The following opcodes are used for **addition**:

- ADDI — Add Signed Integer
- ADDU — Add Unsigned Integer
- ADDF — Add Floating Point

ADDI — *Add Signed Integer* {#ADDI}

???+ note "Properties"

```
    |-----------------|---------------------------|
    | Property        | Value                     |
    |-----------------|---------------------------|
    |  `Opcode`       | #13                       |
    |  `Type`         | *Arithmetic*              |
    |  `Operand Type` | Signed 64-bit integer     |
    |  `Destination`  | L2 (implicit)             |
    |-----------------|---------------------------|
```

=== "ADDI Algorithm"

```
    ```
        L2 = L2 + <signed_imm>
        L2 = L2 + <reg_val>
        L2 = L2 + <const>
    ```
```

=== "ADDI Example"

```
    ```linenums="1" hl_lines="1 3 5 7"
    ; imm +ve
        ADDI    1
    ; imm -ve
        ADDI    -123
    ; reg val
        ADDI    val(QT)
    ; const
        ADDI    SOME_CONST_VAL
    ```
```

ADDU — *Add Unsigned Integer* {#ADDU}

Use ADDU to add an unsigned value to whatever value
is stored within the L3 register. If the register L3

is not set, then initial value of L3 is assumed to be
0, and not a garbage value.

```
    |-----------------|---------------------------|
    | Property        | Value                     |
    |-----------------|---------------------------|
    |  `Opcode`       | #18                       |
    |  `Type`         | *Arithmetic*              |
    |  `Operand Type` | Unsigned 64-bit value     |
    |  `Destination`  | L3 (implicit)             |
    |-----------------|---------------------------|
```

=== "ADDU Algorithm"

```
``
    L3 = L3 + <unsigned_imm>
    L3 = L3 + <reg_val>
    L3 = L3 + <const>
``
```

=== "ADDU Example"

```
``linenums="1" hl_lines="1 3 5"
; imm +ve
    ADDU    1
; reg val
    ADDU    val(QT)
; const
    ADDU    SOME_CONST_VAL
``
```

ADDF — *Add Float value* {#ADDF}

Use ADDF to add a floating point value to whatever value
is stored within the L1 register. If the register L1
is not set, then initial value of L1 is assumed to be
0, and not a garbage value.

```
    |-----------------|---------------------------|
    | Property        | Value                     |
    |-----------------|---------------------------|
    |  `Opcode`       | #23                       |
    |  `Type`         | *Arithmetic*              |
    |  `Operand Type` | 64-bit float value        |
```

```
    |  `Destination`  | L1 (implicit)             |
    |-----------------|---------------------------|
```

=== "ADDF Algorithm"

```
    ```
        L1 = L1 + <float>
        L1 = L1 + <reg_val>
        L1 = L1 + <const>
    ```
```

=== "ADDF Example"

```
    ```linenums="1" hl_lines="1 3 5"
    ; imm float
        ADDF    3.14
    ; reg val
        ADDF    val(QT)
    ; const
        ADDF    SOME_CONST_VAL

    ```
```
