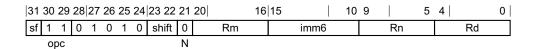
C6.2.14 ANDS (shifted register)

Bitwise AND (shifted register), setting flags, performs a bitwise AND of a register value and an optionally-shifted register value, and writes the result to the destination register. It updates the condition flags based on the result.

This instruction is used by the alias TST (shifted register). See *Alias conditions* for details of when each alias is preferred.



32-bit variant

```
Applies when sf == 0.

ANDS <Wd>>, <Wn>>, <Wm>{, <shift> #<amount>}
```

64-bit variant

```
Applies when sf == 1. 
 ANDS <Xd>, <Xn>, <Xm>{, <shift> #<amount>}
```

Decode for all variants of this encoding

```
integer d = UInt(Rd);
integer n = UInt(Rn);
integer m = UInt(Rm);
integer datasize = if sf == '1' then 64 else 32;

if sf == '0' && imm6<5> == '1' then ReservedValue();

ShiftType shift_type = DecodeShift(shift);
integer shift_amount = UInt(imm6);
```

Alias conditions

Alias	is preferred when
TST (shifted register)	Rd == '11111'

Assembler symbols

<wd></wd>	Is the 32-bit name of the general-purpose destination register, encoded in the "Rd" field.
<wn></wn>	Is the 32-bit name of the first general-purpose source register, encoded in the "Rn" field.
<wm></wm>	Is the 32-bit name of the second general-purpose source register, encoded in the "Rm" field.
<xd></xd>	Is the 64-bit name of the general-purpose destination register, encoded in the "Rd" field.
<xn></xn>	Is the 64-bit name of the first general-purpose source register, encoded in the "Rn" field.
<xm></xm>	Is the 64-bit name of the second general-purpose source register, encoded in the "Rm" field.
<shift></shift>	Is the optional shift to be applied to the final source, defaulting to LSL and encoded in the "shift" field. It can have the following values:
	LSL when shift = 00

```
LSR when shift = 01
ASR when shift = 10
ROR when shift = 11
```

<amount>

For the 32-bit variant: is the shift amount, in the range 0 to 31, defaulting to 0 and encoded in the "imm6" field.

For the 64-bit variant: is the shift amount, in the range 0 to 63, defaulting to 0 and encoded in the "imm6" field,

Operation

```
bits(datasize) operand1 = X[n];
bits(datasize) operand2 = ShiftReg(m, shift_type, shift_amount);
result = operand1 AND operand2;
PSTATE.<N,Z,C,V> = result<datasize-1>:IsZeroBit(result):'00';
X[d] = result;
```