🧠 Instruction Set

> **Version**: `0.5.0`
> **Total Instructions Supported**: `73`
> **Operand Size**: Up to **2 operands**, each of **64-bits**

🧩 Operand Types

Instructions in SASM accept a variety of operand types:

??? example "Immediate Values"

```
0, 1, 2, 3
```

??? example "Constants"

```
M, N, O, P
```

??? example "Registers"

```
L0, L1, L2, L3
```

🗂️ Register Dereferencing

SASM registers are identified by **unique IDs**. Each register can be accessed either as a **reference** or as a **value** using *compile-time functions*.

```
ref(L1)  ; reference to register L1
val(L1)  ; value stored in register L1
```

> These are **compile-time functions** that tell the assembler how to interpret operands.

🔗 Related References

- 📄 Compile-time Functions
- 📄 Register Reference

Instruction Set - Quick Reference

=== "ARITHMETIC"

```
=== "INTEGER"

    | OPCODE | INST |
    |--------|------|
    | **13** | [ADDI](Arithmetic/Addition.md#ADDI) |
    | **14** | [SUBI](Arithmetic/Subtraction.md#SUBI) |
    | **15** | [MULI](Arithmetic/Multiplication.md#MULI) |
    | **16** | [DIVI](Arithmetic/Division.md#DIVI) |
    | **17** | [MODI](Arithmetic/Modulus.md#MODI) |

=== "UNSIGNED"

    | OPCODE | INST |
    |--------|------|
    | **18** | [ADDU](Arithmetic/Addition.md#ADDU) |
    | **19** | [SUBU](Arithmetic/Subtraction.md#SUBU) |
    | **20** | [MULU](Arithmetic/Multiplication.md#MULU) |
    | **21** | [DIVU](Arithmetic/Division.md#DIVU) |
    | **22** | [MODU](Arithmetic/Modulus.md#MODU) |

=== "FLOAT"

    | OPCODE | INST |
    |--------|------|
    | **23** | [ADDF](Arithmetic/Addition.md#ADDF) |
    | **24** | [SUBF](Arithmetic/Subtraction.md#SUBF) |
    | **25** | [MULF](Arithmetic/Multiplication.md#MULF) |
    | **26** | [DIVF](Arithmetic/Division.md#DIVF) |
```

=== "UNDOCUMENTED"

```
=== "RELATIONAL"

    === "INTEGER"

        | OPCODE | INST |
        |--------|------|
        | **35** | [EQI](../reference/missing) |
        | **36** | [GEI](../reference/missing) |
        | **37** | [GTI](../reference/missing) |
        | **38** | [LEI](../reference/missing) |
        | **39** | [LTI](../reference/missing) |
        | **40** | [NEI](../reference/missing) |

    === "UNSIGNED"

        | OPCODE | INST |
```

```
        |--------|------|
        | **41** | [EQU](../reference/missing) |
        | **42** | [GEU](../reference/missing) |
        | **43** | [GTU](../reference/missing) |
        | **44** | [LEU](../reference/missing) |
        | **45** | [LTU](../reference/missing) |
        | **46** | [NEU](../reference/missing) |

    === "FLOAT"

        | OPCODE | INST |
        |--------|------|
        | **47** | [EQF](../reference/missing) |
        | **48** | [GEF](../reference/missing) |
        | **49** | [GTF](../reference/missing) |
        | **50** | [LEF](../reference/missing) |
        | **51** | [LTF](../reference/missing) |
        | **52** | [NEF](../reference/missing) |

=== "TYPE CASTING"

    | OPCODE | INST |
    |--------|------|
    | **57** | [I2F](../reference/missing) |
    | **58** | [U2F](../reference/missing) |
    | **59** | [F2I](../reference/missing) |
    | **60** | [F2U](../reference/missing) |

=== "MEMORY ACCESS"

    | OPCODE | INST |
    |--------|------|
    | **61** | [READ1U](../reference/missing) |
    | **62** | [READ2U](../reference/missing) |
    | **63** | [READ4U](../reference/missing) |
    | **64** | [READ8U](../reference/missing) |
    | **65** | [READ1I](../reference/missing) |
    | **66** | [READ2I](../reference/missing) |
    | **67** | [READ4I](../reference/missing) |
    | **68** | [READ8I](../reference/missing) |
    | **69** | [WRITE1](../reference/missing) |
    | **70** | [WRITE2](../reference/missing) |
    | **71** | [WRITE4](../reference/missing) |
    | **72** | [WRITE8](../reference/missing) |

=== "SEQUENCE CONTROL"

    | OPCODE | INST |
    |--------|------|
    |  **1** | [INVOK](../reference/missing) |
    |  **2** | [RETVL](../reference/missing) |
    |  **8** | [CALL](../reference/missing) |
    |  **9** | [LOOP](../reference/missing) |
```

```
    | **27** | [JMPU](../reference/missing) |
    | **28** | [JMPC](../reference/missing) |
    | **33** | [RET](../reference/missing) |

=== "BINARY OPERATIONS"

    | OPCODE | INST |
    |--------|------|
    | **34** | [NOT](../reference/missing) |
    | **53** | [ORB](../reference/missing) |
    | **54** | [XOR](../reference/missing) |
    | **55** | [SHR](../reference/missing) |
    | **56** | [SHL](../reference/missing) |
    | **29** | [ANDB](../reference/missing) |
    | **30** | [NOTB](../reference/missing) |

=== "MISC"

    | OPCODE | INST |
    |--------|------|
    |  **0** | [DONOP](../reference/missing) |
    |  **3** | [PUSHR](../reference/missing) |
    |  **4** | [SPOPR](../reference/missing) |
    |  **5** | [SHUTS](../reference/missing) |
    |  **6** | [SETR](../reference/missing) |
    |  **7** | [GETR](../reference/missing) |
    | **10** | [PUSH](../reference/missing) |
    | **11** | [SPOP](../reference/missing) |
    | **12** | [SWAP](../reference/missing) |
    | **31** | [COPY](../reference/missing) |
    | **32** | [DUPS](../reference/missing) |
```

INSTRUCTION DOCUMENTATION TEMPLATE:

[OPCODE] — *[OPERATION]* {#[OPCODE]}

```

```

=== "[OPCODE] Example"

```
  ```linenums="1" hl_lines="1 3 5"

  ```
```

=== "[OPCODE] Properties"

```
| Opcode | Operand Type        | Destination   |
|--------|---------------------|---------------|
| 00     | 64-bit       Value | L_ (implicit) |

Identified as memonic [#[OPCODE]](#[OPCODE]), [OPCODE] is used to
```