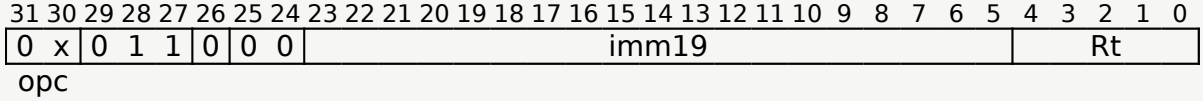
LDR (literal)

Load Register (literal) calculates an address from the PC value and an immediate offset, loads a word from memory, and writes it to a register. For information about memory accesses, see *Load/Store addressing modes*.



32-bit (opc == 00)

```
LDR <Wt>, <label>
```

64-bit (opc == 01)

```
LDR <Xt>, <label>
```

```
integer t = UInt(Rt);
MemOp memop = if opc == '11' then MemOp_PREFETCH else MemOp_LOAD;
constant integer size = 4 << UInt(opc<0>);
boolean signed = opc == '10';

bits(64) offset = SignExtend(imm19:'00', 64);
```

Assembler Symbols

<Wt>	Is the 32-bit name of the general-purpose register to be loaded, encoded in the "Rt" field.
<Xt>	Is the 64-bit name of the general-purpose register to be loaded, encoded in the "Rt" field.
<label>	Is the program label from which the data is to be loaded. Its offset from the address of this instruction, in the range +/-1MB, is encoded as "imm19" times 4.

Operation

```
bits(64) address = PC64 + offset;
bits(size*8) data;
boolean privileged = PSTATE.EL != EL0;

AccessDescriptor accdesc = CreateAccDescGPR(memop, FALSE, privileged, F
case memop of
    when MemOp_LOAD
        data = Mem[address, size, accdesc];
        if signed then
            X[t, 64] = SignExtend(data, 64);
        else
            X[t, size*8] = data;
```

```
    when MemOp_PREFETCH
        Prefetch(address, t<4:0>);
```

Operational information

If PSTATE.DIT is 1, the timing of this instruction is insensitive to the value of the data being loaded or stored.

Internal version only: isa v33.64, AdvSIMD v29.12, pseudocode no_diffs_2023_09_RC2, sve v2023-06_rel ; Build timestamp: 2023-09-18T17:56