UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
ELE0517 - SISTEMAS DIGITAIS - T01
DISCENTES: LEVY GABRIEL E NICHOLAS MEDEIROS

Documentação de um microcontrolador de 8 bits (Nickel)

1. SUMÁRIO DO CONJUNTO DE INSTRUÇÕES

Mnemônica	Operandos	Descrição	Operação	Flags	#Clk
ADD	RD, RS1, RS2	Soma dois registradores	RD = RS1 + RS2	Z, S, C, V	4
SUB	RD, RS1, RS2	Subtrai dois registradores	RD = RS1 - RS2	Z, S, C, V	4
AND	RD, RS1, RS2	AND entre bits dos registradores	RD = RS1 x RS2	Z, S, V	4
OR	RD, RS1, RS2	OR entre bits dos registradores	RD = RS1 ^ RS2	Z, S, V	4
XOR	RD, RS1, RS2	XOR entre bits dos registradores	RD = RS1 ⊕ RS2	Z, S, V	4
NAND	RD, RS1, RS2	NAND entre bits dos registradores	RD = !(RS1 ^ RS2)	Z, S, V	4
NOR	RD, RS1, RS2	NOR entre bits dos registradores	RD = !(RS1 x RS2)	Z, S, V	4
NXOR	RD, RS1, RS2	NXOR entre bits dos registradores	RD = RS1 ⊙ RS2	Z, S, V	4
RTR	RD, RS1, RS2	Desvio à direita recuperando bit perdido	RD[6:0, 7] = RS1[7:1, 0] >> RS2	Z, S, V	4
RTL	RD, RS1, RS2	Desvio à esquerda recuperando bit perdido	RD[7:1, 0] = RS1[6:0, 7] << RS2	Z, S, V	4
SLL	RD, RS1, RS2	Desvio à esquerda	RD[7:1] = RS1[6:0] << RS2, RD[0] = 0	Z, S, C, V	4
SRL	RD, RS1, RS2	Desvio à direita	RD[6:0] = RS1[7:1] >> R, S2, RD[7] = 0	Z, S, C, V	4
SWAP	RD, RS1	Troca nibbles	RD = RS1 << 4	Z, S, V	4
COM	RD, RS1	Complemento de 1	RD = RS1 ⊙ R0	Z, S, C, V	4
NEG	RD, RS1	Complemento de 2	RD = R0 - RS1	Z, S, C, V	4
MOV	RD, RS1	Move de um registrador para outro	RD = RS1 + R0	Z, S, V	4
SBR	RD, W	Seta alguns bits do registrador	RD[W] = 1	S, V	3
CBR	RD, W	Limpa alguns bits do	RD[W] = 0	Z, S,	3

		registrador		V	
SER	RD	Seta todos os bits do registrador	RD = R0 ⊙ R0		4
CLR	RD	Limpa todos os bits do registrador	RD = R0 + R0	Z, S, V	4
CFR		Limpa o registrador de *flags*	FR = R0		3
MIR	RD, RS1	Espelha o *byte*	RD[7:0] = RS1[0:7]	Z, S, V	4
INC	RD	Incrementa 1 no registrador	RD = RD +1	Z, S, C, V	2
DEC	RD	Decrementa 1 do registrador	RD = RD - 1	Z, S, C, V	4
NOP		Não faz nada			3
JMPI	W	Desvio incondicional	PC = W		3
JMPR	W	Pula do endereço	PC = PC + W		3
ADDI	RD, RS1, W	Soma registrador imediatamente	RD = RS1 +W	Z, S, C	4
SUBI	RD, RS1, W	Subtrai registrador imediatamente	RD = RS1 - W	Z, S, C	4
XORI	RD, RS1, W	Operação XOR imediata	RD = RS1 ⊕ W	Z, S	4
BEQ	RS1, RS2, W	Desvio se dois registradores são iguais	IF(E) PC = PC + W		4
BLT	RS1, RS2, W	Desvio se um registrador é menor que o outro	IF(L) PC = PC + W		4
LD	RD, RS1, W	Carrega a um registrador a partir da memória de dados	RD = [RS1] + W		4
ST	RS1, RS2, W	Armazena na memória de dados a partir de um registrador	W + [RS1] = RS2		5
IN	RD, P	Recebe dados de uma entrada externa	RD = P		3
OUT	RS1, P	Envia dados a uma saída externa	P = RS1		3
SBI	P, W	Seta alguns bits no	P[W] = 1		3

		registrador de E/S			
CBI	P, W	Limpa alguns bits no registrador de E/S	P[W] = 0		3

2. CODIFICAÇÃO DAS INSTRUÇÕES

		000	001	010	011	100	101	110	111
00	00	NOP	SUB	AND	OR	XOR	NAND	NOR	XNOR
	01	RTR	RTL	SLL	SRL	SWAP	COM	NEG	MOV
	10	SBR	CBR	SER	CLR	CFR	MIR	INC	DEC
	11	ADD	JMPR						

	00	01	10	11
01	ADDI	SUBI	XORI	JMPI
10	BEQ	BLT	LD	ST
11	IN	OUT	SBI	CBI

Flags:

FLZ(Z): zero flag
FLS(S): signal flag
FLC(C): carry flag
FLV(V): overflow flag
FLE(E): equal flag
FLL(L): lower than flag

3. FORMATO DAS INSTRUÇÕES

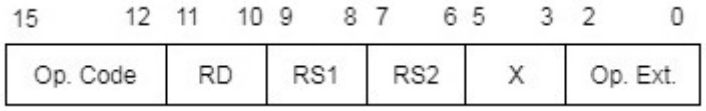


Figura 1 - Formato 3R (usa três registradores).

Instruções que obedecem o formato 3R:

- ADD
- SUB
- AND
- OR
- XOR
- NAND
- NOR
- NXOR
- RTR
- RTL
- SLL
- SRL

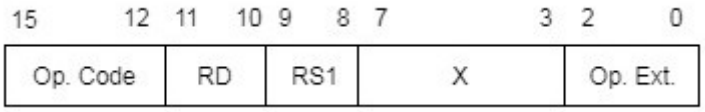


Figura 2 - Formato 2R (usa dois registradores).

Instruções que obedecem o formato 2R:

- SWAP
- COM
- NEG
- MOV
- MIR

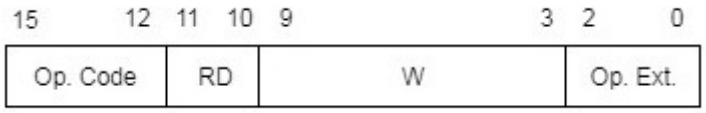


Figura 3 - Formato SRW (usa um registrador e palavra imediata curta).

Instruções que obedecem o formato SRW:

- SER
- CLR
- INC
- DEC
- SBR
- CBR

Figura 4 - Formato W (não usa registradores e palavra imediata média).

Instruções que obedecem o formato W:

- JMPR
- NOP
- CFR

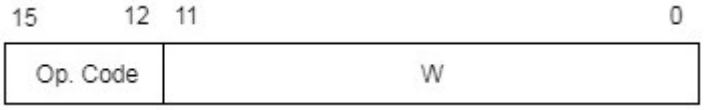


Figura 5 - Formato WW (não usa registradores e palavra imediata longa).

Instruções que obedecem o formato WW:

- JMPI

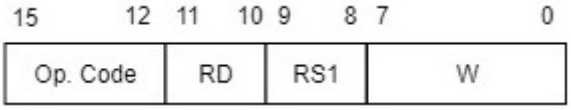


Figura 6 - Formato 8W2D (usa 8 bits para W e dois registradores, com um de destino).

Instruções que obedecem o formato 8W2D:

- ADDI
- SUBI
- XORI
- BEQ
- BLT
- LD

Figura 7 - Formato 8W2 (usa 8 bits para W e dois registradores).

Instruções que obedecem o formato 8W2:

- ST

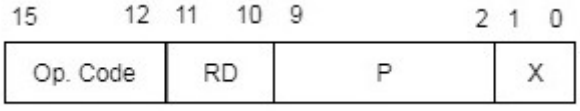


Figura 8 - Formato PIN (interage com a entrada a partir de *bytes*).

Instruções que obedecem o formato PIN :

- IN

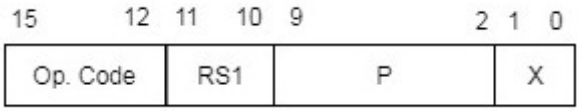


Figura 9 - Formato POUT (interage com a saída a partir de *bytes*).

Instruções que obedecem o formato POUT :

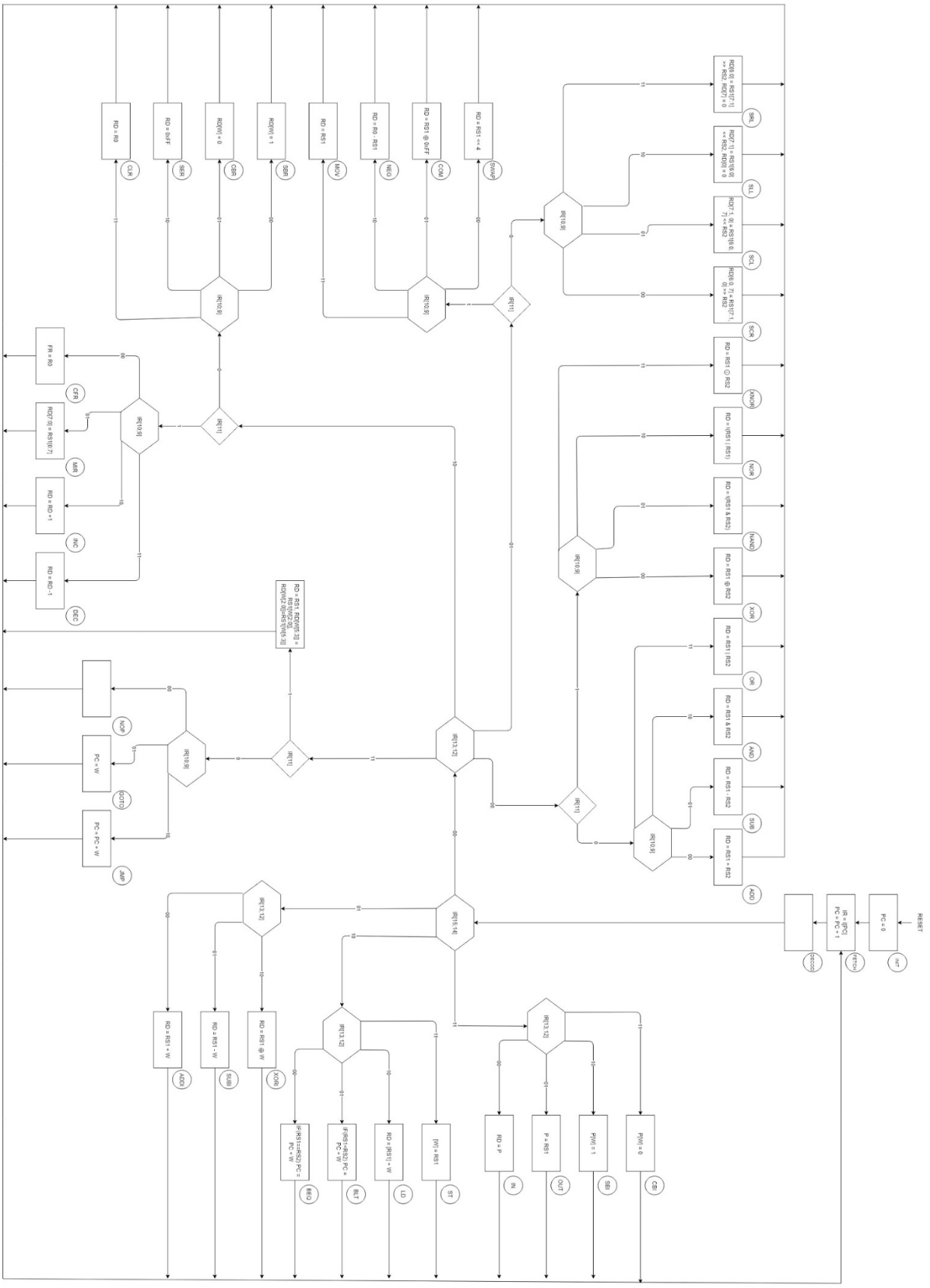
- OUT

Figura 10 - Formato WP (interage com a entrada/saída a partir de *bits*).

Instruções que obedecem o formato WP:

- SBI
- CBI

4. ASM CHART


RESET
PC = 0
INT
IR = @PC; PC = PC + 1
FETCH
DECODE
IR[15:14]
IR[13:12]
IR[11]
IR[10:9]
ADD
SUB
AND
OR
XOR
NAND
XNOR
SRL
SLL
SCL
SCR
MOV
NEG
COM
SWAP
CLR
SSR
CBR
SBR
FR = R0
CFR
MBR
RD = RD + 1
INC
RD = RD - 1
DEC
NOP
PC = W
GOTO
PC = PC + W
JMP
RD = RS1 + W
ADD
RD = RS1 - W
SUB
RD = RS1 @ W
XOR
IF RS1 = RS2 PC = PC + W
IFEQ
IF RS1 < RS2 PC = PC + W
BLT
RD = @RS1 + W
LD
[W] = RS1
ST
RD = P
IN
P = RS1
OUT
PSW = 1
SBI
PSW = 0
CBI

5. COMPONENTES DO *DATAPATH*

- ALU (8 bits):
    - Somador;
    - Subtrator;
    - AND *bitwise*;
    - OR *bitwise*;
    - XOR *bitwise*;
    - NAND *bitwise*;
    - NOR *bitwise*;
    - XNOR *bitwise*;
    - Deslocador/rotacionador à direita;
    - Deslocador/rotacionador à esquerda;
    - Espelhamento;
    - *Set bit*;
    - *Clear bit*;
- Registradores:
    - 1 x 6 bits (State Register)
    - 5 x 8 bits (1 de I/O e 4 internos);
    - 1 x 12 bits (Program Counter);
    - 1 x 16 bits (Instruction Register);
- Multiplexadores:
    - 4 x (4x1);
- Blocos avulsos
    - 2 x somadores (12 bits);
    - 1 x comparador (8 bits);
    - 1 x AND *bitwise* (8 bits);
    - 1 x OR *bitwise* (8 bits);
- Bloco de controle;
- *Datapath*;
- Memória de programa (4096x16);
- Memória de dados (256x8);

6. OPERAÇÕES DA ALU

- Soma (x0);
- Subtração (x1);
- AND (x2);
- OR (x3);
- XOR (x4);
- NAND (x5);
- NOR (x6);
- XNOR (x7);
- Deslocar à esquerda (x8);
- Deslocar à direita (x9);
- Rotacionar à esquerda (xA);
- Rotacionar à direita (xB);
- Espelhar (xC);
- Setar bits (xD);
- Limpar bits (xE);

7. MÁQUINA DE ESTADOS DE BAIXO NÍVEL

SAÍDAS DOS ESTADOS

Estado atual	Estado seguinte	Saídas
INIT	FETCH	PC_clr = 1
FETCH	DECOD	I_rd = 1; PC_ld = 1; IR_ld = 1, MUX_PC=00
DECOD	--	--
ADD	FETCH	MUX_alu = 0000, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
SUB	FETCH	MUX_alu = 0001, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
AND	FETCH	MUX_alu = 0010, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
OR	FETCH	MUX_alu = 0011, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
XOR	FETCH	MUX_alu = 0100, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
NAND	FETCH	MUX_alu = 0101, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
NOR	FETCH	MUX_alu = 0110, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
XNOR	FETCH	MUX_alu = 0111, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
RTR	FETCH	MUX_alu = 1000, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
RTL	FETCH	MUX_alu = 1001, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
SLL	FETCH	MUX_alu = 1000, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
SRL	FETCH	MUX_alu = 1001, MUX_sel = 11, RS1_add = IR[9:8], RS2_add = IR[7:6], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld =

		IR[11]IR[10]
SWAP	FETCH	MUX_alu = 100X, MUX_sel = 11, RS1_add = IR[9:8], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
COM	FETCH	MUX_alu = 0111, MUX_sel = 11, RS1_add = IR[9:8], RS2_add =00, MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
NEG	FETCH	MUX_alu = 0001, MUX_sel = 11, RS1_add = 00, RS2_add = IR[9:8], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
MOV	FETCH	MUX_alu = 0000, MUX_sel = 11, RS1_add = IR[9:8], RS2_add =00, MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
SBR	FETCH	MUX_alu = 1011, MUX_imm = 10, MUX_sel = 11, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10], RS1_add = IR[11:10]
CBR	FETCH	MUX_alu = 1100, MUX_imm = 10, MUX_sel = 11, R1_ld = !IR[11]IR[10], R2_l]d =IR[11]!IR[10] , R3_ld = IR[11]IR[10], RS1_add = IR[11:10]
SER	FETCH	MUX_alu = 0111, MUX_sel = 11, RS1_add = 00, RS2_add =00, MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
CLR	FETCH	MUX_alu = 0000, MUX_sel = 11, RS1_add = 00, RS2_add =00, MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
CFR	FETCH	SR_clr = 1
MIR	FETCH	MUX_alu = 1010, MUX_sel = 11, RS1_add = IR[9:8], MUX_imm = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
INC	FETCH	MUX_alu = 0000, MUX_sel = 11, RS1_add = IR[11:10], MUX_imm = 11, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
DEC	FETCH	MUX_alu = 0001, MUX_sel = 11, RS1_add = IR[11:10], MUX_imm = 11, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
NOP	FETCH	--
JMPI	FETCH	MUX_PC = 10 , PC_ld = 1
JMPR	FETCH	MUX_PC = 11, PC_ld = 1
ADDI	FETCH	MUX_alu = 0000, MUX_sel = 11, RS1_add = IR[9:8], MUX_imm = 01, W = IR[7:0], R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
SUBI	FETCH	MUX_alu = 0001, MUX_sel = 11, RS1_add = IR[9:8], MUX_imm = 01, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
XORI	FETCH	MUX_alu = 0100, MUX_sel = 11, RS1_add = IR[9:8], MUX_imm = 01, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
BEQ	JMPR/ FETCH	IF(E) JMPR ELSE FETCH
BLT	JMPR/ FETCH	IF(L) JMPR ELSE FETCH

LD	FETCH	MUX_alu = 0000, MUX_sel = 10, RS1_add = IR[9:8], MUX_imm = 01, D_rd = 1, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
ST	FETCH	MUX_alu = 0000, MUX_sel = 11, RS1_add = IR[11:10], MUX_imm =01, RS2_add = IR[9:8], D_wr = 1
IN	FETCH	MUX_sel = 01, IN_ld = 1, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10]
OUT	FETCH	RS1_add = IR[11:10], OUT_ld = 1
SBI	FETCH	MUX_alu = 1011, MUX_imm = 10, MUX_sel = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10], RS1_add = IR[11:10], OUT_ld = 1
CBI	FETCH	MUX_alu = 1100, MUX_imm = 10, MUX_sel = 00, R1_ld = !IR[11]IR[10], R2_ld =IR[11]!IR[10] , R3_ld = IR[11]IR[10], RS1_add = IR[11:10], OUT_ld = 1

TRANSIÇÃO DOS ESTADOS

Estado		Transição
De	Para	
INIT	FETCH	--
FETCH	DECOD	--
DECOD	ADD	IR[15:14] = 00 & IR[13:12,2] = 110 & IR[1:0] = 00
DECOD	SUB	IR[15:14] = 00 & IR[13:12,2] = 000 & IR[1:0] = 01
DECOD	AND	IR[15:14] = 00 & IR[13:12,2] = 000 & IR[1:0] = 10
DECOD	OR	IR[15:14] = 00 & IR[13:12,2] = 000 & IR[1:0] = 11
DECOD	XOR	IR[15:14] = 00 & IR[13:12,2] = 001 & IR[1:0] = 00
DECOD	NAND	IR[15:14] = 00 & IR[13:12,2] = 001 & IR[1:0] = 01
DECOD	NOR	IR[15:14] = 00 & IR[13:12,2] = 001 & IR[1:0] = 10
DECOD	XNOR	IR[15:14] = 00 & IR[13:12,2] = 001 & IR[1:0] = 11
DECOD	RTR	IR[15:14] = 00 & IR[13:12,2] = 010 & IR[1:0] = 00
DECOD	RTL	IR[15:14] = 00 & IR[13:12,2] = 010 & IR[1:0] = 01
DECOD	SLL	IR[15:14] = 00 & IR[13:12,2] = 010 & IR[1:0] = 10
DECOD	SRL	IR[15:14] = 00 & IR[13:12,2] = 010 & IR[1:0] = 11
DECOD	SWAP	IR[15:14] = 00 & IR[13:12,2] = 011 & IR[1:0] = 00
DECOD	COM	IR[15:14] = 00 & IR[13:12,2] = 011 & IR[1:0] = 01

DECOD	NEG	IR[15:14] = 00 & IR[13:12,2] = 011 & IR[1:0] = 10
DECOD	MOV	IR[15:14] = 00 & IR[13:12,2] = 011 & IR[1:0] = 11
DECOD	SBR	IR[15:14] = 00 & IR[13:12,2] = 100 & IR[1:0] = 00
DECOD	CBR	IR[15:14] = 00 & IR[13:12,2] = 100 & IR[1:0] = 01
DECOD	SER	IR[15:14] = 00 & IR[13:12,2] = 100 & IR[1:0] = 10
DECOD	CLR	IR[15:14] = 00 & IR[13:12,2] = 100 & IR[1:0] = 11
DECOD	CFR	IR[15:14] = 00 & IR[13:12,2] = 101 & IR[1:0] = 00
DECOD	MIR	IR[15:14] = 00 & IR[13:12,2] = 101 & IR[1:0] = 01
DECOD	INC	IR[15:14] = 00 & IR[13:12,2] = 101 & IR[1:0] = 10
DECOD	DEC	IR[15:14] = 00 & IR[13:12,2] = 101 & IR[1:0] = 11
DECOD	NOP	IR[15:14] = 00 & IR[13:12,2] = 000 & IR[1:0] = 00
DECOD	JMPI	IR[15:14] = 01 & IR[13:12] = 11
DECOD	JMPR	IR[15:14] = 00 & IR[13:12,2] = 110 & IR[1:0] = 01
DECOD	ADDI	IR[15:14] = 01 & IR[13:12] = 00
DECOD	SUBI	IR[15:14] = 01 & IR[13:12] = 01
DECOD	XORI	IR[15:14] = 01 & IR[13:12] = 10
DECOD	BEQ	IR[15:14] = 10 & IR[13:12] = 00
DECOD	BLT	IR[15:14] = 10 & IR[13:12] = 01
BEQ	NOP	E = 1
BLT	NOP	L = 1
DECOD	LD	IR[15:14] = 10 & IR[13:12] = 10
DECOD	ST	IR[15:14] = 10 & IR[13:12] = 11
DECOD	IN	IR[15:14] = 11 & IR[13:12] = 00
DECOD	OUT	IR[15:14] = 11 & IR[13:12] = 01
DECOD	SBI	IR[15:14] = 11 & IR[13:12] = 10
DECOD	CBI	IR[15:14] = 11 & IR[13:12] = 11
ADD… CBI	FETCH	--

8. DIAGRAMA DE BLOCOS DO NICKEL