Proiect FIC- Set de instrucțiuni

Instrucțiuni care transferă valoarea din AC în registrele X și Y

|  |  |  |
| --- | --- | --- |
| Opcode | Nume instrucțiune | Altele<10> |
| 000000 | TRX | 0000000000000000 |
| 000001 | TRY | 0000000000000000 |

TRX – transfera valoarea din ACC in X

TRY – transfera valoarea din ACC in Y

Toți ceilalți biti sunt 0 deoarece sunt nefolosiți

Instructiuni de memorie:

Adresa de registru: 0 – X

1 - Y

Valoarea imediata este pe 9 biti, deci memoria de date trebuie sa aiba maxim 512 locații, fiecare locatie are 16 biți.

Instructiuni Load and Store:

|  |  |  |  |
| --- | --- | --- | --- |
| Opcode | Nume instrucțiune | Adresa de registru (bits) | Valoare imediata (bits) |
| 000010 | LDR | <1> | <9> |
| 000011 | STR | <1> | <9> |

Instructiuni Stack:

|  |  |  |  |
| --- | --- | --- | --- |
| Opcode | Nume instrucțiune | Adresa de registru (bits) | Valoare imediata (bits) |
| 000100 | PSH | <2> | 00000000 |
| 000101 | POP | <2> | 00000000 |

Instructiunile Stack pot face push/pop valorilor din registrele următoare pe stiva:

00 – X

01 – Y

10 – ACC

11 - PC

Instructiuni Branch:

Adresă este pe 10 biti, deci instructiunea de memorie are nevoie de maxim 1024 locații, fiecare locatie are 16 biți.

|  |  |  |
| --- | --- | --- |
| Opcode | Nume instrucțiune | Adresă |
| 000110 | BRZ | <10> |
| 000111 | BRN | <10> |
| 001000 | BRC | <10> |
| 001001 | BRO | <10> |
| 001010 | BRA | <10> |
|  | JMP | <10> |
|  | RET | 0000000000 |

JMP și RET vor fi implementate ca pseudoinstructiuni. Un JMP [adr] trebuie sa genereze:

PSH PC

BRA [adr]

O instrucțiune RET trebuue sa genereze POP PC

Instrucțiuni Arithmetic and logic:

|  |  |  |  |
| --- | --- | --- | --- |
| Opcode | Nume instrucțiune | Adresa de registru (bits) | Valoare imediata (bits) |
| 001101 | ADD | <1> | <9> |
| 001110 | SUB | <1> | <9> |
| 001111 | LSR | <1> | <9> |
| 010000 | LSL | <1> | <9> |
| 010001 | MUL | <1> | <9> |
| 010010 | DIV | <1> | <9> |
| 010011 | MOD | <1> | <9> |
| 010100 | CMP | <1> | <9> |
| 010101 | INC | <1> | 000000000 |
| 010110 | DEC | <1> | 000000000 |
| 010111 | AND | <1> | <9> |
| 011000 | OR | <1> | <9> |
| 011001 | XOR | <1> | <9> |
| 011010 | NOT | <1> | 000000000 |
| 011011 | RSR | <1> | <9> |
| 011100 | RSL | <1> | <9> |
| 011101 | FCT | <1> | <9> |

Instrucțiunea de mutare a valorii immediate in registru:

|  |  |  |  |
| --- | --- | --- | --- |
| Opcode | Nume instrucțiune | Adresa de registru (bits) | Valoare imediata (bits) |
| 011110 | MOV | <1> | <9> |

Exemplu:

ADD X => ACC = ACC + X

ADD X, val. imediată => X = X + val. imediată

SUB X => ACC = ACC – X

SUB X, val. imediată => X = X - val. imediată

LSR X => ACC = ACC >> X

LSR X, val. imediată => X = X >> val. imediată

LSL X => ACC = ACC << X

MUL X => ACC = ACC \* X

DIV X => ACC = ACC / X

MOD X => ACC = ACC % X

INC X => X = X + 1

DEC X => X = X - 1

MOV X, val. imediată => X = val. imediată

CMP X => compara ACC cu X

Asemanator pentru AND, OR, XOR, RSR, RSL.

Resetarea acumulatorului poate fi făcută astfel: TRX, SUB X

Factorial: rezultatele pot fi salvate doar în registrele X or Y

Ex: FCT X 3 -> muta in registrul X valoarea lui 3! Care este 6.