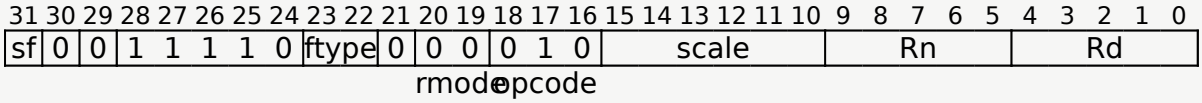
## SCVTF (scalar, fixed-point)

Signed fixed-point Convert to Floating-point (scalar). This instruction converts the signed value in the 32-bit or 64-bit general-purpose source register to a floating-point value using the rounding mode that is specified by the *FPCR*, and writes the result to the SIMD&FP destination register.

A floating-point exception can be generated by this instruction. Depending on the settings in *FPCR*, the exception results in either a flag being set in *FPSR*, or a synchronous exception being generated. For more information, see *Floating-point exception traps*.

Depending on the settings in the *CPACR_EL1*, *CPTR_EL2*, and *CPTR_EL3* registers, and the Security state and Exception level in which the instruction is executed, an attempt to execute the instruction might be trapped.




**32-bit to half-precision (sf == 0 && ftype == 11)**
**(FEAT_FP16)**

```
SCVTF <Hd>, <Wn>, #<fbits>
```

**32-bit to single-precision (sf == 0 && ftype == 00)**

```
SCVTF <Sd>, <Wn>, #<fbits>
```

**32-bit to double-precision (sf == 0 && ftype == 01)**

```
SCVTF <Dd>, <Wn>, #<fbits>
```

**64-bit to half-precision (sf == 1 && ftype == 11)**
**(FEAT_FP16)**

```
SCVTF <Hd>, <Xn>, #<fbits>
```

**64-bit to single-precision (sf == 1 && ftype == 00)**

```
SCVTF <Sd>, <Xn>, #<fbits>
```

**64-bit to double-precision (sf == 1 && ftype == 01)**

```
SCVTF <Dd>, <Xn>, #<fbits>
```

```
if ftype == '10' || (ftype == '11' && !IsFeatureImplemented(FEAT_FP16))
```

```
integer d = UInt(Rd);
integer n = UInt(Rn);

constant integer intsize = 32 << UInt(sf);
constant integer decode_fltsize = 8 << UInt(ftype EOR '10');
FPRounding rounding;

if sf == '0' && scale<5> == '0' then UNDEFINED;
integer fracbits = 64 - UInt(scale);

rounding = FPRoundingMode(FPCR[]);
```

**Assembler Symbols**

| | |
|---|---|
| <Dd> | Is the 64-bit name of the SIMD&FP destination register, encoded in the "Rd" field. |
| <Hd> | Is the 16-bit name of the SIMD&FP destination register, encoded in the "Rd" field. |
| <Sd> | Is the 32-bit name of the SIMD&FP destination register, encoded in the "Rd" field. |
| <Xn> | Is the 64-bit name of the general-purpose source register, encoded in the "Rn" field. |
| <Wn> | Is the 32-bit name of the general-purpose source register, encoded in the "Rn" field. |
| <fbits> | For the 32-bit to double-precision, 32-bit to half-precision and 32-bit to single-precision variant: is the number of bits after the binary point in the fixed-point source, in the range 1 to 32, encoded as 64 minus "scale".<br><br>For the 64-bit to double-precision, 64-bit to half-precision and 64-bit to single-precision variant: is the number of bits after the binary point in the fixed-point source, in the range 1 to 64, encoded as 64 minus "scale". |

**Operation**

```
CheckFPEnabled64();

FPCRType fpcr = FPCR[];
constant boolean merge = IsMerging(fpcr);
constant integer fltsize = if merge then 128 else decode_fltsize;
bits(fltsize) fltval;
bits(intsize) intval;

intval = X[n, intsize];
fltval = if merge then V[d, fltsize] else Zeros(fltsize);
Elem[fltval, 0, decode_fltsize] = FixedToFP(intval, fracbits, FALSE, fpcr
V[d, fltsize] = fltval;
```

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