STLUR

Store-Release Register (unscaled) calculates an address from a base register value and an immediate offset, and stores a 32-bit word or a 64-bit doubleword to the calculated address, from a register.

The instruction has memory ordering semantics as described in *Load-Acquire, Load-AcquirePC, and Store-Release*

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

Unscaled offset
(FEAT_LRCPC2)

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	x	0	1	1	0	0	1	0	0	0	imm9	0	0	Rn	Rt
size								opc							

32-bit (size == 10)

```
STLUR <Wt>, [<Xn|SP>{, #<simm>}]
```

64-bit (size == 11)

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

```
integer scale = UInt(size);
bits(64) offset = SignExtend(imm9, 64);
```

Assembler Symbols

<Wt>	Is the 32-bit name of the general-purpose register to be transferred, encoded in the "Rt" field.
<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, in the range -256 to 255, defaulting to 0 and encoded in the "imm9" field.

Shared Decode

```
integer n = UInt(Rn);
integer t = UInt(Rt);

constant integer datasize = 8 << scale;
boolean tagchecked = n != 31;
```

Operation

```
bits(64) address;
bits(datasize) data;

AccessDescriptor accdesc;
accdesc = CreateAccDescAcqRel(MemOp_STORE, tagchecked);

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

address = address + offset;

data = X[t, datasize];
Mem[address, datasize DIV 8, accdesc] = data;
```

Operational information

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