

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

(le CPU 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 :

15	14	13	12	11	10	9	8	7	6	Indicateurs (Flags)					
0	0	0	0	0	0	0	0	0	0	WF	IF	ZF	VF	CF	NF

FLAGS	SIGNIFICATION	USAGE
ZF	Zero Flag	=1 ⇔ Le résultat de la dernière instruction est nul
CF	Carry Flag	Retenue de la dernière instruction (0 si pas retenue)
NF	Negative Flag	Bit de signe (de gauche, MSB, n°N-1) du résultat de la dernière instruction.
VF	oVerflow Flag	=1 ⇔ Le résultat de la dernière instruction a débordé de l'intervalle autorisé pour un entier signé en code complément à 2 : $[-2^{N-1} .. 2^{N-1}-1]$
IF	Interrupt Flag	=1 ⇒ Les interruptions sont validées; 0 ⇒ inhibées
WF	Wait Flag	=1 ⇔ Le CPU est arrêté et attend une interruption

GROUPE I : OPÉRATIONS À TROIS REGISTRES OPÉRANDES

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	OP3				CRsa				CRsb				CRd		

CHAMP	SIGNIFICATION
OP3	Code d'opération à trois opérands
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	ADC	ADd Carry	$Rsa + Rsb + CF \rightarrow Rd$		*	*	*	*	*
001	XOR	EXclusive OR	$Rsa \vee Rsb \rightarrow Rd$		*	0	0	*	*
010	DIV	DIVision <i>signée & reste</i>	$Rsa \div Rsb \rightarrow Rd$ $Rsa \% Rsb \rightarrow Rsa$		*	*	*	*	*
011	MUL	MULTiplication <i>signée</i>	$Rsa \times Rsb \rightarrow Rd$		*	*	*	*	*
100	AND	AND	$Rsa \wedge Rsb \rightarrow Rd$		*	0	0	*	*
101	OR	OR	$Rsa \vee Rsb \rightarrow Rd$		*	0	0	*	*
110	ADD	ADD	$Rsa + Rsb \rightarrow Rd$		*	*	*	*	*
111	SUB	SUBtract	$Rsa + \neg Rsb + 1 \rightarrow Rd$		*	*	*	*	*

GROUPE II : OPÉRATIONS À DEUX REGISTRES OPÉRANDES

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	OP2				CRs				CRd			

CHAMP	SIGNIFICATION
OP2	code d'opération à eux opérands
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	RLC	Rotate Left through Carry	$(Rs \ll 1) + CF \rightarrow Rd$		*	0	*	*	*
0001	RRC	Rotate Right through Carry	$(Rs \gg 1) + (CF \ll (N-1)) \rightarrow Rd$		*	0	*	*	*
0010	SRL	Shift Right Logical	$Rs \gg 1 \rightarrow Rd$		*	0	*	*	*
0011	SRA	Shift Right Arithmetic	$Rs / 2 \rightarrow Rd$		*	0	*	*	*
0100	NOT	Not	$\neg Rs \rightarrow Rd$		*	0	0	*	*
0101	SBB	SuBtract Borrow	$Rs + \neg CF + 11...11 \rightarrow Rd$		*	*	*	*	*
0110	SHL	SHift Left	$Rs \ll 1 \rightarrow Rd$		*	0	*	*	*
0111	NEG	NEGate	$\neg Rs + 1 \rightarrow Rd$		*	*	*	*	*
1000	INP	INPut data	$IO[Rs] \rightarrow Rd$		*	0	0	*	*
1001	OUT	OUTput data	$Rs \rightarrow IO[Rd]$		*	0	0	*	*
1010	SWB	SWap Bytes	$Rs.LSByte \rightarrow Rd.MSByte$ $Rs.MSByte \rightarrow Rd.LSByte$		*	0	0	*	*
1011	RLB*	Rotate Left Barrel	rotation de 4 bits à gauche		*	0	0	*	*
1100	ANI	ANd Immediate	$Rs \wedge IE \rightarrow Rd$		*	0	0	*	*
1101	EXT*	EXTend sign	$extend(Rs.LSByte) \rightarrow Rd$		*	0	0	*	*
1110	ADI	ADd Immediate	$Rs + IE \rightarrow Rd$		*	*	*	*	*
1111	CMP	CoMPare	$Rs + \neg Rd + 1$		*	*	*	*	*

Groupe III: Transferts

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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; 0000 si direct ou immédiat

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

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

D	MNÉMO- NIQUE	SIGNIFI- CATION	ACTION PRINCIPALE	INDICATEURS					
				W	I	Z	V	C	N
0	ST	STore	$Ra \rightarrow \text{Opérande B}$		*	0	0	*	*
1	LD	LoaD	$Ra \leftarrow \text{Opérande B}$		*	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]; $Rb \leftarrow Rb + \text{taille}(\text{type})$
100	Indirect- Pré-décrémenté	$Rb - \text{taille}(\text{type})$	$Rb \leftarrow Rb - \text{taille}(\text{type});$ 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]]

GROUPE IV : SAUT RELATIF LONG JCC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0					1							

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

Jump if Condition : Saut à une instruction si la condition est vérifiée.

Action principale: Condition vérifiée $\Rightarrow PC \leftarrow PC + \text{déplacement}$

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

GROUPE V : INSTRUCTIONS À UN REGISTRE OPÉRANDE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1											

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

OP1	MNÉMO	SIGNIFICATION	ACTIONS
001	JEA	Jump to Effective Address <i>saut inconditionnel absolu long (saute à l'instruction opérande)</i>	$PC \leftarrow EA$
010	JSR	Jump to SubRoutine	$SP \leftarrow SP - T;$ $M[SP] \leftarrow PC; PC \leftarrow EA$
011	TRP	TRaP Trappe logicielle: <i>Appelle une fonction système dont le n° est indiqué par l'opérande A</i>	$SP \leftarrow SP - T;$ $M[SP] \leftarrow SR;$ $SP \leftarrow SP - T;$ $M[SP] \leftarrow PC;$ $PC \leftarrow M[4 \times A] \wedge 11..10$
100	TST	TeST	$CF \leftarrow 0, VF \leftarrow 0,$ $ZF \leftarrow is_zero(A), NF \leftarrow A_{N-1}$
101	TSR	TeSt and Reset	$CF \leftarrow 0, VF \leftarrow 0,$ $ZF \leftarrow is_zero(A), NF \leftarrow A_{N-1};$ $A \leftarrow 0$
110	MSR	Move Status Register	$A \leftarrow SR$
111	MPC	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 _n	Le bit n°n de R2 (bit de droite poids faible = R2 ₀)
R1 ← R2 ou R2 → R1	Le registre R1 est chargé avec le contenu de R2
a1, a2; a3	action a1 <i>simultanée</i> à action a2 <i>puis</i> action a3
X << 2 X >> 2	X décalé de de 2 bits à gauche; idem à droite
A / B A % B	quotient de A divisé par B ; reste de A sur B
+ ×	addition <i>fixée sur N bits</i> ; multiplication
¬ ∧ ∨ ∇	opérateurs bit à bit: NOT, OR, AND, XOR

GROUPE VI : INSTRUCTIONS SANS REGISTRE OPÉRANDE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0											

CHAMP	SIGNIFICATION
OP0	Code d'opération sans opérande

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

GROUPE VII : OPÉRATIONS RAPIDES

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	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	LDQ	LoaD Quick	$R \leftarrow \text{ext}(\text{ValueCode})$			*	0	0	*
1	ADQ	Add Quick	$R \leftarrow R + \text{ext}(\text{ValueCode})$			*	*	*	*

GROUPE VIII: BRANCHEMENT RELATIF COURT BCC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1												

CHAMP	SIGNIFICATION
CC	Code de la Condition de branchement
DispCode	Code complément à 2 du déplacement sur 8 bits. (déplacement de -128 à 127)

Branch if Condition: saute à une instruction si la condition est vérifiée

Action principale : Condition vérifiée $\Rightarrow PC \leftarrow PC + \text{déplacement}$

EXCEPTIONS MATÉRIELLES

NOM	SIGNIFI- CATION	DÉCLEN- CHEMENT	ACTIONS
Reset	initialisation	ligne /RST=0	Efface SR; inhibe les exceptions ; Lance programme adresse démarrage.
Interrupt	interruption sur ligne n°1	ligne /IRQi ↓ (périphérique)	<i>Termine</i> l'instruction en cours; empile SR; inhibe les exceptions ; appelle le programme de vecteur n°n = i + 32 . <i>Exécutera instruction suivante après</i>
Trap	trappe n°h	CPU	Empile SR; inhibe les exceptions ; appelle le programme de vecteur n°h <i>Exécutera instruction suivante après</i>
Fault	faute n°h	CPU	<i>Ne finit pas</i> l'instruction en cours; empile SR; inhibe les exceptions ; appelle programme de vecteur n°h . <i>Réexécutera instruct. en cours après.</i>
Error	erreur n°h	CPU	<i>Arrête</i> l'instruction en cours; inhibe les exceptions ; appelle programme de vecteur n°h . <i>Initialisera le CPU après.</i>

- 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