Opcodes

| Opcode Name Action Opcode bitfields Arithmetic Logic Unit ADD rd,rs,rt Add rd=rs+rt 0000000 rs rt ADDI rt,rs,imm Add Immediate rt=rs+imm 001000 rs rt ADDIU rt,rs,imm Add Immediate Unsigned rt=rs+rt 000000 rs rt ADDU rd,rs,rt Add Unsigned rd=rs+rt 0000000 rs rt AND rd,rs,rt And rd=rs&rt 0000000 rs rt ANDI rt,rs,imm And Immediate rt=rs&imm 001100 rs rt LUI rt,imm Load Upper Immediate rt=imm< 00 00000 rs rt NOR rd,rs,rt Nor rd=~(rs rt) 0000000 rs rt OR rd,rs,rt Or rd=rs rt 0000000 rs rt ORI rt,rs,imm Or Immediate rt=rs imm 001101 rs rt SLT rd,rs,rt Set On Less Than rd=rs <rt< td=""> 000000 rs rt SLTI rt,rs,imm Set On Less Than Immediate rt=rs<imm< td=""> 001010 rs rt</imm<></rt<> | imr imr rd rd imr imr | n 00000 00000 n | 100000 |
|---|--|--|--------------------------|
| ADD rd,rs,rt Add rd=rs+rt 000000 rs rt ADDI rt,rs,imm Add Immediate rt=rs+imm 001000 rs rt ADDIU rt,rs,imm Add Immediate Unsigned rt=rs+imm 001001 rs rt ADDU rd,rs,rt Add Unsigned rd=rs+rt 000000 rs rt AND rd,rs,rt And rd=rs&rt 000000 rs rt ANDI rt,rs,imm And Immediate rt=rs&imm 001100 rs rt LUI rt,imm Load Upper Immediate rt=imm< | imr rd rd imr imr rd rd | n 00000 00000 n | 100000 |
| ADDI rt,rs,imm Add Immediate | imr rd rd imr imr rd rd | n 00000 00000 n | 1 |
| ADDIU rt,rs,imm Add Immediate Unsigned rt=rs+imm 001001 rs rt ADDU rd,rs,rt Add Unsigned rd=rs+rt 000000 rs rt AND rd,rs,rt And rd=rs&rt 000000 rs rt ANDI rt,rs,imm And Immediate rt=rs&imm 001100 rs rt LUI rt,imm Load Upper Immediate rt=imm<<16 001111 rs rt NOR rd,rs,rt Nor rd= \sim (rs rt) 000000 rs rt OR rd,rs,rt Or rd=rs rt 000000 rs rt ORI rt,rs,imm Or Immediate rt=rs imm 001101 rs rt SLT rd,rs,rt Set On Less Than rd=rs <rt 000000="" rs="" rt<="" td=""><td>imr rd rd imr imr rd</td><td>n 00000 00000 n</td><td></td></rt> | imr rd rd imr imr rd | n 00000 00000 n | |
| ADDU rd,rs,rt Add Unsigned rd=rs+rt 000000 rs rt AND rd,rs,rt And rd=rs&rt 000000 rs rt ANDI rt,rs,imm And Immediate rt=rs&imm 001100 rs rt LUI rt,imm Load Upper Immediate rt=imm<<16 | rd rd imr imr rd rd | 00000 00000 n | |
| AND rd,rs,rt And rd=rs&rt 000000 rs rt ANDI rt,rs,imm And Immediate rt=rs&imm 001100 rs rt LUI rt,imm Load Upper Immediate rt=imm<<16 | rd imr imr rd rd | 00000 n n | 100001 |
| ANDI rt,rs,imm And Immediate rt=rs&imm 001100 rs rt LUI rt,imm Load Upper Immediate rt=imm<<16 | imr imr rd rd | m m | 100100 |
| $ \begin{array}{c ccccccccccccccccccccccccccccccccccc$ | imr rd rd | n | |
| $\begin{array}{cccccccccccccccccccccccccccccccccccc$ | rd rd | | |
| OR rd,rs,rt Or rd=rs rt 000000 rs rt ORI rt,rs,imm Or Immediate rt=rs imm 001101 rs rt SLT rd,rs,rt Set On Less Than rd=rs <rt< td=""> 000000 rs rt</rt<> | rd | 00000 | 100111 |
| ORI rt,rs,imm Or Immediate rt=rs imm 001101 rs rt SLT rd,rs,rt Set On Less Than rd=rs <rt< td=""> 000000 rs rt</rt<> | _ | 00000 | 100101 |
| SLT rd,rs,rt Set On Less Than rd=rs <rt 0000000="" rs="" rt<="" td=""><td></td><td></td><td>1</td></rt> | | | 1 |
| | | 00000 | 101010 |
| per registricità per en 2000 main inimicatate per 15 mini | imr | | 101010 |
| SLTIU rt,rs,imm Set On < Immediate Unsigned rt=rs <imm 001011="" rs="" rt<="" td="" =""><td>imr</td><td></td><td></td></imm> | imr | | |
| SLTU rd,rs,rt Set On Less Than Unsigned rd=rs <rt 000000="" rs="" rt<="" td=""><td>_</td><td>00000</td><td>101011</td></rt> | _ | 00000 | 101011 |
| SUB rd,rs,rt Subtract rd=rs-rt 000000 rs rt | | 00000 | 100010 |
| SUBU rd,rs,rt Subtract Unsigned rd=rs-rt 000000 rs rt | | 00000 | 100011 |
| XOR rd,rs,rt Exclusive Or rd=rs^rt 000000 rs rt | | 00000 | 100110 |
| XORI rt,rs,imm Exclusive Or Immediate rt=rs^imm 001110 rs rt | imr | | 1100110 |
| Shifter | 1 | •• | |
| SLL rd,rt,sa Shift Left Logical rd=rt< <sa 000000="" rs="" rt="" td="" ="" <=""><td>rd</td><td>sa</td><td>000000</td></sa> | rd | sa | 000000 |
| SLLV rd,rt,rs Shift Left Logical Variable rd=rt< <rs 000000="" rs="" rt<="" td=""><td></td><td>00000</td><td>000100</td></rs> | | 00000 | 000100 |
| SRA rd,rt,sa | | sa | 000011 |
| SRAV rd,rt,rs Shift Right Arithmetic Variable rd=rt>>rs 000000 rs rt | | 00000 | 000111 |
| SRL rd,rt,sa Shift Right Logical rd=rt>>sa 000000 rs rt | | sa | 000010 |
| SRLV rd,rt,rs Shift Right Logical Variable rd=rt>>rs 000000 rs rt | | 00000 | 000110 |
| Multiply | | 100000 | 000110 |
| DIV rs,rt Divide HI=rs%rt; LO=rs/rt 000000 rs rt | Too | 00000000 | 011010 |
| DIVU rs,rt Divide Unsigned HI=rs%rt; LO=rs/rt 000000 rs rt | _ | 0000000 | |
| MFHI rd Move From HI rd=HI 000000 0000000000 | | 000000 | 010000 |
| MFLO rd Move From LO rd=LO 000000 0000000000000000000000000000 | _ | 00000 | 010010 |
| | | | 010001 |
| | | 0000000 | |
| MULT rs,rt Multiply HI,LO=rs*rt 000000 rs rt | _ | 0000000 | |
| MULTU rs,rt Multiply Unsigned HI,LO=rs*rt 000000 rs rt | | 0000000 | |
| Branch | 1000 | 30000000 | 011001 |
| BEQ rs,rt,offset Branch On Equal if(rs==rt) pc+=offset*4 000100 rs rt | offs | set | |
| | 1 offs | | |
| | 1 offs | | |
| | 0 offs | | |
| | 0 offs | | |
| | _ | | |
| BITZ is offset Branch $0n < 0$ $if(rs < 0)$ $nc + = offset *4$ $nc = 0.00001 rs$ | | J-L | |
| BLTZ rs, offset Branch On < 0 if $(rs<0)$ pc+=offset*4 000001 rs 0000 | _ | set | |
| BLTZAL rs, offset Branch On < 0 And Link r31=pc; if(rs < 0) pc+=offset*4 000001 rs 1000 | 0 offs | | |
| BLTZAL rs,offset Branch On < 0 And Link | _ | | 001101 |
| BLTZAL rs,offsetBranch On < 0 And Linkr31=pc; if(rs<0) pc+=offset*4000001 rs1000BNE rs,rt,offsetBranch On Not Equalif(rs!=rt) pc+=offset*4000101 rsrtBREAKBreakpointepc=pc; pc=0x3c000000 code | 0 offs | | 001101 |
| BLTZAL rs,offsetBranch On < 0 And Linkr31=pc; if(rs<0) pc+=offset*4000001 rs1000BNE rs,rt,offsetBranch On Not Equalif(rs!=rt) pc+=offset*4000101 rsrtBREAKBreakpointepc=pc; pc=0x3c000000 codeJ targetJumppc=pc_upper (target<<2) | 0 offs | | 001101 |
| BLTZAL rs,offsetBranch On < 0 And Link $r31=pc$; if(rs<0) $pc+=offset*4$ 000001 rs1000BNE rs,rt,offsetBranch On Not Equalif(rs!=rt) $pc+=offset*4$ 000101 rsrtBREAKBreakpoint $epc=pc$; $pc=0x3c$ 000000 codeJ targetJump $pc=pc_upper (target<<2)$ 000010 targetJAL targetJump And Link $r31=pc$; $pc=target<<2$ 000011 target | 0 offs | set | |
| BLTZAL rs,offset Branch On < 0 And Link r31=pc; if(rs<0) pc+=offset*4 000001 rs 10000 BNE rs,rt,offset Branch On Not Equal if(rs!=rt) pc+=offset*4 000101 rs rt BREAK Breakpoint epc=pc; pc=0x3c 000000 code J target Jump pc=pc_upper (target<<2) 000010 target JAL target Jump And Link r31=pc; pc=target<<2 000011 target JALR rs Jump And Link Register rd=pc; pc=rs 000000 rs 00000 rs 000000 rs 00000000 | 0 offs offs 0 rd | o0000 | 001001 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs offs 0 rd 0000 | 00000 000000 | 001001 001000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs offs 0 rd 0 0000 rd | 00000 000000 0000000 | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs 0 rd 0000 rd rd | 00000 0000000 0000000 0000000 | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs 0 rd 0000 rd rd | 00000 0000000 0000000 0000000 | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs 0 rd 0000 rd rd 00000 | 00000 0000000 0000000 0000000 0000000 | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs 0 rd 0000 rd rd 00000 | 00000 0000000 0000000 0000000 0000000 | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs 0 rd 0000 rd rd 00000 offs offs | 00000 0000000 0000000 0000000 000000 | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs 0 rd 0000 rd rd 00000 offs offs | 00000 0000000 0000000 0000000 000000 | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | O rd OOOO rd rd rd Ooffs offs offs | 00000 0000000 0000000 0000000 0000000 set set | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs offs offs 0 rd 00000 rd rd offs offs offs offs | 00000 0000000 0000000 0000000 0000000 set set set | 001001 001000 0000 |
| BLTZAL rs,offset Branch On < 0 And Link | 0 offs offs 0 rd 00000 rd rd cooooo offs offs offs offs | 00000 0000000 0000000 0000000 0000000 set set set set | 001001 001000 0000 |
| BLTZAL rs, offset Branch On < 0 And Link r31=pc; if(rs<0) pc+=offset*4 000001 rs 1000 BNE rs, rt, offset Branch On Not Equal if(rs!=rt) pc+=offset*4 000101 rs rt BREAK Breakpoint epc=pc; pc=0x3c 000000 code 000000 code J target Jump pc=pc_upper (target < 2) | 0 offs offs offs 0 rd 00000 rd rd offs offs offs offs | 00000 0000000 0000000 0000000 0000000 set set set set set | 001001 001000 0000 |

Notes: The immediate values are normally sign extended.
The opcodes ADD and ADDU are equivalent in the Plasma CPU since ALU operations don't cause exceptions.
The program counter (pc) points to eight bytes past the currently executing instruction.