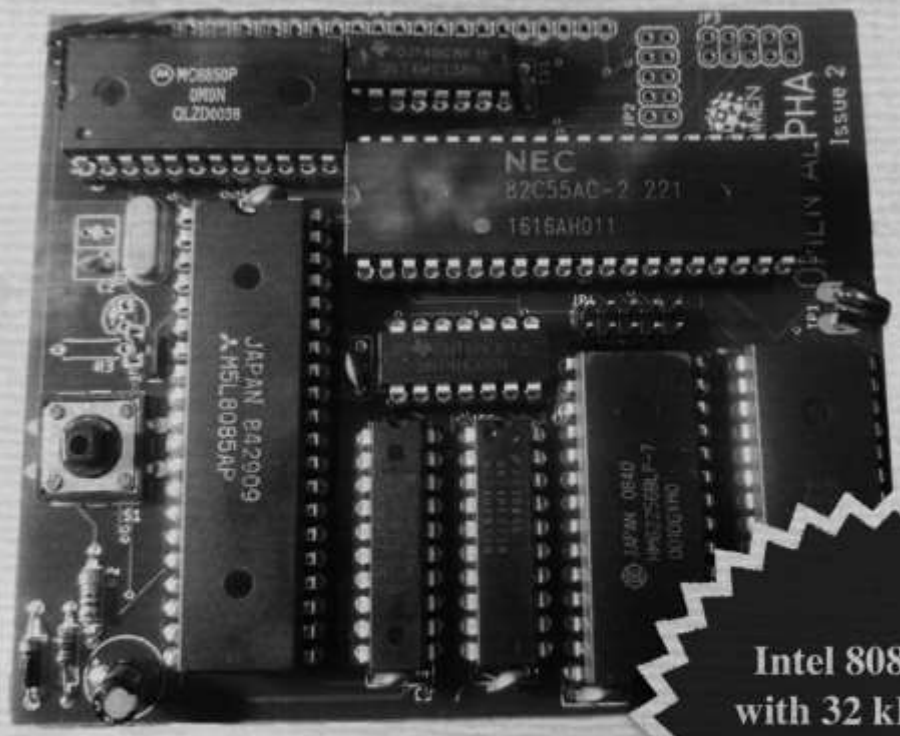


Intel 8085 CPU with 32 kB RAM and 32 kB EEPROM

OMEN ALPHA

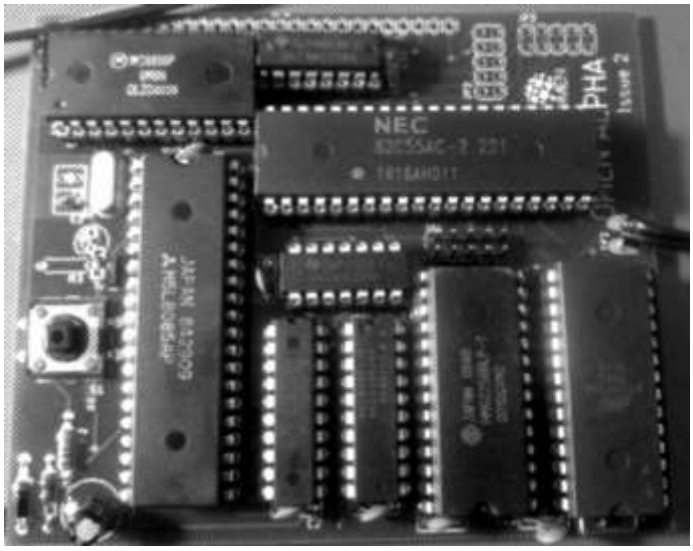
Simple, powerful, extendable

A perfect 80's computer for the true geeks! You can design and build your own peripherals in the 80's spirit. Your fantasy is not limited by hardware anymore. The OMEN Alpha is a perfect choice when you want to learn more about computers, microelectronics and digital circuits, as well as to learn an assembler or other languages (BASIC, C). Embedded parallel and serial interface allows you to use the Alpha as a control unit for other devices. Powered by Intel 8085 CPU with 32 kB RAM and 32 kB EEPROM, this computer suits perfect as an entry small computer for an 8 bit world.


OMEN

OMEN Alpha / issue 4

Technical Documentation



INTRODUCTION

==========

The OMEN Alpha computer kit is a low-cost computer trainer, based on the Intel 80C85 CPU. It has these features:

- 80C85 CPU working at 1.8432 MHz
- 32 kB RAM
- 32 kB EEPROM
- Serial port up to 115.200 Bd / MC6850
- 3 parallel ports / Intel 8255
- Application system bus

ASSEMBLY INSTRUCTIONS

==========

1. Solder sockets for the integrated circuits
2. Test all soldered connections
    a. Test if all pins are well connected
    b. Check if GND is not short connected to Vcc
    c. Check if each IC has properly connected GND and Vcc
3. Solder all passive parts /capacitors, diode, resistors, push button, crystal/
4. Connect the power adapter and check
5. Insert the CPU into its socket /keep the proper orientation!/ and try to power it up. Check if oscillator lives /at CPU pin 37/
6. Insert the essential ICs: 74245, 74573, 7400, 62256, 6850 and AT28C256. Again: keep the proper orientation! Bad orientation can damage the IC!
7. Connect the serial pins TxD, RxD and GND /pinhead JP1/ to the TTL-to-USB converter
8. Start the serial terminal on your PC, select proper serial port and set the parameters to 115.200 Bd, 8 data bits, no parity, 1 stop bit.
9. Power your Alpha and check the terminal.

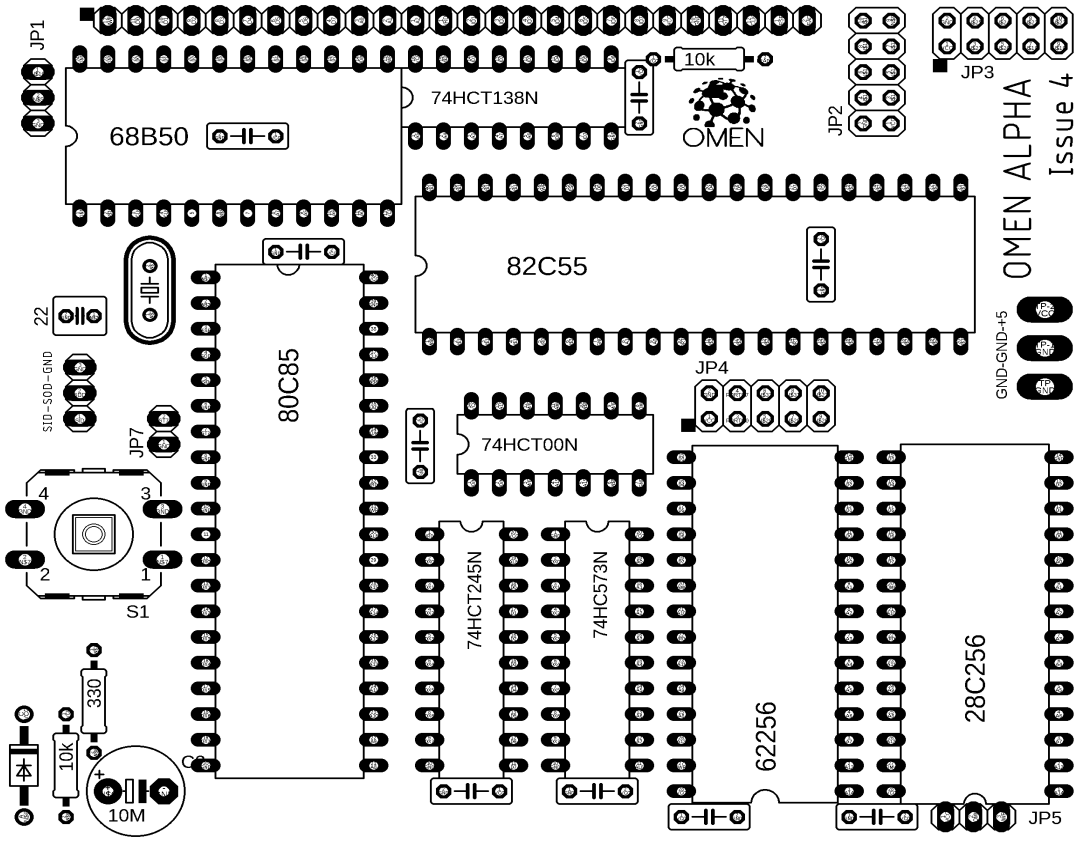
MONITOR

=======

The main software for OMEN Alpha issue 4 is the Dave Dunfield's MON85 Serial Monitor, improved by Roman Borik.

Usage is described on the https://8bt.cz/mon85 or see the Appendix 1

You can invoke the installed Tiny BASIC v.2 by jumping to the 1000h - just enter "G1000" /no quotes/ and press the ENTER key.



Jumpers and pin headers

JP1: Serial port. Pins are GND, TxD, RxD from left to right, RxD is next to the JP1 label

JP2, JP3, JP4: Parallel ports. See below. The square marks the pin 1.

JP5: EEPROM write enable. Short pins 1-2 to disable writing, or short pins 2-3 to enable writings. Pin 1 is on the left, pin 3 is next to the JP5 label

JP7: RST7.5 enabler. Close to disable interrupts, open to leave it on the application bus. Connect the 10k resistor to make it "open-collector signal".

JP8: Serial signals SID and SOD

PARALLEL PORTS

==========

JP2:

GND	PA7	PA6	PA5	PA4
Vcc	PA0	PA1	PA2	PA3

JP3:

GND	PB7	PB6	PB5	PB4
Vcc	PB0	PB1	PB2	PB3

JP4:

GND	PC7	PC6	PC5	PC4
Vcc	PC0	PC1	PC2	PC3

SYSTEM APPLICATION CONNECTOR

==========

This connector is on the upper edge of board. Pin 1 is on the left side.

Pins:

1 /WR
2 D0
3 D1
4 D2
5 D3
6 D4
7 D5
8 D6
9 D7
10 A0
11 A1
12 A2
13 /RD
14 IO1 --- 20h - 27h
15 IO2 --- 10h - 17h
16 IO3 --- 30h - 37h
17 IO4 --- 08h - 0Fh
18 IO5 --- 28h - 2Fh
19 IO6 --- 18h - 1Fh
20 IO7 --- 38h - 3Fh
21 Vcc
22 GND
23 RST7.5 -- see JP7
24 CLK
25 RESET -- Output signal
26 READY -- Pull it low to force a WAIT state
27 /BUSRQ /not connected on Alpha/
28 /BUSACK /not connected on Alpha/
29 /not connected on Alpha/
30 /not connected on Alpha/

IOx signals are decoded by 74138

PERIPHERAL IC ADDRESSES

ACIA 6850:

DEh - Controll Register / Status Register

DFh - Data Register

PPI 8255:

04h - PORT A

05h - PORT B

06h - PORT C

07h - Control port

MEMORY MAP
==========

0000h - 7FFFh - EEPROM 32k

8000h - FFFFh - System RAM

APPENDIX 1 - The MON85 Manual

==========

MON85

A
Software Debug Monitor

For the 8085/8080

Dunfield Development Systems

High quality tools for
Embedded Development
at low prices.

http://www.dunfield.com

MON85

TABLE OF CONTENTS

1. INTRODUCTION

MON85 is a ROMable interactive debugging program for the 8085 processor, which contains a full complement of commands to monitor and control the execution of your program. It may also be used on an 8080 processor, as long as you do not attempt to use the 8085 specific SIM and RIM instructions, or the '.5' type interrupt vectors.

MON85 must be installed beginning at location $0000 in the 8085 processor memory map, allowing it to intercept the interrupt vectors. Provision has been made to re-vector interrupts to locations within the user program.

All functions of MON85 are performed via software without hardware assist. The only hardware specific subroutines required by the monitor are used to communicate with the console terminal, and are located at the very end of the monitor source code listing:

INIT - Called to initialize any hardware required for I/O.

OUT - Write the character in A to the console. No processor registers should be modified by this routine.

IN - Test for a character from the console, and return it in A if one is available. Otherwise clear A to zero. No other registers should be modified by this routine.

1.1 8085 Restart Interrupts

MON85 reserves two "restart" interrupts: RST 0 is similar to a physical RESET, causes a cold restart of the monitor. This instruction should be used with CAUTION during a debugging session because any breakpoints set in the user program will be "forgotten", and not properly removed. RST 1 is used by MON85 to regain control at breakpoints. Opcodes of breakpointed instructions are replaced by RST 1 whenever the user program is executed, and are restored whenever the monitor is re-entered. This insures that the operation of breakpoints is transparent to you during the debugging session.

If for any reason a 'RST 1' ($CF) instruction is encountered in the user program (and is not a breakpoint), command mode will be entered without the '** Breakpoint' message.

All other "restart" interrupts are re-vectored to the corresponding locations in the first page of memory occupied by the user program (as identified by the 'U' command).

1.2 Breakpoints

MON85 allows you to set breakpoints in your program, such that you will be given control whenever the program reaches a breakpoint, and can examine or change things before proceeding. MON85 also allows you to TRACE a program, so that you can see each instruction and register contents as it executes.

MON85 is completely transparent to the program being tested (unless timing loops are interrupted with BREAKPOINTS or TRACE). When a breakpoint is encountered, or an instruction is traced, MON85 uses one stack entry on the user program stack. Since this consists of a PUSH and a POP, it should not affect any information stacked by the user program. However, you should be aware of this, in case you are examining the stack, or your program trys to reclaim data already popped from the stack. (It is very poor practice, to write programs which depend on the stack contents below the stack pointer).

When a breakpoint is encountered, the message '** Breakpoint n' is printed where n is the number of the breakpoint (0-7). If TRACE is ON, no other action is taken, otherwise command mode is entered (If the 'A' switch (see below) is ON the registers are also displayed).

2. MON85 COMMANDS

The following commands are recognized by MON85, and may be entered whenever MON85 is in COMMAND MODE, which is indicated by the 'C> ' prompt:

2.1 A ON|OFF

This switch turns ON or OFF the automatic register display which occurs whenever a breakpoint is encountered or an instruction is stepped in trace mode (see below). The default value for 'A' is ON.

2.2 B [0-7 address]

Sets one of eight breakpoints [0-7] at the specified address. Once

set, a breakpoint remains in effect until it is removed by setting it to address ZERO (0). Breakpoints are completely invisible, and may be added, removed or changed at any time without adverse affects. If the 'B' command is issued with no operands, the current breakpoints, and the settings of the 'A', 'S' and 'T' flags (see below) are displayed. A displayed address of '****' indicates that a breakpoint is not set.

2.3 C <source> <destination> <size>

Copies <size> bytes of memory from the <source> to the <destination> address.

2.4 D <address>

Displays memory in assembly listing format, starting at <address>, continuing until an ESCAPE character is entered. Output can be temporarily stopped with the SPACE BAR, and restarted with the RETURN key (Additional SPACE's will output single lines). The output is displayed in the following form:

'<address> <opcodes> <ascii> <instruction>'

<address> is the current memory location, <opcodes> is the instruction opcodes (1, 2 or 3 bytes), <ascii> is ASCII representation of <opcodes> (all non-printable characters are displayed as dots), and <instruction> is the assembly language instruction and operands which <opcodes> represents.

2.5 E <address>

Edits memory, starting at <address>. The address, and its contents are displayed, followed by a '=' prompt. Sub commands are:

nn [nn ...] - Replace memory contents with HEX data bytes
'text ... - Replace memory contents with ASCII text
- - Backup to previous locations
<blank line>- Advance to next location with change
<ESCAPE> - Return to MON85 command prompt

2.6 F <start> <end> <value>

Fills memory between the <start> and <end> addressses with the specified <value>.

2.7 G [address]

Loads the user program registers, and begins execution at the specified address. If no [address] is given, execution begins at the address contained in the user program program counter (PC). A simple 'G' with no operands, is all that is needed to resume execution after a breakpoint interrupt, or to resume trace mode exection.

2.8 I <port>

Reads the specified 8085 I/O port, and displays the data.

2.9 L [address]

Downloads code from the console port. MON85 will recognize and accept either INTEL or MOTOROLA hex format download records. The

address in the FIRST download record is recorded as the program BASE address (See 'U' command). If an [address] is given, the address fields of each record are adjusted so that the code image will be loaded into memory beginning at that address, otherwise it is placed in memory at the absolute address contained in each record.

If you accidently enter this command, you may enter either 'S9' or ':00' to signify a null download file and return to the command prompt.

2.10 M <address>

Displays memory in HEX/ASCII dump format starting at the specified address. ESCAPE, SPACE, and CARRIAGE-RETURN can be used to control the listing, the same is with the 'D' command.

2.11 O <port> <data>

This command writes the <data> byte to the specified 8085 I/O <port>.

2.12 R [rp value]

Changes the contents of the user program register pair <rp> to the specified <value>. Valid pairs are BC, DE, HL, SP, PC, and PSW. If no operands are given, the contents of the user program registers are displayed.

2.13 S ON|OFF

Controls the handling of subroutine calls when in trace mode (see below). When 'S' is set ON, subroutines will be be traced in the normal fashion. When set OFF, subroutines calls are not traced, and trace will resume at the next instruction following the CALL (After the subroutine executes).

WARNING: DO NOT SET BREAKPOINTS IN THE SUBROUTINES WHEN 'S' IS SET OFF. When 'S' is set off, MON85 considers the subroutine (and all of it's embedded instructions) to be one single operation. If a breakpoint is encountered inside the subroutine, TRACE will lose the address at which to resume following the subroutine call. This will cause unpredictable action when the subroutine returns. Otherwise, 'S' may be turned on and off at any time with no adverse effects. The default value for 'S' is ON.

2.14 T ON|OFF

Turns TRACE mode ON and OFF. When TRACE is set ON, and the GO command is issued, MON85 will first prompt with 'T>' before beginning program execution.

Entering a space (' ') will display and execute one instruction, and return to the 'T>' prompt.

Entering '?' will display the processor registers. (If the 'A' switch is set ON, they will always be displayed following every instruction that is traced).

An ESCAPE charecter will cause MON85 to return to the usual command prompt (TRACE remains ON, and will resume with the next 'G' command).

Trace may be turned on and off at any time in a program, with no adverse effects. (If you begin execution with TRACE off, you will have to hit a breakpoint to get you back to command mode before you can turn TRACE on).

When a breakpointed instruction is encountered by the tracer, The message '** Breakpoint n' will be issued at the end of the previous instruction, indicating that the breakpointed instruction occurs NEXT. Pressing the SPACE BAR would then execute the breakpointed instruction. The default value for 'T' is OFF.

2.15 U [address]

Identifies the starting address of a user program. MON85 uses this address to re-map the "restart" interrupt vectors. When a 'L'oad command is performed, MON85 initializes 'U' to the address contained in the first download record. If no [address] is specified, MON85 will display the current user program starting address.

2.16 ?

This command displays a short summary of the other MON85 commnds.

3. COMMAND SUMMARY

A ON/OFF - Enables/Disables auto register display.

B [0-7 address] - Sets/Removes/Displays breakpoints.

C <src> <dest> <size> - Copy memory

D <address> - Displays memory in disassembly format.

E <address> - Edit memory contents.

F <start> <end> <value> - Fill memory with a value.

G [address] - Begin/Resume program execution.

I <port> - Input from port

L [address] - Download from host

M <address> - Displays memory in dump format.

O <port> <data> - Output to port

R [rp value] - Sets/Displays register contents.

S ON|OFF - Enables/Disables subroutine traceing.

T ON|OFF - Enables/Disables TRACE mode.

U [address] - Set/Displays program base address.

? - Display command summary

MON85 Improvements by Martin Borik:

- Support for undocumented 8085 instructions DSUB B, ARHL, RDEL, LDHI d8, LDSI d8, LHLX D, SHLX D, JNK a16, JK a16, RSTV
- Command R displays all flags of F register (SZKA3PVC). If flag is not set dash '-' is displayed.
- Added restart vector RST 8 (0040h) for possibility to handle RSTV call.
- Changed TRACE mode. After entering TRACE mode, instruction on actual PC and content of registers (if it is switched on) are displayed. Entering a space ' ' executes this instruction, and returns to the 'T>' prompt with the next instruction.
- Instructions LXI, DAD, INX, DCX displays argument 'SP' rather than 'S'.
- Commands that requires 1 byte parameter raises error if entered value not fit to 1 byte.
- Command 'C' checks overlap of source and destination block and for copying uses appropriate direction.
- Command 'F' checks <start> and <end> parameters and raises error, if <end> is lower than <start>.
- Added command 'H' to send out memory content in Intel HEX format.
- Sending of LF and CR characters were reversed and are sent in the usual order - CR first and followed by LF.