ABACUS I OPERATOR'S MANUAL

ABACUS I
M: COM0001SP
SN: 00000001
D: 28-10-2017

TABLE OF CONTENTS

PART 1 INTRODUCTION

A. BACKGROUND

The *Abacus I* was created as a research and development project to explore how computers work at the lowest level. It is based on *Ben Eater's SAP-1* breadboard computer. The *Abacus I* is built on three Perfboards (DOT PCBs) with extra boards for extra components (e.g. voltage protector). ~~*Abacus I* is Turing complete.~~ It is reminiscent of 1970s microcomputers, i.e. MITS' Altair 8800. The case opens to allows access to the internals. LED's have been added in circuits where they would normally be unnecessary (e.g. Bus and Registers), to allow the user to understand what the computer is doing at any given moment.

B. FOR BEGINNERS

This operating manual will describe the functions and processes of the *Abacus I*. To help inexperienced or new "bare-bones computer" users understand the workings of the *Abacus I*, we recommend reading part one of the *Altair 8800 Operator's Manual*. MITS' manual goes into extreme detail regarding logic, electrical logic, number systems, the binary system, the octal system, computer programming, a simple program, and computer languages. Despite being published in 1974, the information is still valid and applies to the concepts of the *Abacus I*'s functions.

C. SPECIFICATIONS

Name:	*Abacus I*
Model:	*COM0001SP*
Began Develop.:	*07/10/2017* (DD/MM/YYYY)
Code Name:	*Project I*

CPU:	TTL/Non-microprocessor, 8bit bus
Clock:	A/Monostable, 0.7-742Hz
Architecture:	*SAP-1 (Modified)*
Instruction Set:	*Simplex 8.1* (8bit, Version 1)
ALU:	Add & Subtract Capable
Memory:	16 8bit words, 4bit address, Static RAM
Input:	XX switches and buttons (Front), X switches (Internal)
Output/Display:	4 7-Segment Decimal Digits (Front), XX LED’s (Front), XX LED’s (Internal)

PART 2 ORGANISATION OF THE ABACUS I

A. COMPUTER DIAGRAM

Figure 2-1 shows the PCB of the *Abacus I* with labelled modules (green), control pins (yellow), input pins (blue), and output LEDs (red).
Figure 2-2 shows the front panel of the *Abacus I* with labelled input switches and output LEDs.

This manual goes into further detail regarding modules, control pins, and input/output switches/LEDs in parts two, four, and three respectively.

FIGURE 2-1

FIGURE 2-2

B. ARCHITECTURE

The *Abacus I*’s modified *SAP-1* (Simple as Possible 1) architecture is centred around the bus and is extremely modular.

The *SAP-1* architecture is shown in Figure 2-3.
The *Abacus I*’s *SAP-1 (modified)* architecture is shown in Figure 2-4.

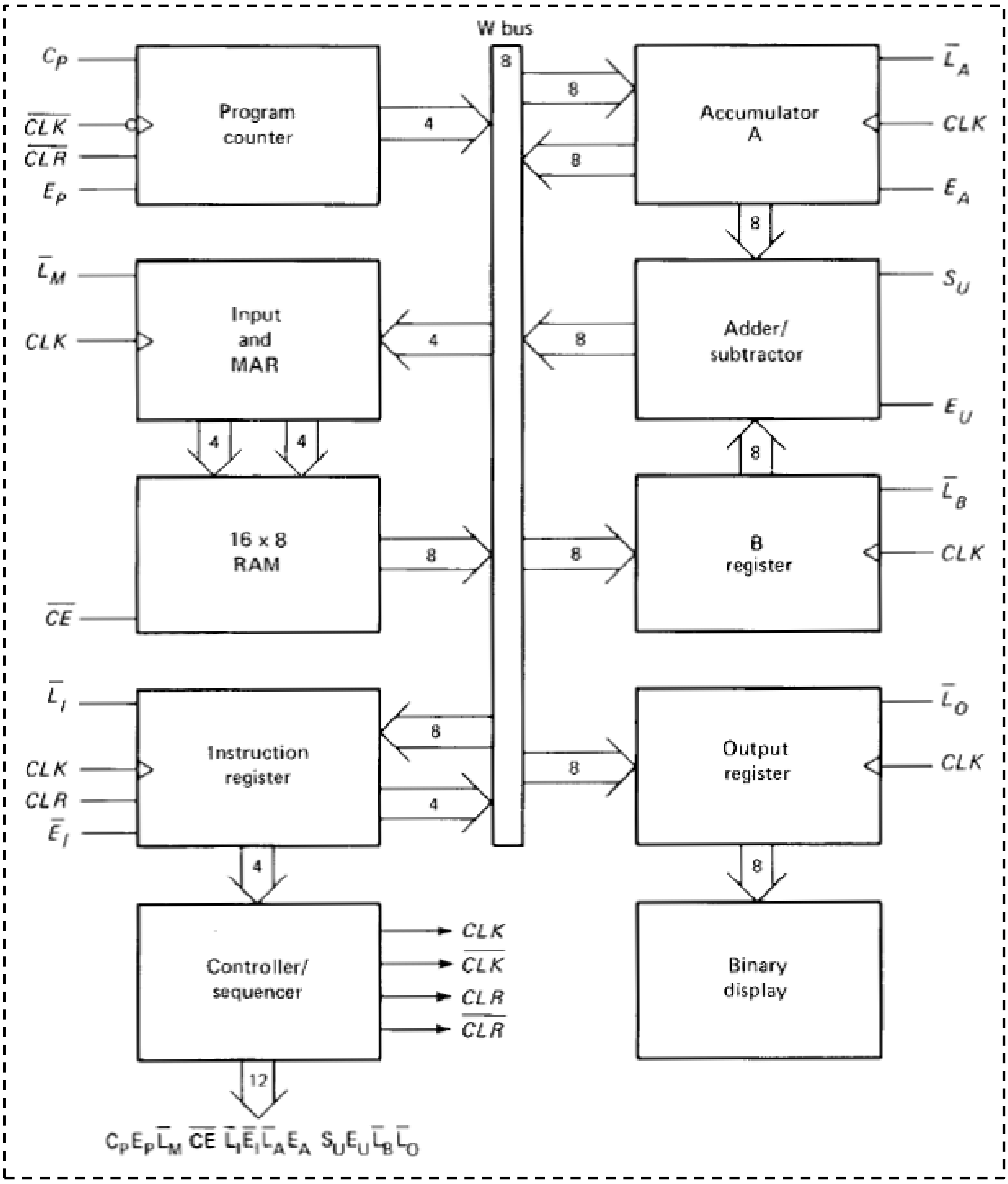

FIGURE 2-3

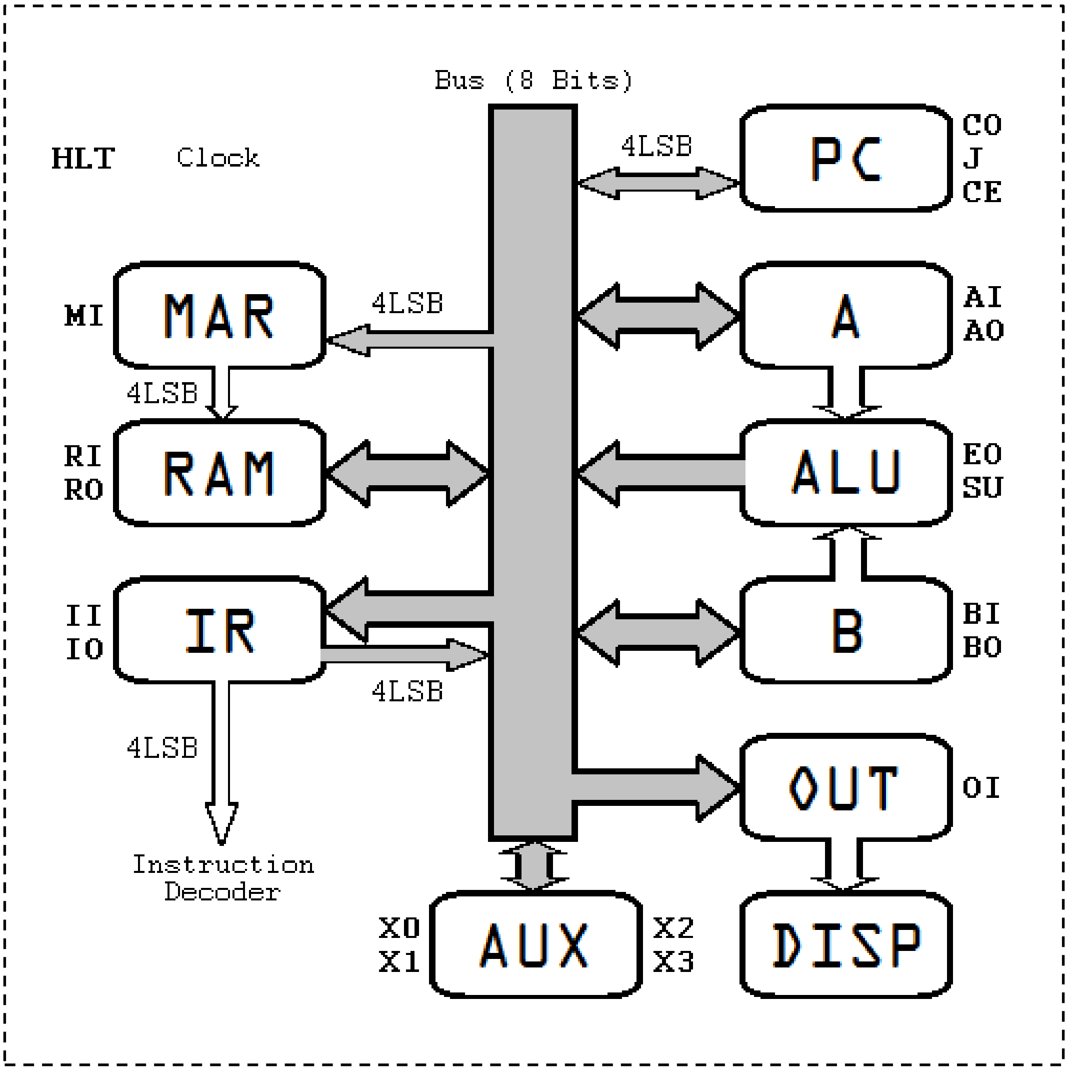

FIGURE 2-4

C. MODULES

Each block represents one of the computer's modules, and the arrows represent how these modules interact.

4LSB: Four least significant bits.
4MSB: Four most significant bits.

The bus is 8 bits wide. The bus also contains headers for an auxiliary module to be attached post production.

CLK	Clock
PC	Program Counter
MAR	Memory Address Register
RAM	Random Access Memory
IR	Instruction Register
A	Register A
B	Register B
ALU	Arithmetic Logic Unit
OUT	Output
DISP	Decimal Display
INST	Instruction Decoder
AUX	Auxiliary Module

It is not necessary to understand the detailed electronic operation of each part of the computer to make effective use of the machine. However, a general understanding on each of the various operating sections is important.

1. CLOCK--The clock keeps the whole system in time. Using the dial labelled *Clock Speed* the frequency at which the computer completes operations can be adjusted. The speed ranges from 0.7Hz to 742Hz. The LED labelled *Clock* shows the pulsing of the clock, and will reflect the computer's current speed. The clock also has two functioning modes, Astable Pulse Mode (Run Mode) and Manual Pulse Mode (Step Mode). Pulse mode is considered the normal operating mode where the clock rhythmically pulses at its set speed. If you need to, for example, slow the computer's speed further than 0.7Hz or are trying to find a bug in your program, you can switch to Step Mode. In this mode, the clock pauses and awaits the programmer's input via the button labelled *Step Clock* before progressing the computer's processes by one pulse. The HLT (halt) instruction links directly to the clock and forces the system to pause itself. This allows programs to halt the computer once they have finished running and allows the programmer to view the output without the display refreshing.

2. PROGRAM COUNTER--The program counter keeps track of which memory location the computer is reading/executing from. Using the CE (counter enable) control progresses the counter forward one location. The counter only ranges from 0 to 15, however this is not an issue because the *Abacus I* only supports 16 memory addresses. The *PC* will loop continuously through unless the clock is halted or a jump is specified. The J (jump) control can be used to input a value from the bus to the *PC*. This allows programs that occupy less than 16 memory addresses to loop and remain in control of the computer. When operating the computer manually (as opposed to through a program in memory) the CO (counter out) control allows you to put a number between 0 and 15 on the bus for other modules to make use of.

3. MEMORY ADDRESS REGISTER--The *MAR* takes a memory address from the bus (4 least significant bits) and instructs the *RAM* module to navigate to that address. The *RAM* module then inputs or outputs the contents. This is completed using the MI (memory in) control.

D. INPUT/OUTPUT

The front panel of the *Abacus I* contains all the switches and LEDs required to control the computer. These are soft-wired to the PCB using friction headers. This allows for the components to be disassembled but can cause confusion when accessing the innards of the computer. Refer to part two's computer diagram to locate the I/O headers and their front panel counterparts.

E. AUXILARY MODULES

On the board of the *Abacus I* are 8 friction headers in the bottom right corner. These headers are attached to the bus and allow for an auxiliary module to be connected to the computer. In addition, X control lines have been left vacant to allow for control of an auxiliary module. Auxiliary control codes will have to be programmed into the microcode EEPROMs. More on the custom controls of the auxiliary module in part four and appendix one. ~~{Custom controls have two control pins active and are AND gated to the one output. Use address 16-19 on first EEPROM}~~

An example of an auxiliary module could be an EEPROM storage module to save and load programs.

All bus and control headers are labelled in part two's computer diagram.