Objective:

Clearly explain the scope of the work, determine materials needed for the development, and also create the environment needed for the development of the virtual CPU.

Tasks:

1. Define Virtual CPU Features Elaborate on the big components of the virtual CPU:

Basic Registers - Accumulator, program counter, and stack pointer. Instruction Set Architecture that would contain basic operations like Arithmetic, Logical Operations, Data Transferring, and Branches. Memory Management Capability. I/O Capability if needed. Advanced features-interrupt handling or floating-point arithmetic, depending on project complexity.

• Select Architecture: 8-bit, 16-bit, or 32-bit CPU.

2. Choose a Programming Language and Development Tools:

• Programming Language:

• Python: more convenient for quick prototyping because it is easier to do, especially for beginners or in cases when the high-level work of CPU is more important.

• Development Tools:

• IDEs like VS Code Python.

• Debuggers: PDB or integrated VSCode debugger for Python.

• Testing frameworks, for example, Pytest for Python .

3. Version Control Setup:

• Version Control System: Git for source control and interaction among multiple users.

• Repository Hosting: GitHub repository for ease of access, collaboration, and tracking progress.

• Create feature branches for different features to keep development organized; for example, feature/registers, feature/instructions, bugfix/memory-handling.

The following steps are a foundation for the Virtual CPU Project: they ensure that there are objectives and that an environment is prepared for development. If further elaboration on any of these steps is needed, just let me know!