|  |  |  |
| --- | --- | --- |
| bit C | uOP | meaning |
| c0 | CAR<=CAR+1 | Control Address increment |
| c1 | CAR<=\*\*\*\* | address redirection, depends on  position of uOP |
| c2 | CAR<=0 | reset car |
| c3 | MBR<=memory | Read memory content to MBR |
| c4 | IR<=[15:8] | copy MBR opcode |
| c5 | MAR<=MBR[7:0] | copy MBR address |
| c6 | PC<=PC+1 | PC increment |
| c7 | BR<=MBR | copy MBR to BR |
| c8 | ACC<=0 | reset ACC |
| c9 | ALU<=ACC+BR | ADD BR to ACC |
| c10 | MAR<=PC | read PC to MAR for next address |
| c11 | MBR<=ACC | copy ACC content to MBR |
| c12 | memory<=MBR | write MBR content to memory |
| c13 | ALU<=ACC-BR | sub BR from ACC |
| c14 | pc<=MBR[7:0] | copy MBR content address to PC |
| c15 | ALU<=ACC and BR | ACC && BR |
| c16 | ALU<=ACC or BR | ACC || BR |
| c17 | ALU<= not ACC | !ACC |
| c18 | ALU<=SHR ACC | logical right shift |
| c19 | ALU<=SHL ACC | logical left shift |
| c20 | CAR<=CAR+1+FLAG(1) | JMPGEZ |
| c21 | ALU<=BR | input content of BR to ALU |
| c22 | ALU<=ACC | input content of ACC to ALU |
| c23 | ACC<=ALU | output result to ACC |
| c24 | ALU<=RESULT[15:0] MR<=RESULT[31:16] | ALU gets the lower part of the mult result while MR gets the higher |
| c25 | ACC<=MBR | input content of MBR to ACC |
| c26 | ALU<=BR\*ACC | multiply BR by ACC |
| c27 | CAR<=CAR+FLAG(3) | multiply control |
| c28 | \*\*\*\*\* |  |
| c29 | \*\*\*\*\* |  |
| c30 | \*\*\*\*\* |  |
| c31 | \*\*\*\*\* |  |