C6.2.6 ADDS (extended register)

Add (extended register), setting flags, adds a register value and a sign or zero-extended register value, followed by an optional left shift amount, and writes the result to the destination register. The argument that is extended from the `<Rm>` register can be a byte, halfword, word, or doubleword. It updates the condition flags based on the result.

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

31	30	29	28	27	26	25	24	23	22	21	20 16	15 13	12 10	9 5	4 0
sf	0	1	0	1	0	1	1	0	0	1	Rm	option	imm3	Rn	Rd
	op	S													

32-bit variant

Applies when `sf == 0`.

```
ADDS <Wd>, <Wn|WSP>, <Wm>{, <extend> {#<amount>}}
```

64-bit variant

Applies when `sf == 1`.

```
ADDS <Xd>, <Xn|SP>, <R><m>{, <extend> {#<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;
ExtendType extend_type = DecodeRegExtend(option);
integer shift = UInt(imm3);
if shift > 4 then ReservedValue();
```

Alias conditions

Alias	is preferred when
CMN (extended register)	`Rd == '11111'`

Assembler symbols

`<Wd>`	Is the 32-bit name of the general-purpose destination register, encoded in the "Rd" field.
`<Wn|WSP>`	Is the 32-bit name of the first source general-purpose register or stack pointer, encoded in the "Rn" field.
`<Wm>`	Is the 32-bit name of the second general-purpose source register, encoded in the "Rm" field.
`<Xd>`	Is the 64-bit name of the general-purpose destination register, encoded in the "Rd" field.
`<Xn|SP>`	Is the 64-bit name of the first source general-purpose register or stack pointer, encoded in the "Rn" field.
`<R>`	Is a width specifier, encoded in the "option" field. It can have the following values:

W	when option = 00x
W	when option = 010

X	when option = x11
W	when option = 10x
W	when option = 110

`<m>` Is the number [0-30] of the second general-purpose source register or the name ZR (31), encoded in the "Rm" field.

`<extend>` For the 32-bit variant: is the extension to be applied to the second source operand, encoded in the "option" field. It can have the following values:

UXTB	when option = 000
UXTH	when option = 001
LSL|UXTW	when option = 010
UXTX	when option = 011
SXTB	when option = 100
SXTH	when option = 101
SXTW	when option = 110
SXTX	when option = 111

If "Rn" is '11111' (WSP) and "option" is '010' then LSL is preferred, but may be omitted when "imm3" is '000'. In all other cases <extend> is required and must be UXTW when "option" is '010'.

For the 64-bit variant: is the extension to be applied to the second source operand, encoded in the "option" field. It can have the following values:

UXTB	when option = 000
UXTH	when option = 001
UXTW	when option = 010
LSL|UXTX	when option = 011
SXTB	when option = 100
SXTH	when option = 101
SXTW	when option = 110
SXTX	when option = 111

If "Rn" is '11111' (SP) and "option" is '011' then LSL is preferred, but may be omitted when "imm3" is '000'. In all other cases <extend> is required and must be UXTX when "option" is '011'.

`<amount>` Is the left shift amount to be applied after extension in the range 0 to 4, defaulting to 0, encoded in the "imm3" field. It must be absent when <extend> is absent, is required when <extend> is LSL, and is optional when <extend> is present but not LSL.

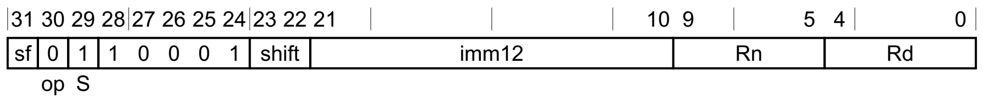
Operation

```
bits(datasize) result;
bits(datasize) operand1 = if n == 31 then SP[] else X[n];
bits(datasize) operand2 = ExtendReg(m, extend_type, shift);
bits(4) nzcv;

(result, nzcv) = AddWithCarry(operand1, operand2, '0');

PSTATE.<N,Z,C,V> = nzcv;

X[d] = result;
```

C6.2.7 ADDS (immediate)

Add (immediate), setting flags, adds a register value and an optionally-shifted immediate 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 CMN (immediate). See *Alias conditions* for details of when each alias is preferred.



32-bit variant

Applies when `sf == 0`.

```
ADDS <Wd>, <Wn|WSP>, #<imm>{, <shift>}
```

64-bit variant

Applies when `sf == 1`.

```
ADDS <Xd>, <Xn|SP>, #<imm>{, <shift>}
```

Decode for all variants of this encoding

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

case shift of
    when '00' imm = ZeroExtend(imm12, datasize);
    when '01' imm = ZeroExtend(imm12:Zeros(12), datasize);
    when '1x' ReservedValue();
```

Alias conditions

Alias	is preferred when
CMN (immediate)	`Rd == '11111'`

Assembler symbols

`<Wd>`	Is the 32-bit name of the general-purpose destination register, encoded in the "Rd" field.
`<Wn|WSP>`	Is the 32-bit name of the source general-purpose register or stack pointer, encoded in the "Rn" field.
`<Xd>`	Is the 64-bit name of the general-purpose destination register, encoded in the "Rd" field.
`<Xn|SP>`	Is the 64-bit name of the source general-purpose register or stack pointer, encoded in the "Rn" field.
`<imm>`	Is an unsigned immediate, in the range 0 to 4095, encoded in the "imm12" field.
`<shift>`	Is the optional left shift to apply to the immediate, defaulting to LSL #0 and encoded in the "shift" field. It can have the following values:

`LSL #0`	when `shift = 00`
`LSL #12`	when `shift = 01`

The encoding `shift = 1x` is reserved.

Operation

```
bits(datasize) result;
bits(datasize) operand1 = if n == 31 then SP[] else X[n];
bits(4) nzcv;

(result, nzcv) = AddWithCarry(operand1, imm, '0');

PSTATE.<N,Z,C,V> = nzcv;

X[d] = result;
```