ARM INSTRUCTION SETS

Introduction

Definition:

An Advanced RISC Machine (ARM) processor is one of a family of Central Processing Units (CPUs) based on the Reduced Instruction Set Computer (RISC) architecture for computer processors.

ARM processors are used in music players, smartphones, wearables, tablets, and other consumer electronic devices. This needs a very few instruction sets and transistors.

Advantages:

The advantages of ARM processor are,

(i) Due to its very small size, it is perfectly fit for small-sized devices.

(ii) It has less power consumption along with reduced complexity in its circuits,

(iii) It is easy to program at the assembly level.

ARM processor can be applied to various designs such as 32-bit devices and embedded systems and we can even be upgraded according to the user needs.

ARM instructions are written one per line, starting after the first column. Comments begin with a semicolon and continue to the end of the line. A label gives a name to a memory location, comes at the beginning of the line, starting in the first column:

Processor and Memory Organization

Different versions of the ARM architecture are identified by the number. ARM7 is a Von Neumann architecture machine and ARM9 uses a Harvard architecture. A The ARM architecture supports two basic types of data: - Standard ARM word is 32 bits long, and - Word may be divided into four 8-bit bytes.

ARM7 allows addresses to be 32 bits long and an address refers to a byte, not a word. The word 0 in the ARM address space is at location 0, the word 1 is at 4, the word 2 is at 8, and so on. As a result, the PC is incremented by The ARM processor, can be configured at power-up to address the bytes in a word in

either little-endian or big-endian mode, as big-endian stores the Most Significant Bytes (MSBs) first, whereas little-endian stores the Least Significant Bytes(LSBs) first.

Data Operations

In the ARM processor, arithmetic and logical operations cannot be performed directly on memory locations. While some processors allow such operations to be directly link to the reference of main memory.

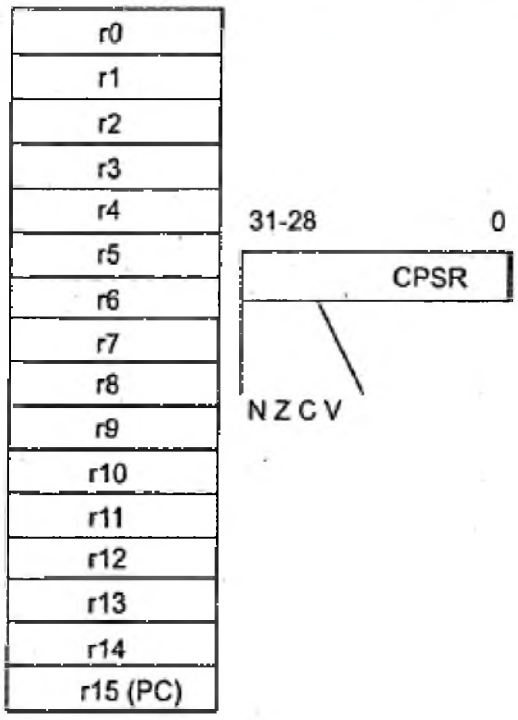
ARM Programming Model

The registers in the basic ARM programming model. ARM has 16 general-purpose registers, r0 through r15. The rl5 register has the same capabilities as the other registers, but it is also used as the Program Counter (PC).

The properties of a general-purpose register allows the PC value to be used as an operand in computations, which can make certain programming tasks easier.

(1)CPSR:

The basic register in the programming model is the Current Program Status Register (CPSR) which is used by the ARM core to monitor and control the internal functions.



This register is setup automatically during every arithmetic, logical, Of shifting operation. The top four bits of the CPSR hold the following usefully information about the results of that arithmetic/logical operation.

(i) The negative (N) bit is set when the result is negative in two's complement arithmetic.

(ii) The zero (Z) bit is set when every bit of the result is zero.

(iii) The carry (C) bit is set when there is a carry out of the operation.

(iv) The overflow (V) bit is set when an arithmetic operation results in an overflow.

These bits can be used to easily check the results of an arithmetic operation.

(2) Data Processing Instructions

The following instructions are used for data processing in ARM processor:

(i) Arithmetic Instructions,

(ii) Multiply instructions,

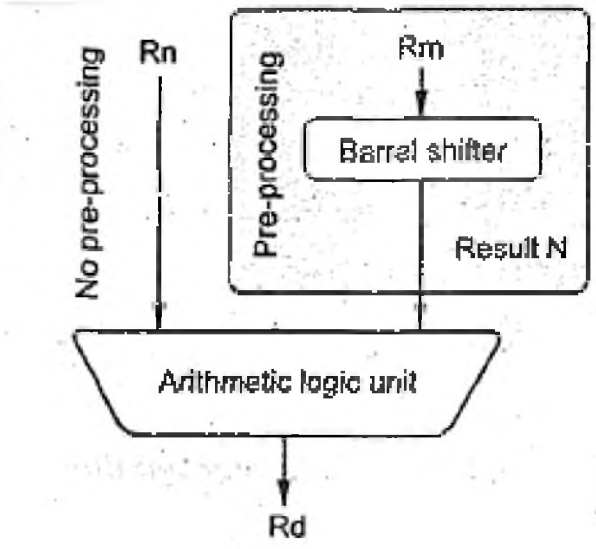
(iii) Logical instructions,.

(iv) Shift and rotate instructions,

(v) Comparison instructions, and

(vi) Load and store instructions.

Most of the data processing instructions can process one of their operands using the barrel shifter.



One operand to ALU is routed through the Barrel shifter. Thus, the operand can be modified before it is used which is useful for fast multiplication and dealing with lists, table and other complex data structure.

The basic form of a data instruction is simple as,

ADD r0, rl,r2

This instruction sets register r0 to the sum of the values stored in rl and r2.

(i) Arithmetic Instructions

The arithmetic operations perform addition and subtraction and with carry versions include the current value of the carry bit in the computation.

ADD	Add
ADC	Add with carry
SUB	Subtract
SBC	Subtract with carry
RSB	Reverse Subtract
RSC	Reverse Subtract with carry

ADD	Add two 32-bit values	$Rd = Rn + N$
ADC	Add two 32-bit values and carry	$Rd = Rn + N + carry$
SUB	Subtract two 32-bit values	$Rd = Rn - N$
SBC	Subtract with carry of two 32-bit values	$Rd = Rn - N - !$ *(carry flag)*
RSB	Reverse subtract of two 32-bit values	$Rd = N - Rn$
RSC	Reverse subtract with carry of two 32-bit values	$Rd = N - Rn - !$ *(carry flag)*

Examples:

ADD RO, R1,R2 @RO = R1+R2

ADC RO, R1,R2 @RO = R1+R2 + C

SUB RO,R1,R2 @RO = R1-R2

SBC RO,R1,R2 @RO = R1-R2-! C

RSB RO,R1,R2 @RO = R2-R1

RSC RO,R1,R2 @RO = R2-R1-!C

(ii) Multiplication Instructions

The basic ARM provides two multiplication instructions. Here, Rs is the source register and Rd and Rm cannot be the same register.

Example:

MLA R4,R3,R2,R1 @R4 = R3xR2 + Rl

(iii) Logical Instructions

The bit-wise logical operations perform logical AND, OR, and XOR that is, exclusive -or (EOR) operations.

Example

AND RO, R1,R2 @RO = R1 ANDR2

ORR RO, RI, R2 @RO = R1 ORR2

EOR RO, RI, R2 @ RO = RI XOR R2

(iv)Shift and Rotate Instructions

Shifting means to move bits right and left inside an operand. All of the shift and rotate instructions affect Overflow Flag(OF) and Carry Flag(CF).

LSL	Logical shift left (zero fill)
LSR	Logical shift right (zero fill)
ASL	Arithmetic shift left
ASR	Arithmetic shift right
ROR	Rotate right
RRX	Rotate right extended with C

The shift modifier is always applied to the second source operand. The LSL and LSR modifiers perform left and right logical shifts, filling the leastsignificant bits of the operand with zeroes.

(v) Compare (or) Comparison Instructions:

The compare instructions are used to compare or test a register with a 32- bit value. They update the CPSR flag bits according to the result, but do not affect other registers.

After the bits have been set, the information can then be used to change program flow by using conditional execution.

Examples:

CMP R1,R2 @R1-R2

CMN R1,R2 @R1+R2

TST R1,R2 @ R1ANDR2

TEQ R1,R2 @R1EORR2

(vi) Move Instructions

Move is the simplest ARM instruction which copies the value (N) of second operand into Rd(destination register), where N is the register or immediate value. This instruction is useful for setting an initial values and transferring the data between registers.

(vii) Load -Store Instructions: Memory Instructions

ARM uses load-store instruction only for memory access. LDR is used to load something from memory into a register, and STR is used to store something from a register to a memory address.