**Minimal Processor Design Document**

# Design Goals

The goals for the processor can be summarized in the following hierarchy with importance in descending order:

1. Remaining all basic computing functionality (Remaining Turing complete)
2. Using basic components and available ICs
3. Minimizing component count and hardware complexity
4. Minimizing clock cycles per command
5. Keeping instructions simple and functional
6. Minimizing instructions for important codes

# Methodology

To achieve the required design goals while prioritizing the most important goals first, we use subtractive design, where we start from a functional example and remove from it until we reach the minimum functional product that still aligns with our most important goal which is remaining Turing complete.

The processor we start from is the xxxxx processor originally designed by xxxxx for educational purposes and described in his book xxxxx. We then remove any instructions that can be achieved through alternative methods in a reasonable amount of time.

The less complicated the commands are the less hardware the control unit takes, which forms the bottleneck when it comes to assembling a physical version of this processor without the use of PCBs. As such moving functionality from hardware to software forms the basis of most of the following changes.

# Changes

Multiple skips are replaced with just a singular skip instruction. Constructing any of the other instructions then requires some logical or arithmetic manipulation of the subject to achieve the desired condition.

Increment and clear are removed, as they can be achieved using a load or add respectively. The AND instruction is replaced with a NND (Logical NAND operation) to allow the removal of the CMA (complement) instruction and using the NND command as the basis of all logical operations.

Multistage commands such as ISZ (Increment and skip if zero) and BSA (Branch and save address) have been removed as they can both be replicated through multiple commands. Their removal also facilitates the decrease of maximum clock cycles per command while increasing instruction count, which lines up with the importance hierarchy of our design goals.

Another important change is to the flag handling, as all flags and flip flops can now be interfaced together through LDF (load flags) and STF (store flags) commands. This forms the basis of all flag-based interactions from flag-specific skips, to manipulating flags including the interrupt enable and the S flag used to halt the processor. However, to maintain interrupt responsiveness, another necessary command is XRF which allows the processor to flip specific bits without affecting others that may change during the processing of information. A couple more commands were added out of simplicity but can be removed which are an and (ANF) and an or (ORF) for clearing and setting flags respectively

Another major change affects the addressing from memory. The memory addressing is no longer automatic at the beginning of each memory instruction execution and is instead done manually through software. This change takes indirect referencing from the hardware side to the software side minimizing the hardware.

Another change made was removing multiple registers and connections that were unnecessary in the original design, having a minimal data bus with just 2 possible data line options for memory, and 4 data lines connected to the bus internally from memory output. Every other register I connected directly to relevant registers.

And an address bus has also been added to facilitate indirect addressing and direct parameter pulling in the new design. The added bus is less of an addition and more of a result for the splitting of the original bus to 2 separate buses.

One final feature that was added, not out of necessity, but as a reasonable adjustment, is the replacement of one of the circulation operations (CIR) with a reverse operation (REV). This allows performing the other operation through more commands, taking slightly more time; however, it also takes less time to derive the second circulate operation from the inverse operation than vice versa.