

(3.1) Why is the program counter a pointer and not a counter?

Counters are relative to a certain point, which would have to be specified in another point. The pointer is similar to a counter, only with the starting point located at the start or end of memory depending on the system. A pointer is also used because of branch / jump statements that cause the pointer to move in a non-sequential manner.

(3.2) Explain the function of the following registers

- (a) PC: program counter, keeps track of the location of the next instruction to be executed
- (b) MAR: Memory Address Register: stores the address of the data to be retrieved or stored
- (c) MBR: Memory Buffer Register: stores the data to be loaded or stored
- (d) IR: Instruction Register: stores the current fetched instruction

(3.3) (a) $c=0$ $z=0$ $v=0$ $n=0$

(b) $c=1$ $z=1$ $v=0$ $n=0$

(c) $c=0$ $z=0$ $v=0$ $n=0$

(d) $c=1$ $z=0$ $v=0$ $n=0$

(e) $c=0$ $z=0$ $v=0$ $n=1$

(f) $c=1$ $z=0$ $v=1$ $n=0$

(3.10) Why does ARM implement a reverse subtract?

Because literals can only be stored operand 2, so this allows the programmer to subtract from a literal.

(3.17) What are the advantages and disadvantages of storing the 12 bit literal as an 8 bit literal and a 4 bit shift?

This allows for a greater range of values the literal, but gives less precision, especially when moving further from zero.

(3.18) Write one or more ARM instructions that will clear bits 20 to 25 inclusive in register 0.

ANDR0, R0, #0xfe0fffff

(3.19) Swap without an extra register.

EORR0, R0, R1EORR1, R0, R1EORR0, R0, R1

(3.25) (a) xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

(b) xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

(c) xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

(d) xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

(3.39) *START : LDRBr2, [r0], #1STRBr2, [r1], #1TEQR2, #0BRNESTART*

(3.51) Why is the program counter a pointer and not a counter?

START : MOVr0, #1TEQr1, r2BREQENDLDRBr3, [r1], #1LDRBr4, [r2], #-1TEQr3, r4BRNENOBSTART
NO : movr0, #0END :