SWP, SWPA, SWPAL, SWPL

Swap word or doubleword in memory atomically loads a 32-bit word or 64-bit doubleword from a memory location, and stores the value held in a register back to the same memory location. The value initially loaded from memory is returned in the destination register.

- If the destination register is not one of WZR or XZR, `SWPA` and `SWPAL` load from memory with acquire semantics.
- `SWPL` and `SWPAL` store to memory with release semantics.
- `SWP` has neither acquire nor release semantics.

For more information about memory ordering semantics, see *Load-Acquire, Store-Release*.
For information about memory accesses, see *Load/Store addressing modes*.

Integer
(FEAT_LSE)

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	1	1	1	0	0	0	A	R	1	Rs					1	0	0	0	0	0	Rn					Rt				
size																															

32-bit SWP (size == 10 && A == 0 && R == 0)

```
SWP <Ws>, <Wt>, [<Xn|SP>]
```

32-bit SWPA (size == 10 && A == 1 && R == 0)

```
SWPA <Ws>, <Wt>, [<Xn|SP>]
```

32-bit SWPAL (size == 10 && A == 1 && R == 1)

```
SWPAL <Ws>, <Wt>, [<Xn|SP>]
```

32-bit SWPL (size == 10 && A == 0 && R == 1)

```
SWPL <Ws>, <Wt>, [<Xn|SP>]
```

64-bit SWP (size == 11 && A == 0 && R == 0)

```
SWP <Xs>, <Xt>, [<Xn|SP>]
```

64-bit SWPA (size == 11 && A == 1 && R == 0)

```
        SWPA <Xs>, <Xt>, [<Xn|SP>]
```

64-bit SWPAL (size == 11 && A == 1 && R == 1)

```
        SWPAL <Xs>, <Xt>, [<Xn|SP>]
```

64-bit SWPL (size == 11 && A == 0 && R == 1)

```
        SWPL <Xs>, <Xt>, [<Xn|SP>]
```

```
if !IsFeatureImplemented(FEAT_LSE) then UNDEFINED;

integer t = UInt(Rt);
integer n = UInt(Rn);
integer s = UInt(Rs);

constant integer datasize = 8 << UInt(size);
integer regsize = if datasize == 64 then 64 else 32;
boolean acquire = A == '1' && Rt != '11111';
boolean release = R == '1';
boolean tagchecked = n != 31;
```

Assembler Symbols

<Ws>	Is the 32-bit name of the general-purpose register to be stored, encoded in the "Rs" field.
<Wt>	Is the 32-bit name of the general-purpose register to be loaded, encoded in the "Rt" field.
<Xs>	Is the 64-bit name of the general-purpose register to be stored, encoded in the "Rs" field.
<Xt>	Is the 64-bit name of the general-purpose register to be loaded, 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.

Operation

```
bits(64) address;
bits(datasize) data;
bits(datasize) store_value;
AccessDescriptor accdesc = CreateAccDescAtomicOp(MemAtomicOp_SWP, acqui

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

store_value = X[s, datasize];

bits(datasize) comparevalue = bits(datasize) UNKNOWN;    // Irrelevant
```

```
data = MemAtomic(address, comparevalue, store_value, accdesc);

X[t, regsize] = ZeroExtend(data, regsize);
```

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