Sep 10, 2012 Severity: Recommended Eugeny Brychkov, RU

Storage subsystem v.3.0 Manual

This DN0003 design note is applicable to the storage subsystem microcode version 3.0 and above.
This DN0003 design note is licensed to GR8BIT Engineering Community, and put into public access as an evaluation of the quality of product and related documentation and support services for potential customers.

Contents

1. Introduction 3

2. Quick guide to upgrade to v.3.0 4

3. Definitions 5

4. Interesting facts about GR8BIT storage subsystem 7

5. Operating GR8BIT storage subsystem 9
5.1. Configuration of the storage subsystem 9
5.1.1. Operation mode 9
5.1.2. Boot message 10
5.1.3. Media configuration 11
5.1.4. Diagnostic outputs 11
5.2. Keyboard shortcuts 12
5.3. How much free memory do I have? 13
5.4. Starting GR8BIT with storage subsystem installed 14
5.5. Booting GR8BIT with storage subsystem installed 16
5.6. Configuring storage media to be used within GR8BIT 17
5.6.1. Configuring floppy drives and floppy disks 17
5.6.2. Configuring hard drives and logical volumes (LVOLs) 18

6. Storage subsystem design 22
6.1. Floppy disk subsystem design 22
6.2. Hard disk subsystem design 24
6.3. DOS2 mini-mapper design 25

7. Programming GR8BIT storage subsystem 32
7.1. Application API 32
7.2. System level API 32
7.2.1. ECHO command 33
7.2.2. INIHRD command 34
7.2.3. DSKIO commands 34
7.2.4. DSKCHG command 35
7.2.5. DSKFMT command 35
7.2.6. DRIVES command 36
7.2.7. VERSION command 36
7.2.8. SOFTWARE command 36
7.2.9. GR8STORM command 37
7.2.10. HDDSTAT command 37
7.2.11. INIDPB command 38
7.2.12. PROMPT command 38
7.2.13. GETDPB command 39
7.2.14. HDDID command 39
7.2.15. BUFACC commands 39
7.2.16. Completion status (DSKIO, DSKFMT and HDDID) 40
7.3. Organization of the dialogue between CPU and MC 41

Special thanks to **Raymond van der Meulen** (testing), **Dominique Chuderski** (testing), **Jan Wilmans** (DOS2 ROMs) and **Albert Beevendorp** (volume serial numbers) for help and support during development and testing of the GR8BIT storage subsystem.

1. Introduction

Purpose of this document: This document is intended to provide to GR8BIT Engineering Community members, as well as other interested users and hardware, software and firmware developers who want to use GR8BIT storage subsystem v.3.0 and above, and use its API to connect and operate their own devices.

Overview of features: GR8BIT Storage subsystem version 3.0 supports the following configurations:

- 3.5" floppy disk drives – up to 2;
- IDE hard disk drives – up to 2;
- Logical volumes (LVOLs) at one time – up to 2;
- MSX-DOS 1.0 mode (no FAT16 partition support);
- MSX-DOS 2.20 mode (optional **built-in** FAT16 partition support);
- Bootable from LVOL (hard disk) in DOS1 and DOS2 modes.

Storage subsystem now has *built-in* **GR8STORM™** *(GR8BIT storage manager)* utility for subsystem configuration and hard disk preparation for use/recovery. Standalone GR8HDD.COM utility was discontinued.

* **Note**: all references to FAT16 partitions assume usage of GR8BIT storage subsystem in MSX-DOS 2.20 mode. FAT16 partitions are not available for read/write in MSX-DOS 1.0 mode.

Compatibility considerations: There're several efforts made in order to maintain compatibility with MSX standard as well as to support MS-DOS/Windows (hereinafter referred as PC systems) formats:

- GR8BIT formatted diskettes are readable and writeable by PC systems; PC systems' formatted diskettes are readable and writeable by GR8BIT, but not bootable.
- GR8BIT hard disk drive volume structures are compliant with PC systems conventions. GR8BIT can configure, read and write FAT12 and FAT16 file systems, created by PC systems. GR8BIT hard disk portion of the driver code uses LBA fields within MBR to get logical volume (LVOL) configuration, and the converts LBA into C/H/S in order to support hard disks with no LBA capabilities (32-bit mathematics are embedded into microcontroller's microcode).

Supported storage media configurations:

- Diskettes formatted with 9 sectors per track (360K/720K), and diskettes formatted with 18 sectors per track (1.44MB). GR8BIT disk ROM does not follow *Media Descriptor* preset diskette configuration conventions, and it gets diskette configuration information from its boot sector. Storage subsystem v.3.0 automatically updates physical configuration of the diskette in its tables, and operates proper diskette information even if system does not invoke DSKCHG or GETDPB system calls (e.g. Metal Gear 2 bootable 720K diskette).
- HDDs operating in C/H/S mode; maximal configuration is 65535/16/63.
- Logical drives (up to 2) located on the HDDs with up to 65536 sectors in size (32MB) in 16-bit addressing mode and up to 4,194,304 sectors in size (2GBytes)

in 23-bit addressing mode. There's no limitation on the location of LVOL on the hard disk as all 4 bytes (32 bits) are used to address LVOL base. Both FAT12 and FAT16 partitions are supported, no additional driver (e.g. FAT16.COM) is required.

Limitations of functionality: Some functionality which may be desired is still missing (partially due to scarce disk driver ROM space):

- Sector interleaving during diskette format operation is not implemented;
- Floppy disk media bad block management is not implemented;
- In diskette format dialog, if you choose to write system onto diskette being formatted, all system files – DOS1 and DOS2 – will be written. If you do not need DOS2 system files, just delete them afterwards;
- Diskette format routine will always write DOS2-type boot sector.

2. Quick guide to upgrade to v.3.0

In order to get storage subsystem version 3.0 up and running, please perform the following steps:

1. Wire the floppy disk controller's DMA and IRQ lines, as shown and described in subchapter 6.1;
2. Build DOS2 memory mini-mapper into your GR8BIT, as shown in subchapter 6.3. If you'll not do it, you will able to operate system in DOS1 mode only if I/O board's jumper JP3 "ROMSEL" is open;
3. Apply two resistor packs of 10KOhm nominal to the HDD/FDC data lines as shown in subchapter 6.2 to pull them up during periods of signal transitions and inactivity. If you will not do it and have no HDDs connected to the storage controller, it may incorrectly think that there's hard disk attached, and wait 8 seconds for each to become ready (you lost 16 seconds each on boot);
4. Flash ROM microcode version 3.0 to the GR8BIT storage subsystem EEPROM chip (IC3, W24C512, located on the I/O board), and flash PIC18F46K22 microcontroller's EEPROM with microcontroller's firmware version 3.0.

We recommend partitioning first (master) hard disk in the following way:

- Partition 0 – FAT12 to have 3 sectors per FAT copy (first tunable parameter), with 16 sectors per cluster (second parameter = 4). You will have LVOL0 of 8Mbytes in size;
- Partition 1 – FAT12 to have 12 sectors per FAT copy (max), and 16 sectors per cluster (max). This LVOL1 will have size of 32Mbytes;
- Partitions 2 and 3 – of your choice, either FAT12/FAT16, of any configuration.

The rationale for such recommendation is to have LVOL0 with minimal number of FAT sectors with DOS1-supported OS to save high memory space when you boot into DOS1 mode, but at the same time have fully functional and bootable LVOL located on the hard disk. At any time you can use GR8STORM to deactivate LVOL0, and have LVOL1 of 32Mbytes of size appear in the GR8BIT drive map as first logical device located on the hard disk.

3. Definitions

B

Built-in means that you do not need any additional software or hardware in order to get declared functionality

Boot sector is the 512-byte chunk of data and code at the beginning of logical area of the media. It contains vital information like media configuration, and boot loader. Floppy disk can have only one logical area (at logical sector 0, or CHS=0/0/1), hard drive may have several logical areas (partitions), and each partition is addressed relatively to the start of the hard disk having boot sector at its beginning

C

Compatible format is a format of floppy disk with 3 sectors per FAT copy and having usable space of 1.44Mbytes. Floppy disks having compatible format can be read/written by GR8BIT and PC. Format is called "Compatible" because it is accepted by GR8BIT storage subsystem in compatible mode, and such disk's FAT copy in memory in DOS1 mode consumes only 1536 bytes. It differs with Legacy format in the total disk usable space size

Compatible mode is the state of the storage subsystem when it can accept only floppy disks in Compatible and Legacy formats

Cylinder/Head/Sector (CHS) is the method of addressing floppy and hard disk media. Contemporary floppy media are divided into circular cylinders (or tracks), and may have one of two heads. Each head can read track it is positioned on. Each track is divided into sectors (small arcs of the circular track). This method may be considered as obsolete for hard disk drives, however GR8BIT storage subsystem uses it to access disk hardware to support older drive models

D

Driver is a program which runs either in RAM or ROM and acts as an interface between hardware and application software. It translates high level commands of application software to the low level commands recognizable by hardware

Drive letter is an storage subsystem's identifier for storage device, e.g. A:, B: , C:

DOS1 is a first generation of the kernels. Its maximal size, together with driver, is 16Kbytes, it has limited memory management (no use of mapper) keeping all its data structures in the high memory area

DOS2 is a second generation of the kernels. Its maximal size, together with driver, is 64Kbytes, and it uses mini-mapper to switch between 16Kbyte pages. It has support for memory mapper management, and stores most its data into the upper pages of the RAM

F

File allocation table (FAT) is a structure which allows storage subsystem to find data written for specific file on the media. Floppy disks and LVOLs usually keep 2 FAT copies

to be able to restore data in case of one copy get damaged. If FAT becomes invalid, part of the data, or whole data read from the disk, becomes invalid, and restoration may require expert knowledge or tools

Floppy disk controller (FDC) is an entity (chip or emulating device) which controls floppy disk drive(s) and decides which information is written on the diskette and at which location. It usually connects to the FDD using 34-wire ribbon cable

Floppy disk drive (FDD) is a device where you insert floppy disks. This device is responsible for reading and writing media, and detecting its presence

K

Kernel is a core of the storage subsystem BIOS (basic I/O system), which serves as intermediary between application software and custom hardware drivers. While drivers are responsible for ensuring information storage and access to hardware, kernel is responsible for keeping data stored in the structured format so that it can be easily restored if needed

L

Legacy format is a format of floppy disk with 3 sectors per FAT copy and having usable space of 720Kbytes. Such format of disks is historically used in MSX-compatible computers having WD1793-compatible FDC

Logical block addressing (LBA) is a method of addressing of contiguous space of the media or volume, with logical block 0 being located at the beginning of media, and block 1 following logical block 0 and so on

Logical drive is a user-interface device usually addressed by the drive letter

Logical volume is a contiguous area on the media (usually hard disk or disk array), which can be addressed by LBA method

M

MSX-DOS is an operating system for MSX compatible computers. It is very similar to MS-DOS, but tailored for MSX computers.

Master boot record (MBR) is the first sector (sector 0 in LBA terms) of the hard disk, which contains information about LVOLs (partitions) created on the hard disk. In GR8BIT it does not contain boot code, but contains LVOL recovery information. MBR is not directly accessible by the GR8BIT's CPU

S

Standard format is a format using a maximal configuration for floppy disk. If you format floppy disk on PC, it will by default receive Standard format of 1.44Mbytes in size and 9 sectors per FAT copy. This format is compatible with PC

Standard mode is the state of the storage subsystem when it can accept floppy disks in all the formats: Standard, Compatible and Legacy

4. Interesting facts about GR8BIT storage subsystem

Here're quick facts which may help you understanding how system behaves, and why it behaves in certain way. Understanding of the matters will allow you to perform initial troubleshooting and investigation.

- Most of the functionality is residing in the PIC18F46K22 microcontroller, with storage system ROM using this functionality;
- You still have about 22Kbytes of the microcontroller's flash ROM free for implementation of your ideas and drivers for new devices. There's also 1Kbyte free in the storage system ROM in DOS1 kernel, and 670 bytes free in page 0 (9Kbytes in page 1) in DOS2 kernel;
- Floppy disk configuration (including physical configuration like sector per track and number of heads) is sourced from the boot sectors of the floppy drives and LVOLs. It means you can use disks of any configuration. However if boot sector gets corrupt you will need to restore it before using the disk successfully;
- Diskette physical configuration (sectors per track and number of heads) is updated on any read or write to the floppy disk's boot sector. Actuality of this information does not depend on the storage ROM invoking DSKCHG command (which senses change of the diskette and automatically updates DPB);
- Volume serial numbers are calculated basing on the values of two timers within microcontroller, using sophisticated transformation algorithm so that serial numbers not appear linear or sequential;
- Microcontroller is responsible for turning floppy disk drives' motors off, and it does that through high priority interrupt routine. If you notice that floppy disk drive's motor does not turn off within 2 seconds, then microcontroller has hung or doing something with its interrupts disabled;
- Microcontroller keeps most of the storage subsystem variables (a workset), and you can view them in hex through GR8STORM utility's *Diagnostic outputs* menu;
- GR8BIT storage ROM operates with logical identifiers of the drives, translation to the physical identifiers is performed by microcontroller;
- LVOL drive parameter blocks (DPB) are updated only once, on the subsystem initialization. While it is practically possible to have hot-pluggable IDE drives in the GR8BIT system, you can only initialize its hardware using "Enumerate" function of GR8STORM, but not update its logical configuration within GR8BIT storage ROM;
- If in Compatible floppy subsystem mode you will try reading floppy disk with more than 3 sectors per FAT copy, storage subsystem will mark diskette inserted as invalid and display "Disk I/O error" for all operations for the diskette (without even accessing it) until you change the diskette with the one having 3 sectors per FAT;
- If GR8BIT will try reading data to the memory space occupied by the current CPU stack, it will flash screen border and return error condition;
- You can create sophisticated configuration of the storage subsystem of your GR8BIT using "Operation mode" menu of the GR8STORM;

- It is possible to invoke GR8STORM from the BASIC prompt by “call gr8storm”, but also at the end of storage subsystem initialization during boot up by holding CTRL-GRAPH (CTRL-ALT keys on PS/2 keyboard);
- Holding “1” key at the *beginning* of the GR8BIT storage subsystem initialization after reset or power up will force it to initialize in DOS1 mode;
- GR8STORM is a system application built into the microcontroller, and when you invoke it, you have direct dialogue with microcontroller rather than with GR8BIT system;
- If GR8BIT is not able boot from the floppy disk properly, it will prompt to insert another diskette to boot from. If it is not able to boot from LVOL, it will just skip this volume;
- Microcontroller keeps LVOL structure for two LVOLs, and can perform 32-bit and 16-bit multiplication and division;
- FAT16-configured volumes, by default, receive dummy DOS1 DPB. Only DOS2 portion of ROM with FAT16 patch can operate FAT16 volumes using extended *drive table structure*;
- Floppy disk controller works using DMA mechanism, while hard disk drive interface works in PIO mode;
- All data transfers between microcontroller and GR8BIT system are CRC protected. Microcontroller can read or write maximum 7 sectors of 512 bytes size at once;
- MSX-DOS 1.03 and MSX-DOS 2.20 file images are embedded into the microcontroller, and are written to respective media when choosing option 4 (write system files) in FORMAT command dialogue. These images are not bootable from the microcontroller;
- FAT16 partition is identified by the “FAT16” signature in its boot sector. Its size in sectors is located in the “Big-Size” area at the 0x30 offset to the base of boot sector;
- If you use COMMAND2.COM version 2.20, you will have wrong volume size reported if it is larger than 32Mbytes (happens for FAT16 volumes). This issue is solved in patched version of COMMAND2.COM version 2.44 by T.N.I. http://www.tni.nl/products/command2.html ;
- When using DOS1 kernel mode driver explicitly marks all the invalid volumes, including FAT16 volumes, with special media descriptor 0C3h, and such volume becomes inaccessible. This mechanism is implemented in order for system not to damage device’s structures and data in DOS1 mode.

5. Operating GR8BIT storage subsystem

You can configure GR8BIT storage subsystem in smart ways which fit your immediate requirements. The configuration information is stored within microcontroller's EEPROM and is kept when you switch off the system.

5.1. Configuration of the storage subsystem

Configuration is performed using built-in GR8STORM™ utility. During any prompt for your input you may press CTRL-C or CRTL-BREAK keys to cancel the operation, get out to the higher level menu, or exit the utility. Please note that after specific changes to the configuration of LVOLs you are prompted to reboot the system. Structure of GR8STORM's menus is presented in table 5.1.

Table 5.1. Structure of GR8STORM's menus

Main menu	Submenus
1. Operation mode	Floppy subsystem
	HDD subsystem
	Device order
	FDD mode
	Phantom floppy
	Write to EEPROM
2. Boot message	Clear message
	Set new message
3. Media configuration	Reset and enumerate hardware
	Update floppy boot sector to DOS2
	Partition hard disk
	Change partition activation status
	Initialize/recover Logical Volume (LVOL)
4. Diagnostic outputs	Display MC's workset
	Display MC's buffer contents
	Display FDD0's floppy disk's boot sector
	Display HDD0's MBR and LVOLs' boot sectors
	Display HDD1's MBR and LVOLs' boot sectors

5.1.1. Operation mode

Operation mode defines the way how storage subsystem functions. It has several settings, which can be combined together to create a set of handy configurations for various applications.

- Floppy subsystem (*Enable/Disable*): enables or disables floppy subsystem. If disabled, floppy disk drives are not initialized, and do not appear in the device map;

- HDD subsystem (*Enable/Disable*): enables or disables hard disk drive subsystem. If disabled, hard disks are not queried and/or initialized, and LVOLs contained on them do not appear in the device map;
- Device order (*Floppy first/HDD first*): will change order of initialization of subsystems and order of appearance of related devices in the device map;
- FDD mode (*Standard/Compatible*): enabling Standard mode will allow storage subsystem to access Standard-formatted floppy diskettes with 9 sectors per FAT copy. In Compatible mode FAT size is limited to 3 sectors;
- Phantom floppy (*Enable/Disable*): if disabled and you have connected only one physical floppy disk drive, phantom floppy device will not appear in the device map. In this case you will not be able to copy from floppy to floppy, but it will free up 1536 or 4608 Bytes of user memory in DOS1 mode;
- The last option Write to EEPROM writes settings to the microcontroller's EEPROM. In order for changes to take effect you will need to reboot the system.

Examples of applications: operating GR8BIT in DOS1 mode and lacking of user memory, you can disable phantom floppy. To boot from LVOL in DOS1 mode prioritize HDD first, and then drive A: will appear to be first LVOL configured. Disable floppy subsystem if you do not use it.

5.1.2. Boot message

You may have message of maximum 24 characters displayed at the boot of the GR8BIT during initialization of the storage subsystem (fig. 5.1). This message is stored in microcontroller's EEPROM, and is not lost when you power system off. We advise not to put any text there for which you would feel uncomfortable or judged inappropriately.

Figure 5.1. Boot message appearance during storage subsystem initialization

5.1.3. Media configuration

This section of GR8STORM allows you to re-initialize storage subsystem, update floppy boot sector and configure hard disks for use.

- Reset and enumerate hardware invokes hardware initialization routine according to the settings set up by the “Operation mode” section. It is helpful when power-up sequence does not complete properly, and during troubleshooting steps you need to have storage subsystem and its workset to be re-initialized;
- Update floppy boot sector to DOS2 reads floppy’s boot sector, and updates its boot loader to DOS2 version. Media configuration is preserved;
- Choosing Partition hard disk starts the process of the hard disk partitioning and initialization. It initializes whole the media, to re-initialize LVOL please use “Initialize/recover” submenu.
- Change partition activation status changes flag of the partition in the master boot record (MBR). If flag for specific partition is set, it will be configured as LVOL during storage subsystem initialization. Those partitions which have this flag reset will be skipped;
- Initialize/recover Logical Volume (LVOL) will display two submenus: initialize LVOL – also known as *quick format*, and recover LVOL structures. Former uses LVOL recovery information contained in MBR in order to reconstruct and restore all the LVOL’s control structures (original boot sector, clear FAT, and clear directory), but latter will just fix boot sector, and signature of the first FAT (without clearing FATs and directory).

5.1.4. Diagnostic outputs

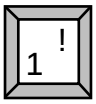
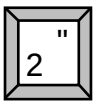
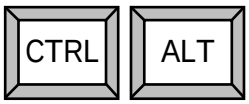
If you have issues with storage subsystem, or want to view control data of the microcontroller and storage devices, you will use diagnostic outputs for troubleshooting.

- Display MC’s workset displays internal microcontroller set of register, and it used to troubleshoot the operation of microcontroller and analysis of its state;
- Display MC’s buffer contents displays information present in the microcontroller’s I/O buffer. Analyzing contents of the buffer you may get an idea what was read or written last, and what was actually read or written;
- Display FDD0's floppy disk's boot sector reads boot sector of the floppy and displays it, and is very useful in troubleshooting diskette boot problems. If you notice that boot sector’s loader is corrupt, you can use *Update floppy boot sector to DOS2* menu to recover it;
- Display HDD0's MBR and LVOLs' boot sectors displays master boot record (MBR) of the hard drive 0 (master IDE), and boot sectors of all the LVOLs. Analyzing this information you may get a clear picture on how hard disk space is partitioned, and where boot problems may happen.
- Display HDD1's MBR and LVOLs' boot sectors displays the same as previous menu item, but for hard drive 1 (slave IDE).

5.2. Keyboard shortcuts

There're several very useful shortcuts available at the beginning and at the end of the GR8BIT storage subsystem initialization. Please refer to the table 5.2 for details. Remember that you should allow GR8BIT PS/2 keyboard controller to initialize within 4 seconds after system power on, not pressing any keys during this period of time.

Table 5.2. Keyboard shortcuts

Key (keys)	Description
	Disk-ROM will not install. Works in DOS1 mode only
	Forces storage subsystem to initialize in DOS1 mode. FAT16 volumes are not available. Devices' structures (DPBs and buffers) are located in the upper memory
	Cancels installation of the FAT16 patch. Works in DOS2 mode only
	Disables floppy disk subsystem
	Disables hard disk subsystem
	Forces storage subsystem to configure in Compatible mode
	**Starts GR8STORM™ built-in utility

**This key/key combination is sampled *at the end of* the subsystem initialization

5.3. How much free memory do I have?

Answer to this question is very important because CPU visible memory is limited even with GR8BIT having mapped 1MB of main SRAM. Ability to run programs and games is dependent on how much memory is available, and where system's control structures are located in the memory.

If you have no storage subsystem installed or configured in the system (e.g. if you are in DOS1 and no media was detected), then you will have about **28.8 Kbytes** free memory.

If you start system in DOS2 mode (not necessarily boot into the MSXDOS), DOS2 kernel will have advanced memory mapping capabilities, and disk buffers, control structures and other stuff will be located in the last pages of the *mapper* RAM. It will not consume much of valuable space in the high visible memory space (in bank 3, from 0C000h...0FFFFh), and you always will have **25 Kbytes** user space free in BASIC and **57 Kbytes** free with operating system loaded, regardless of number of logical drives and their sizes. However such an advanced configuration may not be supported by earlier software designed for DOS1 with no memory management, software performing its own memory management. Such software may not work properly in DOS2 environment.

If you start system in DOS1 mode (by booting from DOS1-type floppy disk or holding Backspace key on storage subsystem initialization), you will get DOS1 kernel with no memory management, and information about drives and buffers will be allocated in the memory bank 3 and user available memory will be automatically reduced by the amount allocated. Consider configurations described in table 5.3.

Table 5.3. Examples of FDD configurations and memory available

Configuration	Bytes used
1 FDD Phantom = disabled Compatible mode	In Compatible mode space for 3 FAT sectors per logical drive is allocated, and you will get **1536** bytes (512*3) less free space
2 FDDs Phantom = n/a Compatible mode	1536 bytes are allocated twice, for every floppy drive's logical device, so minus **3072** bytes
1 FDD Phantom = *enabled* Compatible mode	Phantom drive is a second logical floppy drive using the same physical floppy drive. Two logical drives = minus **3072** bytes
2 FDDs Phantom = n/a Standard mode	In Standard mode space for 9 FAT sectors per logical drive is allocated (4608 bytes). With 2 logical drives in Standard mode you will get **9216** bytes user free space less

For every LVOL configured system will allocate space equal to 512 bytes multiplies by the number of sectors in file allocation table (FAT). See table 5.4.

Table 5.4. Examples of LVOL configurations and memory available

Configuration	Bytes used
FAT16 volume	Whatever number of FAT sectors it has, it will be allocated dummy number of sectors equal to 3, and will cause minus **1536** bytes. FAT16 volumes are not supported by DOS1 kernel
FAT12 volume 3 sectors per FAT	3 sectors by 512 bytes = **1536** bytes less
FAT12 volume 12 sectors per FAT	12 sectors by 512 bytes = **6144** bytes less

***** Please note that in DOS1 mode if there're no devices found to configure, floppy and LVOLs, storage subsystem ROM will not be installed.

5.4. Starting GR8BIT with storage subsystem installed

Before GR8BIT will get to operating system (OS) load procedure, or to the BASIC if OS in not present on the first media system finds, disk ROM will perform initialization of the storage subsystem and enumeration of the logical volumes (LVOLs) located on the hard disk drives.

Figure 5.2. Storage subsystem initialization screen

Figure 5.2 shows messages of the GR8BIT storage subsystem initialization screen. Let's consider every part, as each provides you with important information:

1. Copyright/Copyleft notice. For DOS2 kernel it will credit OKEI for FAT16 partition support code;
2. Version of the GR8BIT driver and kernel (in GR8BIT storage ROM flashed in IC3 on the I/O board). If you will hold Backspace before initialization starts, you will get DOS1 kernel;
3. Version of the microcode in the microcontroller. You should ensure that it is compatible with ROM version. In case MC is not installed, hangs or malfunctions, the "none" message will be displayed;
4. Identification of the operation mode of the microcontroller. It is bitmap byte:

Bit #	Function
7	0 = Disable Hard disk drive subsystem
6	0 = Disable Floppy disk drive subsystem
2	1 = floppy drive devices go first
1	1 = Phantom floppy disk device is disabled
0	1 = Floppy disk drive subsystem is in Compatible mode

5. Initialization status of the floppy disk subsystem. It displays if drives are installed (recognized), and number of physical and logical drives associated with floppy disks. If you will have phantom drive option enabled, or have 2 physical floppy disk drives recognized, you will have 2 logical floppy drives;
6. List of hard drives detected and partition entries found on them. Letter "A" at the beginning of partition line designates if the partition is market for configuration. If trailing message is not "OK", LVOL will not be configured. If subsystem senses that there's a hard disk, but cannot get its identification information (e.g. disk is faulty, unsuccessfully completed master/slave diagnostics, or incorrect disk master/slave jumper configuration), ROM will count down 8 seconds before timing out on the hard disk device. Note that on the fig. 5.2 you see disk CHS configuration of 65535/16/63, which means CHS-addressable space of 8Gbytes only, but the NTFS logical volume is listed as 19Gbytes. This is the valid case because Windows OS host created partition not using and limiting itself by CHS addressing, but using LBA addressing. GR8BIT operates in CHS mode, thus has disk size limit of 8Gbytes (4 FAT16 partitions by 2Gbytes each);
7. Number of hard disk drives identified and LVOLs configured. Note that having a LVOL configured does not mean you will have full access to data on it (it should have correct data structured and be of supported OS type);
8. Any diagnostic messages from LVOL configuration microcode. Error detected during LVOL DPB (drive parameter block) configuration are:
    - *DIRENT truncated:* number of directory entries for FAT12 exceeds allowed 254: directory entries will be truncated to 254 in DPB;
    - *DPB Error:* boot record of the volume is invalid, thus DPB cannot be constructed out of it. LVOL will be configured using dummy DPB.

5.5. Booting GR8BIT with storage subsystem installed

In the DOS1 mode subsystem will only attempt to boot from first drive configured in the system drive A:. If you have floppy disk system configured first, it will boot from floppy drive 0, if hard disk system first, then from first LVOL is configured from hard disk 0.

In DOS2 mode GR8BIT will try every storage device available (max 8) to see if it can boot from it, thus if you have configured floppies first, with single floppy and phantom drive disabled, it will first try booting from floppy drive (A:), and if unsuccessful will boot from first LVOL (B:).

The following prerequisite should be met to boot to the MSX-DOS operating system from the specific device:

1. Device is operational and ready for system read requests – it is attached to the controller, passes its internal diagnostics, and is configured properly;
2. For hard disks partition table (MBR) is valid – please use GR8STORM utility to initialize the MBR;
3. Boot sector contains valid media configuration – valid boot sector is written during format operation (diskettes: *format* in OS and *call format* in BASIC; GR8STORM utility for hard disk volumes during partitioning or volume recovery);
4. There're operating system files on the media – at least MSXDOS.SYS and COMMAND.COM (you can use FORMAT routine's option 4 or copy them manually).

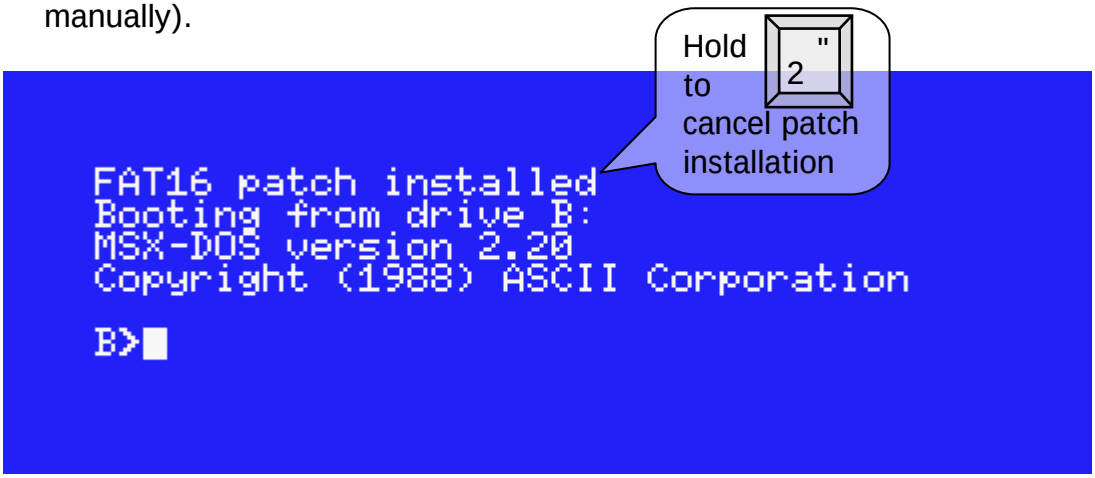


Figure 5.3. Operating system's Boot messages

When GR8BIT starts booting, it automatically applies FAT16 patch by OKEI (fig. 5.3). If, for some reason, you do not want FAT16 patch to be applied, hold "2" key at the start of the boot process.

If storage subsystem detects no floppy drives, first LVOL becomes drive 0 (A:) and can be booted from even in DOS1 mode. Note that DOS1 kernel requires at least one logical drive to be configured. If you detach all storage devices from controller, or disable floppy and hard disk subsystems via operation mode menu, system will not install DOS1 kernel at all.

5.6. Configuring storage media to be used within GR8BIT

Important feature of the GR8BIT storage subsystem version 3.0 is that it has original versions of MSX-DOS and MSX-DOS2 flashed into its microcontroller. You may have MSX-DOS operating system without having any media containing required files. Use FORMAT built-in utility from either MSX-BASIC or from MSX-DOS. ***Important***: FORMAT utility was relocated to the end of the page 2 of the DOS2 storage ROM, and will work with built-in FAT16 patch installed. However if you will run external FAT16.COM application with /R key (uninstall) and then install it back, FORMAT utility will stop working until you remove FAT16 patch using this external FAT16.COM application. It happens because FAT16.COM disables FORMAT utility (it does not know that GR8BIT storage subsystem has FORMAT utility code relocated to the safe place in its ROM).

5.6.1. Configuring floppy drives and floppy disks

There's nothing about configuration of the floppy drives other than just attaching them to the respective connector on the floppy cable. Ensure that you use regular FDDs you used to use with your PC with DC (disk change) option (pre-) set. Disk drives which are configured with RDY option will not work.

Configuring diskettes involves formatting them – laying out sector information in the raw media space, and filling these sectors with proper information (boot sector, FAT, directory).

There're two options for formatting diskettes:

1. Format floppy disk on your PC, using 1.44M size (for HD diskette only, PC command is "format /t:80/n:18", will create *Standard* formatted diskette in GR8BIT terminology) or 720K size (for DD and HD diskette, PC command is "format /t:80/n:9", will create *Legacy* formatted diskette in GR8BIT terminology). You can't create diskettes in *Compatible* format with PC's "format" command. Disks created this way will be readable and writeable on GR8BIT, but will *not* be bootable due to PC boot code in the boot sector;
2. Format floppy disk using GR8BIT's MSX-DOS ***format*** command or BASIC's ***call format*** command. Both invoke the same formatting microcode residing in the ROM and microcontroller. Usage of these commands in scripts/applications is not recommended.
   When you invoke the ROM-embedded formatting command (fig. 5.4), you will select logical drive (and media in it) to perform operation on (1).
   You will see options available (2), and dialogue string. If you select option other than 4 (Write system files) for LVOL, command will abort with message "Use GR8STORM to initialize LVOL". Choosing option 4 will load DOS1 and DOS2 system files onto the selected media.
   You can abort the command any time until it starts formatting by pressing CTRL-STOP or CTRL-C key combination.
3. You can load image onto the diskette. Note that it can be done only after diskette was properly formatted (e.g. physically prepared for user data load).

If you have formatted floppy disk on the PC, please use GR8STORM™ *Update floppy boot sector to DOS2* submenu instead of FIXDISK.COM to make it bootable by the GR8BIT (FIXDISK will return *Invalid MSX-DOS call*).

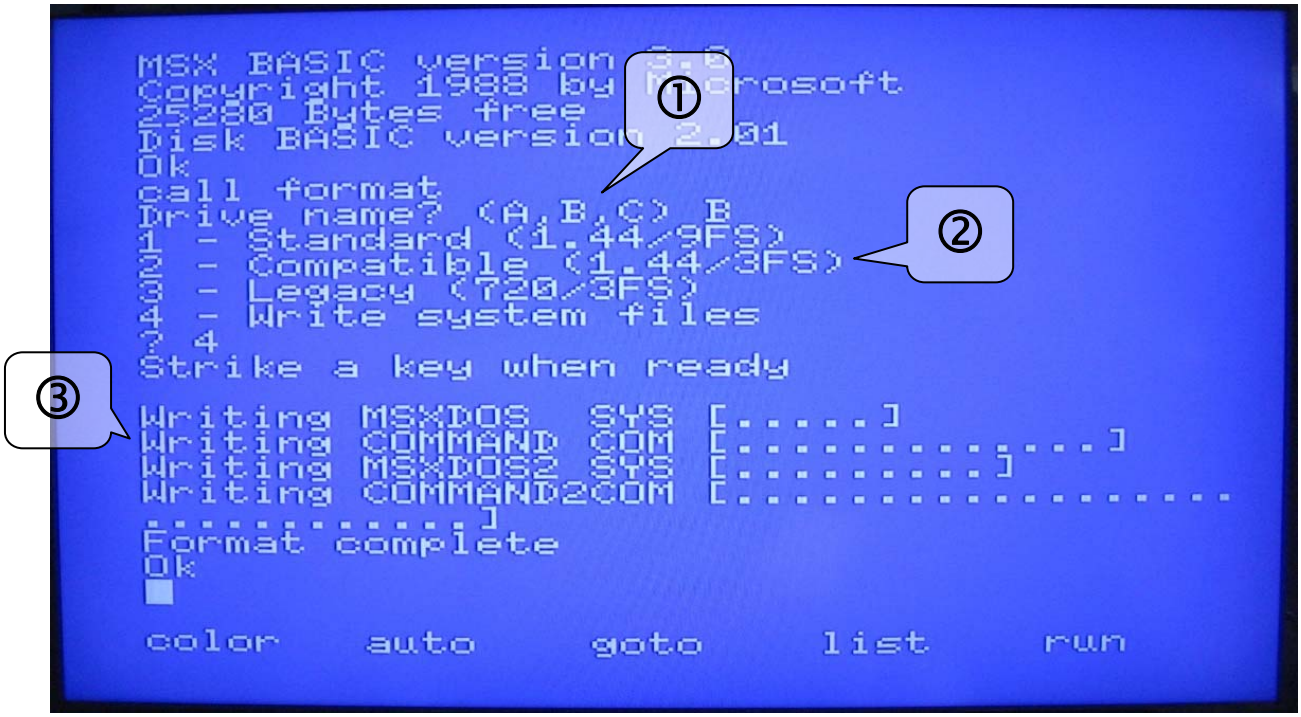


Figure 5.4. Diskette formatting dialogue

5.6.2. Configuring hard drives and logical volumes (LVOLs)

You will use GR8STORM built-in utility to partition the hard disk, initialize or recover volume structure on its LVOLs. The utility works in 80-column screen mode in order to provide the best layout of the information on the screen.

Invoke GR8STORM using "call gr8storm" BASIC command or holding CTRL+GRAPH during storage subsystem initialization. Fig. 5.5 shows start of the dialogue; first ensure that hard disk subsystem is enabled (1) and then follow to the "Partition hard disk" menu (2).

You select hard disk drive number, and have its partition listed (1 at fig. 5.6). Then you confirm that you want to proceed further, and get into the partition configuration menu. You see disk size listed (2), and note that it is maximal supported size in CHS mode and not actual hard disk drive size which can be larger. First partition will always be FAT12 (3).

There're only two tunable options:

- Sectors per FAT copy (SpF) identify how many clusters LVOL may have. Number of clusters is calculated as follows:
  - FAT12: Clusters = SpF*512*2/3, maximum is 12*512*2/3=4096 (1000h) clusters.
  - FAT16: Clusters = SpF*512/2, maximum is 256*512/2=65536 (10000h) clusters.

- In terms of media format, cluster numbers 0, 1, 0FF8..0FFF for FAT12 and 0FFF8...FFFF for FAT16 have special purpose, and are not used for definition of data allocation.

- Sectors per Cluster (SpC) identify how many 512-byte sectors are allocated for each cluster. More SpC means larger LVOL, but also means more usable space may be lost on remainders of the files or small files.

After the calculation the LVOL you are going to create will have the size equal to Clusters $*$ 2^{SpC} $*$ 512 bytes and maximal value for FAT12 volumes equals to 4096 $*$ 2^4 $*$ 512 = 32Mbytes, and for FAT16 volumes equals 65536 $*$ 2^6 $*$ 512 = 2Gbytes. You will see this calculated value in the “Size =” section.

You can play with the settings to create different sizes of the volumes, and create multiple, up to 4 volumes choosing menu option “Accept and next entry”. LVOLs 1, 2 and 3 created on the hard disk may be either of FAT12 or FAT16 type.

When you filled up all 4 entries or chosen “Accept and create”, you are again asked for prompt if you really want to proceed. If you choose uppercase “Y”, it will rewrite all the control structures on the disk and previous structures will be lost.

LVOLs need not to be “formatted” after you finish their configuration with GR8STORM. You can view the structures tool has created for you by going to *Diagnostic outputs* menu and choosing option 4 *Display HDD0's MBR and LVOLs' boot sectors* (fig. 5.7 and 5.8).

After reboot you may copy system files and application to the newly configured LVOL, and use it as any regular storage device addressing it by the respective drive letter.

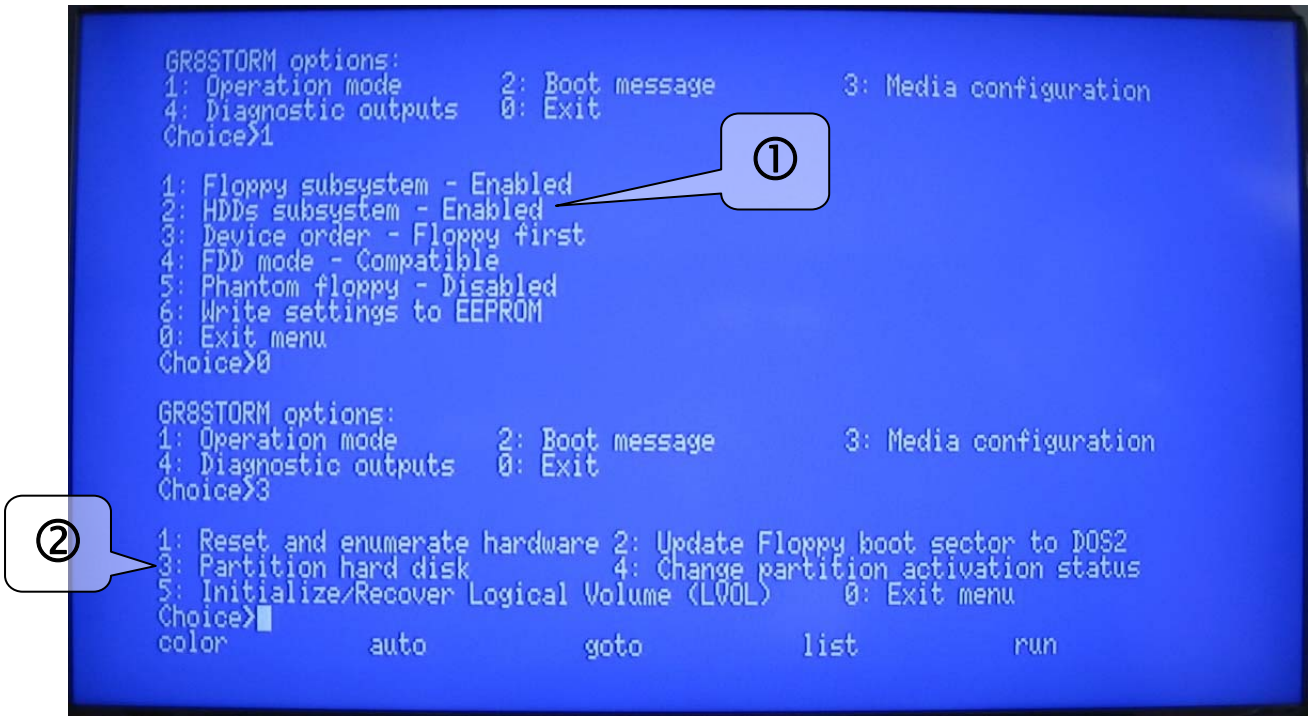


Figure 5.5. Start of GR8STORM dialogue

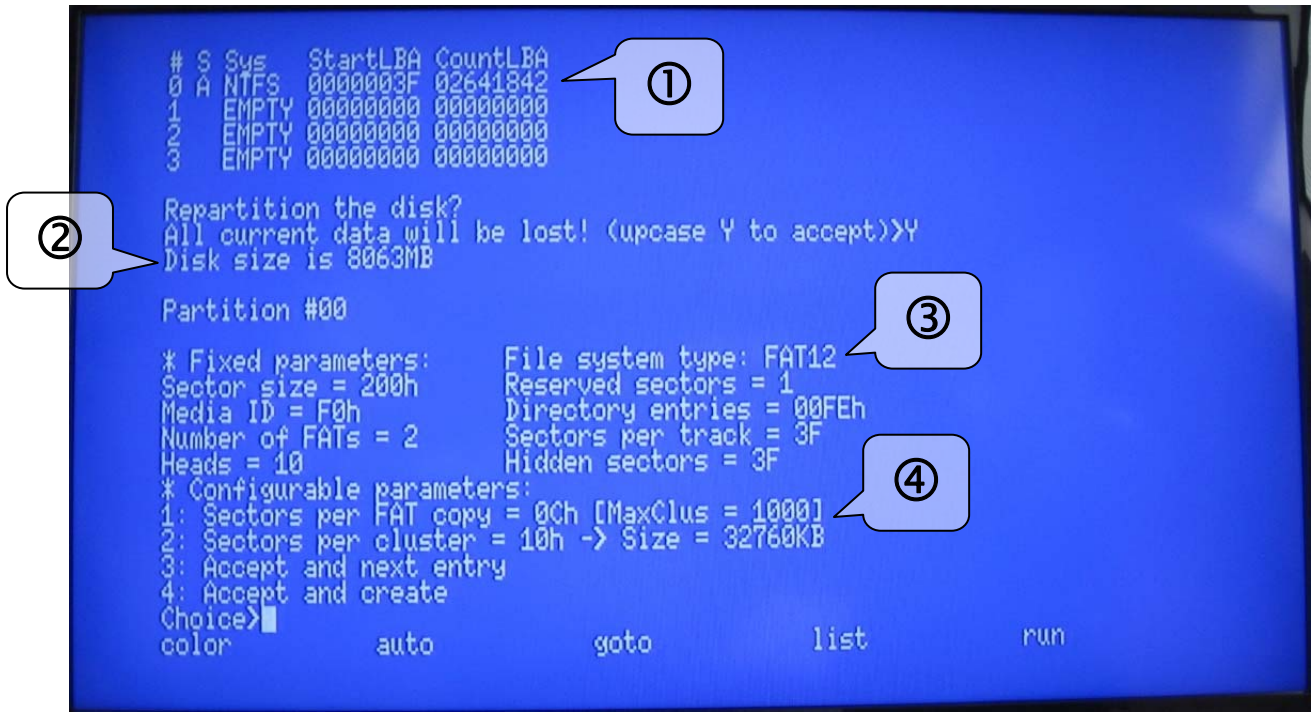


Figure 5.6. Further option selection dialogue of the GR8HDD

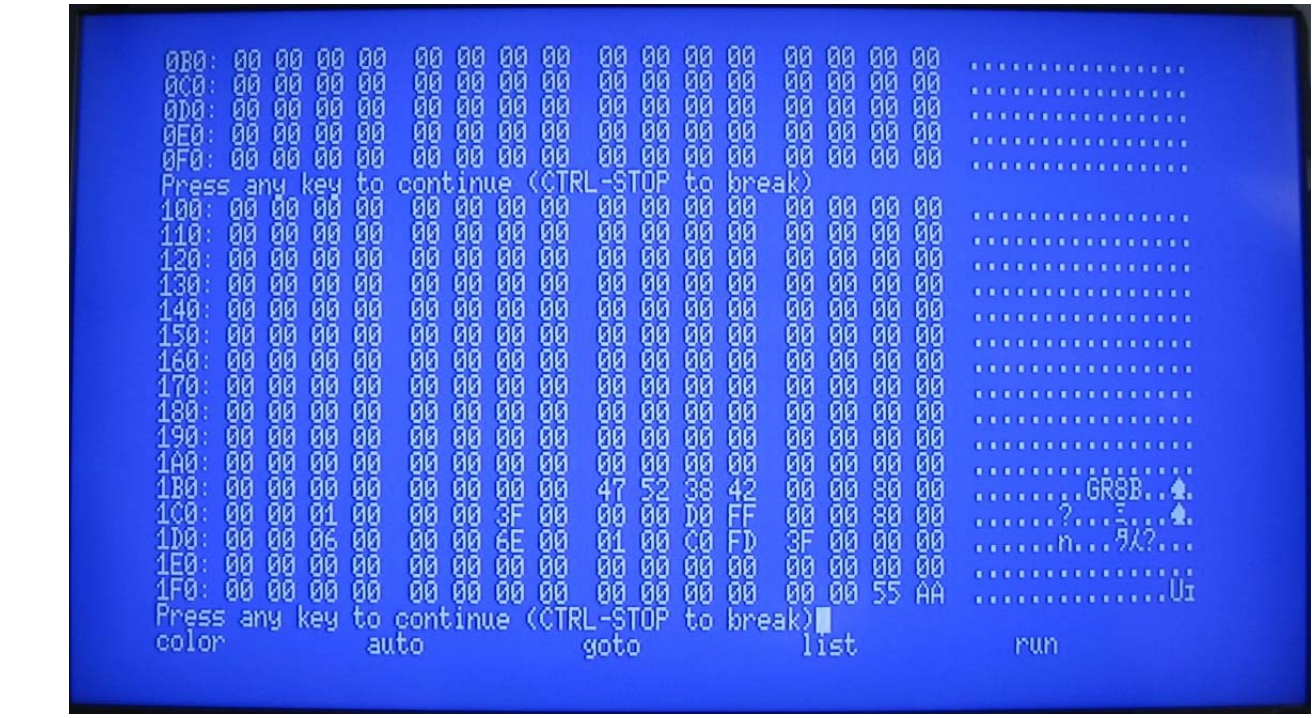


Figure 5.7. Master boot record (MBR) dump (useful data from 1BE..1DD)

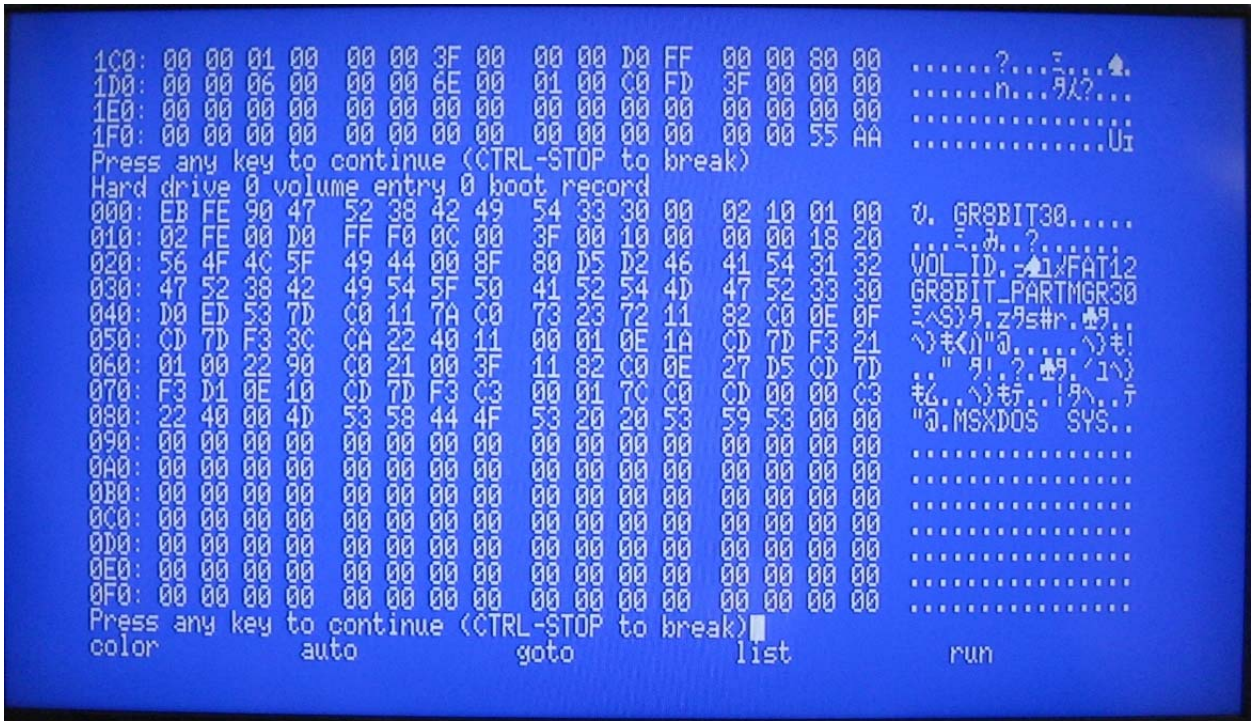


Figure 5.8. LVOL 0 boot sector dump

Limitations: GR8STORM (a) only writes LBAs into the MBR LVOL configuration, and does not write corresponding C/H/S values of start and end of partition; (b) it does not write PC bootstrap code into MBR (PC will not be able to boot from this hard drive), however hard drive is expected to be readable if attached to the PC's IDE controller.

6. Storage subsystem design

Subsystem is based on several components (fig. 6.1), which provide specific API for developers to extend its functionality. Microcontroller with multi-stage architecture was introduced in order to have reliable access to the media, and have unified entry point to any attached storage device. Without MC GR8BIT, and any other MSX-compatible computer, is unable to reliably control floppy disk controller data flow at 500KBit/s speed due to insufficient CPU speed at 3.58MHz system clock.

If you want microcontroller to have more functionality for floppy disk controller, you develop modifications and add-ons to the microcontroller microcode. As an example, it is not really required to have IDE disk drive connected to the 40-pin connector – you may design and build adapter to connect any device and alter microcontroller functionality appropriately.

If you want to change API between GR8BIT CPU and microcontroller, you will need first have this API defined from MC side, and then implement CPU-side routines modifying GR8BIT storage ROM.

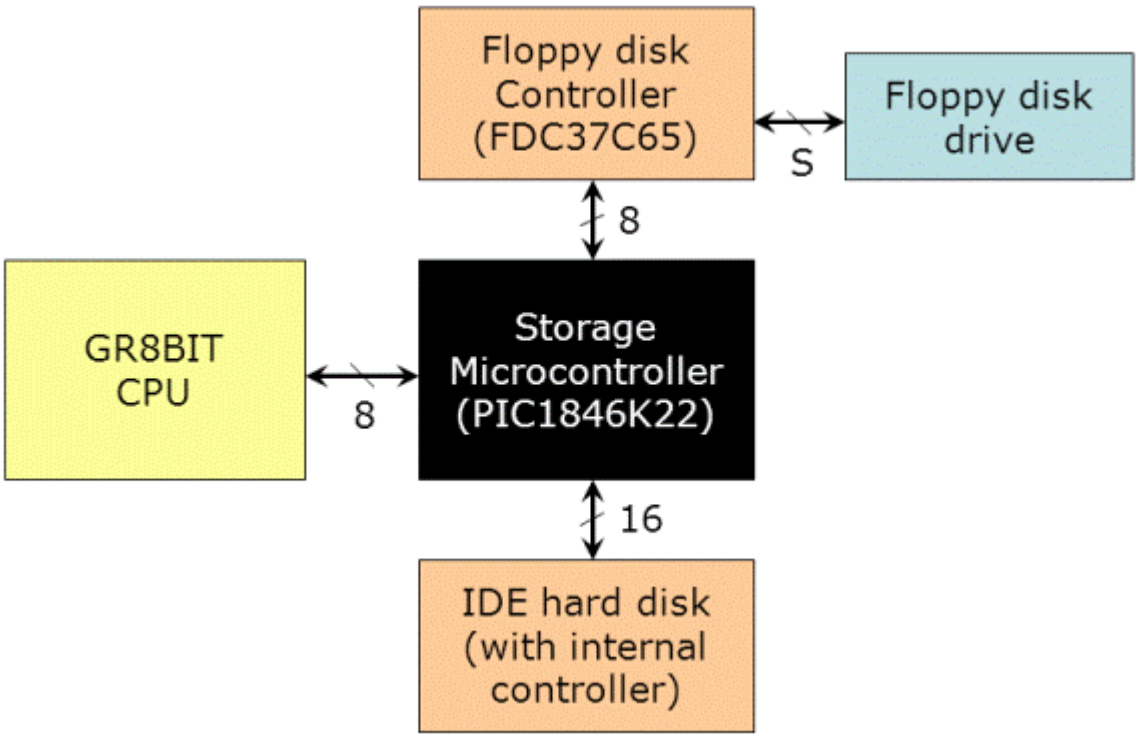


Figure 6.1. Architecture of the GR8BIT storage subsystem

6.1. Floppy disk subsystem design

Floppy disk controller was initially designed to work in PIO mode; unfortunately all the attempts to make it work reliable in PIO mode did not succeed. We decided to implement DMA transfers, and this way of interfacing succeeded. In order for your GR8BIT storage hardware to support DMA floppy disk controller interface and current

release of microcode, you will need to add 4 (four) air-wires connecting IRQ, DMA, $\overline{\text{DACK}}$ and TC pins of FDC to the specified pins of port B of MC (fig. 6.2 and 6.3).

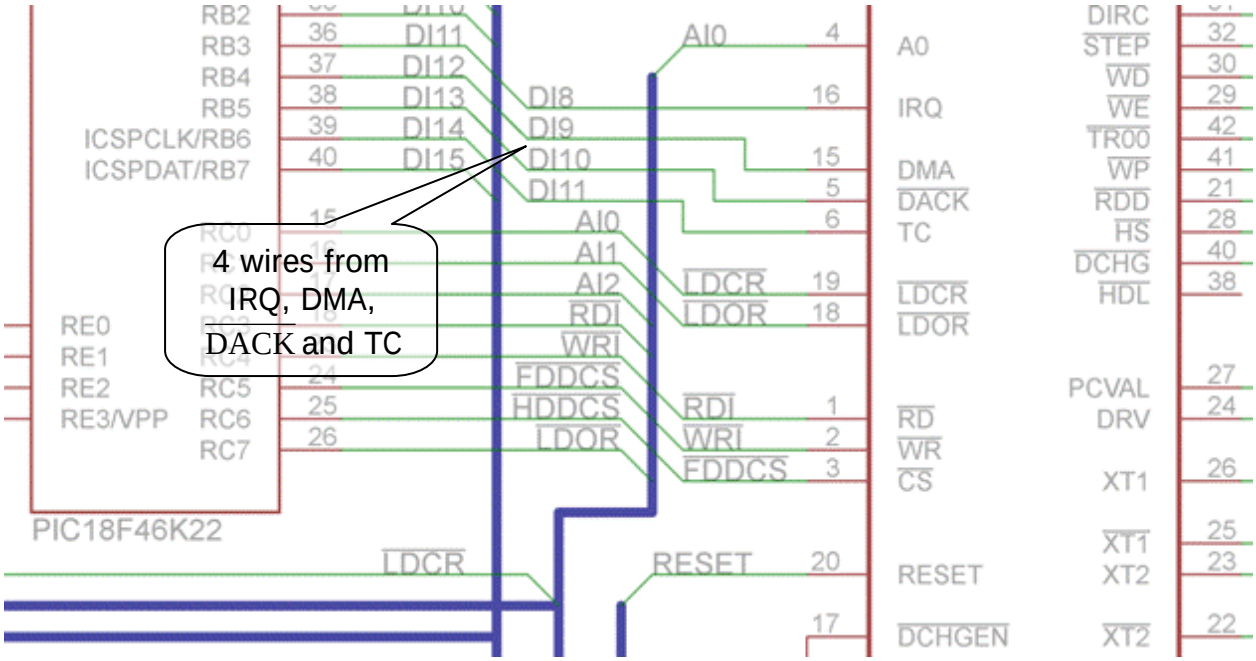


Figure 6.2. Amended circuit diagram

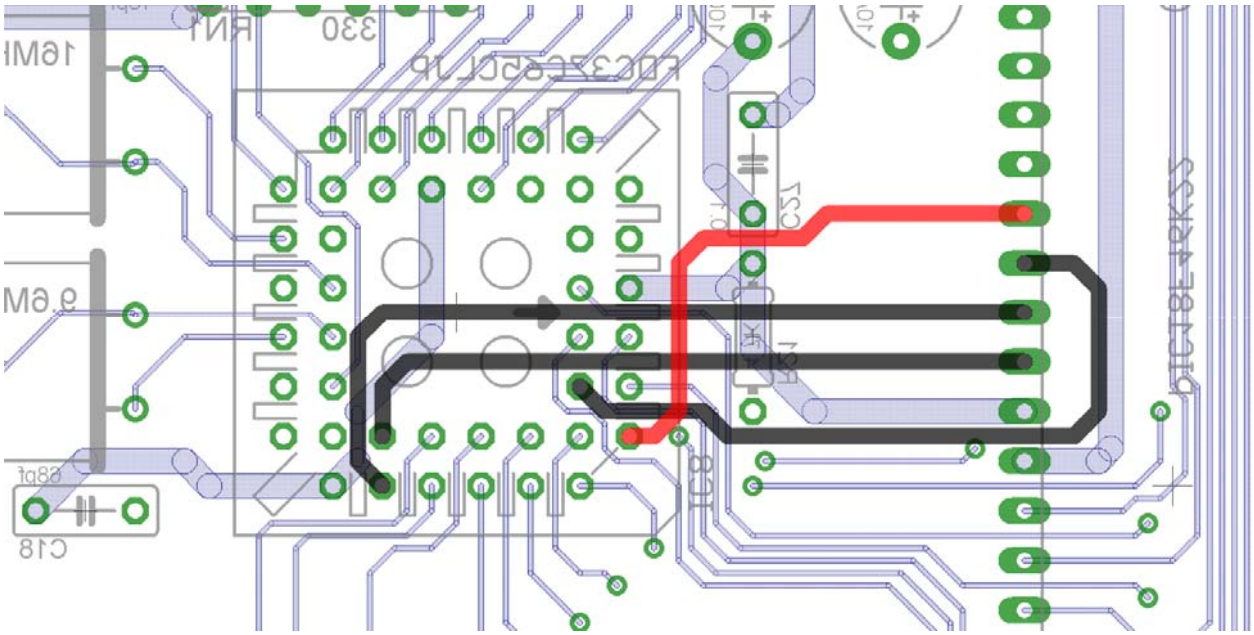


Figure 6.3. Air-wires to solder (view from PCB's solder side)

Unlike initially chosen PIC16F1939, new microcontroller PIC18F46K22 (which is a direct replacement for former) is twice faster and is able to serve GR8BIT CPU requests without additional wait stated introduced by slowdown circuit.

PIC18F46K22 has 3896 bytes of SRAM, and it can read up to 7 sectors per one DSKIO API call. After the read operation, MC returns number of sectors actually read for

GR8BIT CPU to read valid user data from the MC's data buffer. Such a technique, coupled with DMA access method, dramatically increased floppy disk transfers speed relative to previous implementations.

6.2. Hard disk subsystem design

IDE interface uses PIO transfers Mode 0 only and CHS media access method to maintain compatibility with older ATA/IDE hard disk drives. If you wish you can change timing of the IDE bus modifying MC's related microcode to speed up HDD data I/O port access.

Unfortunately initial electrical design of the IDE interface has non-critical flaw: it should have pull-up resistors on its data lines. The symptoms can only be experienced if no HDD is connected – due to no pull-up resistors and no active-output device on the bus, storage microcontroller does not read valid value (either 00h or 0FFh), and thinks there's something attached to the IDE interface. This flaw can be easily fixed by adding at least one resistor pack, 10KOhms in nominal resistance.

We recommend following the following steps:

1. Prepare resistor packs as shown on fig. 6.4 – bend pins 1 and 2 completely up (pins 1 will connect to +5V and pins 2 will be unused), and slightly bend other pins up so that when resistor pack's body is put onto the PCB pins 2-10would touch IDE connector's protruding pins;
2. Solder resistor packs as shown on the fig. 6.5.

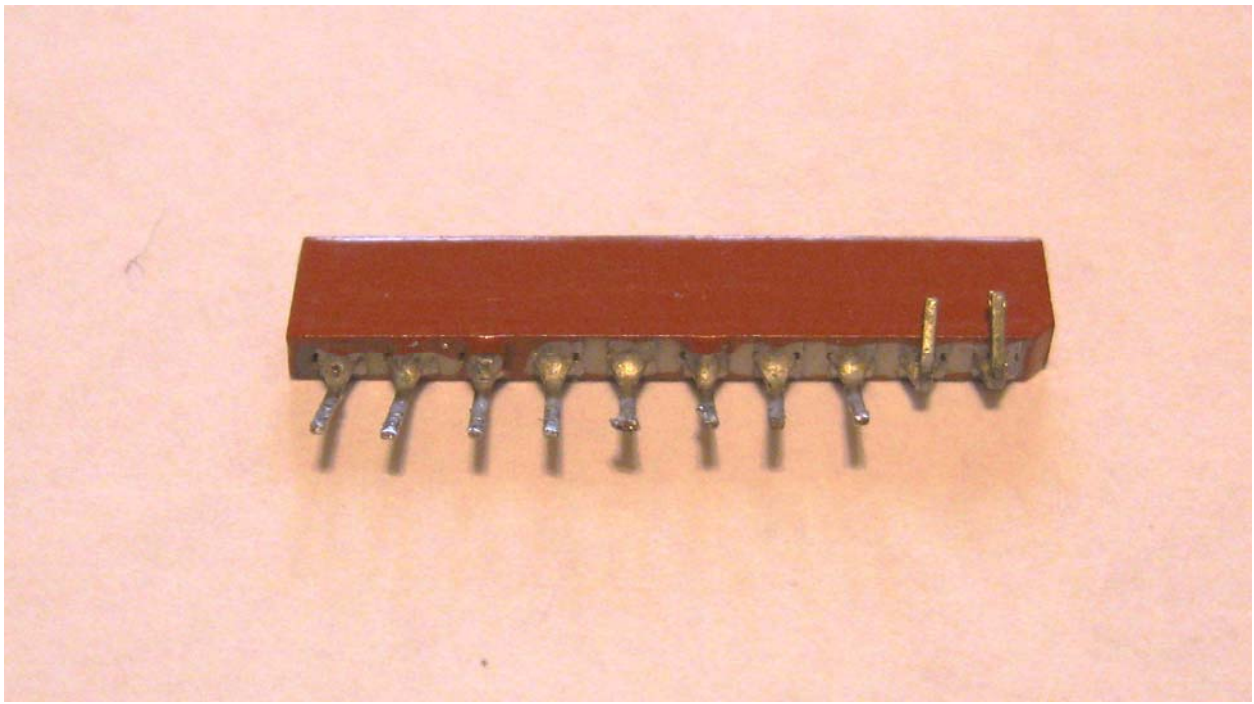


Figure 6.4. Resistor pack's pins 1 and 2 bent completely and others slightly up

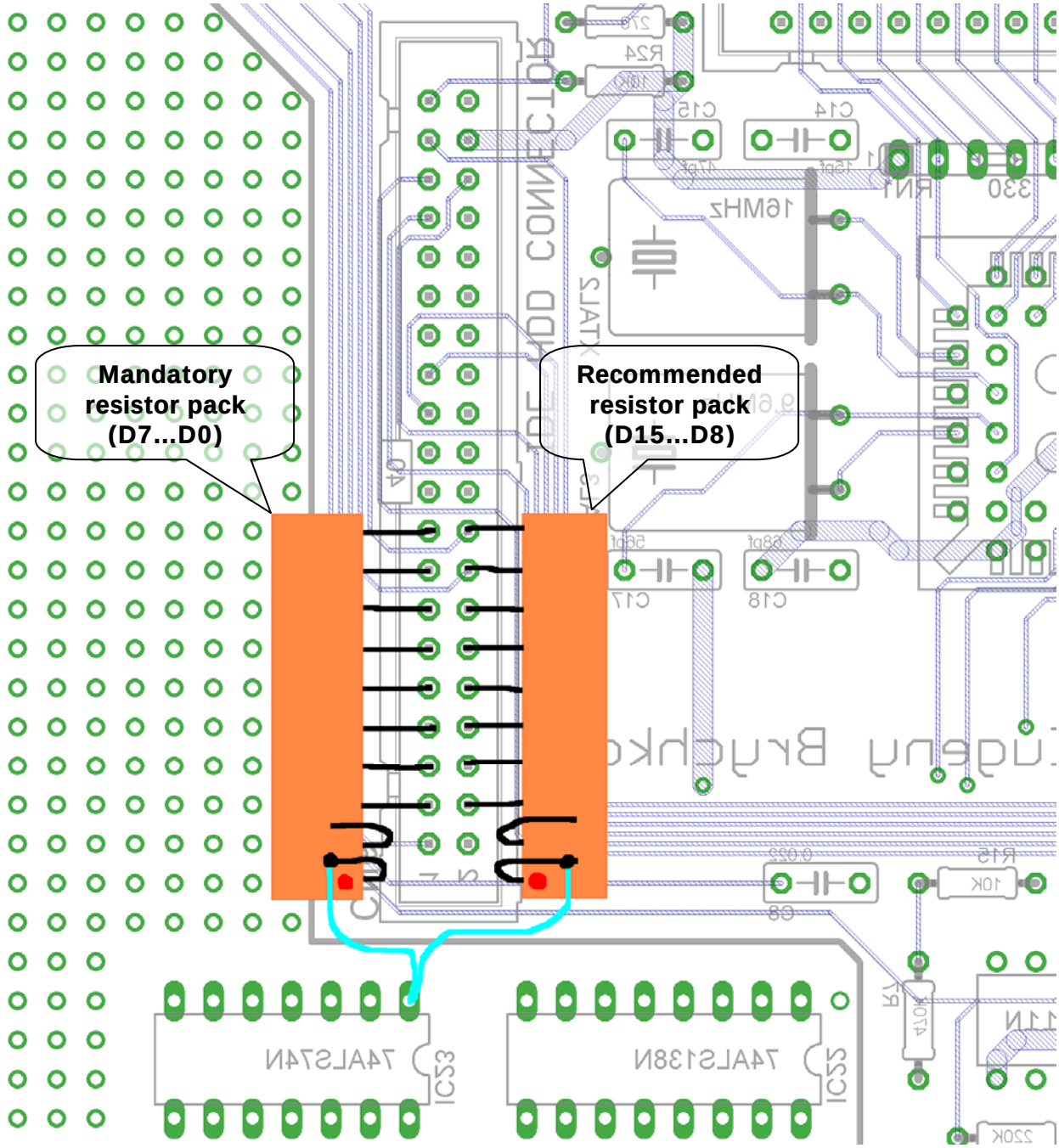


Figure 6.5. Resistor packs soldered to the IDE connector (board's solder side)

6.3. DOS2 mini-mapper design

In default design GR8BIT allows its CPU to "see" storage subsystem ROM's content in the banks 1 (4000-7FFF) and 2 (8000-BFFF), first half of the 64Kbyte IC3 chip if jumper JP3 is closed, and second half if it is open. The 16Kbyte pages within 32Kbyte visible space appear exchanged, i.e. if you have JP3 open, then system will see chip's contents in the range C000-FFFF in the page 1 (addresses 4000-7FFF) and contents in the range 8000-BFFF in page 2 (addresses 8000-BFFF). This way, if you do not implement DOS2 mini-mapper, you may have DOS1 ROM image (which is located in the

ROM chip in space C000-FFFF) visible to the CPU at addresses 4000-7FFF and allow it to function properly in DOS1 mode.

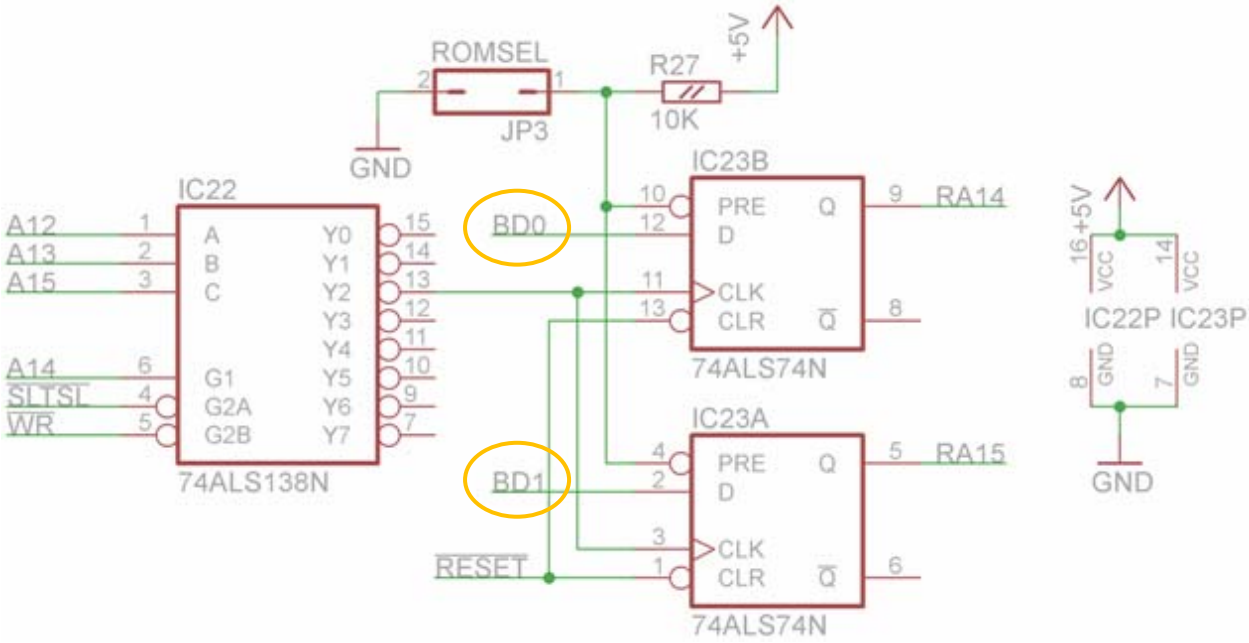


Figure 6.6. Circuit diagram of the DOS2 mini-mapper

Figure 6.6 shows circuit diagram of the DOS2 mini-mapper. You will add two chips, and cut three tracks on the board. If JP3 is closed, both IC23 trigger Q outputs are set to logical 1, and IC3's A14 and A15 lines are steadily set to 11 (page 3), causing GR8BIT to always see DOS1 ROM image. If JP3 is open, triggers' outputs after system reset is 00 (page 0) GR8BIT can switch between banks writing to the address range 6000-6FFF when storage ROM is visible in the CPU space. Page change is performed when the following conditions are met at the inputs of IC22 (fig. 6.6): A15=A12=0, A14=A13=0 (address space 6xxxh), $\overline{WR}$ signal is active (system is writing to the memory or port), and $\overline{SLTSL}$ is active (CPU address points to the sub-slot space identified by the respective SLTSL signal – SJ1 or SJ3 jumpers of the I/O board).

ROM chip select signal need also be changed from $\overline{CS12}$ to $\overline{CS1}$ (fig. 6.7) so that storage ROM pages would appear only in the bank 1 (addresses 4000-7FFF). If we would leave $\overline{CS12}$ signal, then CPU may erroneously find something "useful" in the bank 2 and try working with it – and such a condition may yield unexpected results and affect system stability.

It is important to wire data input lines of the IC23 trigger elements directly to the GR8BUS data lines (BDx), not to the I/O board buffered data lines. I/O board's data buffer IC1 is activated only for EPROM read using $\overline{CS1}$ and $\overline{SLTSL}$ signals (IC5D), but $\overline{CS1}$ signal only activates on read from the bank 1 (see IC31B on the main board's schematic – it uses $\overline{RD}$ signal to activate de-multiplexer element).

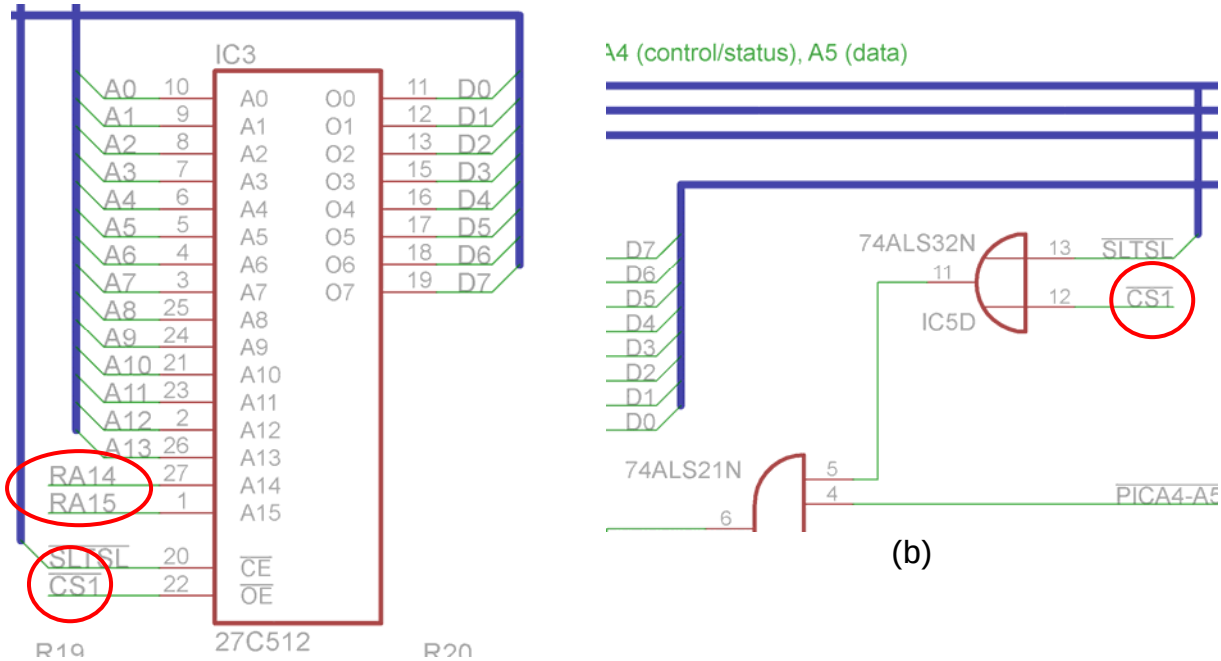


Figure 6.7. Modifications to the "chip select" line – from $\overline{\text{CS12}}$ to $\overline{\text{CS1}}$

Here's the list of steps to implement changes in order to have DOS2 mini-mapper in your GR8BIT system.

1. Cut one conductor at the bottom of the I/O board, and two at the top (fig. 6.10). One at the bottom is A14 line coming from the GR8BUS edge connector, and two the top are $\overline{\text{CS12}}$ from edge connector and A15 input of IC3 from JP3;
2. Solder two sockets into the development area (any place you have available), wire them to the power rails, and also wire some other close-up pads (fig. 6.11);

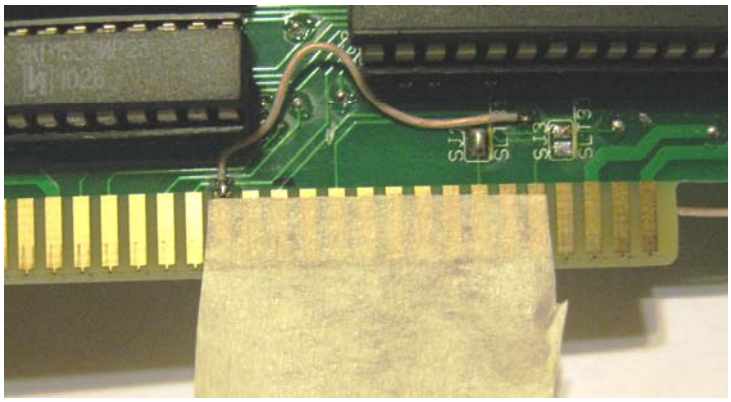


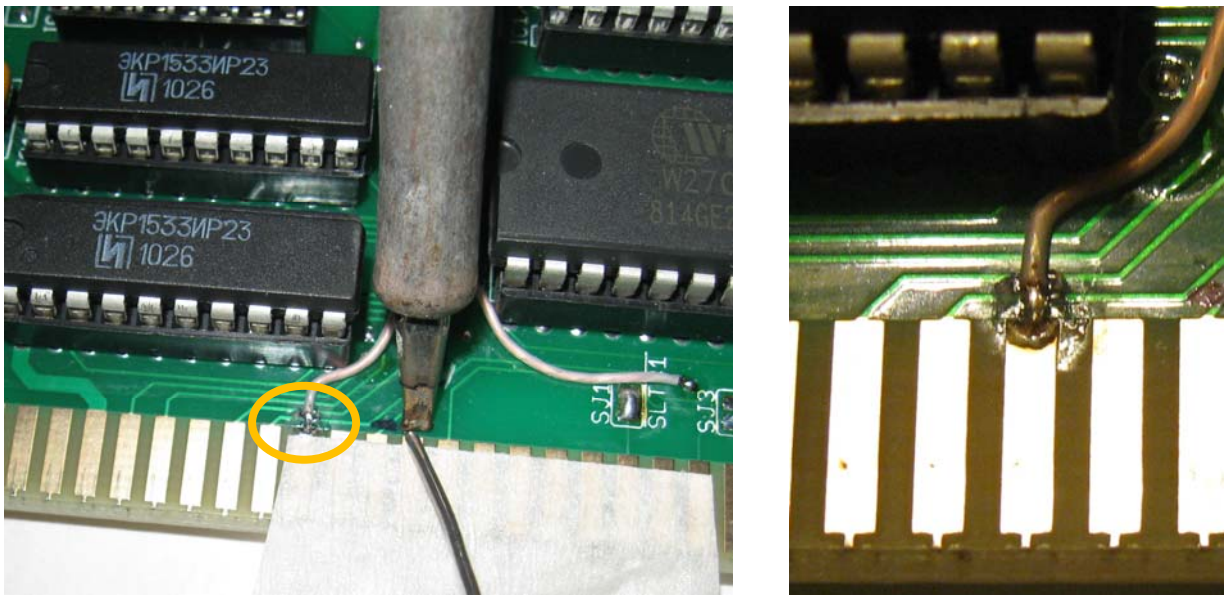
Figure 6.8. Apply paper-based adhesive tape to the edge connector at the place of tinning

3. Prepare to solder three SMD pads of the GR8BUS edge connector – signals A15, A14 and $\overline{\text{CS1}}$:
   a. apply paper-based adhesive tape to the pads leaving their very top open – this tape will ensure that solder will not flow down the pad *quickly*

filling its part which inserts into the GR8BUS connector on the main board (fig. 6.8);

b. Very carefully tin top of the pads (fig. 6.9a), applying limited quantity of the tin within small period of time
c. Connect air-wires to them (check fig. 6.12 and 6.13 for wire lengths);
d. Remove tape, and clean pads with spirit.

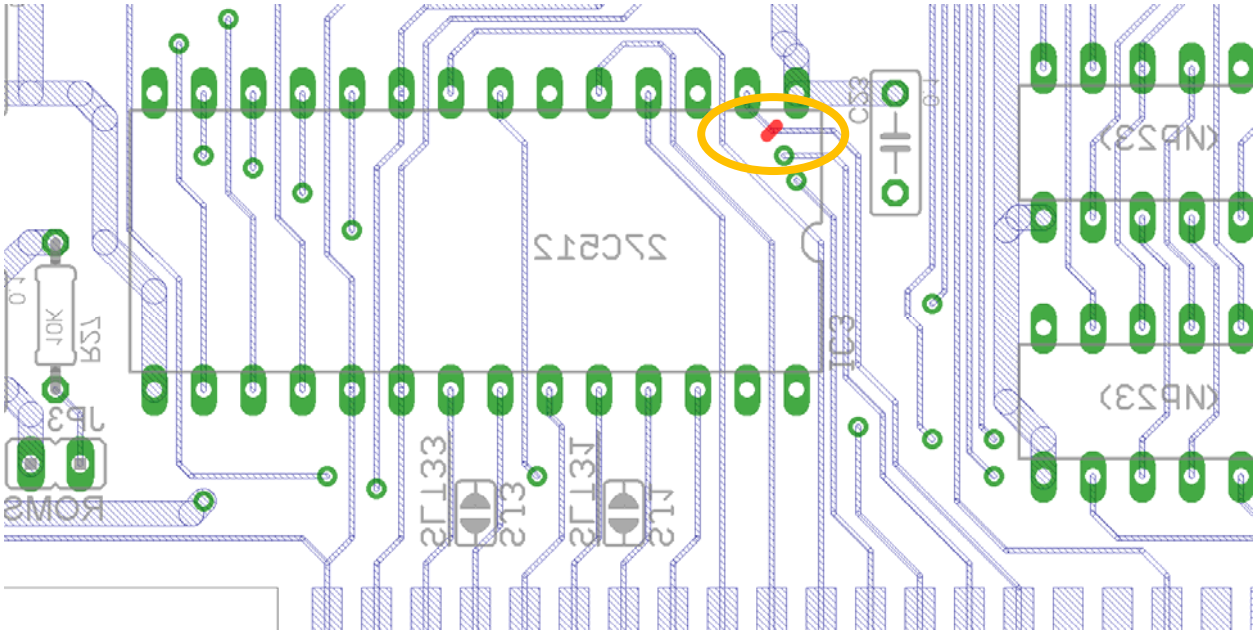
4. Solder wires as shown on the fig. 6.13. Keep them as clustered as possible.



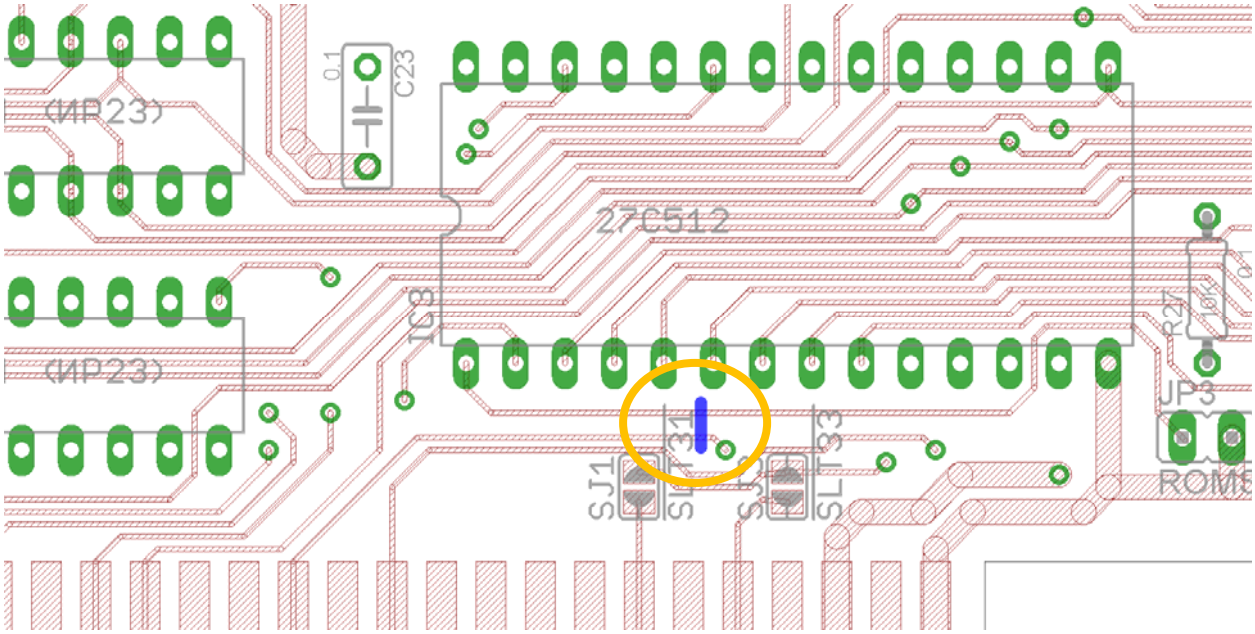
(a) (b)

Figure 6.9. Tinning edge connector's pad (a), and having air-wire connected (b)

Note: $\overline{CS1}$ signal is circled orange on the picture to the left. In this example we tin another pad as $\overline{CS1}$ was already tinned before.



(a) Solder side view, one cut



(b) Component side view, 2 conductors in one cut

Figure 6.10. Conductors to cut at the bottom (a) and at the top (b)

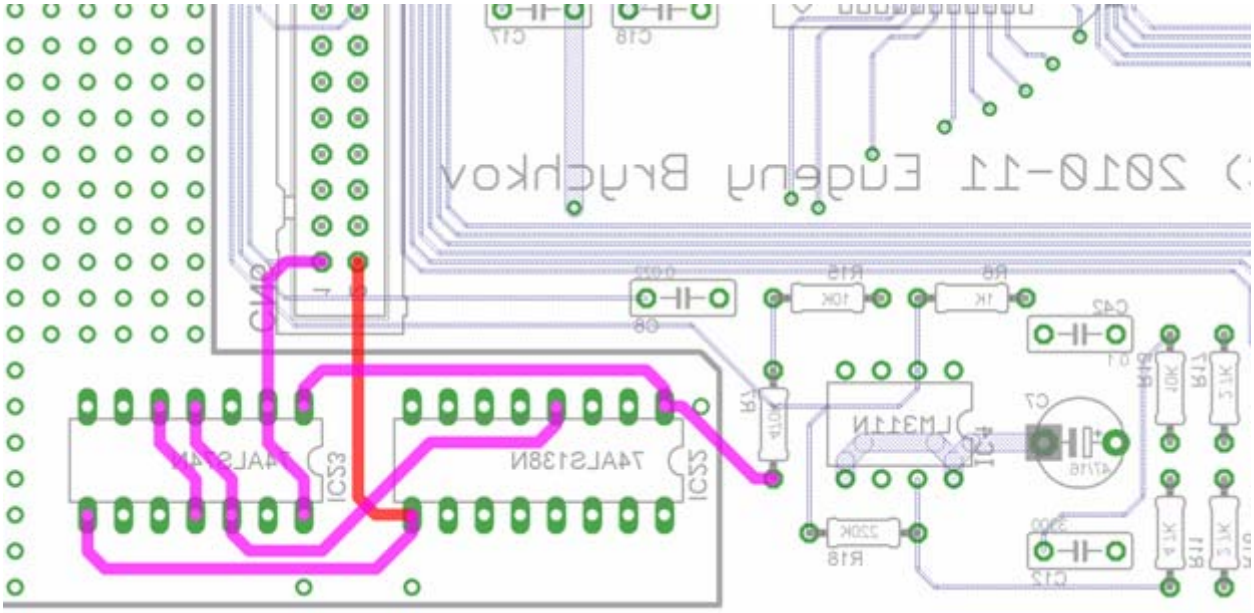


Figure 6.11. Wire power rails and some other close-up pads

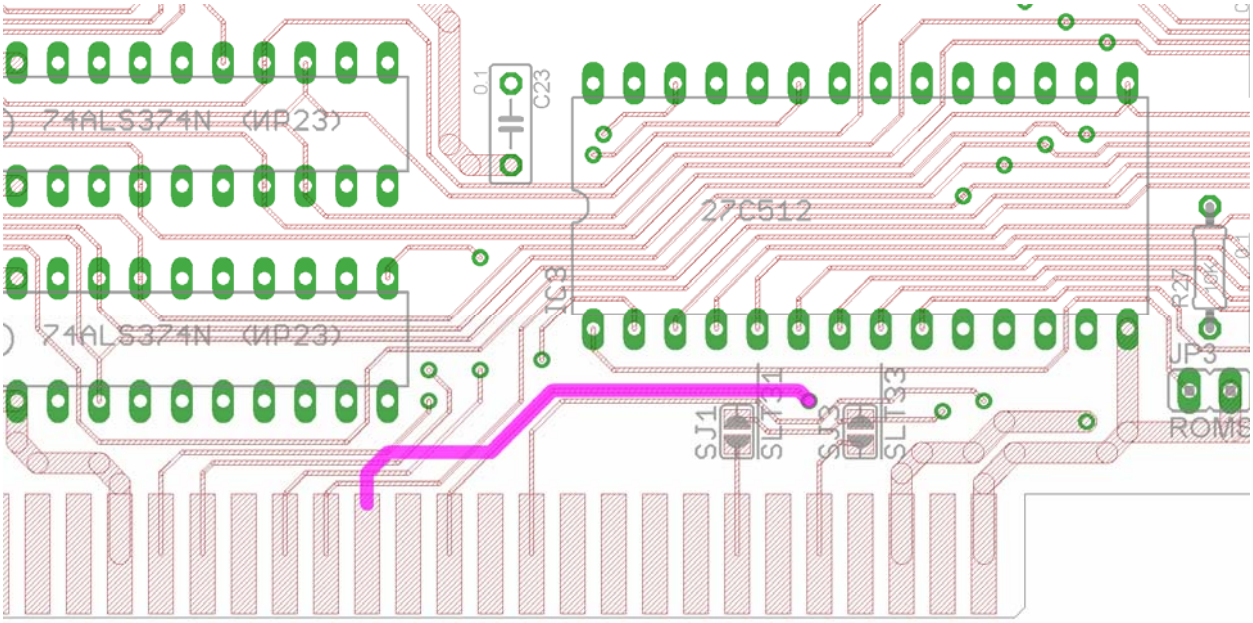
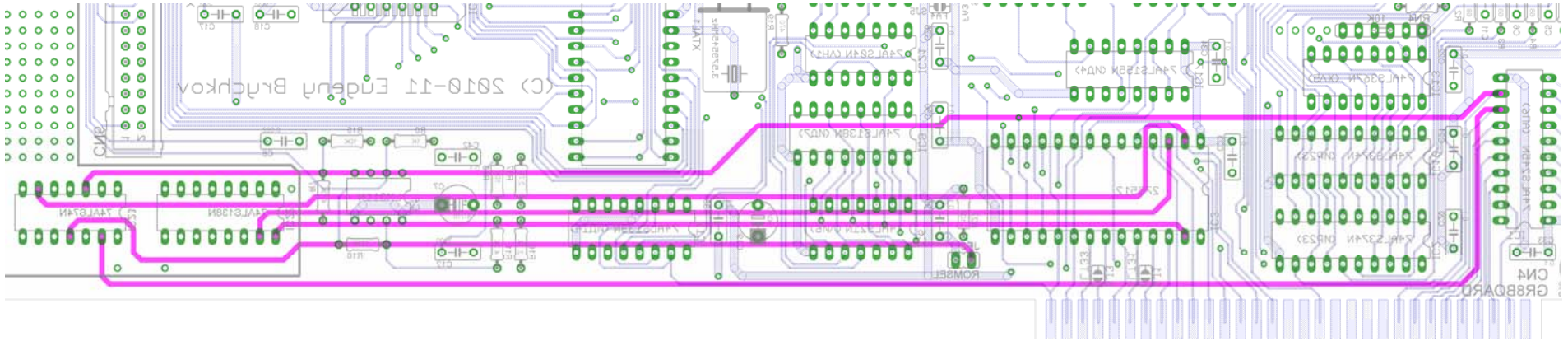
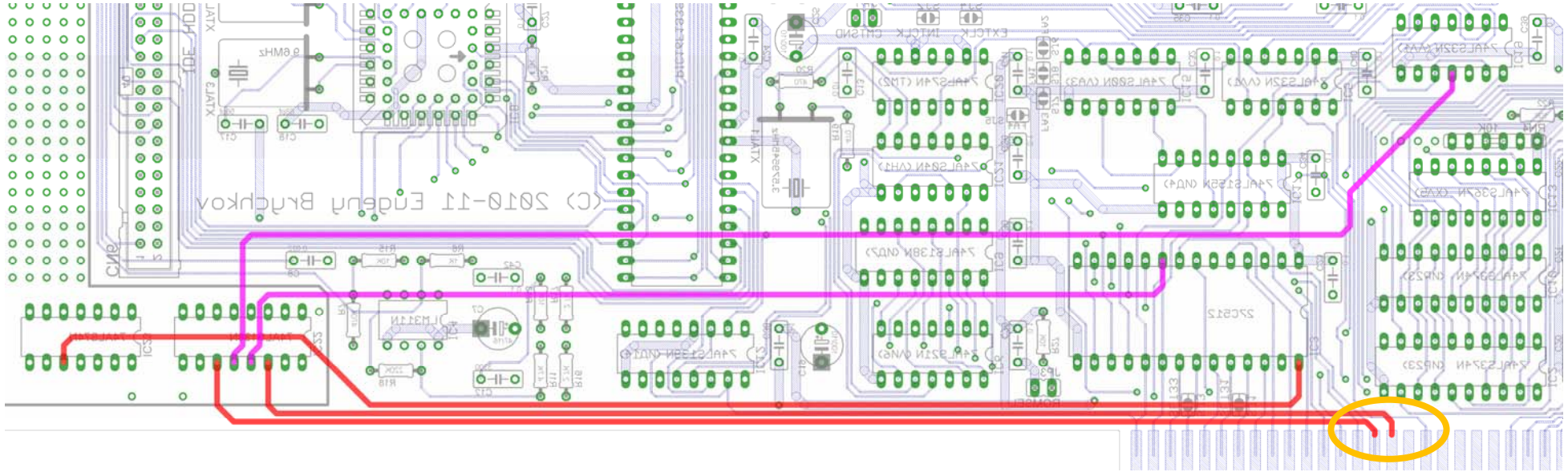


Figure 6.12. One air-wire at the component side of the I/O board – connecting $\overline{CS1}$ pad of the GR8BUS edge connector and a via



(a) Solder side view



(b) Solder side view

Figure 6.13. Long wires connecting address, data and control signals

7. Programming GR8BIT storage subsystem

7.1. Application API

Application API is used by the GR8BIT and other compatible application programs to access resources attached to the GR8BIT storage subsystem; they will use standard system (BASIC or MSX-DOS) calls. Disk subsystem calls (such as INIHRD, DSKIO, DRIVES, DSKCHG, GETDPB) are not expected to be used by application software (are used by system software).

The minor, but important change was introduced to the DSKIO system call to support 23-bit sector numbers: if bit 7 of register C (media descriptor) is reset, then remaining 7 bits are treated as upper sector number. With such a mechanism system and application software can address volumes with up to 8,388,608 sectors (4Gbytes). In terms of DSKIO system call, this 23-bit sector address is a sector offset relative to the base of the LVOL, which is always 32-bit. Hard disks' MBR is not directly addressable by application or system calls because MBR is located out of LVOL addressable space.

GR8BIT will flash alternate screen color if application or system software tries to load data onto the stack memory location, with DSKIO returning *Write Protected* error for read operation. This functionality is very useful in DOS1 mode in multiple disk drive configurations when significant portion of memory is used by the disk buffers and programs may not fit into available free memory.

7.2. System level API

System API is used by the GR8BIT storage subsystem ROM and system utilities to control storage microcontroller (MC). Following is the list of the commands available for the system software (table 7.1). Detailed explanation of the API and CPU-microcontroller data exchange is provided in the next section.

Table 7.1. List of the available MC commands

Cmd	Name	Purpose
00	INIHRD	Initialize storage system. Enumerates physical floppy drives, recalibrates them and populates LVOL structures in its workset. Returns number of logical devices configured (use by DOS1 ROM to cancel installation of the storage ROM)
01	DSKIO-R	Read of a set of sectors to the MC buffer, maximal number of sectors per call: 7 (limited by the size of internal MC's SRAM). Logical sector access (23-bit). Returns number of sectors actually read
02	DSKIO-W	Write a set of sectors from the MC buffer, maximal number of sectors per call: 7 (limited by the size of internal MC's SRAM). Logical sector access (23-bit). Returns number of sectors actually written
03	DSKCHG	Get floppy drive's disk change status
04	FMTDSK	Format floppy disk
05	DRIVES	Get number of logical devices served by storage subsystem

Table 7.1 continued

Cmd	Name	Purpose
06	VERSION	Get MC's identification string
07	SOFTWARE	Get a chunk of the system software (MSXDOS.SYS, MSXDOS2.SYS, COMMAND.COM, COMMAND2.SYS)
08	GR8STORM	Perform user dialogue with GR8BIT storage management utility
09	HDDSTAT	Get contents of the specified HDD unit's status and error register
0D	INIDPB	Is used by the GR8BIT storage ROM to get drive parameter block (DPB) information about the logical device
0E	PROMPT	Check if floppy drive host is about to access has actual diskette according to the phantom logical drive, and return flag is host should prompt to insert actual diskette
0F	GETDPB	Get DPB of the media in the specified logical device. DPBs of LVOLs are considered always intact
10	HDDID	Get HDD identification information (Identify command). Full 512 bytes of identification information are returned
11	BUFACC-R	Read from MC buffer
12	BUFACC-W	Write to MC buffer
AA	ECHO	Tests whenever MC is present and/or available within the system

In the following section **Port 0** is I/O port at address 0xA4, and **Port 1** is I/O port at address 0xA5. The following operations read or write to these ports:

	Port 0		Port 1	
Read	DB A4:	IN A,(0A4h)	DB A5:	IN A,(0A5h)
Write	D3 A4:	OUT (0A4h),A	D3 A5:	OUT (0A5h),A

7.2.1. ECHO command

This command is used to test if MC is ready to accept commands. CPU sends 0xAA to the MC, and MC replies 0xAA. Next byte read after 0xAA should be 0xFF.

Type	Description	Byte	Port	Host R/W
Command (microcontroller)	ECHO Check MC ready	0xAA	Port 0	W
	Response	0xAA	Port 1	R

7.2.2. INIHRD command

INIHRD initializes storage subsystem (floppy disk and hard disk – according to configuration set in EEPROM) and its variables.

Type	Description	Byte	Port	Host R/W
Command (floppy)	INIHRD Initialize storage subsystem	0x00	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Confirmation code	0xEC	Port 1	W
	Perform dialogue with the GR8BIT CPU			
	Read number of logical devices configured	0x01	Port 1	R

7.2.3. DSKIO commands

These commands read or write logical sector. DSKIO allows programmer to address logical devices (not physical ones). Maximal number of contiguous sectors MC can read or write is 7 (512 bytes * 7 = 3584 bytes). Reads or writes are performed to/from MC's buffer, thus programmer should successfully load valid data using BUFACC-W command before writing, and get valid data from the buffer using BUFACC-R command after reading. C/H/S values for floppy drives and hard disks are calculated by the MC basing on the values of *sector per track* and *heads* obtained by the MC from respective device's boot sector.

Type	Description	Byte (exm)	Port	Host R/W
Command	DSKIO* Sector read (write) with retries	0x01 (0x02)	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Logical device identifier 0=drive A:	0x01 (B:)	Port 1	W
	Number of sectors to process	0x07 (maximum)	Port 1	W
	Media ID/Sector # bits 22..16* **	0xF9	Port 1	W
	Sector # byte 0**	0x00	Port 1	W
	Sector # byte 1**	0x00	Port 1	W
	Command execution phase (read or write), filling MC buffer			
	Result phase flag	0xAF	Port 1	R
	Number of sectors successfully processed	0x01 (remaining 6 were not)	Port 1	R
	Completion status read	See table 7.2	Port 1	R

* If bit 7 (MSB) of the Media ID is 0, then parameter is treated as upper value of sector number
** Relative to the LVOL base in LVOL table for logical volumes, absolute for floppy disk devices.

7.2.4. DSKCHG command

This command reads status of the floppy disk change line off the specified logical device. It returns 0xFF is media was changed since last call, or 0x01 if media was not changed. Calling DSKCHG resets disk change flag. For hard disk-based LVOLs DSKCHG command always returns 0x01 (not changed).

Type	Description	Byte (exm)	Port	Host R/W
Command (floppy)	DSKCHG Check device's change status	0x03	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Device# 0=A:	0x01 (B:)	Port 1	W
	Command execution phase			
	Result phase flag	0xAF	Port 1	R
	Disk change status read	0xFF (chg) 0x01 (n-chg)	Port 1	R

7.2.5. DSKFMT command

DSKFMT command formats floppy disk. It does not partition or format HDD – please use GR8STORM™ built-in utility for these tasks.

Type	Description	Byte (exm)	Port	Host R/W
Command	DSKFMT Format floppy media	0x04	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Device# 0=A:	0x02 (C:)	Port 1	W
	Format parameter	See table below	Port 1	W
	Confirmation code	0xEC	Port 1	W
	Format command start flag	0xAE	Port 1	R
	Drive initialization, disk format, and boot/FAT/directory sectors initialization			
	Result phase flag	0xAF	Port 1	R
	Completion status read	See table 7.2	Port 1	R

Format parameter definition

Value	Description
0	Standard 1.44 (9 sectors per FAT copy, 18 sectors per track)
1	Compatible 1.44 (3 sectors per FAT copy, 18 sectors per track)
2	Legacy 720 (3 sectors per FAT copy, 9 sectors per track)

7.2.6. DRIVES command

Returns number of logical devices configured by the storage subsystem.

Type	Description	Byte (exm)	Port	Host R/W
Command (system)	DRIVES Get number of logical devices	0x05	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Number of logical devices	0x01 (1drv)	Port 1	R

7.2.7. VERSION command

This command returns MC's identification string, terminated with 0xFF. All the characters returned are ASCII ones, except terminating character.

Type	Description	Byte (exm)	Port	Host R/W
Command (microcontroller)	VERSION Get version of MC firmware	0x06	Port 0	W
	Command accept flag	0xAB	Port 1	R
	MC firmware ID string (read in loop)	Terminated with 0xFF	Port 1	R

7.2.8. SOFTWARE command

MC has 64Kbytes of the internal flash ROM, and it keeps copies of original MSXDOS.SYS (v1.03), COMMAND.COM (v.1.11), MSXDOS2.SYS and COMMAND2.COM (v2.20). Images of this software can be downloaded using SOFTWARE command in 512-byte chunks. By default these images are not designed to be bootable off the MC.

Chunk #0 contains package identification information: file name (11 bytes), 0x00 (initial extent #), size in bytes (2 bytes) and remaining fill bytes up to 512 byte chunk size. Maximal size of the package (memory space which can be addressed with chunk ID) is 2^6 * 512bytes = 32Kbytes. It is practically possible to read contents of MC's flash ROM beyond the last valid chunk of the package. Package image starts at chunk #1. Bytes within last chunk beyond package size are not invalid.

Type	Description	Byte (exm)	Port	Host R/W
Command	SOFTWARE Get chunk of the system SW	0x07	Port 0	W
512 times	Command accept flag	0xAB	Port 1	R
	Package ID and chunk ID	See table below	Port 1	W
	Data exchange is required flag	0xAE	Port 1	R
	Data read from flash ROM	Data byte	Port 1	R
	Sending CRC for checking	8-bit CRC	Port 1	W
	Result phase flag	0xAF	Port 1	R
	Completion status read	0xCC (success) 0x00 (CRC err)	Port 1	R

SOFTWARE command parameter format

Bit	Description
7..6	00 Return 512 byte chunk of data from image of MSXDOS.SYS 01 Return 512 byte chunk of data from image of MSXDOS2.SYS 10 Return 512 byte chunk of data from image of COMMAND.COM 11 Return 512 byte chunk of data from image of COMMAND2.COM
5..0	Data chunk number (in 512 byte increments, thus maximal size of the software package can be 32KBytes)

7.2.9. GR8STORM command

This command invokes dialogue between GR8BIT CPU and microcontroller, with CPU (user) provides input to the GR8STORM application, and MC performing requested actions and giving results back to the CPU (user). Dialogue is text-based. Please review sector 4.3 for description on how dialogue works.

Type	Description	Byte (exm)	Port	Host R/W
Command	GR8STORM Invoke dialogue with built-in GR8BIT storage manager	0x08	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Confirmation code	0xEC	Port 1	W
	Perform dialogue with the GR8BIT CPU			

7.2.10. HDDSTAT command

This command allows host to identify the status of the respective HDD. It supplies status register, identifying readiness of the drive, and error register, which contains relevant information in case of status register's bit 0 ERR is set.

Type	Description	Byte (exm)	Port	Host R/W
Command (hard disk)	HDDSTAT Get HDD status information	0x09	Port 0	W
	Command accept flag	0xAB	Port 1	R
	HDD ID #	0x00 or 0x01	Port 1	W
	HDD status register	Byte	Port 1	R
	HDD error register	Byte	Port1	R

7.2.11. INIDPB command

This command is issued by the host's CPU to perform initial configuration of the drive parameter blocks (DPBs) of the logical devices. DPB data is obtained from boot sector, which is read from the device or volume associated with logical device number supplied. For FAT16-type LVOL command fails with completion status code 0xCD, which signals to host CPU to apply dummy DPB.

Type	Description	Byte (exm)	Port	Host R/W
Command (hard disk)	INIDPB Get hard disk drive ID information	0x0D	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Logical device# 0=A:	0x02 (C:)	Port 1	W
	Read of media's boot sector and construction of the DPB			
	Result phase flag	0xAF	Port 1	R
	Completion status read	See table below	Port 1	R
	If result flag indicated success, return 12 bytes of DPB	(12 bytes)	Port 1	R

INIDPB completion code description

Value	Description
0x00	Fail, unable to construct DPB, no DPB data will be returned
0xCC	Successful completion, DPB will be returned
0xCF	Successful, but number of directory entries was truncated to 0xFE
0xCD	Fail, media has FAT16 identifier, DOS1 DPB does not matter

7.2.12. PROMPT command

This command returns flag to the host CPU it should ask user to insert another diskette into the drive, and allows proper operation of the phantom floppy drive.

Type	Description	Byte (exm)	Port	Host R/W
Command (hard disk)	HDDSTAT Get HDD status information	0x0E	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Logical device# 0=A:	0x00 (A:)	Port 1	W
	Completion status read	See table below	Port 1	R

PROMPT completion code description

Value	Description
0x00	Logical floppy drive requested is different than current one, prompt is required
0xCC	No prompt is required
0xCD	Logical device is LVOL, no prompt is required

7.2.13. GETDPB command

This command is issued by the host's CPU to get current up-to-date version of DPB. In case of success, constructed DPB is located at the offset 0x200 in the MC's buffer.

Type	Description	Byte (exm)	Port	Host R/W
Command (hard disk)	GETDPB Get hard disk drive ID information	0x0F	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Logical device# 0=A:	0x02 (C:)	Port 1	W
	Read of media's boot sector and construction of the DPB			
	Result phase flag	0xAF	Port 1	R
	Completion status read	See table below	Port 1	R
	If result flag indicated success, return 12 bytes of DPB	(12 bytes)	Port 1	R

INIDPB completion code description

Value	Description
0x0?	Fail, media error or timeout (code is the one returned by DSKIO)
0xCC	Successful completion, DPB will be returned
0xCD	Fail, media is LVOL, no DPB update is required

7.2.14. HDDID command

This command reads HDD identification information (such as physical configuration and model string) into the MC's buffer at location 0x000. After successful completion of the command, buffer will contain 512 byte structure returned by the hard disk drive.

Type	Description	Byte (exm)	Port	Host R/W
Command (hard disk)	HDDID Get hard disk drive ID information	0x10	Port 0	W
	Command accept flag	0xAB	Port 1	R
	HDD ID and data type to return	0x00 (HDD0)	Port 1	W
	Identification information is read into the buffer at 0x100			
	Result phase flag	0xAF	Port 1	R
	Completion status read	See table 4.2	Port 1	R

7.2.15. BUFACC commands

BUFACC command is used to access MC's internal buffer. Valid buffer starts at 0x000 and ends at 0xDFF (3584 bytes), but practically programmer can read and write above this space, altering or damaging MC's operations. In the example in the table below host reads (0x11) 18 bytes from MC buffer location 0x200 (e.g. DPB image after successful DPB construction by MAKEDPB command).

Type	Description	Byte (exm)	Port	Host R/W
Command	BUFACC Read from (write to) MC buffer	0x11 (0x12)	Port 0	W
	Command accept flag	0xAB	Port 1	R
	Byte count, LOW	0x12	Port 1	W
	Byte count, HIGH	0x00	Port 1	W
	Buffer offset, LOW	0x00	Port 1	W
	Buffer offset, HIGH	0x02	Port 1	W
	Data ready flag	0xAE	Port 1	R
Byte count	Read data bytes	Data bytes	Port 1	R/W
	8-bit CRC check byte	0xXX	Port 1	W
	Result phase flag	0xAF	Port 1	R
	Completion status read	See table 4.2	Port 1	R

7.2.16. Completion status (DSKIO, DSKFMT and HDDID)

Table 7.2. Completion status bytes for commands DSKIO, DSKFMT and HDDIO

Value	Description	Comments
0x00	Write protected	GR8BIT ROM will return “Write protected” for read operation if there’s insufficient memory to read data to
0x02	Not ready	Returned when microcontroller is unresponsive
0x04	CRC error	
0x06	Seek error	
0x08	Record not found	
0x0A	Write fault	
0x0C	No diskette	No host retries to be performed in this case
0xCC	Success	You can test bit 7 of the return code (1=success)

7.3. Organization of the dialogue between CPU and MC

Some commands, namely GR8STORM and INIHRD, use dialogue method to get information from GR8BIT CPU (user) and inform CPU (user) about its results. GR8BIT CPU receives characters from MC, and displays them. Information exchange is performed in ASCII format, except several control codes. Frequently used codes are listed in the table 7.3 below.

Table 7.3. Control codes used in the exchange in dialogue mode

Value	Description
0x01	Get one character from the host; character 0x03 means CTRL-BREAK
0x02	Get string from the host; max length of the string to get follows
0x80...0x92	Reserved by GR8STORM
0xFE	End of the dialogue, exit
Other char	To be printed on the GR8BIT screen

End of DN0003 “Storage subsystem v.3.0 Manual”.