ARM7 Instruction SET Part 1

Lec 10
Oct 14, 2022

#'B' → for branch
branch means jumping.

- BAL/B

BAL → branch always
it will always jump.

- BNV

BNV → branch never.
→ no operation has been done.
Like "NOP" (no operation)
in 8051.

- BCS

BCS → branch if carry set.
jump if carry set. (JC).

- BCC

BCC → branch carry clear.
jump if carry flag is zero.
(JNC)

- BVS

BVS → branch if overflow set.
jump if overflow flag is 1

- BVC

BVC → branch if overflow clear.
jump if overflow flag 0.

- BMI

BMI → branch if minus.
jump if negative flag is 1.

- BPL

BPL → branch if plus.

- BEQ

BEQ → branch if equal.
→ jump if zero flag is 1.
(it compares the two numbers)
and effects are shown in
zero flags.

- BNQ

BNQ → branch if not equal.
→ jump if zero flag is 0.

- BHI

BHI → branch if higher.
(we compare two numbers
and the first number is
higher.)

- BHS

BHS → branch if higher or same.
(we compare two numbers and
the first number is higher
or same)

- BLO

BLO → branch if lower.
(the first number is lower.)

- BLS

BLS → branch if lower or same.
(we compare two number the first one is lower or same)

- BGT

BGT → branch if grater than (signed) if the first one is greater.

- BGE

BGE → branch if greater than or equal (signed) if first is greater than or equal.

- BLT

BLT → branch is less than (signed)

- BLE

BLE → branch is less than or equal.

In intel processors, the conditions are attached with jumps.

In ARM the conditions can be attached with every instruction.

MOV R0, R1

MOVEQ R0, R1

MOV R0, R1.

value in R1 goes to R0.

MOVEQ R0, R1

(X) if R0 and R1 are equal then data will pass R1 to R0.
This is wrong.

(✓) The effect of just previous instruct produces equal result.
then R1 values moves to R0.

Like JC (jump if carry)
if carry flg is 1 it will jump to the asking location.
otherwise it will not jump.

MOVCS R0, R1.

if we have done previous operation which produces carry then R1 value pass to R0. otherwise skip the instruction and go to next instruction.

MULEQ

if the result of the previous instruction is equal then multiplication will occur. Otherwise it will skip this operation.

ADD R0, R1, R2
ADDS R0, R1, R2

* In intel processors flags are automatically affected by the result of the instruction. But in ARM NO instructions do not affect flags automatically.

The only way to affect the flags by the result of instruction is at the end of the instruction name.

ADD R0, R1, R2

R0 ← R1 + R2

it will not affect any flag. If there is carry will not find it in carry flag.

ADDS R0, R1, R2.

R0 ← R1 + R2.

here these flags will be affected by the result of the operation.

MOV R0, R1

MOVS R0, R1

MOVSEQ R0, R1

MOV R0, R1

value in R1 moves to R0.
the corresponding flags will not be affected.

MOVS R0, R1

the value in R1 goes to R0 and affects the flags

MOVSEQS R0, R1.

The result of previous instruction is equal then
the value in R1 goes to R0.
and also affects the flags.

If the result of previous instruction is not equal then skip (MOVSEQ R0,R1) this instruction.