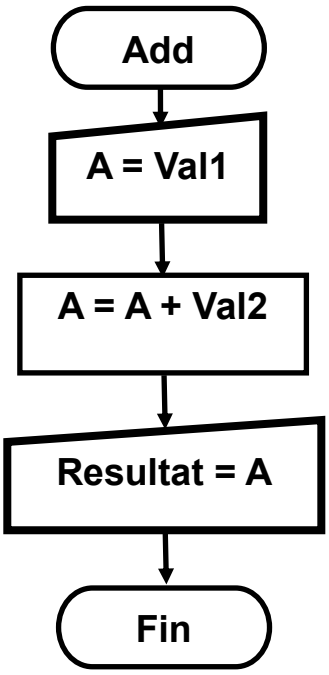
Correction TD-2

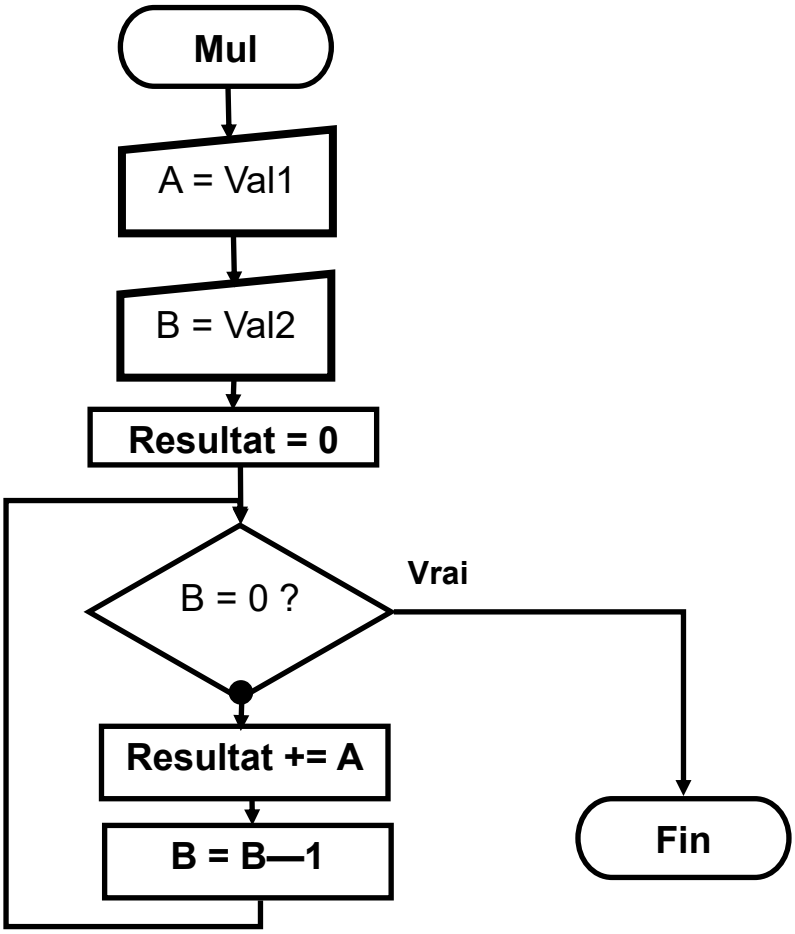
Exercice –1– Addition

Le microprocesseur ne travail que dans ses registres internes (ici A)



Exercice –2– Multiplication

On suppose que ce microprocesseur n'a pas d'instruction MUL dans sa bibliothèque

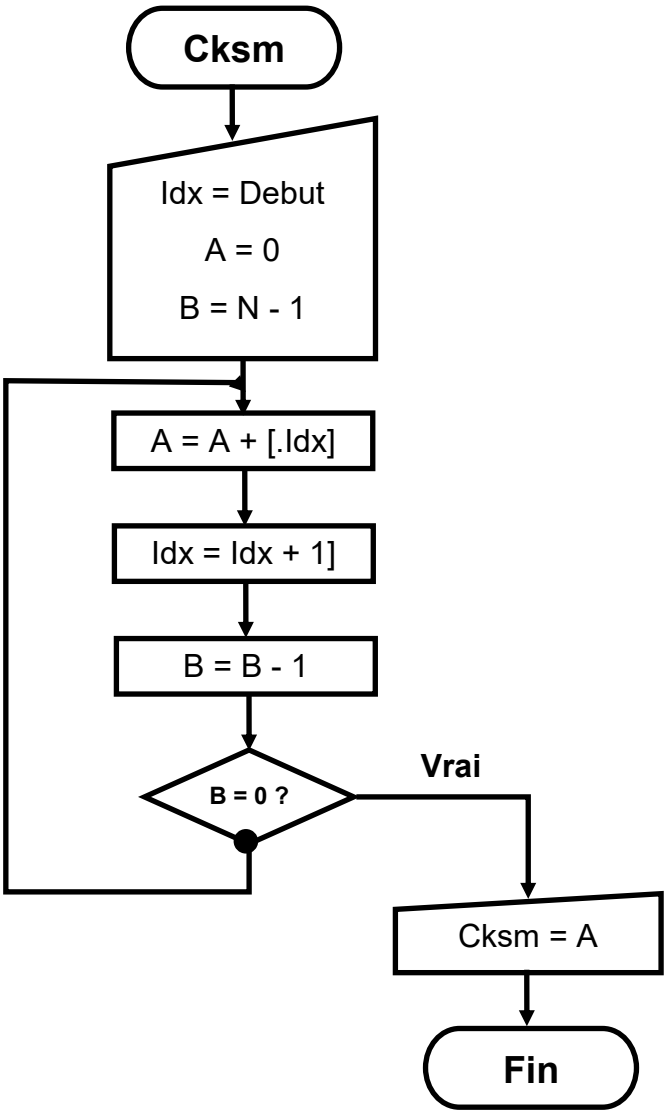


Exercice –3– Checksum

Lors d'une transmission d'une série de valeur on utilise un Checksum pour le contrôle en effectuant la somme d'un tableau de valeurs.

L'utilisation d'un index **Idx** sera nécessaire pour parcourir ce tableau

Debut	Val**[0]**	<— **Idx**
	Val[**1**]	
	Val[**2**]	
	...	
	Val[**N**]	



Exercice –4– Div16

Cet exercice est de démontrer l'utilisation d'algorithme en phase avec l'utilisation d'un microprocesseur:

En plus des opérations de base (addition, soustraction, lecture mémoire écriture mémoire, fonctions logiques) il possède l'opération de décalage bit à bit à droite ou à gauche.

En binaire quand on décale une fois à droite on divise par 2 si on décale 1 fois à gauche on multiplie par 2

Exemple:

0000 0110 = 6 (0x06) après décalage >> 1 0000 0011 = 3 (0x03)

0000 0101 = 5 (0x05) après décalage << 1 0000 1010 = 10 (0x0A)

Donc la division par 16 correspondra à 4 décalages à droite **>> 4**

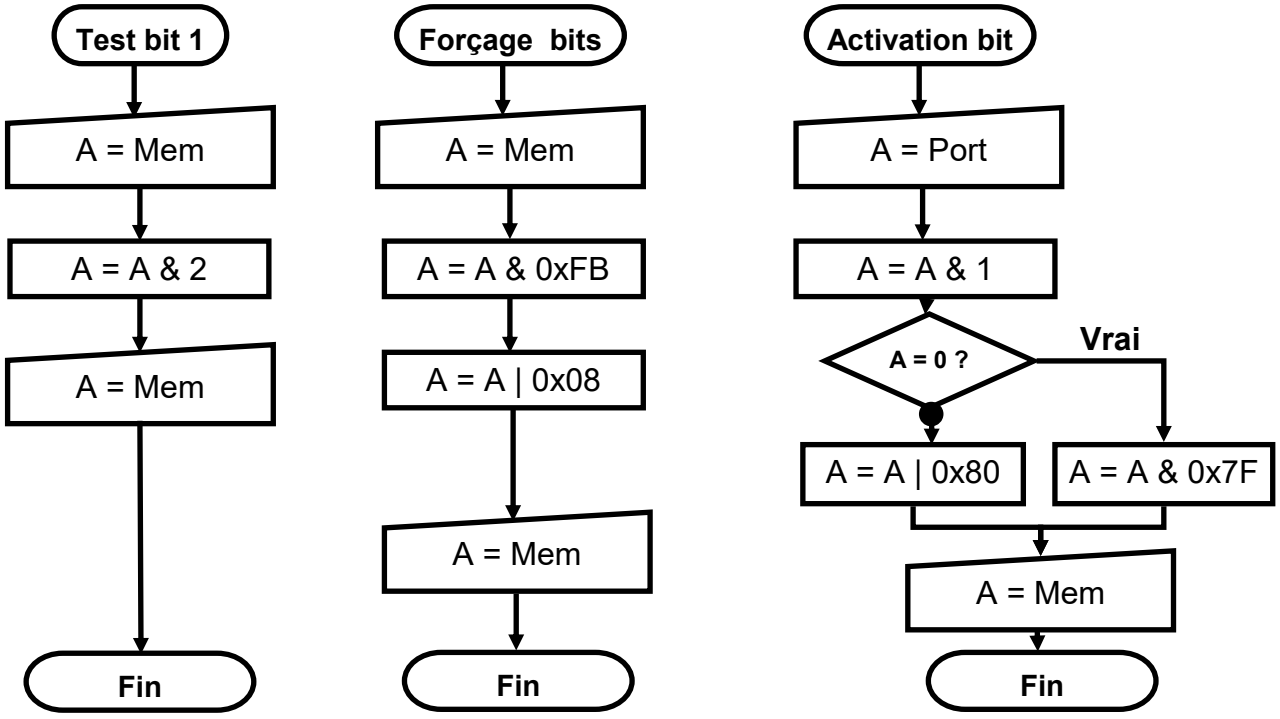
Exercice - 5 - 6 - 7 - Opérations logiques

Cet exercice permet d'utiliser les instructions logiques ET OU réalisables par un microprocesseur. Toujours dans un registre de travail interne on peut réaliser des opérations logiques « bit à bit » c'est-à-dire on effectue l'instruction logique avec chaque bit et son correspondant.

Exemple:

	0110 0111	0110 0010
ET	0000 0010	OU 0000 **1111**
	0000 0010	**0110 1111**

Un résultat à 0 correspond à l'état logique FAUX, et différent de « 0 » à VRAI



Exercice –8– Conversion de 2 Ascii vers Hexa sur 1 octet

Cet exercice permet de faire un sous programme en forme de fonction et de manipuler la table ASCII standard.

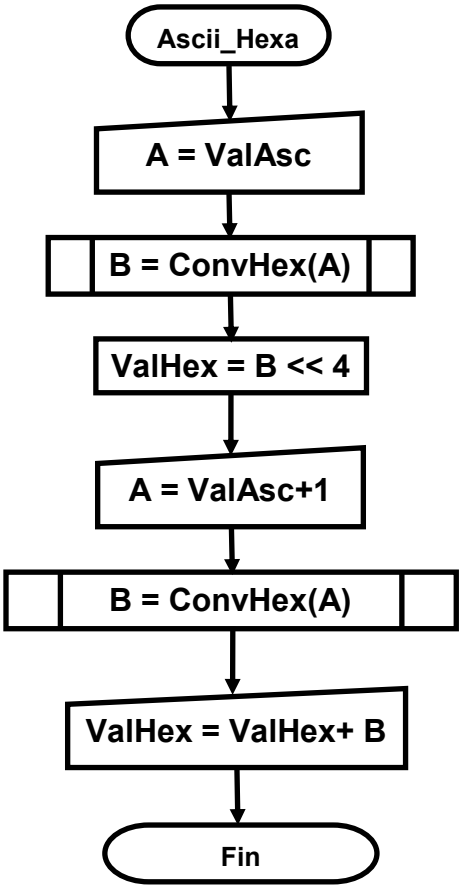
Les chiffres ‘0’ à ‘9’ vont de 0x30 à 0x39 et de ‘A’ à ‘F’ vont de 0x41 à 0x46

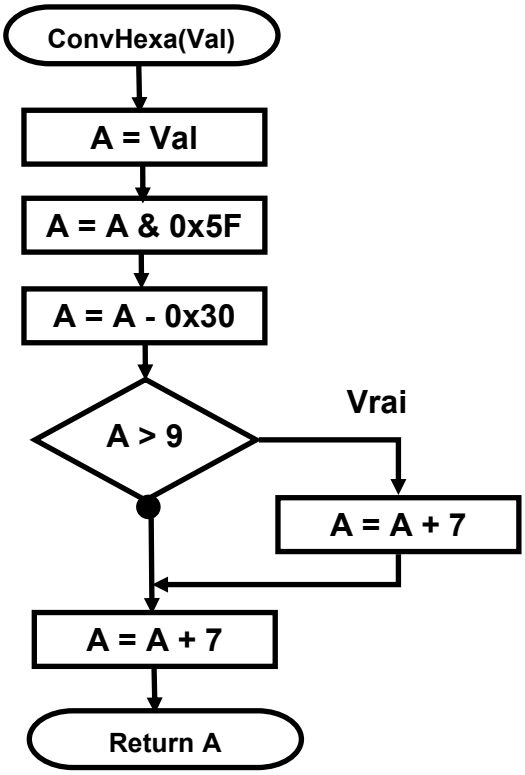
Notez qu’il y a 7 autres caractères entre ces 2 groupes.

LSB/ MSB	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	«	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	‘	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	.	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

ValAsc ValAsc+1 2 caractères Ascii hexadécimal

ValHex 1 octet en hexadécimal correspondant



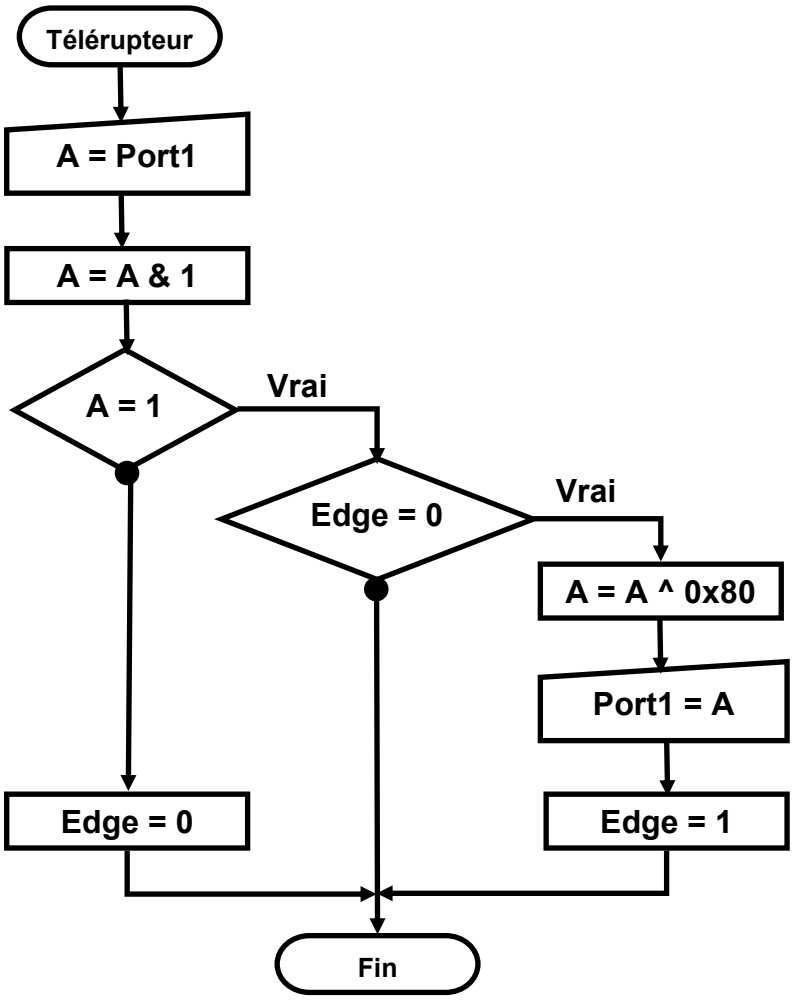


Exercice –9– Fonction télérupteur

Un télérupteur classique est une fonction qui permet d'allumer une lampe sur une pression d'un bouton poussoir, une deuxième pression éteint cette lampe.

Il faut donc détecter le changement d'état du bouton pour faire ces actions à cet instant et non sur l'état du bouton Marche ou Arrêt. On utilise une mémoire : « Edge »

La « Détection de front » est une fonctionnalité très courante en automatisme.



Exercice –10– Horloge numérique ou RTCC *(Real Time Clock Calender)*

Un microcontrôleur peut facilement déclencher périodiquement un programme particulier par l'intermédiaire d'un périphérique interne qui peut être programmé pour générer une interruption toutes les 100ms par exemple. Cette base de temps peut activer un système multitâche. Le but de cet exercice est de faire comprendre un raisonnement de type temporel, c'est-à-dire qui débute ici toutes les 100ms.