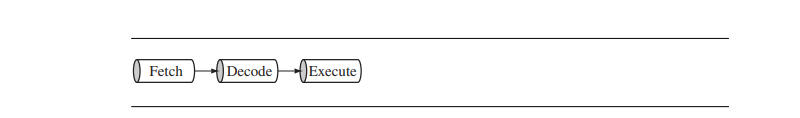
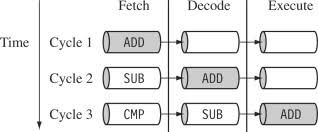
Pipeline is mechanism in a RISC processor, which is used to execute instructions. Pipeline speeds up the execution by fetching the instruction while other instruction are being decoded and executed.

The above Figure shows a three-stage pipeline:

* Fetch loads an instruction from memory.
* Decode identifies the instruction to be executed.
* Execute processes the instruction and writes the result back to a register.

The following Figure illustrates pipeline using a simple example:



The Figure shows a sequence of three instructions being fetched, decoded, and executed by the processor.

* The three instructions are placed into the pipeline sequentially.
* In the first cycle, the core fetches the ADD instruction from memory.
* In the second cycle, the core fetches the SUB instruction and decodes the ADD instruction.
* In the third cycle, both the SUB and ADD instructions are moved along the pipeline. The ADD instruction is executed, the SUB instruction is decoded, and the CMP instruction is fetched.

This procedure is called filling the pipeline.

The pipeline allows the core to execute an instruction every cycle.

* As the pipeline length increases, the amount of work done at each stage is reduced, which allows the processor to attain a higher operating frequency. This in turn increases the performance.
* The increased pipeline length also means increased system latency and there can be data dependency between certain stages.
* The pipeline design for each ARM family differs. For example, the ARM9 core increases the pipeline length to five stages, as shown in Figure.

ARM9 Five-stage Pipeline:

**Fetch → Decode → Execute → Memory → Write**

The ARM9 adds a memory and writeback stage, which allows the ARM9 to –

• process on average 1.1 Dhrystone MIPS per MHz

• increase the instruction throughput in ARM9 by around 13% compared with an ARM7.

The ARM10 increases the pipeline length still further by adding a sixth stage, as shown in the following :

**Fetch →Issue 🡪Decode → Execute → Memory → Write**

ARM10 Six-stage Pipeline

• can process on average 1.3 Dhrystone MIPS per MHz

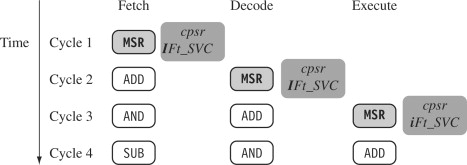
• has about 34% more throughput than an ARM7 processor core

• but again at a higher latency cost.

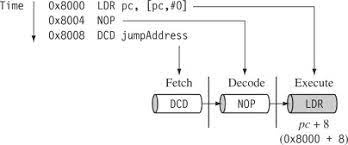
Pipeline Executing Characteristics:

The ARM pipeline will not process an instruction until it passes completely through the execute stage.

* For example, an ARM7 pipeline (with three stages) has executed an instruction only when the fourth instruction is fetched.

The following Figure shows an instruction sequence on an ARM7 pipeline.

* The MSR instruction is used to enable IRQ interrupts, which only occurs once the MSR instruction completes the execute stage of the pipeline. It clears the I bit in the cpsr to enable the IRQ.
* Once the ADD instruction enters the execute stage of the pipeline, IRQ interrupts are enabled.

The following Figure illustrates the use of the pipeline and the program counter PC.

* In the execute stage, the pc always points to the address of the instruction plus 8 bytes. In other words, the pc always points to the address of the instruction being executed plus two instructions ahead.

Note: when the processor is in Thumb state, the pc is the instruction address plus 4.

* There are three other characteristics of the pipeline:

1. First, the execution of a branch instruction or branching by the direct modification of the pc causes the ARM core to flush its pipeline.
2. Second, ARM10 uses branch prediction, which reduces the effect of a pipeline flush by predicting possible branches and loading the new branch address prior to the execution of the instruction.
3. Third, an instruction in the execute stage will complete even though an interrupt has been raised. Other instructions in the pipeline will be abandoned, and the processor will start filling the pipeline.