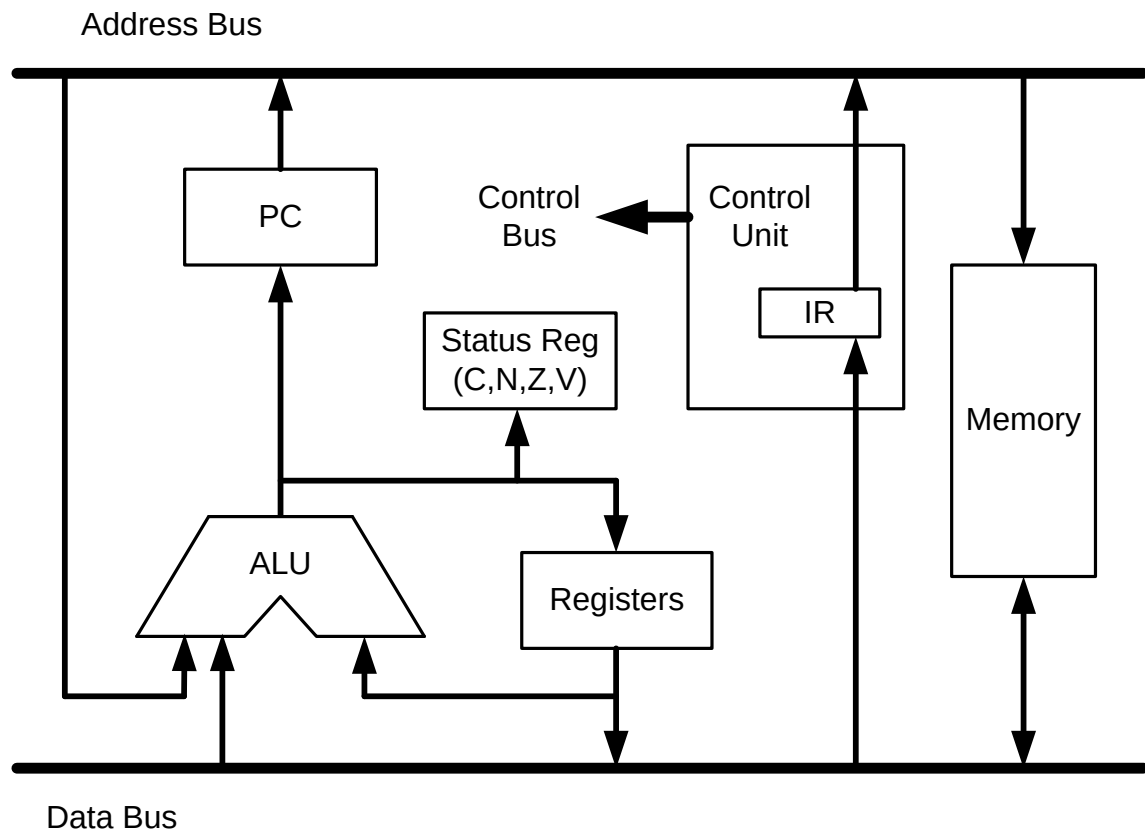


Chapter 2 Introduction to Microprocessor

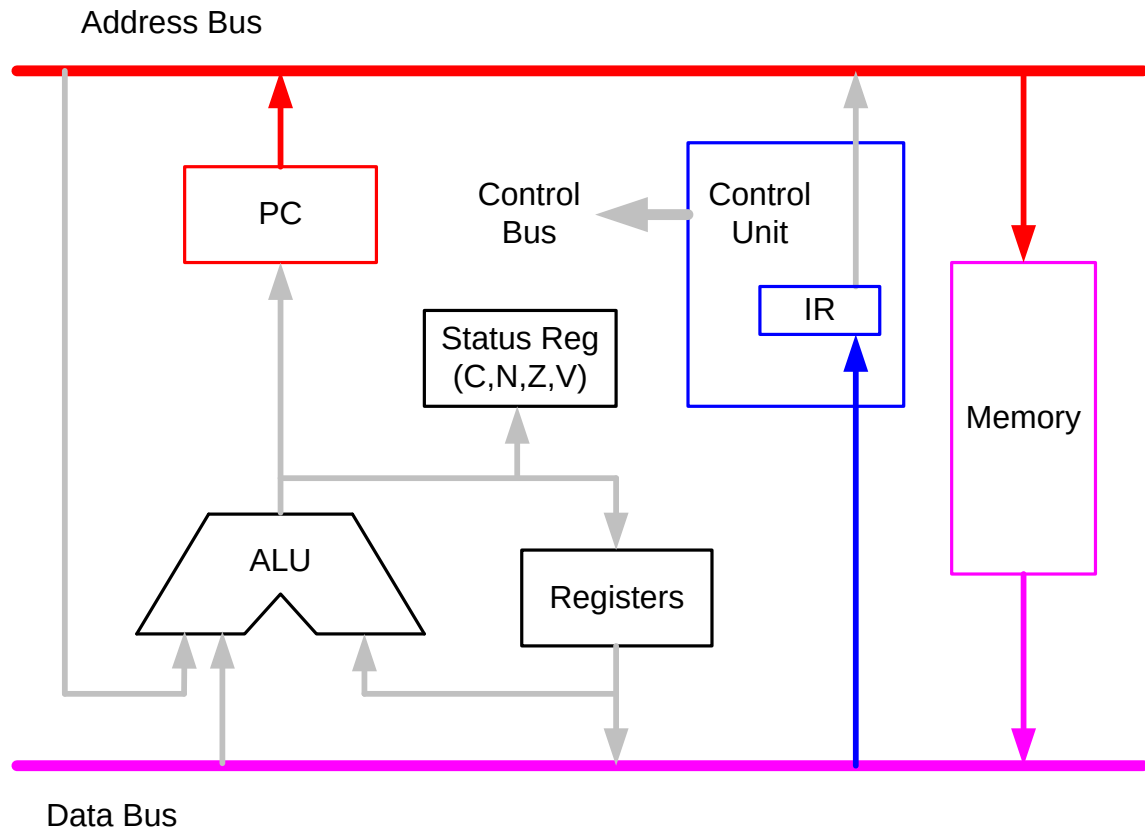
1. A very simple Processor



- ALU: Arithmetic Logic Unit – performs operations on operands
- Registers: holds the data being processed and result from ALU
- Status Register: contains status bits that reflect the result from ALU (e.g. C – result contains carry, N – result is negative, Z – result is 0, V – result is overflow)
- PC: Program counter – hosts address of the next instruction to be executed
- IR: Instruction Register – holds the current instruction being executed
- Control Unit: decodes the instruction in IR and issues control signals to various logic to execute the instruction

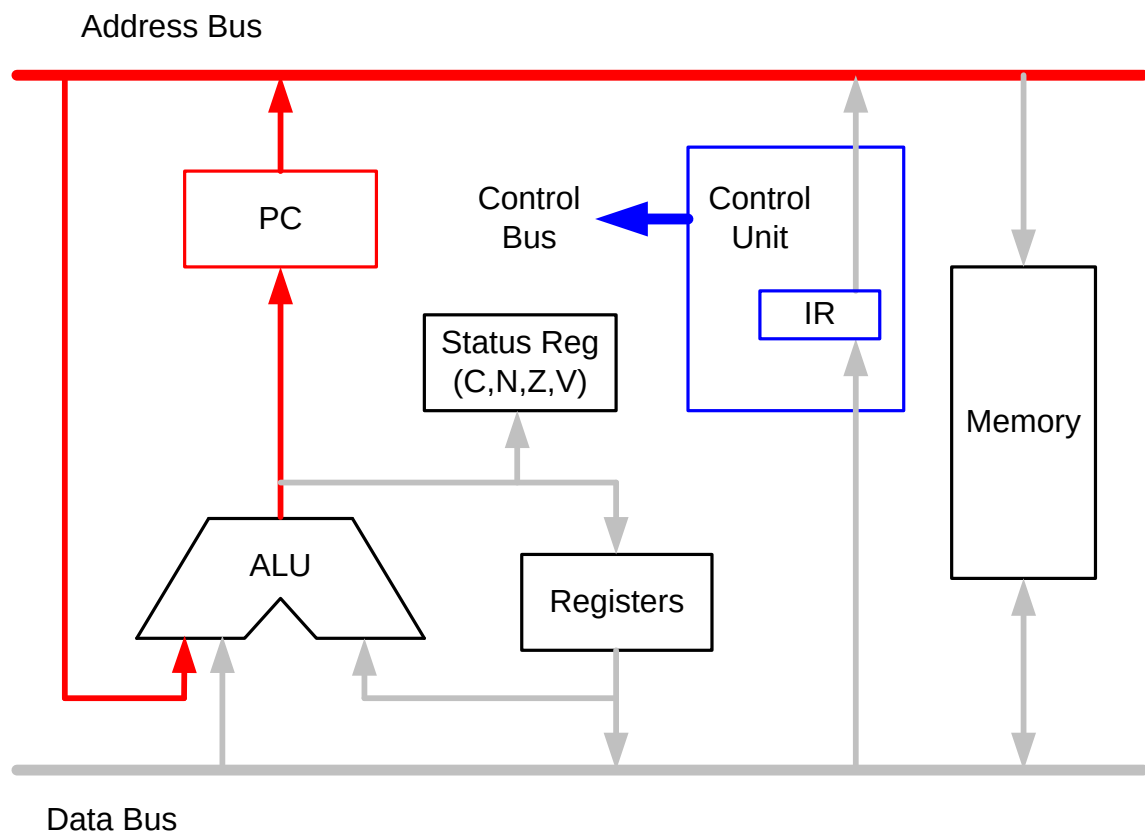
2. Instruction Execution

Step 1: Instruction Fetch – Read instruction from memory



- The content in PC is placed onto the address bus
- Memory places the content (instruction) onto the data bus
- Instruction is stored in the instruction register (IR)

Step 2: Instruction Decode – Determine the action to be taken

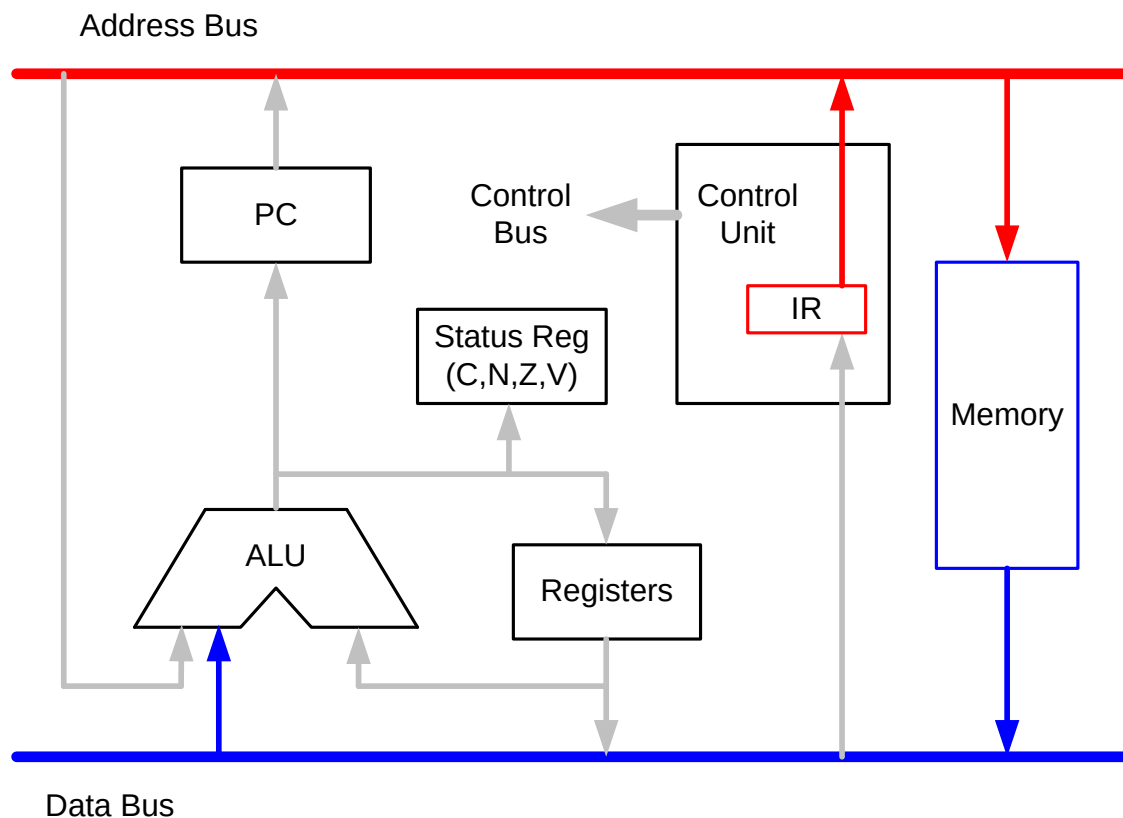


- The instruction stored in IR is decoded by control unit to provide control signals to ALU and other internal logic circuits inside CPU.
- Update the address in PC so that it points to the address of next instruction.

The content in the PC is pushed onto the address bus, the ALU increments this value (dependent on the instruction being executed, it may point to the address of next instruction) and places it back into the PC.

For different CPU design, the PC may have an automatic update function integrated so that PC does not require ALU to update itself.

Step 3: Operand Fetch (optional) – Retrieve data from memory for processing

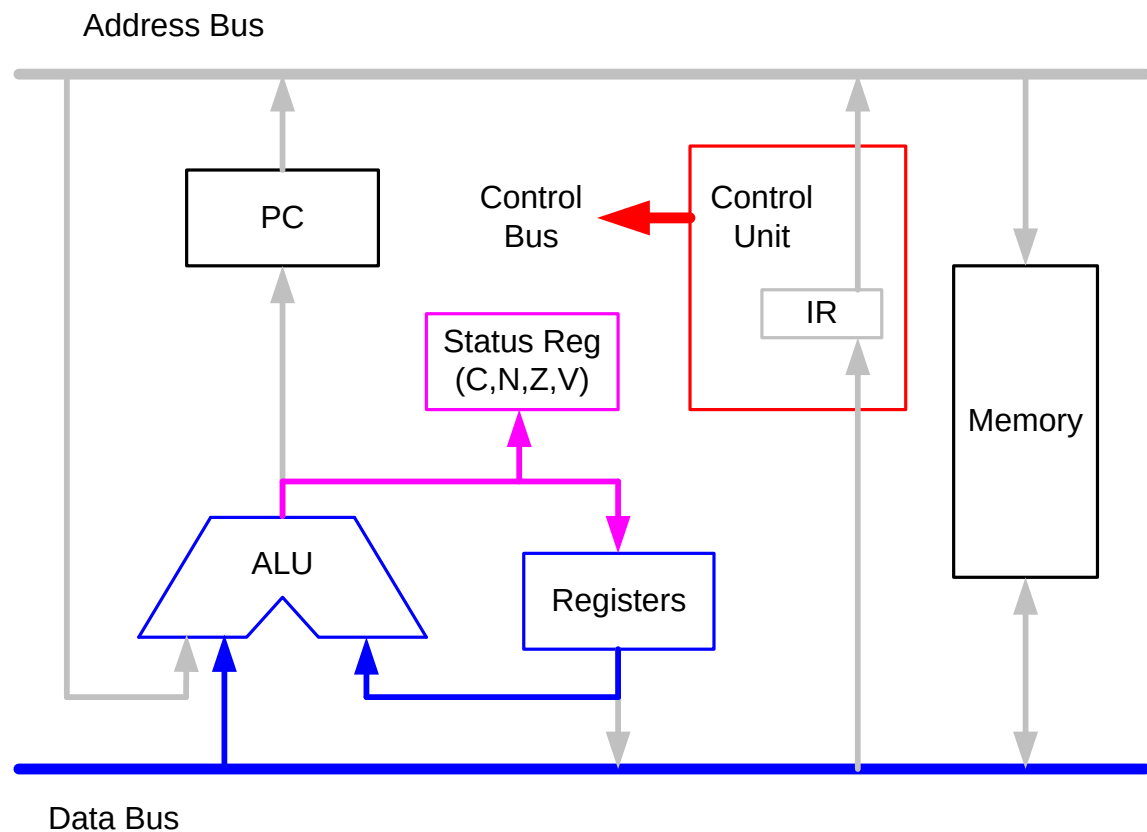


- The instruction register provides the address of the data to be processed (i.e. operand address).
- Memory places the operand data onto the data bus.
- Operand will be stored in the internal register for ALU to process.

IMPORTANT

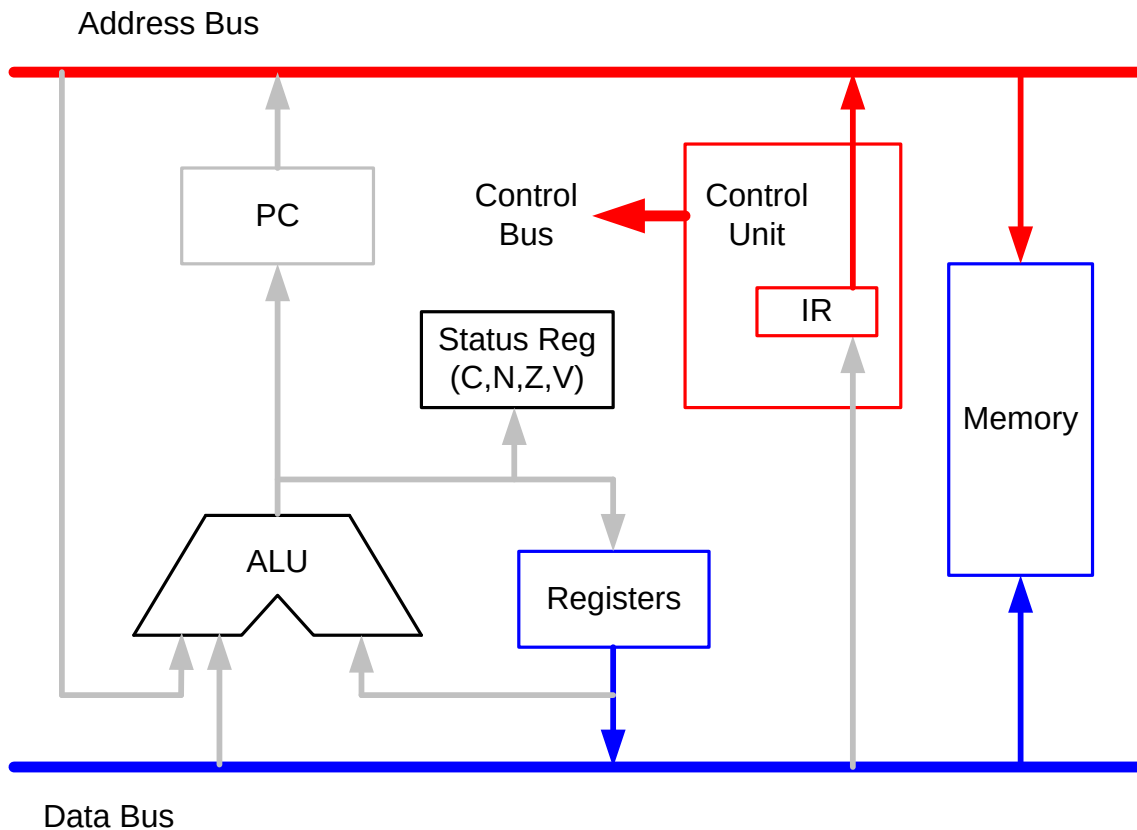
- If operand is embedded in the instruction or resided in internal registers, operand fetch is **NOT** necessary since decode unit can issue control signals to direct the operand into the ALU for execution.

Step 4: Execute Instruction – perform the action



- Control Unit issues control signals to ALU and other logic circuits to execute the instruction.
- ALU performs specific operation according to the instruction register.
- The result is placed back into the register and the status bits in the status register are updated.

Step 5: Write-back (optional) – Store the result to memory



- After executing the instruction, this step is required **if** the result is to be stored in memory.
- The address is placed onto the address bus.
- The result is written back into the memory.
- For some processors, this will require an additional instruction to perform the operation.

3. Instructions

Instruction is coded in binary form and contains information:

- about the *action* which the CPU is required to perform
- the source of data (direct data or data stored in register/memory)
- the destination of data (in register or memory)

<i>Operation</i>	<i>Source Operand</i>	<i>Destination Operand</i>
Op-code (<i>i</i>-bit)	Source (<i>j</i>-bit)	Destination (<i>k</i>-bit)

A simple format of an instruction

There are 2^i operations available (e.g., addition, subtraction, multiplication, division, shift operation, AND, OR, EX-OR, comparison, etc.)

There are 2^j ways to specify how to obtain the source operand.

There are 2^k ways to specify where the result should be stored.

Example (it is *NOT* a real CPU):

3-bit Op-code:

000	001	010	011	100	101	110	111
ADD	SUB	SHL	SHR	AND	OR	NOT	CMP

The control unit will decode these 3-bit of the IR (Instruction register) to determine the operation to be performed.

3-bit source and destination operands:

000	001	010	011	100	101	110	111
R0	R1	R2	R3	Memory address			

If the first bit is “0”, the operand is resided in one of the four registers, R0, R1, R2 or R3.

If the first bit is “1”, the operand is resided in memory and the memory address should be given together with the instruction.

e.g. (1)

IR = 000 001 011

→ R1 + R3

result is stored in R3

e.g. (2)

IR = 100 010 1XX , memory address

→ R2 AND (data stored in memory address)

Result is stored in the memory

e.g. (3)

IR = 001 1XX 1XX , mem addr 1, mem addr 2

→ [mem addr1] – [mem addr 2]

Result is stored in mem addr 2

NOTE: The instruction format of a real CPU is more complicated!!!

4. Non-pipelined processor

A program consists of a sequence of instructions for a processor to perform specific tasks.

Processor executes instructions in a sequential manner.

Processor requires an *instruction cycle* to complete the execution of an instruction, generally including the following:

- Instruction fetch;
- Instruction decode;
- Instruction execution.

Instruction cycle: the time required for processor to complete the execution of an instruction, usually in terms of no of clock cycles.

Different instructions may require different number of clocks to complete.

The same instruction may use different number of clock cycles to complete due to the use of different ways to obtain the source data and store the result.

Obviously, the number of clock cycles will be smaller if both source and destination operands are processor's internal registers.

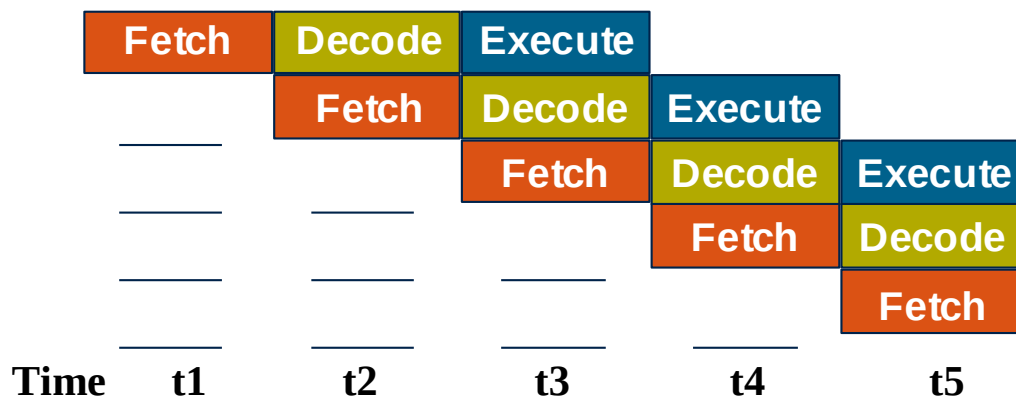
On the other hand, more clock cycles are required if both source and destination operands are resided in external memory.

5. Pipelined processor

Processor can be designed in a way that the instruction fetch, decode and execution can be operated independently by three different units.

Hence, the result from the instruction fetch unit will be fed to the instruction decode unit, and its result will be fed to the instruction execution unit. Therefore, the three units operate on three different instructions concurrently.

Effectively, the speed of the processor is increased by N fold if there are N stages of pipeline involved.



Example of a 3-stage pipeline process

However, if the processor requires data that is not from the pipeline, the advantages due to pipeline will be destroyed since it needs to fetch/store data from/to memory. If this situation does occur frequently, the gain in speed will be decreased.

```

Instr 1
Instr 2
Jump L1    <== pipeline breaks here, PC is loaded a new address
Instr 3
L1 Instr 4
Instr 5
  
```

Therefore, pipelined processor is usually equipped with a large number of internal registers so that data can be brought into the processor in advance. The processor does not to access external memory very often.

The fastest possible instruction execution time is dependent upon the time for fetching the instruction – access time of memory devices is the dominant factor.

6. CISC Versus RISC

Typically CISC (**C**omplex **I**nstruction **S**et **C**omputer) chips have a large amount of different and complex instructions.

The philosophy behind it is that hardware is always faster than software, therefore one should make a powerful instruction set, which provides programmers with assembly instructions to do a lot with short programs.

In general, CISC chips are with relatively long instruction cycle per instruction (a large number of clock cycles), but use less number of instructions (less than RISC) to complete the task.

RISC (**R**educed **I**nstruction **S**et **C**omputers) chips evolved around the mid-1980 as a reaction at CISC chips.

The philosophy behind it is that fewer, simpler and faster instructions would be better than the large, complex and slower CISC instructions. However, more instructions are needed to accomplish a task, *i.e. a longer program*.

An other advantage of RISC is that – in theory – because of the *much* simple instructions, RISC chips require fewer

transistors, which makes them easier to design and cheaper to produce.

Finally, it's easier to write powerful optimized compilers, since fewer instructions involve.

Summary of CISC versus RISC

The main characteristics of CISC microprocessors are:

- Extensive instructions.
- Complex and efficient machine instructions.
- Microencoding of the machine instructions.
- Extensive addressing capabilities for memory operations.
- Relatively fewer registers.

In comparison, RISC processors are more or less the opposite of the above:

- Reduced instruction set.
- Less complex, simple instructions.
- Hardwired control unit and machine instructions.
- Few addressing schemes for memory operands with only two basic instructions, LOAD and STORE
- Many registers.

NOTE:

RISC usually loads the operands into the CPU internal registers first and then processes the data afterward. In this way, CPU does not need to fetch operand from external memory and the pipeline advantage is not destroyed very often.

7. Performance of CPU

CPU performance does not depend on the operating clock frequency.

Faster clock, the shorter time required for the instruction cycle.

But, if the design of the CPU is different, the number of clock cycles required for the instruction can be very different.

Pipeline processor versus non-pipeline processor

CISC versus RISC

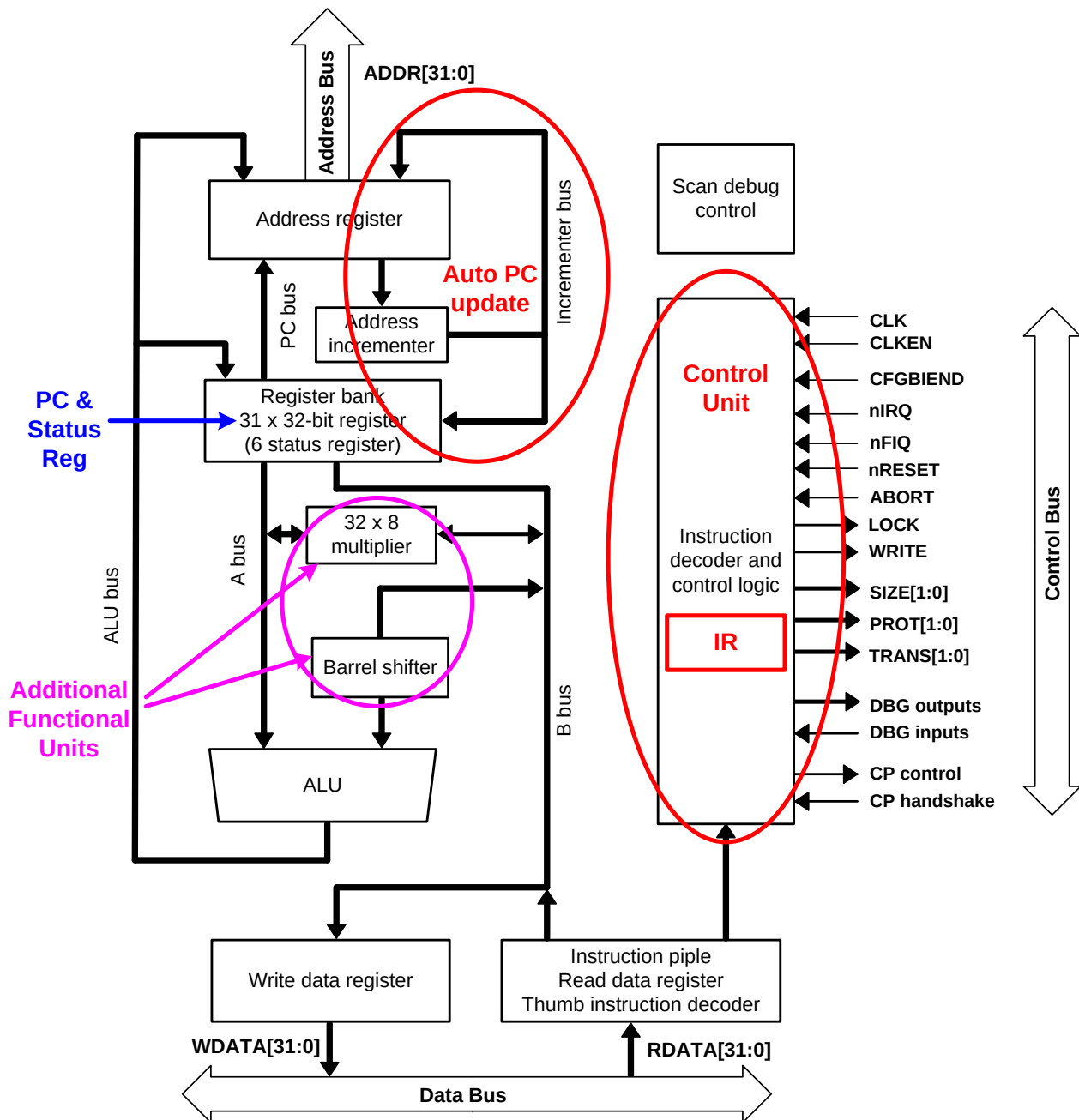
More other factors:

internal cache

hardware multiplier

number of internal general purpose registers

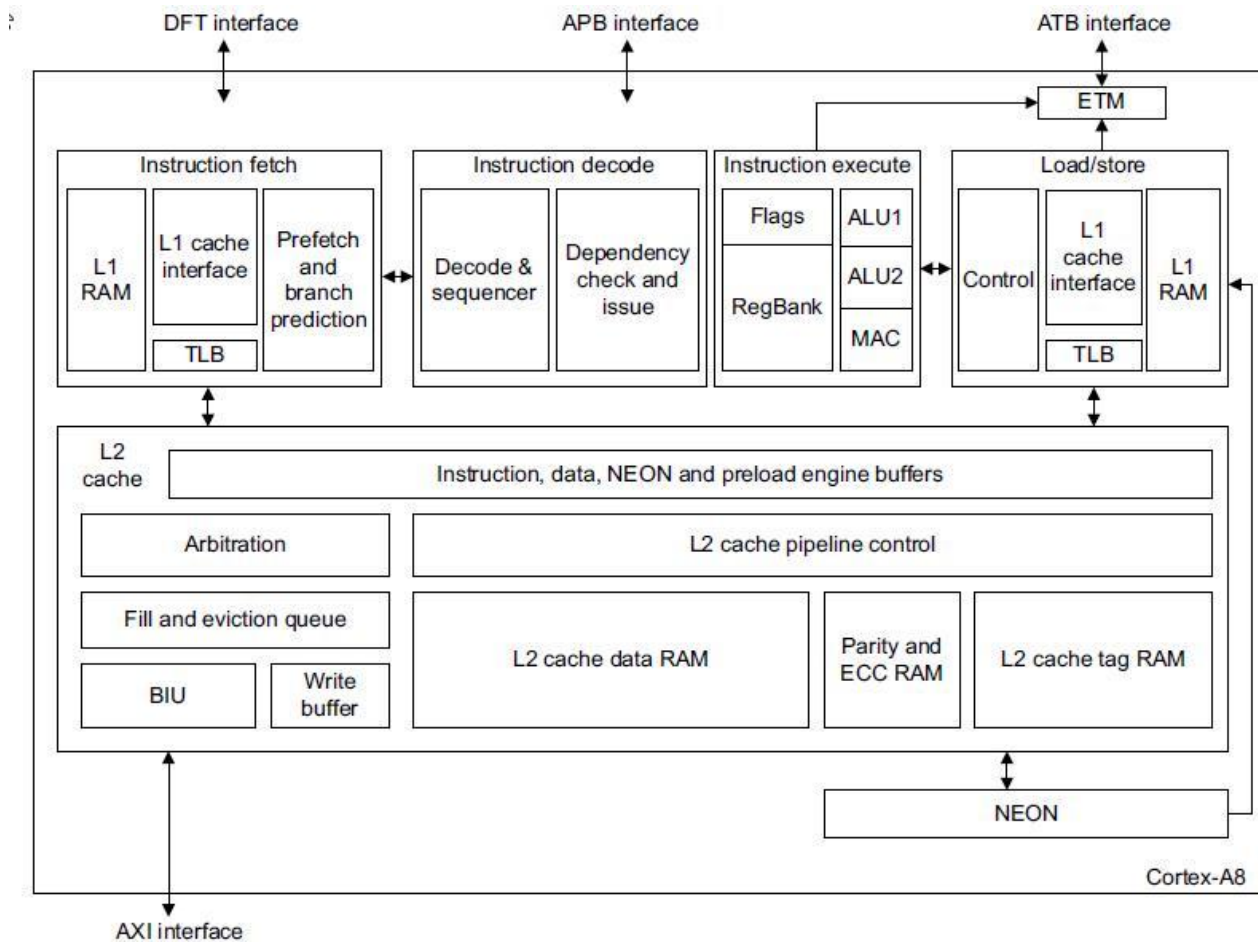
8. The ARM7TDMI Processor



ARM7TDMI-S core

Clock frequency: ~ 66 MHz

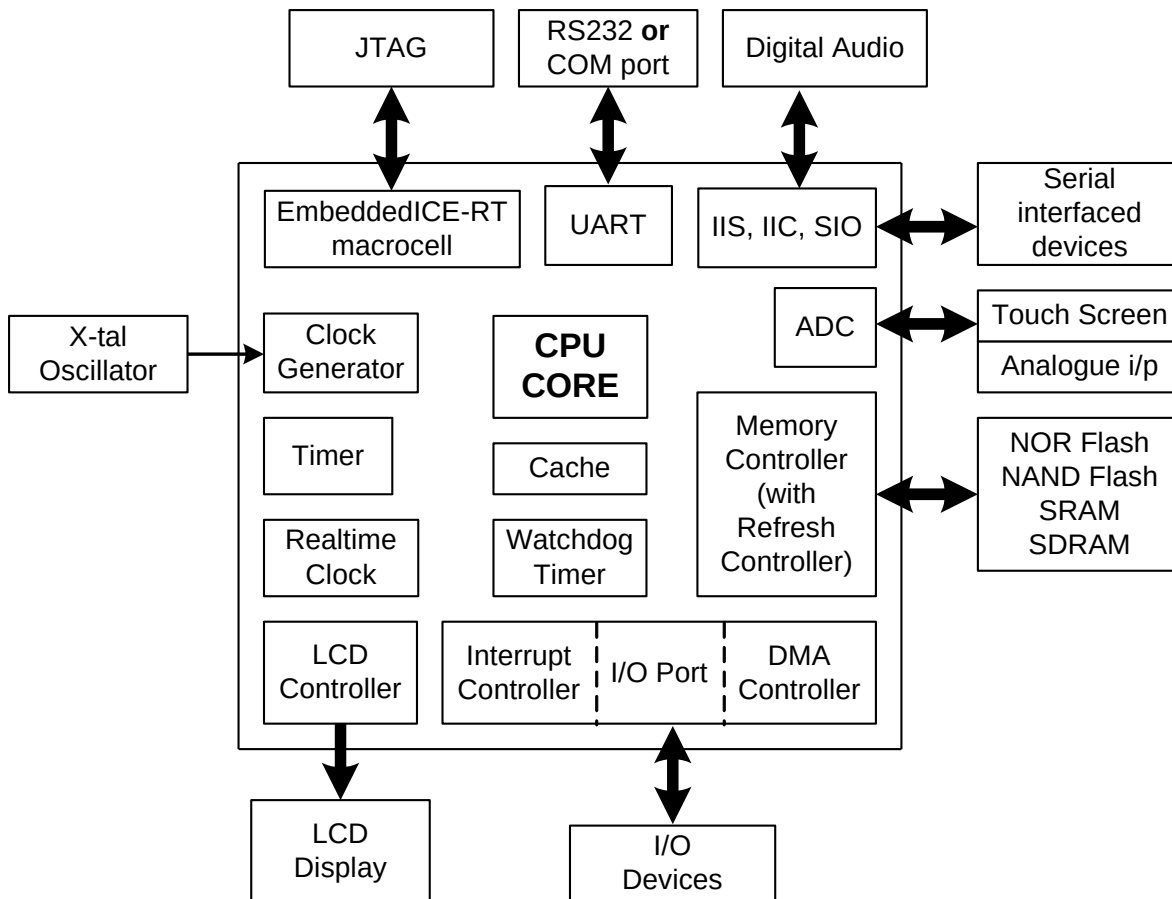
9. The Cortex-A8 Processor



Clock frequency: ~ 1.2 GHz

10. SoC (System on Chip)

Samsung S3C44BOX



Most common peripheral interfaces are integrated into one single chip.