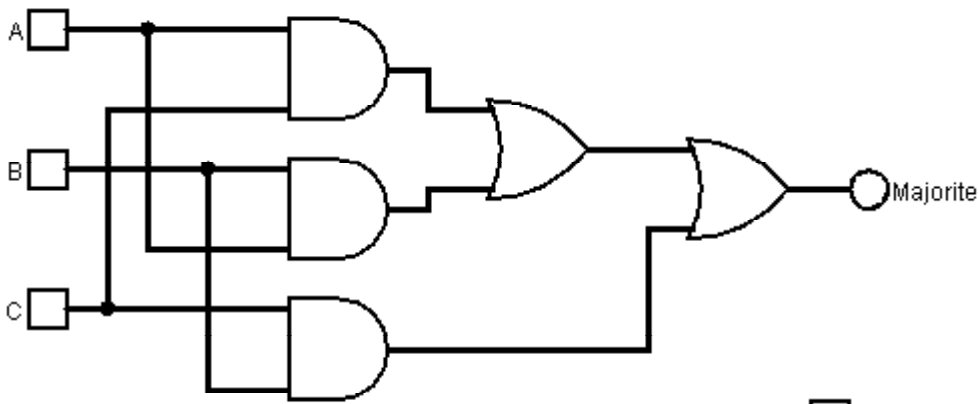
- L'étape suivante consiste à simplifier la fonction majorité par la table de KARNAUGH (K-MAP).



$$\text{MAJORITE} = AC + AB + BC = A(C+B) + BC$$

De la logique combinatoire vers La logique programmée

- Dernière étape est la réalisation de logigramme :

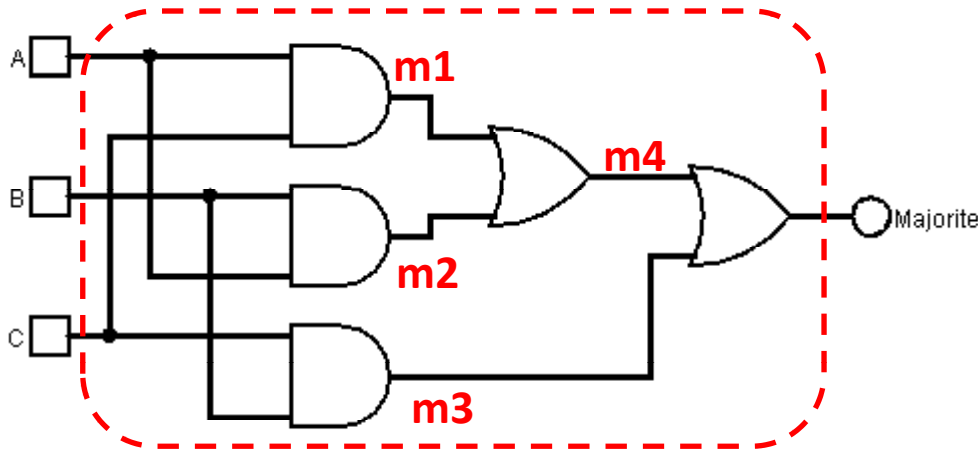


De la logique combinatoire vers La logique programmée

- La question qui se pose, est ce que on peut réaliser le logigramme de la fonction majorité par une **séquence d'opérations élémentaires**, l'une après l'autre (**Algorithme**) ?????

De la logique combinatoire vers La logique programmée

- Absolution, OUI.
- Algorithme de la forme 1 :



0- m1	←	A and C
1- m2	←	A and B
2- m4	←	m1 or m2
3- m3	←	B and C
4- M	←	m3 or m4

Bilan :

Variables internes : m1, m2, m3 et m4

Variables d'entrée : A, B et C

Variables de sortie : M

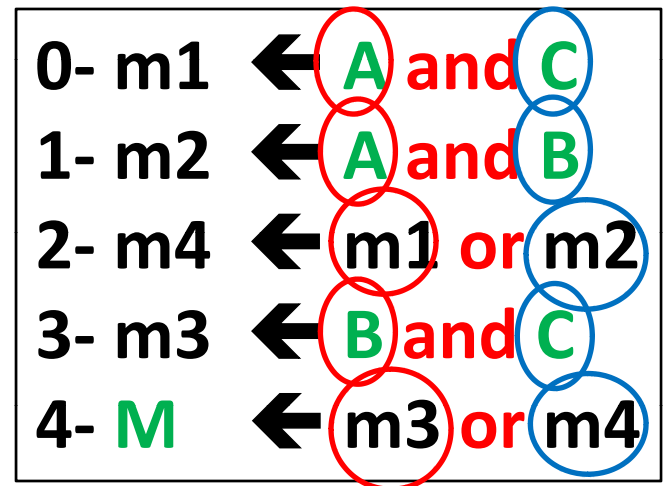
Nombre de lignes dans l'algo : 5

De la logique combinatoire vers La logique programmée

- Vous remarquez que chaque ligne de l'algorithme est composé par les éléments suivants :
 1. **2 opérandes** (source).
 2. **Une opération élémentaire** (OR et AND).
 3. **Résultat** (destination).
- Donc, proposer un chemin de données qui partage les éléments au-dessus sur toutes les lignes de l'algorithme.

De la logique combinatoire vers La logique programmée

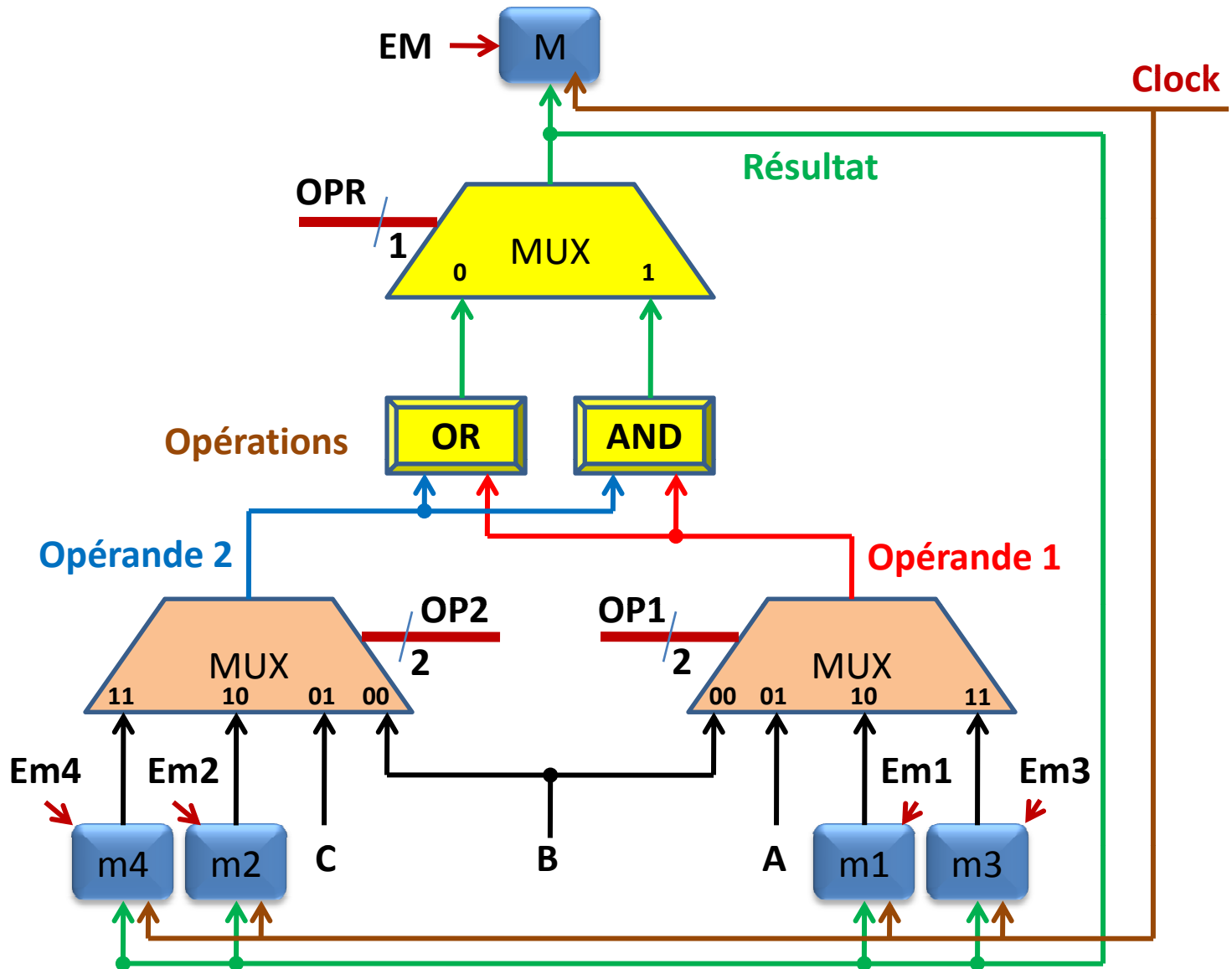
- Opérande 1 : est partagé entre **[A, B, m1, m3]**.
- Opérande 2 : est partagé entre **[C, B, m2, m4]**.
- Opération : est partagé entre **[and, or]**.
- Résultat : est partagé entre **[m1, m2, m3, m4, M]**.



De la logique combinatoire vers La logique programmée

- m1, m2, m3, m4 et M se sont des bascules (des cases mémoires de 1 bit chacune). En plus se sont des destinations dans une ligne bien précise; donc, ces bascules nécessitent un **signal de commande pour autoriser l'écriture**.
- L'élément électronique qui partage une ligne sur plusieurs lignes est le **MUX**.
- Finalement : le chemin de données est un ensemble de bascules avec signal de commande et des MUXs.
 - **Il reste de réaliser !!!???**

De la logique combinatoire vers La logique programmée



De la logique combinatoire vers La logique programmée

- Notre DATA-PATH contient un ensemble de signaux à commandé [**OP1, OP2, OPR, Em1, Em2, Em3, Em4, EM**].
- L'activation des signaux de commande se fait sur la lumière de l'algorithme.
- Donc, on dessine le tableau de commandes, dont chaque ligne présente une opération de l'algorithme et les colonnes sont les signaux à commandé.
- La taille de tableau de commandes = Nombre de ligne dans l'algorithme * Nombre de signaux à commandé.

De la logique combinatoire vers La logique programmée

- Tableau de commandes :

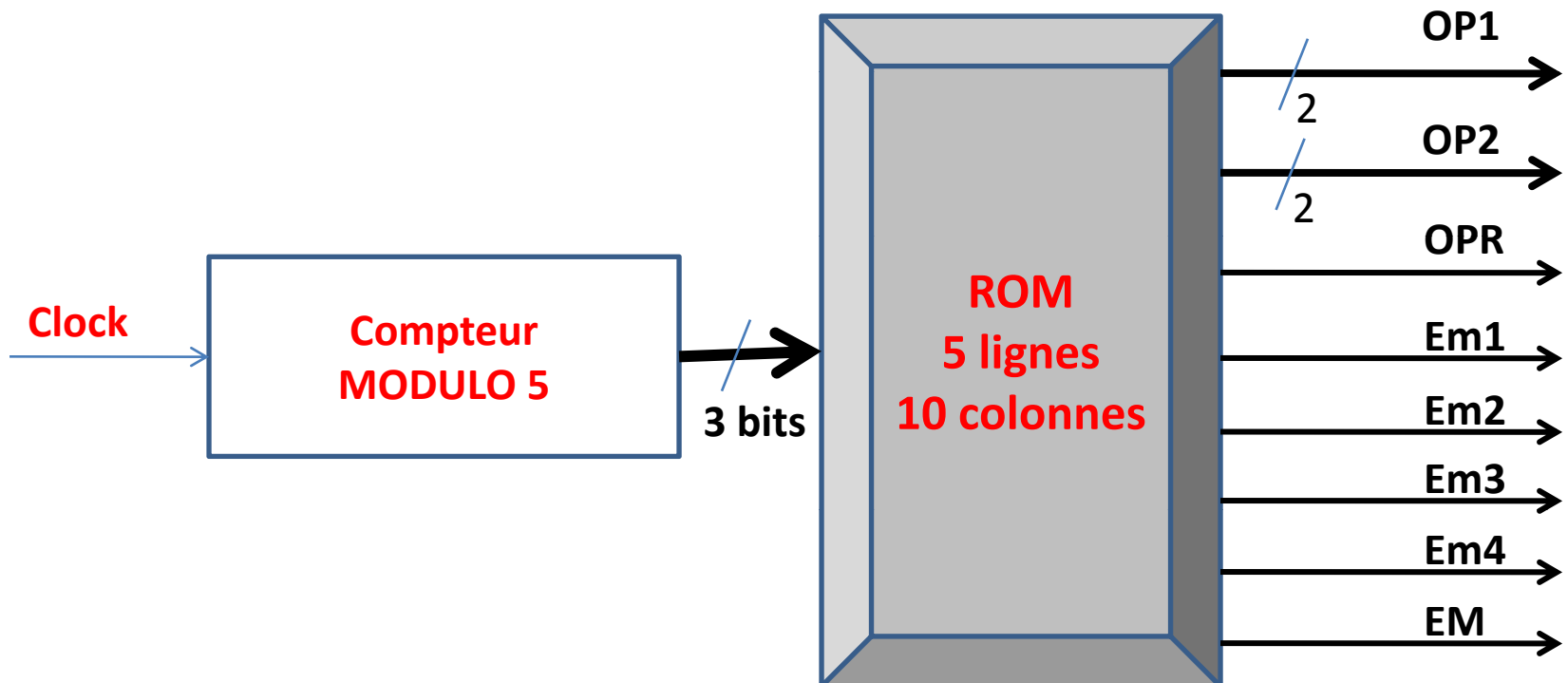
Ligne 0	Opération	Signaux de commandes									
		OP1		OP2		OPR	Em1	Em2	Em3	Em4	EM
0	m1 ← A and C	0	1	0	1	1	1	0	0	0	0
1	m2 ← A and B	0	1	0	0	1	0	1	0	0	0
2	m4 ← m1 or m2	1	0	1	0	0	0	0	0	1	0
3	m3 ← B and C	0	0	0	1	1	0	0	1	0	0
4	M ← m3 or m4	1	1	1	1	0	0	0	0	0	1

De la logique combinatoire vers La logique programmée

- La logique d'activation des signaux de commandes est appelée le **séquenceur**.
- Sur l'axe pratique notre séquenceur est la réalisation du tableau de commandes.
- A votre avis mes chers étudiants, comment réaliser ???

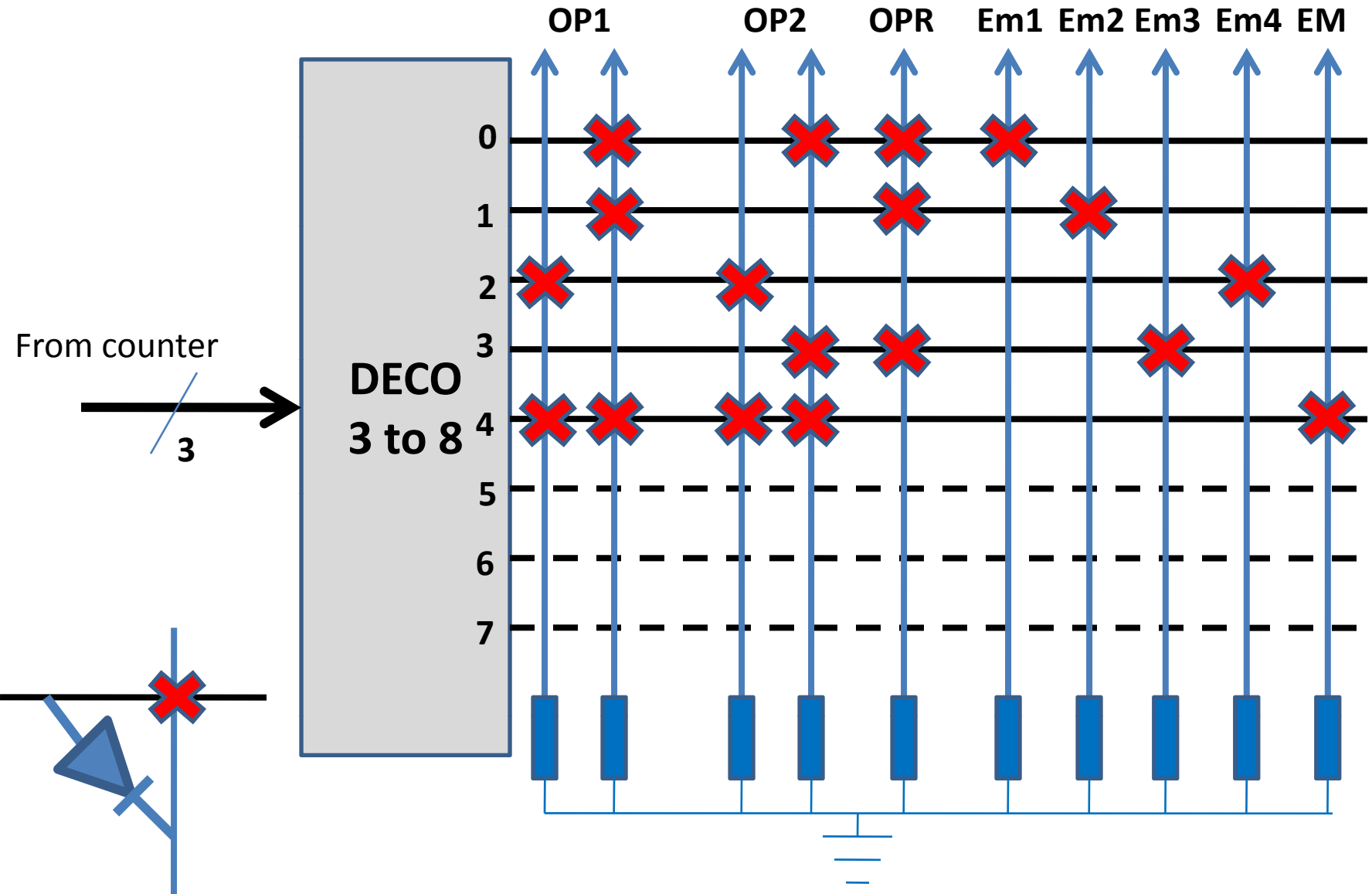
De la logique combinatoire vers La logique programmée

- Tout simplement par un compteur modulo 5 avec un circuit ROM à base de décodeur et réseau de diodes.
- La figure suivante présente la synoptique générale de notre séquenceur.



De la logique combinatoire vers La logique programmée

- Réalisation de la ROM :



De la logique combinatoire vers La logique programmée

- En conclusion la logique programmée de la fonction Majorité est un union entre un chemin de données (DATA-PATH) et un séquenceur cadencé par une horloge (Clock).