

HBI-232MKII

The RS-232C Serial Communication Interface for MSX

User Manual (english version)



www.gr8bit.ru

 **ebsoft.fr**

MSX



(c) 2014-2015 EBSOFT & AGE LABS

REPRO FACTORY
<http://www.repro-factory.com>

HBI-232MKII – User Manual

Board Design and manufacturing by
Eugeny Brychkov
info@gr8bit.ru

BitCOM software development by
Eric Boez
ericb59@ebsoft.fr
&
Eugeny Brychkov

Produced for



www.repro-factory.com

**Download Latest Software, manual and driver for the
HBI-232 MKII cartridge from**

<http://msx.ebsoft.fr/rs232>

Contents

1. Introduction.....	4
2. Purpose.....	4
3. Product specification.....	4
4. Product operation.....	5
5. Programming reference.....	8
6. Testing the HBI-232MKII.....	12
7. BitCOM communication	15

Introduction

This manual gives information about the board operation and design, including programming reference. HBI-232MKII is identical in the specification to the HBI-232, and uses the same set of extended BASIC commands.

Purpose

The purpose of the board is to provide basic connectivity of the MSX computer to the external terminal or other data communication equipment like modem. The device is designed as a cartridge, and to be inserted into the primary slots 1 or 2, but also can operate in the slot expander (subslot).

Product specification

Item	Description
Host communication standard	MSX
DTE/DCE communication standard	RS-232C
Communication connector type	DB-09 female, modem mode
Computer-to-computer communication	Null-modem cable is required
BASIC extended command support	Yes
Communication speed (baud)	50, 75, 110, 300, 600, 1200 , 1800, 2000, 2400 , 3600, 4800, 7200, 9600 , 19200
Chipset	Intel 82C51/82C53 compatible

Product operation

HBI-232 MKII cartridge works out-of-the-box, no setup is required. It contains 8 Kilobytes ROM, which provides BASIC support.

To connect DCE (data communication equipment) like modem use direct connection cable, to connect two computers (MSX to MSX or MSX to PC) use null-modem cable.

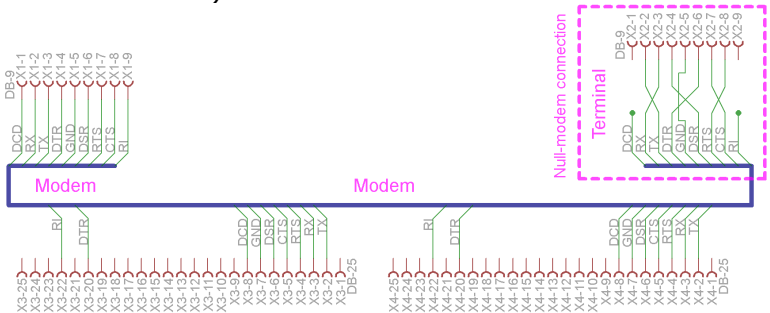


Figure 1. X1, X3 and X4 represent direct connection, X2 represents null-modem connection with former three.

For simple data transfer TX, RX and GND connections are sufficient, however for advanced protocols and handshaking CTS/RTS and DTR/DSR lines may be required.

Using product with MSX BASIC

After successful RS-232C BIOS initialization, you get COM: device, which can be used for *sequential* I/O. It is very important to initialize protocol properly before using the interface. See part 4.2 for information.

Program transfer commands

RUN"COM:"	Load and run BASIC program
-----------	----------------------------

MERGE"COM:"	Merge current BASIC program with loaded from COM port
LOAD"COM:"	Load BASIC program from COM port
SAVE"COM:"	Save BASIC program to COM port

Note: BASIC programs are transmitted and to be received in ASCII format, and they should be terminated with EOF character (0x1A).

Data transfer commands

OPEN"COM:"AS# 1	Open communication port for reading and writing
CLOSE# 1	Close communication port
PRINT# 1 ,	Output data to the communication port
PRINT# 1 ,USING	Output formatted data to communication port
INPUT# 1 ,A	Input variable from communication port. Data should be terminated with CR/LF
LINE INPUT# 1 ,A\$	Read string data from communication port
INPUT\$(5 , 1)	Get five characters from the communication port. Execution is suspended until these 5 characters arrive
LOC(1)	Returns number of bytes in receive buffer
LOF(1)	Returns number of free bytes in receive buffer
EOF(1)	Returns -1 if end of file is reached. End of file is 0x1A character

Note: file number 1 used in table above is an example.

Extended commands

Extended commands may be needed for advanced implementations of the data communication through serial port. To invoke the command, precede it with operator CALL (e.g.

CALL COMINI). Instead of word CALL you can use underscore sign (e.g. _COMINI).

Extended commands

COMINI("0:8N1NHNN", 19200,19200,5)	Initialize communication port with 8 data bits, no parity, 1 stop bit, no XON/XOFF control, CTS/RTS handshake, no auto-LF addition or removal, and no shift-in/shift-out. 19200 baud for receive and transmit, and 5 seconds timeout
COMTERM	User prompt is used as a terminal. Characters typed are sent to the remote device, and characters received from remote device are displayed. Communication port properties set by COMINI are used
COMDTR(,1)	Sets DTR (Data Terminal Ready) signal on. This signal <i>usually</i> means that communication port is ready for information exchange
COMBREAK(,20)	Sends 20 break frames to the transmit channel. Break is a condition of holding TX line low for the period of one character frame
COMSTAT(,A)	A gets 16-bit word of status of the communication system
COM GOSUB(,100)	When interrupt occurs, BASIC program branches to line 100. In HBI-232MKII interrupt, when enabled, is generated when port has character received

COMON	Enables hardware interrupts caused by controller receiving character
COMOFF	Disables hardware interrupts caused by control receiving character. Program still can use pilling of status register to know the status of the controller's buffer
COMSTOP	Suspends execution of interrupt routine until COMON is executed

Note: file number 1 used in table above is an example.

BASIC environment implementation considerations

BASIC is able to process communication at the **1200 baud**, at higher speeds standard BASIC I/O commands may give "Device I/O error" system error. You may need to implement low-level inline assembler routines.

Please review part 5 for programming reference.

Programmin references

When communication subsystem starts, it initializes in 1200 baud rate. This speed is guaranteed to 100% work with I/O using BASIC.

However while usage of BASIC I/O operators at higher speeds is not advised, you still can use extended commands to initialize communication subsystem, and then use it through inline assembler routines at required speeds.

I provide basic information about low-level communication using HBI-232MKII, all management and configuration tasks are expected to be performed with these extended communication BIOS commands.

USART command register

The command register of 8251 USART chip allows you to operate state of the control lines of the RS-232C port. The register is write-only. It is located at the port 0x81.

The format of register as follows:

Bit	Name	Usage
7	EH	Search for sync character in synchronous mode. Keep it 0
6	IR	Internal "software" reset. Keep it 0. Do not set it to 1 otherwise configuration will be reset
5	RTS	Request to Send pin control. Setting this bit to 1 activates $\overline{\text{RTS}}$ pin (sets it to logical 0). RTS signal of the DB9 connector goes high
4	ER	Error reset bit, resets PE, OE and FE errors in status register
3	SBRK	Send Break, forces TX output low. Normally is kept 0
2	RxE	Receive enable. If this bit is 0, 8251 will not receive any incoming data, and data will be lost
1	DTR	Data Terminal Ready pin control. Setting this bit to 1 activates $\overline{\text{DTR}}$ pin (sets it to logical 0). DTR signal of the DB9 connector goes high
0	TxEN	Transmit enable. If this bit is 0, 8251 will not send any data to the communication channel

Under normal conditions this register is loaded with 00100111b, or 0x27.

USART status register

Status register allows you identifying the state of the USART chip. This register is read-only, and is located at the port 0x81.

Bit	Name	Usage
7	DSR	State of Data Set Ready pin. If this bit is 1, remote device has set its $\overline{\text{DSR}}$ signal active.
6	SYNDET/ BRKDET	Break detected condition. This bit is set when remote device has set break condition on its data transmission line.
5	FE	Framing error. Transmission of each byte is performed within frame, and this bit set means frame is corrupt, and data received may be invalid
4	OE	Overrun error. This bit means that CPU did not read data byte, and controller received another byte which overwritten previous byte. Previous byte is lost
3	PE	Parity error. If parity is turned on, this bit set means data is invalid because basic parity check has failed
2	TxEMPTY	Transmitter queue is empty. You can write data byte to the 8251
1	RxRDY	Receive ready. Set means that there's byte in the controller which needs to be read by the CPU. If it will not happen in time, current byte will be lost and OE error condition will be set
0	TxRDY	Tx data register in ready to accept next character. May be used for interrupt generation, but is not used for interrupt in HBI-232MKII

Communication subsystem status register

There're some more additional signals required for DCE (data communication equipment – for example modem), they can be read through separate status register located at port 0x82. CTS signal, instead of connecting it to the USART chip, is routed to this status register so that CPU can read the status of this line instead of letting USART managing it.

Bit	Name	Usage
7	CTS	Status of the Clear To Send line from the remote device
6	CT3	8253 chip has three countdown counters, this bit indicates OUT3's third spare counter (which is not used for USART) pin state
5...2	NC	Not connected, will always be 1
1	RI	Status of the Ring Indicator line from the remote device (e.g. modem)
0	DCD	Status of the Data Carrier Detect from the remote device (e.g. modem)

Data register

8251's data register is located at the port address 0x80 and can be read or written.

Write should happen when TxRDY and TxEMPTY are set, it will trigger start of frame transmission in case TxEN bit is set.

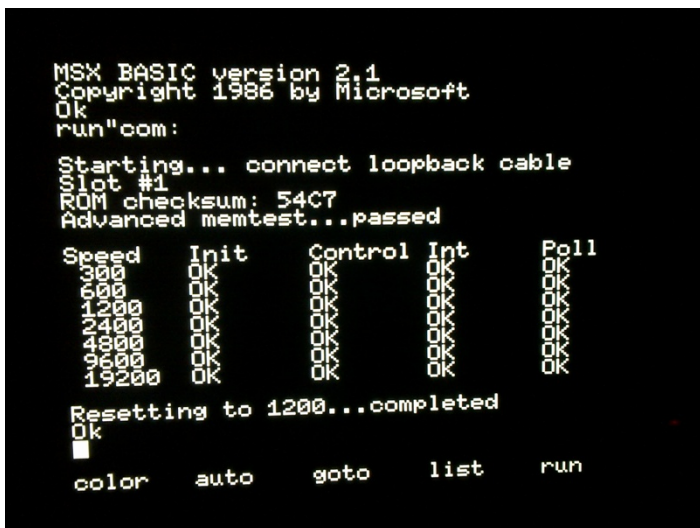
Read should happen when RxE is set and RxRDY is set. If board is having hardware interrupts enabled, RxRDY triggers interrupt, and communication controller's interrupt service routine can identify the source by checking RxRDY bit of the 8251 status register. When data is read, RxRDY bit is reset.

Hardware interrupt

Board provides an option to have its hardware interrupt enabled or disabled. Writing value of 1 into bit 0 of port 0x82 disables interrupt, writing value of 0 into bit 0 enables interrupt. All other bits (7..1) of the port 0x82 do not care.

Testing the HBI-232MKII cartridge

I developed BASIC program called *rstest.bas* for the testing purposes. This program does not require ROM to be present, thus it will test the card even if ROM image is not present or corrupt.



```
MSX BASIC version 2.1
Copyright 1986 by Microsoft
Ok
run"com:

Starting... connect loopback cable
Slot #1
ROM checksum: 54C7
Advanced memtest...passed

Speed      Init      Control  Int      Poll
300        OK        OK       OK       OK
600        OK        OK       OK       OK
1200       OK        OK       OK       OK
2400       OK        OK       OK       OK
4800       OK        OK       OK       OK
9600       OK        OK       OK       OK
19200      OK        OK       OK       OK

Resetting to 1200...completed
Ok
█
color      auto      goto      list      run
```

Figure 2. Photo of the output of the *rstest.bas* BASIC program

The program can be transferred to the MSX computer using RS-232C interface if it's functional, using attached storage subsystem (e.g. floppy disk), or using standard serial cassette port. Program

can be downloaded from the *Downloads* section of the website. Both BAS and WAV formats are available.

As program does not rely onto the BIOS, it searches for HBI-232MKII cartridge through all the slots and available subslots. The match condition is 4000-7FFF being non-writable, while 6000-67FF area within it is writable.

Next is checks validity of the ROM by searching for "COMINI" word in its content. If it is found, ROM is deemed valid, and its checksum is displayed.

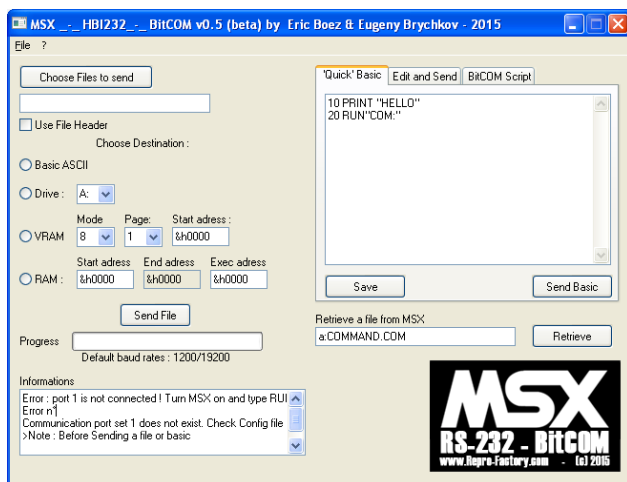
Then program performs advanced RAM test using walking 1 and walking 0 algorithms. The condition is RAM is very important because HBI-232MKII stores all its vital control information in this RAM (in contrast to RAM-less cartridges which are forced to store its data in the stack or other main memory location). RAM test takes approximately 15 minutes on the Z80 running at 3.580 Mhz.

Next program checks communication quality at different standard speeds. To run this test, you will need loopback cable (fig. 3). Test runs for about 10 minutes. The fact of the adapter passing these tests means that board is expected to function properly and you have proper loopback cable. When communicating with other RS-232C compliant equipment, quality of communication also depends on the cable you use and on the properties and settings of the remote communications equipment.



Figure 3. Simple loopback plug. For control signal loopback test connect pins (1, 4 and 6) and (7, 8 and 9) in two groups by three pins. For data test, connect pins 2 and 3.

BitCOM communication application for HBI-232 MKII



What is BitCOM ?

BitCOM is a software application for Windows (Windows XP and Windows 7) that enable you to take control of a MSX computer. Sending images, Rom, files or any data from the PC to the MSX. It can also retrieve files from the MSX drive to the PC.

Purpose of BitCOM

BitCOM was designed to make MSX software developer more easier. Most of the Time developers are creating data on a PC and test their creativity with an emulator.

BitCOM allow quickly test programs, graphics, data, directly on a Real MSX without having to manipulate storage medium as, Floppy disk, SD card... Just click on a button to send data to the MSX and see how it render in real life.

How to use BitCOM

First, connect MSX and PC with a Serial RS232 cable.

Most, PC have a DB9 serial connector.

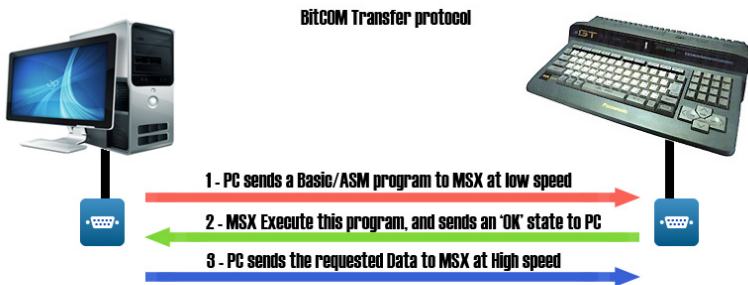
The software works also with USB to Serial adapters, until you install the right driver.

BitCOM will detect default Com port of you PC. Generally Port Com:1

If no Com port is detected, or if there is no connection, an error is showed on the information window.

You don't need to pre-launch any software on the MSX. Just enter this Basic instruction: RUN"COM:" to put the MSX on hold and set it under the command of the PC.

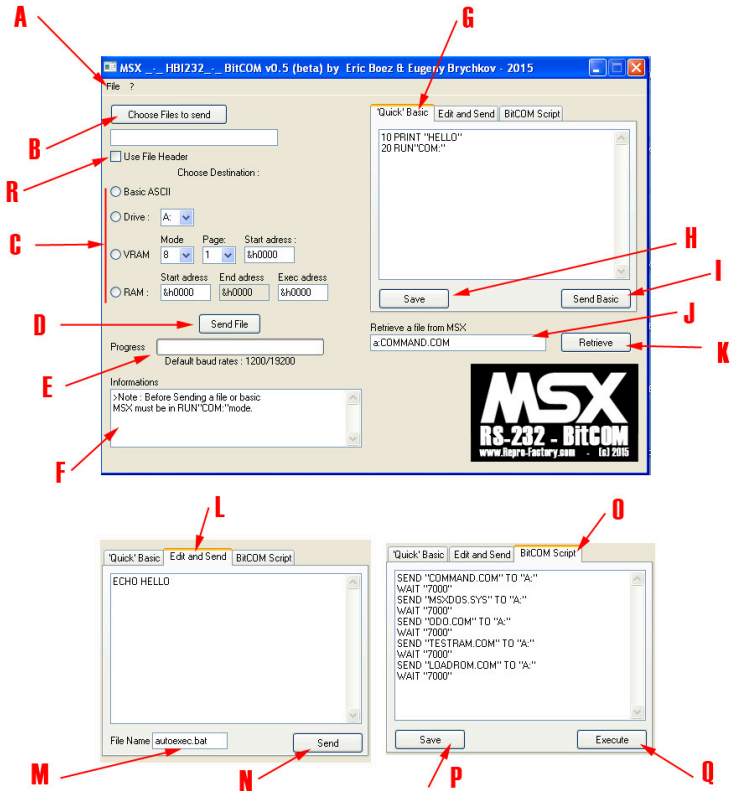
How BitCOM works



BitCOM first sends to MSX a Basic/ASM program. Once this program is received and executed by MSX, PC can tranfer the Data to the MSX at the maximal Speed (19200 bauds).

BitCOM functions

There is a lot of possibilities and features easy to use.



A - **File Menu** let you load a pre-saved script to "Quick Basic" window or to "BitCOM script" window. Let see these functions at points "G" and "L"

B- **Choose file to send.** Click on this button to choose on your PC what is the file you want to send to the MSX. It may be

any file type you want. There is no restriction on the file type you want to send. If the file type you have chosen is recognized by BitCOM as a MSX file type (File image, Bin, or ROM file), BitCOM will activate the logical destination, but you can override this setting. See point "C"

C – **Destination.** You can choose to send the file or data to the Basic Memory, a drive, the Vram, or the Ram.

- 1- If you have chosen "**Basic Ascii**" the file must be an Ascii file, otherwise result is unpredictable.
- 2- **Drive.** Choose the Drive letter when you want to save your file. It can be any functional drive available on your MSX computer (Floppy, SD card, MMC card, Hard drive ...) There is no limitation about the size of the file you want to save.
- 3- **VRAM.** Choose the appropriate screen mode for your image. You can also choose the number of the memory page you want to use. Generally you will upload an image from the beginning of the VRAM, but if you want, you can choose the start address where image will be copied.
- 4- **RAM.** Generally for .BIN or .ROM file. If BitCOM recognizes the header of the file, it will set the "Start address", "End Address" and "Execute address" boxes with the right values. If this doesn't fit your need you can override the setting and enter manually the addresses. In the case of a ROM file, you cannot change the addresses. You can transfer BIN file up to 48K or ROM file up to 32K. If a start address was set,

the ROM or the BIN file will be execute at the start address while it is in MSX memory.

D – **SendFile**. Click on this button to start sending process

E – **Use header file**. This check box enables or disables the read header file feature. (Use to choose right destination, and addresses sets)

F – **Windows information** about process, or errors.

G – **Quick Basic Editor**. You can write here a little Basic program and send it to the MSX. While MSX is in RUN"COM:" state mode, it will execute any Basic instruction you can send it. If you want MSX continue to "heard" about your orders, don't forget to add at the last line RUN "COM:"

H – You can **save** you Quick Basic Script for use it another time. To load a Quick Basic script, see point "A"

I - **Send** the Quick Basic to the MSX for execution.

J – You can **retrieve** a file from the MSX to the PC. Enter here the path, and name of the file you wan to retrieve from a MSX drive. Example "a:command.com" will download to the PC the file "command.com" from the A: drive. By default, downloaded files are save in the "Download" folder of the BitCOM script. You can specify another folder, by adding "> (folder name)" to the name. For example if you enter "a:command.com > MSX1" the

file "command.com" will be downloaded from MSX to the PC, and save in a sub-directory called MSX1.

K - Click on this button to launch the download process

L – **Edit and Save.** In this window you can easily edit a Basic file or a .BAT file and save it on the MSX default drive. Usefull to edit an "autoexec.bat" file for example.

M – **Name** of the file you want to save on MSX.

N - **Save** the file on the MSX drive. The drive set at point "C" will be use.

O – **BitCOM script**, enable you to write script for automation process purpose. By this way you can send to MSX, multiple files. Or send file, and execute program, for a demonstration purpose for example. See the available function in BitCOM Script addendum.

P – **Save** the Script on the PC

Q – **Execute** the BitCOM script

If for a reason you want to cancel a transfer process, just press "F1" on the PC keyboard.

Meanwhile MSX will still be waiting data. There is no possibility to cancel task on the MSX computer. Just make a reset abort all process.

BitCOM Default Configuration

The default configuration is saved in the file **BitCOM_config.ini** you can find in "data" directory.

You can edit "**BitCOM_config.ini**" file with any text editor, and change default values.

Basic_speed=1200

Is the bauds speed for transferring loaders from PC to MSX.

Value can be, 1200,2400,4800,9600,19200

If you change the default speed for other value than 1200, you must say to MSX to use this new speed, before sending the command Run "Com:"

by entering CALL COMINI (,xxx,xxx) where xxx is the default basic speed.

For example, you must enter the commands : **CALL COMINI (,9600,9600):run "COM:"**

to prepare MSX to receive data from PC.

Asm_speed:19200

Is the bauds speed used for data transferring.

Value can be, 1200,2400,4800,9600,19200

PC_port=1

Is the default PC com port to use. If your PC have more than one COM port, you must specify what com port to use.

Value can be 1,2,3,4 etc...

MSX_port=0

Is the default MSX com port to use.

Value can be 0 or 1

DATA_length=8

Is part of the RS232 protocol. Length 8 bits

Value can be 5,6,7,8

Parity_check=N

Is part of the RS232 Protocol. Enable checking parity.

Value can be E(Even), O(odds), I(Ignore), N(No parity)

Stop_bit=1

Is part of the RS232 Protocol.

Value can be 1(1bit), 2(1,5bits), 3(2bits)

XON_control=N

Is part of the RS232 Protocol. Enable XON/XOFF control.

Value can be X(Enable) or N (Disable)

CSRS_handshake=N

Is part of the RS232 Protocol. Enable Handshaking

Value can be H (Enable) or N (disable)

Auto_LF=N

Enable the Automatic LF code insert (When receiving).

Value can be A (Insert LF code) or N (Not insert LF Code)

AutoDel_LF=N

Enable the Automatic LF Code delete (When transmitting).

Value can be A(Delete LF Code) or N(Not delete LF code)

Shift_inout=N

Enable the shift in-out control.

Value can be S (Enable control) or N (Disable control)

QB_Dir= \ToSend

Is the default directory where BitCOM search for Quick Basic Scripts.
(Relative to BitCOM directory)

Script_dir= \script

Is the default directory where BitCOM search for Scripts. (Relative to BitCOM directory)

Download_dir= \Downloads

Is the default directory where BitCOM save files downloaded from the MSX. (Relative to BitCOM directory)

BitCOM Script commands

BitCOM Script enable to automate tasks.

This feature is still in development , few commands are available.

command: **SENDFILE "filename" to "dest"**

Send the file "filename" to a valide "dest" destination.

Valid destinations are :

VRAMxy : where is "x" is the screen mode, "y" the page number

RAM : File must be a BIN or ROM file.

BASIC : file must be a Ascii file

A: to E: file is send to a MSX drive

Command: **WAIT "xx"**

wait "xx" milliseconds, before continuing the next line of the script

Command: **GETFILE "filename"**

Retrieve "filename" on the MSX drive and download it to the PC default download directory.

Examples:

SENDFILE "image.s8" TO "VRAM81"

Send the image .s8 to MSX screen8 page 1

SENDFILE "image,s8" to "A:"

Send the file "image.s8" to drive a:

GETFILE "a:image2.s8"

Donwload the file "image2.s8" from drive A: to PC

Made in Russian Federation
and France



(c) 2014-2015 EBSOFT & AGE LABS



www.grebit.ru

 **ebssoft.fr**

MSX

REPRO FACTORY
<http://www.repro-factory.com>