

## CARTE DE PROGRAMMATION

### MODÈLE DE PROGRAMMATION

R0
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14

**Registres généraux** à N bits (N=16 ou 32)  
contiennent donnée ou adresse

SP R15

**Stack Pointer** (en fait R15)  
pointe sur le dernier mot empilé

PC

**Programming Counter**  
pointe sur le mot suivant l'instruction

SR

**Status Register**  
indique l'état de la machine

### Mot mémoire pour N=16 bits d'adresse 2P

(la machine est "big endian")

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Octet 1 de poids fort (MSB)								Octet 0 de poids faible (LSB)							
Adresse = 2P								Adresse = 2P+1							

### Registre d'état SR :

Indicateurs (Flags)															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FLAGS	SIGNIFICATION	USAGE
ZF	Zero Flag	=1 ⇔ Le résultat de la dernière opération est nul
CF	Carry Flag	Retenue de la dernière opération arithmétique
NF	Negative Flag	Bit de signe (de gauche, msb, n°N-1) du résultat de la dernière opération
VF	oVerflow Flag	=1 ⇔ Le résultat de la dernière opération arithmétique a débordé l'intervalle autorisé en code complément à 2 pour sa taille.
IF	Interrupt Flag	=1 ⇔ Les interruptions sont autorisées
WF	Wait Flag	=1 ⇔ Machine arrêtée et attend une interruption

### GROUPE I : OPÉRATIONS À TROIS REGISTRES OPÉRANDES

1	OP3	CRsa	CRsb	CRd
---	-----	------	------	-----

CHAMP	SIGNIFICATION
OP3	Code d'opération triadique
CRsa	Code de registre source A (e.g. 1111 pour R15)
CRsb	Code de registre source B (e.g. 1010 pour R10)
CRd	Code de registre destination (e.g. 0000 pour R0)

**Actions générales** : Rsa **OP3** Rsb → Rd, nouveaux indicateurs → SR

OP3	MNÉMO- NIQUE3	SIGNIFICATION	ACTION PRINCIPALE	INDICATEURS				
				W	I	Z	V	C N
000	<b>ADC</b>	ADd Carry	Rsa + Rsb + CF → Rd			*	*	*
001	<b>XOR</b>	EXclusive OR	Rsa ∨ Rsb → Rd			*	0	0
010	<b>DIV</b>	DIVision signée & reste	Rsa ÷ Rsb → Rd Rsa % Rsb → Rsa			*	*	*
011	<b>MUL</b>	MULTiplication signée	Rsa × Rsb → Rd			*	*	*
100	<b>AND</b>	AND	Rsa ∧ Rsb → Rd			*	0	0
101	<b>OR</b>	OR	Rsa ∨ Rsb → Rd			*	0	0
110	<b>ADD</b>	ADD	Rsa + Rsb → Rd			*	*	*
111	<b>SUB</b>	SUBstract	Rsa + ¬Rsb + 1 → Rd			*	*	*

### GROUPE II : OPÉRATIONS À DEUX REGISTRES OPÉRANDES

0	1	0	0	OP2	CRs	CRd
---	---	---	---	-----	-----	-----

CHAMP	SIGNIFICATION
OP2	code d'opération diadique
CRs	Code registre source
CRd	Code registre destination

**Actions générales**: Op2(Rs) → Rd, nouveaux indicateurs → SR

OP2	MNÉMO- NIQUE2	SIGNIFICATION	ACTION PRINCIPALE	INDICATEURS				
				W	I	Z	V	C N
0000	<b>RLC</b>	Rotate Left through Carry	(Rs << 1) + CF → Rd			*	0	*
0001	<b>RRC</b>	Rotate Right through Carry	(Rs >> 1) + (CF << (N-1)) → Rd			*	0	*
0010	<b>SRL</b>	Shift Right Logical	Rs >> 1 → Rd			*	0	*
0011	<b>SRA</b>	Shift Right Arithmetic	Rs / 2 → Rd			*	0	*
0100	<b>NOT</b>	Not	¬Rs → Rd			*	0	0
0101	<b>SBB</b>	SuBstract Borrow	Rs + ¬CF + 11...11 → Rd			*	*	*
0110	<b>SHL</b>	SHift Left	Rs << 1 → Rd			*	0	*
0111	<b>NEG</b>	NEGate	¬Rs + 1 → Rd			*	*	*
1000	<b>INP</b>	INPUt data	IO[Rs] → Rd			*	0	0
1001	<b>OUT</b>	OUTput data	Rs → IO[Rd]			*	0	0
1010	<b>SWB</b>	SWap Bytes	Rs.LSByte → Rd.MSByte, Rs.MSByte → Rd.LSByte			*	0	0
1011	<b>RLB*</b>	Rotate Left Bytes	Rs.LSWord → Rd.MSWord, Rs.MSWord → Rd.LSWord			*	0	0
1100	<b>ANI</b>	ANd Immediate	Rs ∧ IE → Rd			*	0	0
1101	<b>EXT*</b>	EXTend sign	Extend(Rs) → Rd			*	0	0
1110	<b>ADI</b>	ADd Immediate	Rs + IE → Rd			*	*	*
1111	<b>CMP</b>	CoMPare	Rs + ¬Rd + 1			*	*	*

### Groupe III: Transferts

0	1	Type	CRa	D	ModeB	CRb
---	---	------	-----	---	-------	-----

CHAMP	SIGNIFICATION
Type	Code de taille d'opérande
D	Direction du transfert
ModeB	Code du Mode d'adressage de l'opérande B
CRa	Code du registre opérande A
CRb	Code du registre B

**Note** : L'opérande B est indiqué par le registre Rb en utilisant le mode d'adressage ModeB. L'opérande A est le contenu du registre Ra.

TYPE	MNÉMO- NIQUE	NOM	TAILLE EN OCTETS	NOTES
01	<b>B</b>	Byte	1	
10	<b>W</b>	Word	2	
11	<b>L</b>	Long word	4	Pour N=32 bits

D	MNÉMO- NIQUE	SIGNIFI- CATION	ACTION PRINCIPALE	INDICATEURS				
				W	I	Z	V	C N
0	<b>ST</b>	STore	Ra → Opérande B			*	0	0
1	<b>LD</b>	LoaD	Ra ← Opérande B			*	0	0

MODEB	NOM	EA	OPÉRANDE B
000	Immédiat	PC	IE
001	Registre		Opérande B = Rb
010	Indirect	Rb	Opérande B = M[Rb]
011	Indirect- Post-incrémenté	Rb	Opérande B = M[Rb]; Rb ← Rb + taille(type)
100	Indirect- Pré-décrémenté	Rb-taille(type)	Rb ← Rb - taille(type); Opérande B = M[Rb]
101	Direct	IE	M[IE]
110	Indexé	IE+Rb	M[Rb+IE]
111	Indirect- pré-indexé	M[IE+Rb]	M[M[Rb+IE]]

CHAMP	SIGNIFICATION
CC	Code de Condition
Mode	Code du Mode d'adressage de l'opérande déplacement
CR	Code du Registre de l'opérande déplacement

**Action :** Condition vérifiée  $\Rightarrow$  PC  $\leftarrow$  PC + déplacement  
Sinon : PC pointe sur instruction suivante

CODE CONDITION	MNÉMO CONDITION	SIGNIFICATION	CONDITION SUR LES INDICATEURS
0001	<b>MP AL</b>	no condition ALways	1
0010	<b>EQ</b>	EQual	ZF
0011	<b>NE</b>	Not Equal	$\neg ZF$
0100	<b>GE</b>	Greater or Equal	$\neg(NF \vee VF)$
0101	<b>LE</b>	Lower or Equal	$(NF \vee VF) \vee ZF$
0110	<b>GT</b>	GreaTer	$\neg(NF \vee VF) \wedge \neg ZF$
0111	<b>LW</b>	LoWer	$NF \vee VF$
1000	<b>AE CC</b>	Above or Equal, Carry Cleared	$\neg CF$
1001	<b>BE</b>	Below or Equal	$CF \vee ZF$
1010	<b>AB</b>	ABove	$\neg CF \wedge \neg ZF$
1011	<b>BL CS</b>	BeLow, Carry Set	CF
1100	<b>VS</b>	oVerflow Set	VF
1101	<b>VC</b>	oVerflow Cleared	$\neg VF$

CHAMP	SIGNIFICATION
OP1	Code d'opération à 1 opérande
Mode	Code du Mode d'adressage de l' <b>opérande A</b>
CR	Code du registre R déterminant A selon le Mode

OP1	MNÉMO	SIGNIFICATION	ACTIONS
000	<b>JPA</b>	Jump Absolute saut inconditionnel absolu long	$PC \leftarrow A$
001	<b>JEA</b>	Jump to Effective Address saut inconditionnel absolu long	$PC \leftarrow EA$
010	<b>JSR</b>	Jump to SubRoutine	$SP \leftarrow SP - T$ ; $M[SP] \leftarrow PC$ ; $PC \leftarrow EA$
011	<b>TRP</b>	TRaP Trappe programmée: <i>Appelle une fonction système dont le n°n est indiqué par l'opérande A</i>	$SP \leftarrow SP - T$ ; $M[SP] \leftarrow SR$ ; $SP \leftarrow SP - T$ ; $M[SP] \leftarrow PC$ ; $PC \leftarrow M[4 \times A] \wedge 11..1L0$
100	<b>TST</b>	TeST	$CF \leftarrow 0$ , $VF \leftarrow 0$ , $ZF \leftarrow is\_zero(A)$ , $NF \leftarrow A_{N-1}$
101	<b>TSR</b>	TeSt and Reset	$CF \leftarrow 0$ , $VF \leftarrow 0$ , $ZF \leftarrow is\_zero(A)$ , $NF \leftarrow A_{N-1}$ ; $A \leftarrow 0$
110	<b>MSR</b>	Move Status Register	$A \leftarrow SR$
111	<b>MPC</b>	Move Program Counter	$A \leftarrow PC$

NOTATION	SIGNIFICATION
IE	"Instruction extension"; mot qui suit l'instruction.
EA	" Effective Address "; adresse de l'opérande
T          N	Taille du mot CPU en octets          idem en bits
M[A]	Case mémoire d'adresse A
IO[A]	Port d'entrée-sortie numéro A
R2 <sub>n</sub>	Le bit n° n de R2 (bit de droite poids faible = R2 <sub>0</sub> )
R1←R2 ou R2→R1	Le registre R1 est chargé avec le contenu de R2
a1 , a2 ; a3	action a1 <b>simultanée</b> à action a2 <b>puis</b> action a3
X << 2      X >> 2	décalage de X de 2 bits à gauche; idem à droite
A / B      A % B	quotient de A divisé par B ; reste de A sur B
+          ×	addition <b>fixée sur N bits</b> multiplication
¬      ^      v      ∨	opérateurs <b>bit à bit</b> : NOT, OR, AND, XOR

CHAMP	SIGNIFICATION
OP0	Code d'opération sans opérande

OP0	MNÉMO0	SIGNIFICATION	ACTIONS
000	<b>NOP</b>	No-OPeration	Aucune action, sauf : $PC \leftarrow PC + 2$
001	<b>HLT</b>	HaLT	Arrête et attend une interruption matérielle: $WF \leftarrow 1$
010	<b>RTS</b>	ReTurn from Subroutine	$PC \leftarrow M[SP]$ ; $SP \leftarrow SP + T$
011	<b>RTI</b>	ReTurn from Interrupt	$PC \leftarrow M[SP]$ ; $SP \leftarrow SP + T$ ; $SR \leftarrow M[SP]$ ; $SP \leftarrow SP + T$
100	<b>CLC</b>	CLear Carry	$CF \leftarrow 0$
101	<b>STC</b>	SeT Carry	$CF \leftarrow 1$
110	<b>DSI</b>	DiSable Interrupts	inhibe les interruptions: $IF \leftarrow 0$
111	<b>ENI</b>	Enable Interrupts	valide les interruptions: $IF \leftarrow 1$

CHAMP	SIGNIFICATION
OPQ	Code d'opération rapide
ValueCode	Code complément à 2 de la valeur sur 8 bits (valeur de -128 à 127)
CR	Code du registre destination

OPQ	MNÉMO- NIQUE	SIGNIFI- CATION	ACTION	INDICATEURS					
				W	I	Z	V	C	N
0	<b>LDQ</b>	LoaD Quick	$R \leftarrow \text{ext}(\text{ValueCode})$			*	0	0	*
1	<b>ADQ</b>	ADd Quick	$R \leftarrow R + \text{ext}(\text{ValueCode})$			*	*	*	*

0	0	0	1	CC	ValueCode
---	---	---	---	----	-----------

CHAMP	SIGNIFICATION
CC	Code de condition
ValueCode	Code complément à 2 du déplacement sur 8 bits. (déplacement de -128 à 127)

## EXCEPTIONS MATÉRIELLES

NOM	SIGNIFICATION	DÉCLENCHEMENT	ACTIONS
Reset	initialisation	ligne /RST=0	Efface SR; inhibe les exceptions ; Lance programme adresse démarrage
Interrupt	interruption sur ligne n°i	ligne /IRQi ↓ (périphérique)	<b>Termine</b> l'instruction en cours; empile SR; inhibe les exceptions ; appelle le programme de vecteur n° INT = f(i) = i + 32 . <b>Exécutera instruction suivante après</b>
Trap	trappe n°n	CPU	Empile SR; inhibe les exceptions ; appelle le programme de vecteur n°n <b>Exécutera instruction suivante après</b>
Fault	faute n°n	CPU	<b>Ne finit pas</b> l'instruction en cours; empile SR; inhibe les exceptions ; appelle programme de vecteur n°n . <b>Réexécutera instruct. en cours après.</b>
Error	erreur n°n	CPU	<b>Arrête</b> l'instruction en cours; inhibe les exceptions ; appelle programme de vecteur n°n . <b>Initialisera le CPU après.</b>

- **adresse de démarrage** = FFFAh *pour N=16*
- **n° vecteur d'exception** = INT = n° ligne de requête d'interruption + 32
- **adresse du vecteur d'exception** n°INT = 4 x INT *pour N=16 et 32*
- **adresse du programme d'exception** = vecteur  $\wedge$  11...110 *pour N=16*