5.5) Branch Instruction

- changes control flow by adding an offset to the PC
- format: B {cond} label
  - cond: only execute if condition true
  - label: assembler replaces with pc-relative offset
- condition code suffixes:

EQ	equal (to zero)	signed
NE	not equal	signed
GT	greater than	signed
GE	greater than or equal	signed
LT	less than	signed
LE	less than or equal	signed
PL	plus (≥0)	ignore overflow
MI	minus (<0)	ignore overflow
HI		unsigned
HS		unsigned
LO		unsigned
LS		unsigned
VS	overflow set	
VC	overflow clear	
AL	always (default)	

(equivalent to GT, GE, LT, LE respectively)

```
e.g. SUBS r2, r2, #1   // r2 ← [r2] - 1
     BGT LOOP          // Go to LOOP if r2 > 0
```

5.6) Pre- and Post-indexed Addressing

- applies to LDR and STR
- pre-indexed: <op>{size}{cond} R_t, [R_n, #offset]! → pre indexed
  - the offset is added to the address in R_n, then the memory access is performed and R_n is updated

```
e.g. LDR r1, [r0, #4]!
     // r1 ← [[r0]+4], r0 ← [r0] + 4
```

- post-indexed: <op>{size}{cond} R_t, [R_n], #offset
  - the memory access is performed with the address in R_n, then R_n is updated by adding the offset.

```
e.g. LDR r1, [r0], #4
     // r1 ← [[r0]], r0 ← [r0] + 4
```

5.7) Compare Instructions

- compares two operands and sets the condition flags (N, Z, C, V) but does not save to a destination register
- CMP {cond} Rn, Operand 2

```
e.g. CMP r1, #1          if r1 = 1, N ← 0, Z ← 0, C ← 1, V ← 0
     // N, Z, C, V ← [r1] - 1    but r1 is not changed
```

- CMN {cond} R_n, Operand2 // compare negative
  - compares [R_n] and −Operand2
- test: TST {cond} R_n, Operand2
  - performs bit-wise AND and updates flags

e.g. TST r1, #0x00008000
// N,Z,C,V ← [r1] • 2_0000 0000 0000 0000 1000 0000 0000 0000

- test equal • TEQ {cond} R_n, Operand2
  - performs bitwise XOR and updates flags
- compare & branch: <op> R_n, label
  - op = CBZ: compare equal to 0
    CBNZ: compare not equal to zero
  - Compares [R_n] with zero and decides on branch

CBZ R_n, label = { CMP R_n, #0 / BEQ label } CBNZ R_n, label = { CMP R_n, #0 / BNE label }

5.8) If-Else

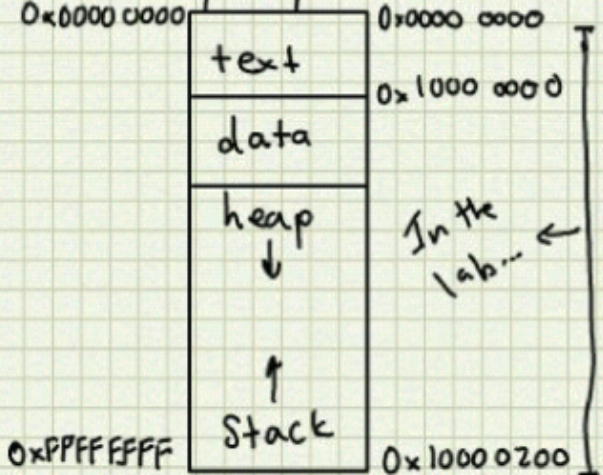
e.g. if(x==0)
  y++;
else
  y--;
// x is r0, y is r1

with branching:
```
    CBZ r0, ADD_
    SUB r1, r1, #2
ADD_ ADD r1, r1, #1
```

without branching:
```
    CMP r0, #0
    ADDEQ r1, r1, #1
    SUBNE r1, r1, #1
```
(SUBNE: condition)

Subroutine

1) The Stack
- LIFO
- each thread/process has a call stack growing from high to low memory addresses
- used for local variables and parameter passing
- typical memory map (32-bit addressing)



- r13 (alias sp) is the stack pointer
- push [r0] onto stack
  STR r0, [sp, #-4]!
- pop from stack into r0:
  LDR r0, [sp], #4