What is an Instruction Set?

An **Instruction Set** is a collection of all the commands a CPU can execute. For an **accumulator-based CPU**, all operations involve the **Accumulator (AC)** by default. The CPU typically uses:

- A single memory address per instruction (*single-address format*)
- The AC as one operand in every operation

Instruction Set Table Breakdown

Type	**Instruction**	**HDL Format**	**Assembly Format**
Data	Load	AC := M(X)	LD X
	Store	M(X) := AC	ST X
Register Move	Move Register	DR := AC	MOV DR, AC
Arithmetic	Add	AC := AC + DR	ADD
	Subtract	AC := AC - DR	SUB
	And	AC := AC `AND` DR	AND
	Not	AC := `NOT` AC	NOT
Control	Branch	PC := M(adr)	BRA adr
	Branch if Zero	if AC = 0 then PC := M(adr)	BZ adr

Arithmetic Operation: Negation ($-X$)

Negation (computing $-X$) is not directly supported with a `NEG` instruction. Instead, it is done using subtraction logic.

Breakdown of Negation:

Let's say `AC` contains X. To compute $-X$, you can do:

HDL Format	**Assembly Format**	**Explanation**
DR := AC	MOV DR, AC	Save X into DR (backup)
AC := AC - DR	SUB	Subtract X from $X \rightarrow$ AC = 0
AC := AC - DR	SUB	Subtract X again $\rightarrow$ AC = -X

So in total:

- First SUB makes accumulator zero
- Second SUB makes the accumulator contain $-X$

This is how negation is implemented using only SUB and the accumulator.

Summary

- This accumulator CPU uses a simple instruction set with single-address instructions.
- Every operation is based on manipulating the accumulator and optionally the data register.
- Even complex operations (like negation) are broken down into multiple basic instructions.
- The instruction set reflects a minimal but powerful set of tools to manipulate data and control flow.