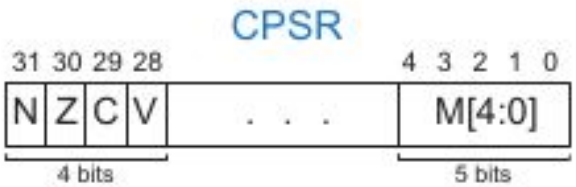
ARMv4 Condition Codes

Instructor: Dr. Vinicius Prado da Fonseca (vpradodafons@online.mun.ca)

Conditional flags

- ARM instructions optionally set condition flags based on whether the result is negative, zero, etc
- Subsequent instructions then execute conditionally, depending on the state of those condition flags
- The ARM condition flags, also called status flags, are negative (N), zero (Z), carry (C), and overflow (V)
- These flags are set by the ALU and are held in the top 4 bits of the 32-bit Current Program Status Register (CPSR)



Conditional flags

- The most common way to set the status bits is with the compare (CMP) instruction
- CMP subtracts the second source operand from the first
- Sets the condition flags based on the result:
  - If the numbers are equal, the result will be zero and the Z flag is set
  - If the first number is an unsigned value that is higher than or the same as the second, the subtraction will produce a carry out and the C flag is set

TST Rd, Rn, Rm	R[m] & R[n], set codes
TEQ Rd, Rn, Rm	R[m] ^ R[n], set codes
CMP Rd, Rn, Rm	R[m] - R[n], set codes
CMN Rd, Rn, Rm	R[n] + R[m], set codes

Conditional flags

- Subsequent instructions can conditionally execute depending on the state of the flags
- The instruction mnemonic (e.g. ADD) is followed by a condition mnemonic (e.g. EQ) that indicates when to execute
- For example (cmp.s):

```
CMP R0, R1 // R0 - R1 & set codes

ADDEQ R0, R0, R1
```

- CMP sets the Z flag if R4 and R5 are equal
- ADDEQ executes only if the Z flag is set (Z=1)
- The conditional field will be used in machine language encodings

./arm_emu.sh cmp.s

Conditional flags

- Other data-processing instructions will set the condition flags when the instruction mnemonic is followed by “S.”
- For example (subs.s):

```
SUBS R2, R3, R4 // R2 <= R3 - R4 & set codes
```

- Will subtract R4 from R3, save the result in R2, and set the condition flags depending on the results
./arm_emu.sh subs.s

Condition Codes

Four condition flags: Negative (N), Zero (Z), Carry (C), and Overflow (V).

- Negative (N) - N is set to 1 if the result of the operation leaves or set the sign bit to 1, otherwise the N is set to 0.
- Zero (Z) - Z set to 1 if the result of the operation is a zero.
- Carry (C) - The C flag is set if the binary add of the sign bit results in a carry, otherwise it is cleared.
- Overflow (V) - If the add/subtract results in number outside the number range, then the V flag is set to 1, otherwise it is set to 0. V = (MSBa == MSBb) && (MSBout != MSBa)

The ARMv4 provides instructions that will change the program counter (branch) if the condition codes are in a given state (e.g., the Z flag is 1).

Z Flag (zflag.s)

- The Z flag is set whenever the ADDS, SUBS, or ANDS results in a zero

```
teq     r0, r0          // r0^r0, sets zero flag
adds    r0, r0, #1      // r0 = 1, clears zero flag
subs    r0, r0, #1      // r0 = 0, sets zero flag, 1-1 causes a carry
add     r0, r0, #1      // r0 = 1, flags not affected
```

- The C flag is set when 1 - 1 occurs
- The hardware performs 1 + (-1) and -1 is 0xffffffff
- When 1 + 0xffffffff happens a carry is generated.

N Flag (nflag.s)

- The N flag is set if the sign bit (B31) is 1 after an instruction that updates the flags is executed

```
mov     r0, #0          // r0 = 0,
subs    r0, r0, #1      // r0 = -1, sets N flag
adds    r0, r0, #1      // r0 = 0, clears N flag, causes a carry
sub     r0, r0, #1      // r0 = -1, flags not affected
```

C Flag (cflag.s)

- The C flag is set any time a carry from bit 31 occurs

```
mov     r0, #-5
adds    r0, r0, #10     // r0 = -5 + 10, causes a carry
mov     r0, #-5
add     r0, r0, #10     // flags not affected
```

V Flag (vflag.s)

- Most positive number is added to itself yielding and overflow and -2
- Most negative number is added to itself
- Two positive numbers add together to produce a negative number
- Two negative numbers add together to produce a positive number

```
mov     r0, #-1
lsr     r0,r0, #1       // r0 is all 1 except the sign bit
adds    r1, r0, r0      // cause an overflow
adds    r0, r0, #0      // reset overflow flag
mov     r2, #1          // create the most negative value
ror     r2, r2, #1      // set the sign bit with rotate ror
adds    r1, r2, r2      // cause an overflow
```

Condition For Equality

- The contents of two register can be tested for equality with

```
cmp     r2, r5        // Compare is the same
```

- For this instruction if the Z flag is set, then the result was 0
- The registers must have had the same value.

CMP should be R[n]-R[m], same SUB. Table is the opposite .
https://developer.arm.com/documentation/dui0068/b/ARM-Instruction-Reference/ARM-general-data-processing-instructions/CMP-and-CMN

Condition For Equality (eq.s)

- cmp r2, r5 // Compare is the same

```
mov     r5, #47
mov     r6, #47
cmp     r5, r6          // r5 - r6, descarding result
adds    r0, r5, r6      // clear Z flag
cmp     r5, r6          // r5 - r6
```

- Why is the C flag set?

Condition For Equality (eq.s)

- Why is the C flag set?

  CMP R5, R6 // R6-R5

$$R5: 0010\ 1111_2\ R6: 0010\ 1111_2$$

```
0010 1111 - 0010 1111 = 0010 1111 + 1101 0001
```

```
 11111 111
    0010 1111
+   1101 0001
----------------
    0000 0000
```

Greater than or Less Than (gtlt.s)

- The following program compares:
  - -3 with -2
  - -2 with -3
  - 5 with 8
  - 8 with 5
  - -3 with 8
  - 8 with -3

```
mov     r3, #-3
mov     r4, #-2
mov     r5, #5
mov     r6, #8
cmp     r3, r4
cmp     r4, r3
cmp     r5, r6
cmp     r6, r5
cmp     r3, r6
cmp     r6, r3
```

Greater than or Less Than (qtlt.s)

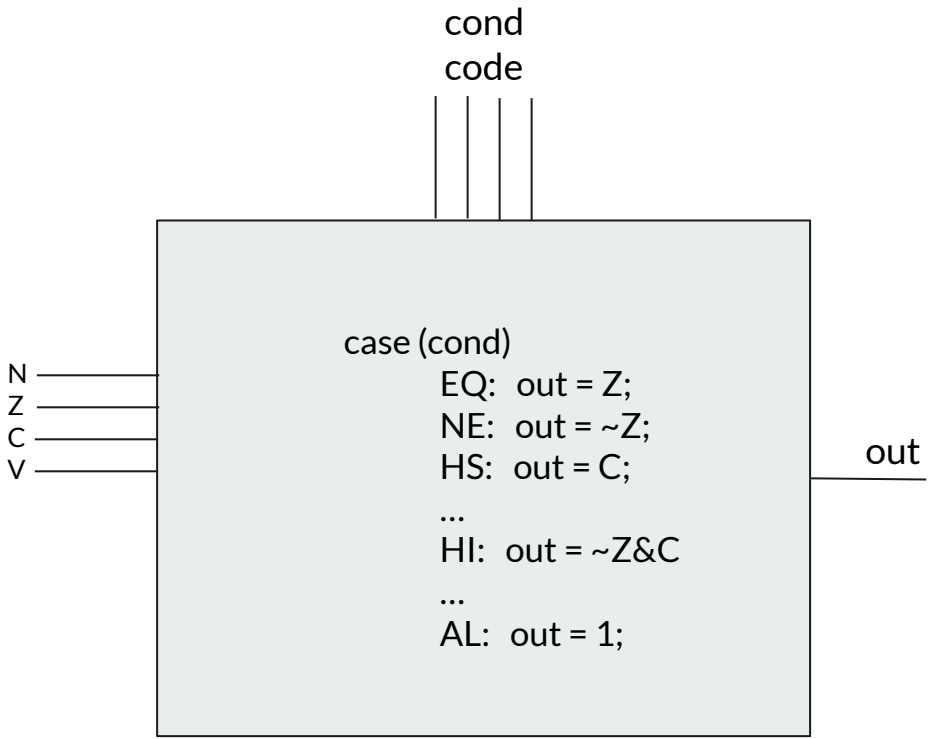
- In the test cmp RA, RB, where A and B are any registers in the following program:
- If RA < RB then N is set
- If RA > RB then N is not set
- The relationship for C is more complicated (e.g. Condition For Equality, slide 13)

```
mov     r3, #-3
mov     r4, #-2
mov     r5, #5
mov     r6, #8
cmp     r3, r4
cmp     r4, r3
cmp     r5, r6
cmp     r6, r5
cmp     r3, r6
cmp     r6, r3
```

Conditions Table (cond_table.s)

Code	Mnemonic	**Long Name**	Condition Expression
0000	EQ	Equal	Z
0001	NE	Not Equal	$\overline{Z}$
0010	CS/HS	Higher Same	C
0011	CS/LO	Low	$\overline{C}$
0100	MI	Minus	N
0101	PL	Plus	$\overline{N}$
0110	VS	V Set	V
0111	VC	V Clear	$\overline{V}$
1000	HI	Higher	$\overline{Z} \cdot C$
1001	LS	Lower Same	$Z + \overline{C}$
1010	GE	Greater Than Equal	$\overline{N \oplus V}$
1011	LT	Less Than	$N \oplus V$
1100	GT	Greater Than	$\overline{Z} \cdot \overline{N \oplus V}$
1101	LE	Less Than Equal	$Z + (N \oplus V)$
1110	AL	Always	true

Conditions Table (cond_table.s)



Questions?