# Overview

CS01 is a thirty-two-bit processor modelled after the RISCV ISA. Only a subset of the RISCV ISA is implemented. The processor features 32, 32-bit integer registers and 32, 32-bit floating-point registers. The processor has the most prominent features from real processors with some of the more complex details left out.

# Programming Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Registers | | | |
|  | 31 0 |  |  | 31 0 |
|  | x0 / zero |  |  | f0 / zero |
|  | x1 / ra |  |  | f1 |
|  | x2 / fp |  |  | … |
|  | x3-x13 / s1-s11 |  |  | … |
|  | x14 / sp |  |  | … |
|  | x15 / tp |  |  | … |
|  | x16-x17 / v0-v1 |  |  | … |
|  | x18-x25 / a0-a7 |  |  | … |
|  | x26-x30 / t0 – t4 |  |  | f30 |
|  | x31 / gp |  |  | f31 |
|  | pc |  |  |  |

x? refers to an integer register. f? refers to a floating-point register

Registers x0 and f0 are always zero. f0 is positive zero.

x14 / sp is the stack pointer.

pc is the program counter.

# Nomenclature

The size of values are referred to as the following:

|  |  |  |
| --- | --- | --- |
| Size | Alternate |  |
| 1 | byte | byte |
| 2 | wyde | half-word |
| 4 | tetra | word |
| 5 | penta |  |
| 8 | octa | double-word |
| 10 | deci |  |
| 16 | hexi |  |

# Instruction Set

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LUI | imm31..12 | | | | | | | | | Rd | 55 |  |
| AUIPC | imm31..12 | | | | | | | | | Rd | 23 |  |
| JAL | 20 | imm10..1 | | | | | 11 | imm19..12 | | Rd | 111 |  |
| JALR | imm11..0 | | | | | | | Rs1 | 0 | Rd | 103 |  |
| BEQ | imm12.10..5 | | | | Rs2 | | | Rs1 | 0 | imm4..1.11 | 99 |  |
| BNE | imm12.10..5 | | | | Rs2 | | | Rs1 | 1 | imm4..1.11 | 99 |  |
| BLT | imm12.10..5 | | | | Rs2 | | | Rs1 | 4 | imm4..1.11 | 99 |  |
| BGE | imm12.10..5 | | | | Rs2 | | | Rs1 | 5 | imm4..1.11 | 99 |  |
| BLTU | imm12.10..5 | | | | Rs2 | | | Rs1 | 6 | imm4..1.11 | 99 |  |
| BGEU | imm12.10..5 | | | | Rs2 | | | Rs1 | 7 | imm4..1.11 | 99 |  |
| LB | imm11..0 | | | | | | | Rs1 | 0 | Rd | 3 | LDB |
| LH | imm11..0 | | | | | | | Rs1 | 1 | Rd | 3 | LDW |
| LW | imm11..0 | | | | | | | Rs1 | 2 | Rd | 3 | LDT |
| LBU | imm11..0 | | | | | | | Rs1 | 4 | Rd | 3 | LDBU |
| LHU | imm11..0 | | | | | | | Rs1 | 5 | Rd | 3 | LDWU |
| FLW | imm11..0 | | | | | | | Rs1 | 2 | FRd | 7 | FLDT |
| SB | imm11..5 | | | | Rs2 | | | Rs1 | 0 | Imm4..0 | 35 | STB |
| SH | imm11..5 | | | | Rs2 | | | Rs1 | 1 | Imm4..0 | 35 | STW |
| SW | imm11..5 | | | | Rs2 | | | Rs1 | 2 | Imm4..0 | 35 | STT |
| FSW | imm11..5 | | | | FRs2 | | | Rs1 | 2 | Imm4..0 | 39 | FSTT |
| ADDI | imm11..0 | | | | | | | Rs1 | 0 | Rd | 19 |  |
| SLTI | imm11..0 | | | | | | | Rs1 | 2 | Rd | 19 |  |
| SLTUI | imm11..0 | | | | | | | Rs1 | 3 | Rd | 19 |  |
| XORI | imm11..0 | | | | | | | Rs1 | 4 | Rd | 19 | EORI |
| ORI | imm11..0 | | | | | | | Rs1 | 6 | Rd | 19 |  |
| ANDI | imm11..0 | | | | | | | Rs1 | 7 | Rd | 19 |  |
| SLLI | 0 | | | | shamt | | | Rs1 | 1 | Rd | 19 | SHLI |
| SRLI | 0 | | | | shamt | | | Rs1 | 5 | Rd | 19 | SHRI |
| SRAI | 16 | | | | shamt | | | Rs1 | 5 | Rd | 19 | ASRI |
| ADD | 0 | | | | Rs2 | | | Rs1 | 0 | Rd | 51 |  |
| SUB | 32 | | | | Rs2 | | | Rs1 | 0 | Rd | 51 |  |
| MUL | 1 | | | | Rs2 | | | Rs1 | 0 | Rd | 51 |  |
| SLL | 0 | | | | Rs2 | | | Rs1 | 1 | Rd | 51 | SHL |
| SLT | 0 | | | | Rs2 | | | Rs1 | 2 | Rd | 51 |  |
| SLTU | 0 | | | | Rs2 | | | Rs1 | 3 | Rd | 51 |  |
| XOR | 0 | | | | Rs2 | | | Rs1 | 4 | Rd | 51 | EOR |
| SRL | 0 | | | | Rs2 | | | Rs1 | 5 | Rd | 51 | SHR |
| SRA | 32 | | | | Rs2 | | | Rs1 | 5 | Rd | 51 | ASR |
| OR | 0 | | | | Rs2 | | | Rs1 | 6 | Rd | 51 |  |
| AND | 0 | | | | Rs2 | | | Rs1 | 7 | Rd | 51 |  |
| FADD | 0 | | 00 | | FRs2 | | | FRs1 | rm | FRd | 83 |  |
| FSUB | 1 | | 00 | | FRs2 | | | FRs1 | rm | FRd | 83 |  |
| FMUL | 2 | | 00 | | FRs2 | | | FRs1 | rm | FRd | 83 |  |
| FDIV | 3 | | 00 | | FRs2 | | | FRs1 | rm | FRd | 83 |  |
| FSQRT | 11 | | 00 | | 0 | | | FRs1 | rm | FRd | 83 |  |
| FEQ | 20 | | 00 | | FRs2 | | | FRs1 | 2 | Rd | 83 |  |
| FLT | 20 | | 00 | | FRs2 | | | FRs1 | 1 | Rd | 83 |  |
| FLE | 20 | | 00 | | FRs2 | | | FRs1 | 0 | Rd | 83 |  |
| FCVT.W.S | 24 | | 00 | | 0 | | | FRs1 | rm | Rd | 83 |  |
| FCVT.WU.S | 24 | | 00 | | 1 | | | FRs1 | rm | Rd | 83 |  |
| FCVT.S.W | 25 | | 00 | | 0 | | | Rs1 | rm | FRd | 83 |  |
| FCVT.S.WU | 25 | | 00 | | 1 | | | Rs1 | rm | FRd | 83 |  |
| FENCE | 0 | | | pred | | succ | | 0 | 0 | 0 | 15 |  |
| FENCE.I | 0 | | | 0 | | 0 | | 0 | 1 | 0 | 15 |  |
| ECALL | 0 | | | | 0 | | | 0 | 0 | 0 | 115 |  |
| ERET | 8 | | | | 0 | | | 0 | 0 | 0 | 115 |  |
| RDCYCLE | 96 | | | | 0 | | | 0 | 2 | Rd | 115 |  |
| RDCYCLEH | 100 | | | | 0 | | | 0 | 2 | Rd | 115 |  |
| RDTIME | 96 | | | | 1 | | | 0 | 2 | Rd | 115 |  |
| RDTIMEH | 100 | | | | 1 | | | 0 | 2 | Rd | 115 |  |
| RDINSTRET | 96 | | | | 2 | | | 0 | 2 | Rd | 115 |  |
| RDINSTRETH | 100 | | | | 2 | | | 0 | 2 | Rd | 115 |  |

## ADD – Addition

**Description**:

Add two values using two’s complement addition, which are in Rs1 and Rs2 or an immediate value and place the sum in the destination register Rd.

**Instruction Format**: R2, RI

**Exceptions**: none

## AUIPC – Add Upper Immediate to PC

**Description**:

This instruction adds the upper 20 bits of the program counter to an immediate supplied by the instruction and stores the result in the destination register Rd. This instruction may be used to generate addresses relative to the program counter.

## FENCE[.I]

**Description**:

With this core the fence instruction is a nop operation. Memory instructions are not buffered and always execute in order. Fencing is used to control order on machines where the order of memory operation may not be in program order.

## LUI – Load Upper Immediate

**Description**:

The LUI instruction sets the upper 20 bits of the destination register Rd to the constant supplied in the instruction and zeros out the lower 12 bits of the destination register. This instruction has a brownfield extension to the RISCV instruction set.

Brownfield Extension

If the target register of a LUI instruction is x0, then the constant in the LUI instruction is applied as the upper 20 bits of the constant for the next instruction.