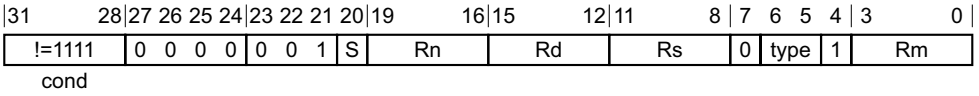
F5.1.47 EOR, EORS (register-shifted register)

Bitwise Exclusive OR (register-shifted register) performs a bitwise Exclusive OR of a register value and a register-shifted register value. It writes the result to the destination register, and can optionally update the condition flags based on the result.

A1



Flag setting variant

Applies when `S == 1`.

```
EORS{<c>}{<q>} {<Rd>,} <Rn>, <Rm>, <type> <Rs>
```

Not flag setting variant

Applies when `S == 0`.

```
EOR{<c>}{<q>} {<Rd>,} <Rn>, <Rm>, <type> <Rs>
```

Decode for all variants of this encoding

```
d = UInt(Rd);  n = UInt(Rn);  m = UInt(Rm);  s = UInt(Rs);
setflags = (S == '1');  shift_t = DecodeRegShift(type);
if d == 15 || n == 15 || m == 15 || s == 15 then UNPREDICTABLE;
```

Notes for all encodings

For more information about the CONSTRAINED UNPREDICTABLE behavior, see Appendix K1 *Architectural Constraints on UNPREDICTABLE behaviors*.

Assembler symbols

`<c>` See *Standard assembler syntax fields* on page F2-2848.

`<q>` See *Standard assembler syntax fields* on page F2-2848.

`<Rd>` Is the general-purpose destination register, encoded in the "Rd" field.

`<Rn>` Is the first general-purpose source register, encoded in the "Rn" field.

`<Rm>` Is the second general-purpose source register, encoded in the "Rm" field.

`<type>` Is the type of shift to be applied to the second source register, encoded in the "type" field. It can have the following values:

- `LSL` when `type = 00`
- `LSR` when `type = 01`
- `ASR` when `type = 10`
- `ROR` when `type = 11`

`<Rs>` Is the third general-purpose source register holding a shift amount in its bottom 8 bits, encoded in the "Rs" field.

Operation

```
if ConditionPassed() then
    EncodingSpecificOperations();
    shift_n = UInt(R[s]<7:0>);
    (shifted, carry) = Shift_C(R[m], shift_t, shift_n, PSTATE.C);
    result = R[n] EOR shifted;
    R[d] = result;
    if setflags then
        PSTATE.N = result<31>;
        PSTATE.Z = IsZeroBit(result);
        PSTATE.C = carry;
        // PSTATE.V unchanged
```