**Nano8 General Purpose Computer**

by Jakob Flocke (TheCodeCurrents)

updated 02.04.2025

**User Manual**

Welcome to the Nano8 General-Purpose Computer Project. The goal of this project is to create a general-purpose computer that is simple enough for beginners to understand while maintaining strong performance. Currently, the project is still in its early stages, so this user manual serves more as a progress report—written as if the product were already complete.

The project aims to encompass a wide range of components, including:

* A custom CPU core
* Various computer designs built around the core
* A fully functional operating system
* An assembler and assembly language
* A simple programming language providing some assembly abstractions
* Multiple OS programs and applications
* A simple GPU for graphical applications

It will be implemented as an FPGA soft-core SoC, an emulator, and using TTL logic chips.

Ultimately, this project aims to be both an educational resource and just an enjoyable experience.

**Table of Contents**

[CPU Architecture 3](#_Toc1)

[Table of Contents 3](#_Toc2)

# **CPU Architecture**

This section covers the CPU architecture, from a high level overview down to the function of each module.

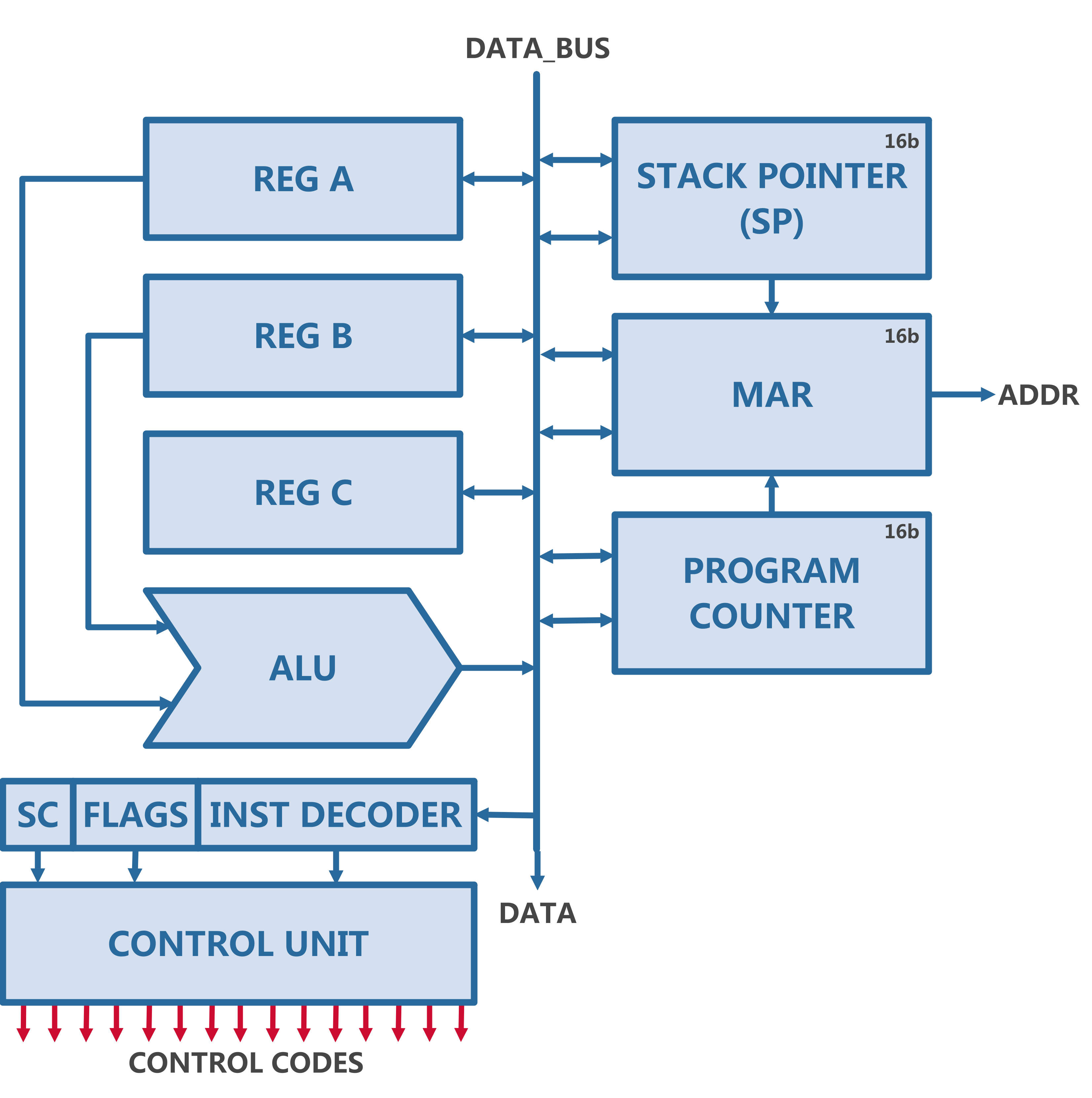


Figure 1: CPU Block Diagram

On a top level the CPU is made up of four individual sections:

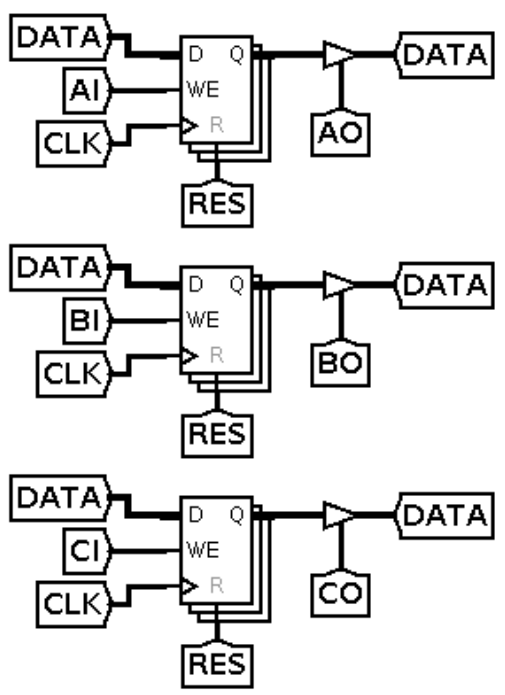
* Data section
  + Register A-C
* Arithmetic section
  + ALU
* Address section
  + stack pointer (SP), program counter (PC) and memory address register (MAR)
* Control section
  + Flags register, step counter (SC), instruction decoder and the control unit (CU)

Note: The clock signal needs to be provided.

## Data Section

The Data Section is the most simple one, it is consists of three 8-bit registers, which are connected to the Data BUS both on the input and on the output. Additionally they all have two control signals each, one for taking data in and the other for outputting the data onto the bus. Once a reset pulse is sent they all reset to their default value 0x00.

Figure 2: CPU Data Section (Logisim)



The behaviour of the registers is simple, they just store a single 8 bit value as long as they are not reset. When the IN (AI, BI or CI) control signal of a register is high, the registers will read the data from the bus on the rising edge of the clock. When the OUT (AO, BO or CO) control signal is high, the value will be outputted to the bus unrelated to the clock signal.

The only instructions that could be done with only the data section (if control signals are controlled manually) are basics like a move operation, where a value is moved from one register to the other or a switch operation, where a third register is used as a buffer to switch the values of two registers.