## LDRAA, LDRAB

Load Register, with pointer authentication. This instruction authenticates an address from a base register using a modifier of zero and the specified key, adds an immediate offset to the authenticated address, and loads a 64-bit doubleword from memory at this resulting address into a register.

Key A is used for `LDRAA`. Key B is used for `LDRAB`.

If the authentication passes, the PE behaves the same as for an `LDR` instruction. For information on behavior if the authentication fails, see *Faulting on pointer authentication*.

The authenticated address is not written back to the base register, unless the pre-indexed variant of the instruction is used. In this case, the address that is written back to the base register does not include the pointer authentication code.

For information about memory accesses, see *Load/Store addressing modes*.

### Unscaled offset
**(FEAT_PAuth)**

| 31 30 | 29 28 27 | 26 | 25 24 | 23 | 22 | 21 | 20 19 18 17 16 15 14 13 12 | 11 | 10 | 9 8 7 6 5 | 4 3 2 1 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 1 | 1 1 1 | 0 | 0 0 | M | S | 1 | imm9 | W | 1 | Rn | Rt |
| size | | | | | | | | | | | |

**Key A, offset (M == 0 && W == 0)**

```
LDRAA <Xt>, [<Xn|SP>{, #<simm>}]
```

**Key A, pre-indexed (M == 0 && W == 1)**

```
LDRAA <Xt>, [<Xn|SP>{, #<simm>}]!
```

**Key B, offset (M == 1 && W == 0)**

```
LDRAB <Xt>, [<Xn|SP>{, #<simm>}]
```

**Key B, pre-indexed (M == 1 && W == 1)**

```
LDRAB <Xt>, [<Xn|SP>{, #<simm>}]!
```

```
if !IsFeatureImplemented(FEAT_PAuth) then UNDEFINED;
integer t = UInt(Rt);
integer n = UInt(Rn);
boolean wback = (W == '1');
boolean use_key_a = (M == '0');
bits(10) S10 = S:imm9;
bits(64) offset = LSL(SignExtend(S10, 64), 3);
boolean tagchecked = wback || n != 31;
```

**Assembler Symbols**

| | |
|---|---|
| <Xt> | Is the 64-bit name of the general-purpose register to be transferred, encoded in the "Rt" field. |
| <Xn\|SP> | Is the 64-bit name of the general-purpose base register or stack pointer, encoded in the "Rn" field. |
| <simm> | Is the optional signed immediate byte offset, a multiple of 8 in the range -4096 to 4088, defaulting to 0 and encoded in the "S:imm9" field as <simm>/8. |

**Operation**

```
bits(64) address;
bits(64) data;
boolean privileged = PSTATE.EL != EL0;
boolean wb_unknown = FALSE;

AccessDescriptor accdesc = CreateAccDescGPR(MemOp_LOAD, FALSE, privileg
if wback && n == t && n != 31 then
    Constraint c = ConstrainUnpredictable(Unpredictable_WBOVERLAPLD);
    assert c IN {Constraint_WBSUPPRESS, Constraint_UNKNOWN, Constraint_
    case c of
        when Constraint_WBSUPPRESS wback = FALSE;     // writeback is su
        when Constraint_UNKNOWN    wb_unknown = TRUE;     // writeback i
        when Constraint_UNDEF      UNDEFINED;
        when Constraint_NOP        EndOfInstruction();

if n == 31 then
    address = SP[];
else
    address = X[n, 64];

if use_key_a then
    address = AuthDA(address, X[31, 64], TRUE);
else
    address = AuthDB(address, X[31, 64], TRUE);

if n == 31 then
    CheckSPAlignment();

address = address + offset;
data = Mem[address, 8, accdesc];
X[t, 64] = data;

if wback then
    if wb_unknown then
        address = bits(64) UNKNOWN;
    if n == 31 then
        SP[] = address;
    else
        X[n, 64] = address;
```

**Operational information**

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