|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| instruction | job | Opc(6) | R0(5) | R1(5) | R2(5)  addr(16) off(16) | Imm(32) | length |
| MTC | Move to float coprocessor | 100000 | - | - | - | - | 32 bits |
| MFC | Move from float coprocessor | 111111 | - | - | - | - | 32 bits |
| ADDF | DST ← SRC1 + SRC2 | 100001 | 5 | 5 | 5 | - | 32 bits |
| SUBF | DST ← SRC1 - SRC2 | 100010 | 5 | 5 | 5 | - | 32 bits |
| MULF | DST ← SRC1 \* SRC2 | 100011 | 5 | 5 | 5 | - | 32 bits |
| DIVF | DST ← SRC1 / SRC2 | 100100 | 5 | 5 | 5 | - | 32 bits |
| INVF | DST ← 1 / SRC | 100101 | 5 | 5 | - | - | 32 bits |
| ABSF | DST ← int32(SRC) | 100110 | 5 | 5 | - | - | 32 bits |
| COMF | SRC1 > SRC2 : DST = 1 , SRC1 < SRC2 : DST = -1 , SRC1 == SRC2 : DST = 0 | 100111 | 5 | 5 | 5 | - | 32 bits |
| MOVIF | DST ← IMM | 110000 | 5 | 5 | - | 32 | 64 bits |
| ADDIF | DST ← SRC + IMM | 110001 | 5 | 5 | - | 32 | 64 bits |
| SUBIF | DST ← SRC - IMM | 110010 | 5 | 5 | - | 32 | 64 bits |
| MULIF | DST ← SRC \* IMM | 110011 | 5 | 5 | - | 32 | 64 bits |
| DIVIF | DST ← SRC / IMM | 110100 | 5 | 5 | - | 32 | 64 bits |
| INVIF | DST ← 1 / IMM | 110101 | 5 | 5 | - | 32 | 64 bits |
| ABSIF | DST ← int32(IMM) | 110110 | 5 | 5 | - | 32 | 64 bits |
| LF = LW | VR ← MEM [$AR+ SIGN EXTEND (Offset)] | 111000 | 5 | 5 | 16 | - | 32 bits |
| SF = SW | MEM [$AR+ SIGN EXTEND (Offset)] ← VR | 111001 | 5 | 5 | 16 | - | 32 bits |
| BEQF | REG1 == REG2 : PC ← PC + SIGN EXTEND ( Address | “00” ) | 111100 | 5 | 5 | 16 | - | 32 bits |
| BLTF | REG1 < REG2 : PC ← PC + SIGN EXTEND ( Address | “00” ) | 111101 | 5 | 5 | 16 | - | 32 bits |
| BGTF | REG1 > REG2 : PC ← PC + SIGN EXTEND ( Address | “00” ) | 111110 | 5 | 5 | 16 | - | 32 bits |
| HLT | STOP PC | 000000 | - | - | - | - | 32 bits |

\*) برنامه به زبان فیبوناچی :

زمان اجرا بر اساس تعداد سیکل :

000010\_00000\_00000\_00000\_00000\_000000

011000\_00001\_00000\_0000000000000000

010010\_00010\_00000\_0000000000000000

010010\_00011\_00000\_0000000000000001

000001\_00100\_00010\_00011\_00000\_000000

010010\_00010\_00011\_0000000000000000

010010\_00011\_00100\_0000000000000000

010011\_00001\_00001\_0000000000000001

011111\_00000\_00001\_1111111111111100

011001\_00010\_00000\_0000000000000001

000000\_00000\_00000\_0000000000000000

sub $r0 $r0 $r0

lw $r1 $r0 0

addi $r2 $r0 0

addi $r3 $r0 1

add $r4 $r2 $r3

addi $r2 $r3 0

addi $r3 $r4 0

subi $r1 $r1 1

blt $r0 $r1 -4

sw $r2 $r0 1

hlt

5\*5 + 5\*5n\* = (n+1)\*25

در مثال تست شده n=10 بوده و زمان اجرا برابر با 275 سیکل پردازنده بوده و

عدد نهایی که 55 یا 00110111 میباشد در آدرس 4 حافظه نوشته خواهد شد.