Table of Contents

[REGISTER ORGANIZATION 2](#_Toc135596700)

[User-Visible Registers 2](#_Toc135596701)

[1. General Purpose Register: 2](#_Toc135596702)

[2. Data Register: 2](#_Toc135596703)

[3. Address Register: 2](#_Toc135596704)

[4. Condition Code Register: 3](#_Toc135596705)

[Control and Status Registers 3](#_Toc135596706)

[Program Status Word (PSW) 3](#_Toc135596707)

[INSTRUCTION CYCLE 5](#_Toc135596708)

[THE INDIRECT CYCLE 5](#_Toc135596709)

# REGISTER ORGANIZATION

The registers within the processor serve as a memory level between the main memory and cache. These registers perform two important roles. First, the user-visible registers allow the programmer to optimize the use of registers to minimize references to main memory. Second, the control and status registers are used by the control unit to manage the operation of the processor and by privileged operating system programs to control the execution of other programs.

## User-Visible Registers

User-visible registers are registers that can be accessed by machine language programs. They are typically used to store data and instructions that are being used by the CPU. There are four main types of user-visible registers:

* General-purpose registers: These registers can be used for any purpose.
* Data registers: These registers are used to store data.
* Address registers: These registers are used to store addresses of memory locations.
* Condition code registers: These registers store information about the state of the CPU, such as whether the last operation was successful.

1. General Purpose Register: General-purpose registers are registers that can be used for any purpose. They can be used to store data, instructions, or intermediate results of calculations. The programmer can assign general-purpose registers to a variety of functions. However, there may be restrictions on how general-purpose registers can be used.
2. Data Register: Data registers are used to store data, but they cannot be used to calculate the address of an operand. This is because data registers do not store addresses.
3. Address Register: Address registers can be either general purpose or dedicated to specific tasks. They can include segment pointers that store the base address of memory segments for segmented addressing, index registers used for indexed addressing that can also be auto indexed, and a stack pointer that points to the top of the stack to allow implicit addressing for stack instructions like push and pop.
4. Condition Code Register: Condition code registers hold bits set by the processor as the result of operations like arithmetic. The condition codes indicate things like positive, negative, zero or overflow results. Condition codes can then be tested as part of conditional branch instructions.

## Control and Status Registers

Control and status registers are registers that are used to control the operation of the processor. They are typically not visible to the user, but they can be accessed by machine instructions that are executed in a control or operating system mode.

Common register types include the **program counter** containing the address of the next instruction to fetch, the **instruction register** holding the currently fetched instruction, the **memory address register** with the address of a memory location, and the **memory buffer register** containing data to be written to or read from memory. Different machines have different register organizations and terminology.

### Program Status Word (PSW)

In many processors, there is a special register or set of registers called the program status word (PSW) that holds important status information. The PSW usually includes condition codes and other relevant flags. Here are some common flags found in the PSW:

* Sign: Indicates the sign (positive or negative) of the result of the previous arithmetic operation.
* Zero: Set when the result of an operation is zero.
* Carry: Set if there was a carry (in case of addition) or a borrow (in case of subtraction) from the high-order bit. Used for multiword arithmetic.
* Equal: Set if the result of a logical comparison is equal.
* Overflow: Indicates an overflow in arithmetic operations.
* Interrupt Enable/Disable: Controls whether interrupts are enabled or disabled.
* Supervisor: Indicates whether the process is running in supervisor or user mode. Certain privileged instructions and memory areas can only be accessed in supervisor mode.

These flags provide important information about the processor's status and influence its behavior during program execution.

# 

# INSTRUCTION CYCLE

The instruction cycle is the basic operation cycle of a computer for executing a single instruction. It consists of two steps - fetch and execute. In the fetch step, the processor reads an instruction from memory, fetching it. In the execute step, the fetched instruction is then executed. The instruction cycle is repeated for each instruction in a program as the program is executed by the processor. The instruction cycle is sometimes called the fetch-and-execute cycle, fetch-decode-execute cycle, or FDX cycle.

## THE INDIRECT CYCLE

The indirect cycle is an additional step in the instruction cycle that is required when an instruction uses indirect addressing. Indirect addressing means that the address of the operand is stored in memory, and the processor must fetch the address from memory before it can fetch the operand.

The indirect cycle consists of the following steps:

* The processor fetches the instruction from memory.
* The processor decodes the instruction and determines if it uses indirect addressing.
* If the instruction uses indirect addressing, the processor fetches the address of the operand from memory.
* The processor fetches the operand from memory.
* The processor executes the instruction using the operand.
* The indirect cycle adds an additional step to the instruction cycle, which can slow down the execution of the instruction. However, indirect addressing can be a useful technique for accessing data that is stored in a variable location.