Nuțu Maria-Nicoleta
Bilet 39

Documentație Proiect PIC24

În cadrul acestui proiect, am avut de implementat un procesor PIC24 care execută următoarele instrucțiuni:

- Instrucțiuni comune:

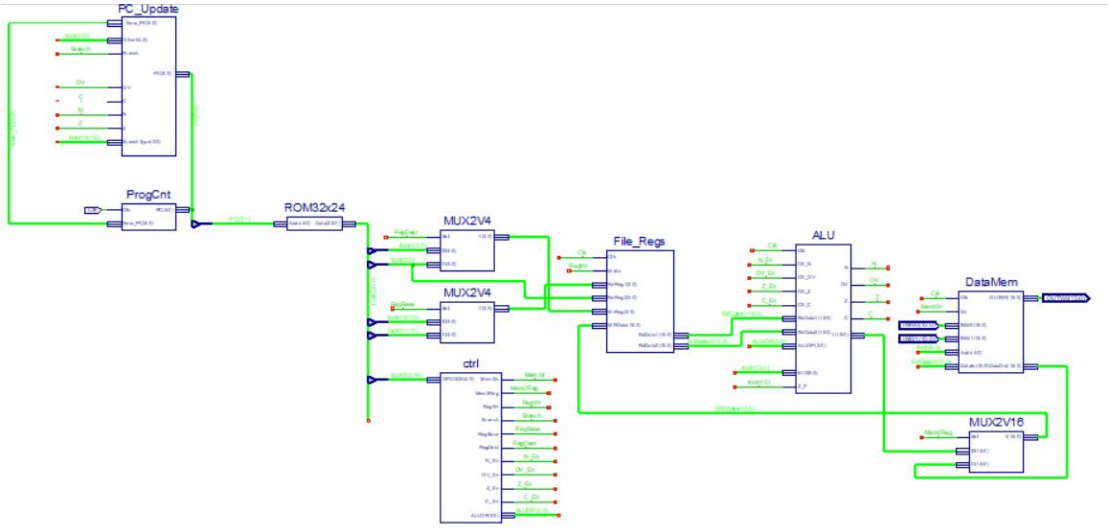
```
ADD   Wb, Ws, Wd
SUB   Wb, Ws, Wd
AND   Wb, Ws, Wd
IOR   Wb, Ws, Wd
MOV   Wns,f
MOV   f, Wns
BRA   Expr
```

- Instrucțiuni specifice:

Proiectul 39

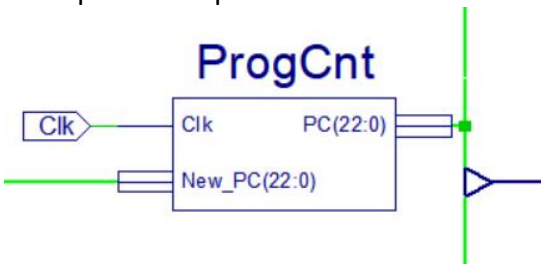
Non-jump Instructions	Flags	Jump Instructions
BSW.Z Ws,Wb	OV	BRA OV,Expr
RRC Ws,Wd	C	BRA C,Expr
SUB #lit10,Wn	N	BRA N,Expr
ZE Ws,Wnd	Z	BRA Z,Expr

SCHEMA BLOC

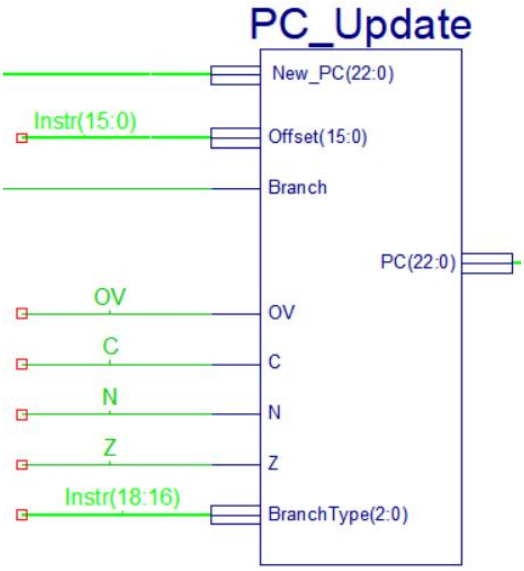


Componente:

1. **ProgCnt** (Program Counter). Acesta este blocul care actualizează PC (program counter) cu o nouă valoare pe frontul pozitiv al semnalului de ceas.

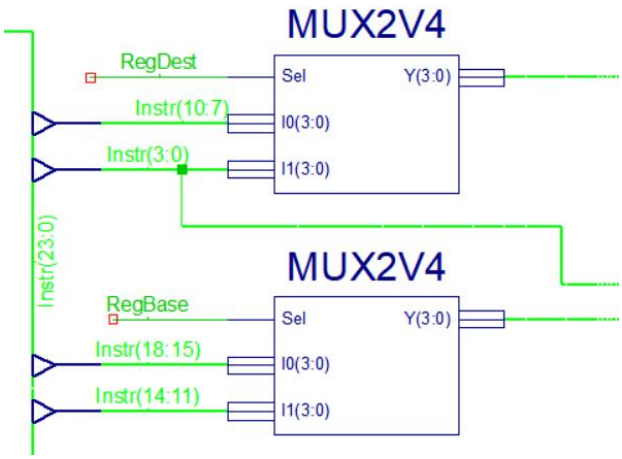


2. **PC_Update**. Acest bloc are rolul de a actualiza program counter-ul cu 2 dacă instrucțiunea nu este de salt sau cu 2 la care se adaugă și deplasamentul dacă instrucțiunea este de salt. Blocul decide dacă instrucțiunea este de salt în funcție de semnalul *Branch*. De asemenea, poate decide ce fel de instrucțiune de salt este după cei 3 biți ai vectorului *BranchType*.

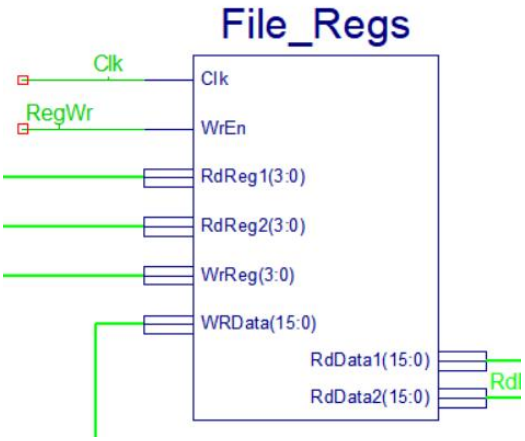


3. **ROM32x24.** Acest bloc reprezintă memoria procesorului. Conține 32 de cuvinte de câte 24 de biți fiecare. Datele se transmit prin portul *Data* folosind valoarea adresei *Addr*.

4. **MUX2V4.** Din tabelul opcode-urilor se observă că nu toate instrucțiunile folosesc aceiași biți pentru regiștrii de bază (w) și cei destinație (d). Instrucțiunile folosesc biții 18:15 pentru regiștrii de bază, exceptând instrucțiunea BSW.Z ce folosește biții 14:11. În ceea ce privește regiștrii destinație, majoritatea instrucțiunilor folosesc biții 10:7. Instrucțiunile MOV f,wnd și SUB folosesc biții 3:0. Pentru a nu se crea confuzii în ce privește preluarea datelor, am folosit 2 MUX-uri cu 2 intrări a câte 4 biți și o selecție, unul pentru registrul de bază și unul pentru registrul destinație.

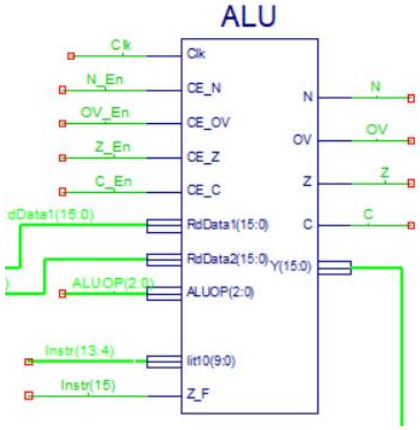


5. **File_Regs.** Acest bloc este responsabil de scrierea datelor în urma realizării operațiilor care au un registru destinație, dar citește și informația pentru operațiile cu registru de bază sau sursă. Fișierul de registre se actualizează la fiecare semnal de ceas pe front ridicător, și cu semnalul *RegWr* activ, ce vine de la unitatea de control.

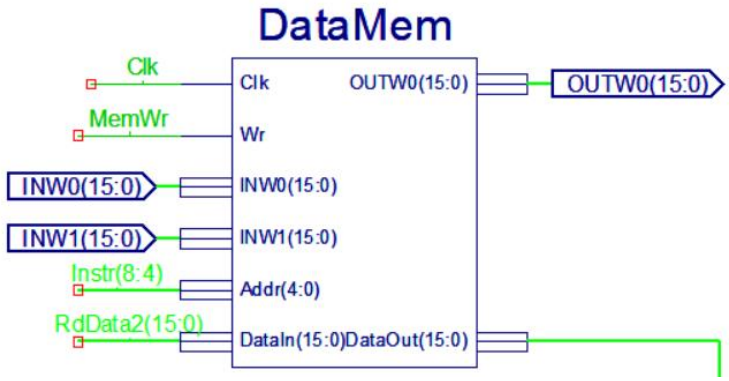


6. **ALU.** Unitatea care se ocupă de realizarea operațiilor logico-aritmetice, implementează fiecare instrucțiune folosindu-se de RdData1 (registrul de bază) și RdData2 (registrul sursă). Instrucțiunile comune sunt instrucțiuni simple, realizate cu ajutorul semnelor “+”, “-”, AND și OR. Mai jos sunt explicate instrucțiunile specifice.

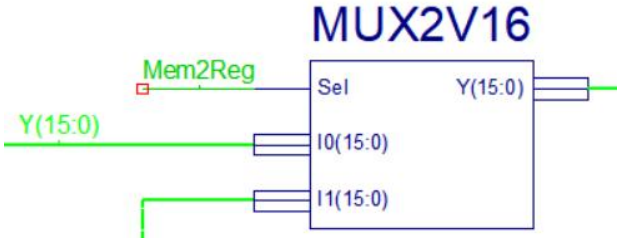
a. BSW.Z Ws, Wb - Bitul (Wb) din registrul Ws este scris cu valoarea flagului Z negată din registrul STATUS. Doar cei 4 cei mai puțin semnificativi biți din Wd se pot folosi pentru a determina poziția bitului ce trebuie scris.
b. RRC Ws, Wd - Rotește conținutul registrului sursă Ws cu un bit la dreapta prin bitul de Carry și stochează rezultatul în registrul destinație Wd. Flagul Carry din registrul STATUS se mută în cel mai semnificativ bit al registrului Wb, apoi e suprascris cu cel mai puțin semnificativ bit al registrului Ws.
c. SUB #lit10, Wn - Scade un operand fără semn (literal) pe 10 biți din conținutul registrului de lucru Wn și salvează rezultatul tot în Wn.
d. ZE Ws, Wnd - Extinde cel mai puțin semnificativ octet al registrului sursă Ws cu zero până la 16 biți și stochează rezultatul în Wnd.



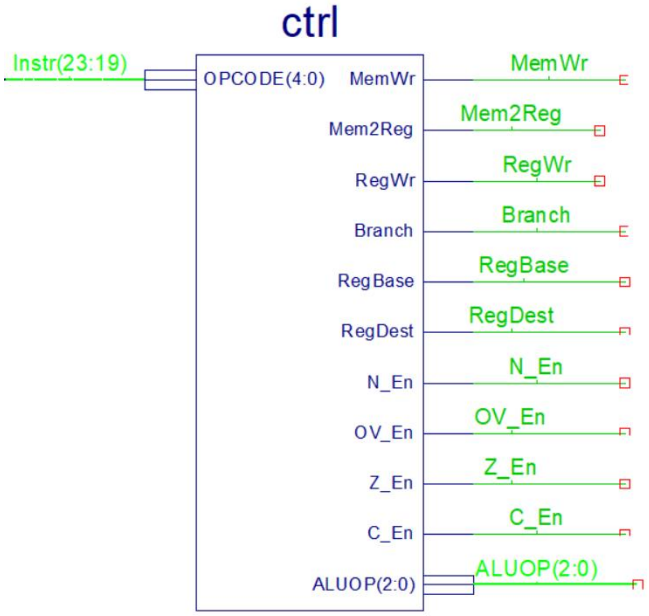
7. **DataMem.** Conține 16x16 module RAM. Instrucțiunea MOV Wns, f e singura care scrie în memorie.



8. **MUX2V16.** Acest MUX are rolul de a decide dacă datele sunt citite din memorie și scrise în registru sau sunt citite din registru și scrise în memorie.



1. **ctrl**. Acest bloc controlează operațiile din celelalte blocuri, deoarece conține toate semnalele specifice fiecărui bloc.
   Semnalele folosite sunt:
   - Pentru instrucțiunile ADD Wb, Ws, Wd și SUB Wb, Ws, Wd avem următoarele flag-uri care sunt semnale folosite în ALU pentru transport, status zero, overflow și status negativ:
     - *N_En* (flag negative)
     - *OV_En* (flag overflow)
     - *Z_En* (flag zero)
     - *C_En* (flag carry)
   - *MemWr* – semnal utilizat în blocul DataMem ce activează scrierea în memorie. Singura instrucțiune care face acest lucru este MOV Wns, f.
   - *Mem2Reg* – semnal utilizat in MUX2V16 pentru scrierea datelor din memorie în registru. Considerăm 0 pentru citirea ALU și 1 pentru citirea memoriei.
   - *RegWr* – semnal utilizat în blocul File_Regs pentru a activa scrierea în registru.
   - *Branch* – semnal utilizat în blocul PC_Update pentru a stabili dacă instrucțiunea este de salt sau nu.
   - *RegBase* – semnal utilizat în MUX_2V4 pentru selectarea biților corespunzători registrului de bază. Selecția se face din biții 18:15 și 14:11.
   - *RegDest* – semnal utilizat în MUX_2V4 pentru selectarea biților corespunzători registrului destinație. Selecția se face din biții 10:7 și 3:0.
   - *ALUOP* – semnal utilizat pentru selecția instrucțiunii curente.
   - *OPCODE* – reprezintă opcode-ul instrucțiunii.



Encoding	2222 3210	1111 9876	1111 5432	11 1098	7654	3210	Flags
ADD	0100	0www	wBqq	qddd	dppp	ssss	N, OV, Z, C
SUB	0101	0www	wBqq	qddd	dppp	ssss	N, OV, Z, C
AND	0110	0www	wBqq	qddd	dppp	ssss	N, -, Z, -
IOR	0111	0www	wBqq	qddd	dppp	ssss	N, -, Z, -
MOV f,wnd	1000	0fff	ffff	ffff	ffff	dddd	none
MOV wns, f	1000	1fff	ffff	ffff	ffff	ssss	none
BSW.Z Ws, Wb	1010	1101	Zwww	w000	0ppp	ssss	none
RRC Ws, Wd	1101	0011	1Bqq	qddd	dppp	ssss	N, -, Z, C
SUB #lit10, Wn	1011	0001	0Bkk	kkkk	kkkk	dddd	N, OV, Z, C
ZE Ws, Wnd	1111	1011	1000	0ddd	dppp	ssss	N, -, Z, C
BRA Expr	0011	0111	nnnn	nnnn	nnnn	nnnn	none
BRA OV, Expr	0011	0000	nnnn	nnnn	nnnn	nnnn	none
BRA C,Expr	0011	0001	nnnn	nnnn	nnnn	nnnn	none
BRA N,Expr	0011	0011	nnnn	nnnn	nnnn	nnnn	none
BRA Z,Expr	0011	0010	nnnn	nnnn	nnnn	nnnn	none

	OPCODE	N_En	OV_En	Z_En	C_En	ALUOP	MemWr	Mem2Reg	RegWr	Branch	RegBase	RegDest
ADD	01000	1	1	1	1	000	0	0	1	0	0	0
SUB	01010	1	1	1	1	001	0	0	1	0	0	0
AND	01100	1	0	1	0	010	0	0	1	0	0	0
IOR	01110	1	0	1	0	011	0	0	1	0	0	0
MOV f, wnd	10000	0	0	0	0	xxx	0	1	1	0	x	1
MOV wns, f	10001	0	0	0	0	xxx	1	x	0	0	x	x
BSW.Z Ws, Wb	10101	0	0	0	0	100	0	0	1	0	1	1
RRC Ws, Wd	11010	1	0	1	1	101	0	0	1	0	x	0
SUB#lit10, Wn	10110	1	1	1	1	110	0	0	1	0	x	1
ZE Ws, Wnd	11111	1	0	1	1	111	0	0	1	0	x	0
BRA expr	00110	0	0	0	0	xxx	0	x	0	1	x	x
BRA OV, Expr	00110	0	0	0	0	xxx	0	x	0	1	x	x
BRA C,Expr	00110	0	0	0	0	xxx	0	x	0	1	x	x
BRA N,Expr	00110	0	0	0	0	xxx	0	x	0	1	x	x
BRA Z,Expr	00110	0	0	0	0	xxx	0	x	0	1	x	x