

M1882

Universal PROM Programmer Operation Manual

M1882 Programmer Confirmation of Accessories

When unpacking the package received from us, check if all the following items necessary for M-1882 had been included. If any item is missing or damaged, please contact our distributor or our sales office in your district.

M1882 Programmer



CD-ROM



M1882 Control software
M1882 Operation Manual
USB, LPT Driver

Power Code



USBCable



ZIP Socket POD1



ISP Connector POD2

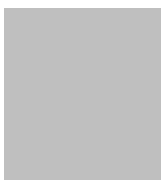


ISP Cable



(For ISP Check)

User Registration Postcard





For Safe Operation

Precaution for Safety

This operation manual includes safety indications here and there so that you can operate M1882 safely and correctly. For safe and correct operation of M1882 and also to prevent you, other operators or workers from injury and property from damage, the following pictographs are used to explain these safety indications.

Before reading this manual, fully understand these pictographs and the meanings. Keep this manual at hand refer to it as occasion arises.

Explanation of Pictographs

 WARNING	Indicates a potential hazardous situation in which the operator would be killed or seriously injured unless this precaution is observed.
 CAUTION	Indicates a potential situation in which the operator would be injured or property would be damaged unless this precaution is observed.



WARNING



Compulsion

When operating this unit, be sure to follow warnings and cautionary instructions given by Minato Electronics Inc.



Do Not Dissassemble

Do not disassemble or modify this unit. A fire may start or you may get and electric shock.



Unplug Power Cord

When finding a smoke,feeling an abnormal smell or hearing an abnormal sound,

Pull out the power plug immediately from the AC plug receptacle.

If keep operating, a fire may start or you may get an electric shock owing to short-circuit.



Unplug Power Cord

If dropped this unit or given a strong shock to the unit, pull out the power plug immediately from the AC plug receptacle.

If keep operating, a fire may start or you may get an electric shock owing to short-circuit. Consult with our repair window.



Unplug Power Cord

If any liquid or foreign matter enters this unit, pull out the power plug immediately from the AC plug receptacle.

If keep operating, a fire may start or you may get an electric shock owing to short-circuit.

Consult with our repair window.



WARNING



Compulsion

This unit shall be operated by an operator who has fully understood the operation manual of M1882.

Miss operation may damage this unit and other devices.



Compulsion

Before touching this unit, be sure to touch nearby large metal and remove static electricity from your body so that this unit is not damaged by static electricity.

Static electricity may damage this unit and other devices.



Compulsion

Clean the unit surface, the device socket and the air filter.

Operation without removing dusts from them will probably result in a fire or a trouble. Try to clean them periodically.



Compulsion

Check the pass/fail judgement not only LED of programmer but also

Check sum on display of PC.



Compulsion

Attached AC cable is only used for Japan.

You need prepare for another suitable cable in your country.

Content

For Safe Opearation

Precaution for safety

Explanation of pictograph

Content

Before Operation

For Users

Copyright of this manual

How to use manual

Chapter 1 Outline of programmer、Specification、 Installation

M1882 programmer outline	4
Component (List of component)	4
Recommneded PC for connecting M1882	5
Feature of M1882	6
Update for programmer	9
Quick start	1 0
Installation of programmer hardware	1 0
Installation of programmer software	1 0
Execution of control software	1 0
How to program device (Programming)	1 1
Part name of M1882	1 2
Connection between M1882 and PC	1 3
In using USB	1 3
In parallel port	1 3
In system programming by M1882	1 4
Selftest and calibration of M1882	1 5
M1882 basic programmer specification	1 6
Hardware	1 6
Socket pin driver	1 6
ISP Connector	1 6
Programming time	1 9
Device operation	1 9
Normal Specification	2 0

Install	2 1
Set up software	2 1
Set up hardware	2 7

Chapter 2 M 1 8 8 2 Control Software Operation Manual

M1882 Control software	3 1
Explanation of Display	3 2
File Command	3 5
Load	3 5
Save	3 7
Load project file	3 8
Save project file	3 8
Recently used file	3 9
Recently used project	3 9
Project option	4 0
e. Load table (encryption table)	4 0
e. Save table (encryption table)	4 0
End	4 0
End as save	4 0
Buffer Command	4 1
Display/Edit	4 1
(Print Buffer, Find text, Replace text, buffer for PLD)	4 2
Fill block	4 3
Copy block	4 3
Move block	4 3
Swap data	4 4
Erase	4 5
Fill random data	4 5
Duplicate buffer	4 5
Check sum	4 5
Display of checksum in main display	4 7
Device command	4 7
Device select/Record	4 7
Device select	4 7
(All, Only selected type, Only selected manufacuter)	4 7
Device select/ ID	4 9
Device option	5 0
(Move option, Syllialize, Count, Related file, Special)	5 0
Blank check	6 8
Read	6 8
Verify	6 9
Program	6 9
Erase	6 9

Test	6 9
IC test	6 9
Jam/VME/SVF/...Player	7 0
Device information	7 4
Programmer command	7 4
Search programmer	7 4
Research programmer	7 5
Module options	7 5
Autmatic YES ! (Auto exersise mode)	7 5
Self test	7 7
Self test Plus	7 7
Selftest ISP Connector	7 7
Calibration test	7 7
Option command	7 8
Normal option	7 8
(File option Hex file Option、 Algo、 Sound、 Log file、 Error 、 Programmer、 Remote control、 Save option、 etc)	8 0
Tool bar	
(Main tool bar、 Expand tool bar、 Device options before operation)	8 3
Protect mode	8 4
Save option setting	8 4
Help · command	8 5
Manual	8 5
Search Manual	8 5
Support Device	8 6
Support Programmer	8 6
Device list(Current programmer、 Total programmer、 Cross · reference)	
Create failure report	8 6
Version information	8 6
Warranty	8 7

Before Operation

For Users,

We thank you for your purchase of our M1882.

The guarantee period of this unit is for one year after delivery to you. Even during the guarantee period, we exclude damages as a result of natural disaster, Misoperation, modification or change of this unit by user and wear of the socket adapter from the guarantee.

Also, please note that we are not obliged to refund for a damaged P-ROM of the programmer due to malfunction.

In case of anything unclear to you, please contact Mianto or Minato distributor.

Specifications are subject to change without prior for further improvement.

How to use this manual

This manual explains how to install the control program and how to use your programmer. It is assumed that the user has some experience with PCs and installation of software. Once you have installed the control program we recommend you consult the context sensitive HELP within the control program rather than the printed User's Manual. Revisions are implemented in the context sensitive help before the printed User's Manual.

Display

Function name using control software is displayed by thick character.

File->Load, File->Save, Buffer ->Display•Edit,

Tool button

Load , **Save** , **Edit** , **Select** ,

Function key used on key board

<F1> , **<F5>** , etc.

Is displayed.

Explanation of word

Device	: Item can be programmed user data.
ZIF socket	: DIP socket on Programmer (ZIF : Zero insertion Force is normal name)
Buffer	: Temporarily memory on PC Reading master ROM and Loading file data are temporarily registered. When file is saved, Data of file in buffer is saved on file.
Printer port	: Printer port on PC
USB port	: USB port of PC Port is connected to M1882 programmer by parallel cable.
HEX data format	: One of datafile format can be checked by text viewer of control software.

INTEL.hex

Motrol.mot are also used as normal data format.

Chapter 1

Outline of programmer、Specification、 Installration

Outline of M1882

M1882 is a high speed programming universal programmer which has USB interface, parallel Interface and 48pin pin driver.
It has also ISP(in system programmer) connector. You can also program device on assembled board.

M1882 can be operated almost PC(DOS-V) has USB port or printer port.

M1882 has universal high function pin driver, voltage pin covered with large voltage area.
And high resolution D/A circuit.
It can support many new device will be released in the future.

Control software has a function which is pull down menu, hot key and help.
It is operated easily.

Please check component before you install control software and operate programmer.

In case there is missing item and defective item, please contact our sales office and local distributor.

List of component

(1) M1882 programmer	1set
(2) Electric cable	1pc
(3) USB cable (1.5m)	1pc
(4) Pod for self check of ZIF socket	1pc
(5) Pod for self check of ISP connector	1pc
(6) Flat cable for ISP check	1pc
(7) CD ROM for control software	1pc

Caution: Attached AC cable is used for domestic.

You need to prepare for AC suitable cable in your country.

PC requirements

Minimam PC requirements

•OS	Windows98
•CPU	Pentiam III 500MHz
•RAM	128MB
•Hard Disk	60MB
•Interface	USB1. 1 or Printer port[PP mode]
•CD Drive	CD-ROM Reader

Reommend PC requirements

•OS	WindowsXP
•CPU	Pentium 4 1GHz
•RAM	512MB
•Hard Disk	150MB
•Interface	USB2. 0 or Printer port[ECP、EPP mode]
•CD Drive	CD-ROM Reader

Feature of M1882 programmer

M1882 is universal programmer which is based next generation of USB/LPT-compatible, Windows PC.

This programmer is fast and reliable universal programmer to meet the strong demand of the small manufacturing and developer's community.

M1882 supports all kinds of types and silicon technologies of today and tomorrow programmable devices without family-specific module. You have freedom to choose the optimal device for your design. Using built-in in-circuit serial programming (ISP) connector, the programmer is able to program ISP capable chips in circuit.

M1882 isn't only programmer, but also tester of TTL/CMOS logic ICs and memories. Furthermore, it allows generating user-definable test pattern sequences.

M1882 provides very competitive price coupled with excellent hardware design for reliable programming. Probably best "value for money" programmer in this class.

M1882 provides very fast programming due to high-speed FPGA driven hardware and execution of time-critical routines inside of the programmer. At least fast than competitors in this category, for many chips much faster than most competitors. As a result, when used in production this one-socket-programmer waits for an operator, and not the other way round.

M1882 interfaces with the IBM PC Pentium compatible or higher, portable or desktop personal computers through USB (2.0/1.1) port or any standard parallel (printer) port. Programmer can utilize power of both USB high-speed port and IEEE1284 (ECP/EPP) high-speed parallel port. Support of both USB/LPT port connection gives you the choice to connect the M1882 programmer to any PC, from latest notebook to older desktop without USB port.

M1882 provides a banana jack for ESD wrist straps connection to easy-to-implement the ESD protection control and also other banana jack for earth wire.

M1882 has a FPGA based totally reconfigurable 48 powerful TTL pindrivers, where provide H/L/pull_up/pull_down and read capability for each pin of socket. Advanced pindrivers incorporate high-quality high-speed circuitry to deliver signals without overshoot or ground bounce for all supported devices. Improved pindrivers operate down to 1.8V so you'll be ready to program the full range of today's advanced low-voltage devices.

M1882 performs device insertion test (wrong or backward position) and contact check (poor contact pin-to-socket) before it programs each device. These capabilities, supported by overcurrent protection and signature-byte check help prevent chip damage due to operator error.

Built-in protection circuits eliminate damage of programmer and/or programmed device due environment or operator failure. All the inputs of the M1882 programmer, including the ZIF socket, connection to PC and power supply input, are protected against ESD up to 15kV.

M1882 programmer performs programming verification at the marginal level of supply voltage, which, obviously, improves programming yield, and guarantees long data retention.

Various socket converters are available to handle device in PLCC, SOIC, PSOP, SSOP, TSOP, TSSOP, TQFP, QFN (MLF), SDIP, BGA and other packages.

M1882 programmer is driven by an easy-to-use control program with pull-down menu, hot keys and on-line help. Selecting of device is performed by its class, by manufacturer or simply by typing a fragment of vendor name and/or part number.

Standard device-related commands (read, blank check, program, verify, erase) are boosted by some test functions (insertion test, signature-byte check), and some special functions (autoincrement, production mode - start immediately after insertion of chip into socket).

All known data formats are supported. Automatic file format detection and conversion during load of file.

The rich-featured autoincrement function enables to assign individual serial numbers to each programmed device - or simply increments a serial number, or the function enables to read serial numbers or any programmed device identification signatures from a file.

The software also provides a many information about programmed device. As a special, the drawings of all available packages, explanation of chip labeling (the meaning of prefixes and suffixes at the chips) for each supported chip are provided.

The software provide a full information for ISP implementation: Description of ISP connector pins for currently selected chip, recommended target design around in-circuit programmed chip and other necessary information.

The remote control feature allows being M1882 software flow controlled by other application – either using .BAT file commands or using DLL file. DLL file, examples (C/PAS/VBASIC/.NET) and manual are part of standard software delivery.

Generated by design software which is provided by manufacturer of respective programmable device. Chips are programmer in-ZIF or through ISP connector (IEEE 1149.1 Joint Test Action Group (JTAG) interface).

VME files are interpreted by VME Player. VME file is a compressed binary variation of SVF file and contains high-level IEEE 1149.1 bus operations. VME files are generated by design software which is provided by manufacturer of respective programmable device. Chips are programmer in-ZIF or through ISP connector (IEEE 1149.1 Joint Test Action Group (JTAG) interface).

Multiple devices are possible to program and test via JTAG chain: JTAG chain (ISP-Jam) or JTAG chain (ISP-VME).

Attaching of more M1882 programmers to the same PC (through USB port) is achieved a powerful multiprogramming system, which support as many chips, as are supported by M1882 programmer and without obvious decreasing of programming speed. It is important to know, there is a concurrent multiprogramming - each programmer works independently and each programmer can program different chip, if necessary.

It is important to remember that in most cases new devices require only a software update due to the M1882 is truly universal programmer. With our prompt service you can have new devices can be added to the current list within hours!

Update for programmer

Why is it important to use the latest version of the control program?

- Semiconductor manufacturers continuously introduce new devices with new package types, manufactured by new technologies in order to support the need for flexibility, quality and speed in product design and manufacturing. To keep pace and to keep you up-to-date, we usually implement more than 500 new devices into the control program within a year.
- Furthermore, a typical programmable device undergoes several changes during its lifetime in an effort to maintain or to improve its technical characteristics and process yields. These changes often impact with the programming algorithms, which need to be upgraded (the programming algorithm is a set of instructions that tells the programmer how to program data into a particular target device). Using the newest algorithms in the programming process is the key to obtaining high quality results. In many cases, while the older algorithm will still program the device, they may not provide the level of data retention that would be possible with an optimal algorithm. Failure to not use the most current algorithm can decrease your programming yields (more improper programmed target devices), and may often increase programming times, or even affect the long term reliability of the programmed device.
- At least, we are making mistakes too

Our commitment is to implement support for these new or modified parts before or as soon as possible after their release, so that you can be sure that you are using latest and/or optimal programming algorithms that were created for this new device.

Quick start


Installing the programmer software

Run the installation program from the CD (Setup.exe) and follow the on-screen instructions.

Installing programmer hardware

- connect the USB (or LPT) port of programmer to a USB (or printer) port of PC using supplied cable
- connect the connector of the power supply adapter to the programmer or turn on programmer by switch.

Run the control program

Double click on  After start, control program automatically scans all existing ports and searches for any connected M1882. Program is common for all the M1882, hence program will try to find M1882.

Menu **File** is used for source files manipulation, settings and viewing directory, changes drives, changes start and finish address of buffer for loading and saving files and loading and saving projects.

Menu **Buffer** is used for buffer manipulation, block operation, filling a part of buffer with string, erasing, checksum and of course editing and viewing with other items (find and replace string, printing...).







Menu **Device** is used for a work with selected programmable device: select, read, blank check, program, verify, erase and setting of programming process, serialization and associated file control.

Menu **Programmer** is used for work with programmer.

Menu **Options** is used to view and change various default settings.

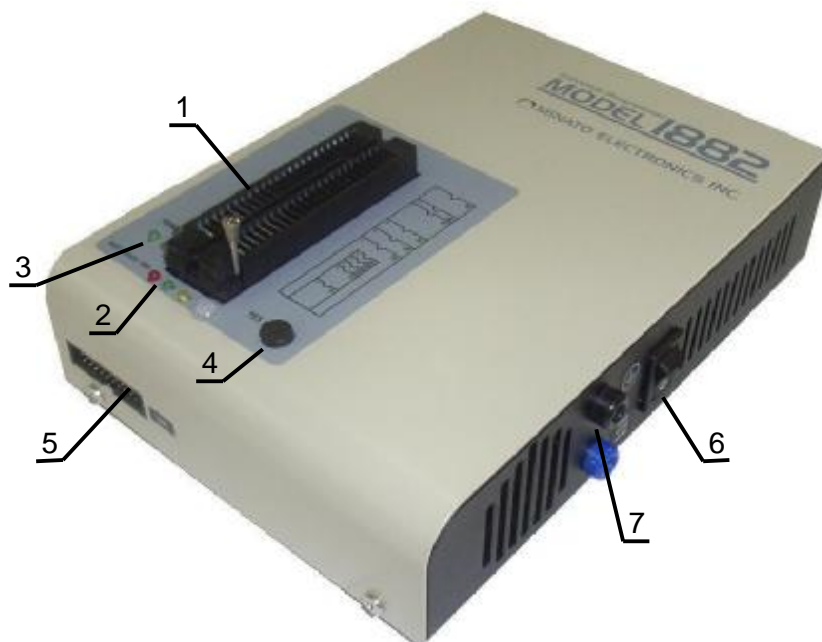
Menu **Help** is used for view supported devices and programmers and information about program version.

Programming a device

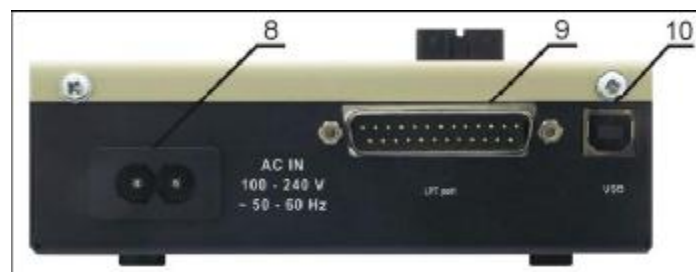
1. select device: click on 
2. load data into buffer:
 - a) from file: click on 
 - b) from device: insert device to ZIF and click on 
3. insert target device to ZIF
4. check, if the device is blank: click on 
5. program device: click on 
6. additional verify of device: click on 

M1882 elements

- 1) 48 pin ZIF socket
- 2) Work result LEDs
- 3) Power/sleep LED
- 4) YES! Button
- 5) ISP connector
- 6) Power switch
- 7) GND connector and connector for ESD wrist strap connection



- 8) Power supply connector
- 9) LPT connector for PC ↔ M1882 communication cable
- 10) USB connector for PC ↔ M1882 communication cable



Connecting M1882 to the PC

Using USB port

In this case, order of connecting USB cable and power supply to programmer is irrelevant.

Using LPT port

Switch off PC and programmer. Insert the communication cable included with your M1882 programmer package to a free printer port on your PC. If your computer is equipped with only one printer port, substitute the programmer cable for the printer cable. Connect the opposite cable end to the programmer. Screw on both connectors to counter-connectors. This is very important. It may be uncomfortable to switch between printer cable and programmer cable, though it is not recommended to operate the M1882 programmer through a mechanical printer switch. Use of an electronic printer switch is impossible. But you can install a second multi-I/O in your computer, thus obtaining a supplementary printer port, says LPT2. So your printer may remain on LPT1 while the programmer on LPT2.

Switch on the PC.

Connect the mains connector of the power supply to a mains plug, and then connect the mini-DIN connector to the programmer's connector labeled "15VDC". At this time all 'work result' LEDs (and 'POWER' LED) light up successive and then switch off. Once the POWER LED lights with low brightness then the M1882 programmer is ready to run.

Next run the control program for M1882.

Caution! If you don't want to switch off your PC when connecting the M1882, proceed as follows:

- When connecting the programmer to the PC: FIRST insert the communications cable and THEN the power-supply connector.
- When disconnecting the programmer from the PC: FIRST disconnect the power-supply connector and THEN the communication cable.

From M1882's point of view the connecting and disconnecting sequence is irrelevant. Protection circuits on all programmer inputs keep it safe. But think of your PC please.

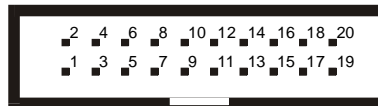
Problems related to the M1882 ↔ PC interconnection, and their removing

If you have any problems with M1882 ↔ PC interconnection, see section Common notes please.

In-system serial programming by M1882

For general definition, recommendation and direction about ISP see section Common notes / ISP please.

Description of M1882 ISP connector



Front view at ISP connector of programmer.

Specification of ISP connector pins depends on the device, which you want to program. You can find it in the control SW for programmer (M1882), menu Device / Device Info (Ctrl+F1). Be aware, the ISP programming way of respective device must be selected. It is indicated by (ISP) suffix after name of selected device.

Note: Pin no. 1 is signed by triangle scratch on ISP cable connectors.

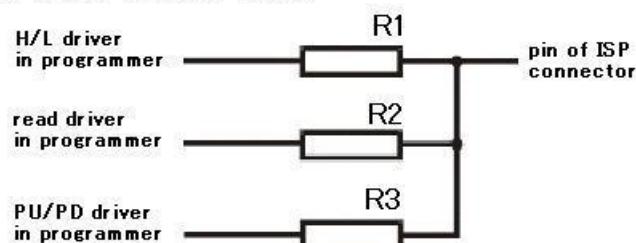


M1882 ISP cable

Warnings:

- When you use M1882 as ISP programmer, don't insert device to ZIF socket.
- When you program devices in ZIF socket, don't insert ISP cable to ISP connector.
- Use only attached ISP cable. When you use other ISP cable (other material, length...), programming may occur unreliable.
- M1882 can supply programmed device (pin 1 of ISP connector) and target system (pin 5 of ISP connector) with limitation (see Technical specification / ISP connector).
- M1882 apply programming voltage to target device and checks his value (target system can modify programming voltage). If the programming voltage is different as expected, no action with target device will be executed.

Note: *H / L / Read M1882 driver*



Selftest and calibration check

If you feel that your programmer does not react according to your expectation, please run the programmer selftest using Diagnostic POD, enclosed with the standard delivery package. See instructions for selftest in the Programmer/Selftest plus menu of M1882.

For optimal results with programmer we recommend you also undertake every 6 months an extended test to check the calibration using 48 Pins Calibration test POD, Type I (optional accessories, ord.no. 70-0438). See instructions for selftest in the Programmer/Calibration test menu of M1882.



Technical specification

HARDWARE

Base unit, DACs

- USB 2.0 high-speed compatible port, up to 480 Mbit/s transfer rate
- FPGA based IEEE 1284 slave printer port, up to 1MB/s transfer rate
- on-board intelligence: powerful microprocessor and FPGA based state machine
- three D/A converters for VCCP, VPP1, and VPP2, controllable rise and fall time
- VCCP range 0..8V/1A
- VPP1, VPP2 range 0..26V/1A
- selftest capability
- protection against surge and ESD on power supply input, parallel port connection
- banana jack for ESD wrist straps connection
- banana jack for connection to ground

Socket, pindriver

- 48-pin DIL ZIF (Zero Insertion Force) socket accepts both 300/600 mil devices up to 48-pin
- pindrivers: 48 universal
- VCCP / VPP1 / VPP2 can be connected to each pin
- perfect ground for each pin
- FPGA based TTL driver provides H, L, CLK, pull-up, pull-down on all pindriver pins
- analog pindriver output level selectable from 1.8 V up to 26V
- current limitation, overcurrent shutdown, power failure shutdown
- ESD protection on each pin of socket (IEC1000-4-2: 15kV air, 8kV contact)
- continuity test: each pin is tested before every programming operation

ISP connector

- 20-pin male type with missinsertion lock
- 6 TTL pindrivers, provides H, L, CLK, pull-up, pull-down; level H selectable from 1.8V up to 5V to handle all (low-voltage including) devices.
- 1x VCCP voltage (range 2V..7V/100mA)
- programmed chip voltage (VCCP) with both source/sink capability and voltage sense
- and 1x VPP voltage (range 2V..25V/50mA)
- target system supply voltage (range 2V..6V/250mA)
- ESD protection on each pin of ISP connector (IEC1000-4-2: 15kV air, 8kV contact)
- two output signals, which indicate state of work result = LED OK and LED Error (active level: min 1.8V)

- input signal, switch YES! equivalent (active level: max 0.8V)

DEVICE SUPPORT

Programmer, in ZIF socket

- EPROM: NMOS/CMOS, 2708*, 27xxx and 27Cxxx series, with 8/16 bit data width, full support for LV series
- EEPROM: NMOS/CMOS, 28xxx, 28Cxxx, 27EExxx series, with 8/16 bit data width
- Flash EPROM: 28Fxxx, 29Cxxx, 29Fxxx, 29BVxxx, 29LVxxx, 29Wxxx, 49Fxxx series, from 256Kbit to 1Gbit, with 8/16 bit data width, full support for LV series
- Serial E(E)PROM: 24Cxxx, 24Fxxx, 25Cxxx, 45Dxxx, 59Cxxx, 25Fxxx, 25Pxxx, 85xxx, 93Cxxx, NVM3060, MDAxxx series, full support for LV series
- Configuration (EE)PROM: XCFxxx, XC17xxxx, XC18Vxxx, EPCxxx, AT17xxx, 37LVxx
- 1-Wire E(E)PROM: DS1xxx, DS2xxx
- PROM: AMD, Harris, National, Philips/Sigmetics, Tesla, TI
- NV RAM: Dallas DSxxx, SGS/Inmos MKxxx, SIMTEK STKxxx, XICOR 2xxx, ZMD U63x series
- PLD: Altera: MAX 3000A, MAX 7000A, MAX 7000B, MAX 7000S, MAX7000AE, MAX II
- PLD: Lattice: ispGAL22V10x, ispLSI1xxx, ispLSI1xxxEA, ispLSI2xxx, ispLSI2xxxA, ispLSI2xxxE, ispLSI2xxxV, ispLSI2xxxVE, ispLSI2xxxVL, LC4xxxB/C/V/ZC, M4-xx/xx, M4A3-xx/xx, M4A5-xx/xx, M4LV-xx/xx
- PLD: Xilinx: XC9500, XC9500XL, XC9500XV, CoolRunner XPLA3, CoolRunner-II
- other PLD: SPLD/CPLD series: AMI, Atmel, AMD-Vantis, Gould, Cypress, ICT, Lattice, NS, Philips, STM, VLSI, TI
- Microcontrollers 48 series: 87x41, 87x42, 87x48, 87x49, 87x50 series
- Microcontrollers 51 series: 87xx, 87Cxxx, 87LVxx, 89Cxxx, 89Sxxx, 89LVxxx, all manufacturers, Philips LPC series
- Microcontrollers Intel 196 series: 87C196 KB/KC/KD/KT/KR/...
- Microcontrollers Atmel AVR: AT90Sxxxx, ATtiny, ATmega series
- Microcontrollers Cypress: CY7Cxxxxx, CY8Cxxxxx
- Microcontrollers ELAN: EM78Pxxx
- Microcontrollers MDT 1xxx and 2xxx series
- Microcontrollers Microchip PICmicro: PIC10xxx, PIC12xxx, PIC16xxx, PIC17Cxxx, PIC18xxx, PIC24xxx, dsPIC series
- Microcontrollers Motorola (Freescale): 68HC05, 68HC08, 68HC11, HCS08, HCS12 series
- Microcontrollers Myson MTV2xx, 3xx, 4xx and 5xx series
- Microcontrollers National: COP8xxx series
- Microcontrollers NEC: uPD78Fxxx series
- Microcontrollers Novatek: NT68xxx series
- Microcontrollers Scenix (Ubicom): SXxxx series

- Microcontrollers SGS-Thomson: ST6xx, ST7xx, ST10xx, STR7xx series
- Microcontrollers TI: MSP430 and MSC121x series
- Microcontrollers ZILOG: Z86/Z89xxx and Z8xxx series
- Microcontrollers other: EM Microelectronic, Fujitsu, Goal Semiconductor, Hitachi, Holtek, Princeton, Macronix, Winbond, Infineon(Siemens), Samsung, Toshiba, ...

Programmer, through ISP connector

- Serial E(E)PROM: IIC series, MW series, SPI series, KEELOQ series, serial data Flash, PLD configuration memories
- Microcontrollers Atmel: AT89Sxxx, AT90Sxxxx, ATtiny, ATmega series
- Microcontrollers Cypress: CY8C2xxxx
- Microcontrollers Elan: EM78Pxxx, EM6xxx series
- Microcontrollers EM Microelectronic: 4 and 8 bit series
- Microcontrollers Microchip PICmicro: PIC10xxx, PIC12xxx, PIC16xxx, PIC17xxx, PIC18xxx, PIC24xxx, dsPIC series
- Microcontrollers Motorola/Freescale: HC11 series, HC908 series (both 5-wire, All-wire), HCS08, HCS12
- Microcontrollers NEC: uPD7xxx series
- Microcontrollers Philips: LPC2xxx series, LPC series, 89xxx series
- Microcontrollers Scenix (Ubicom): SXxxx series
- Microcontrollers TI: MSP430 (both JTAG and BSL series), MSC12xxx series
- PLD: Lattice: ispGAL22xV10x, ispLSI1xxxEA, ispLSI2xxxE, ispLSI2xxxV, ispLSI2xxxVE, ispLSI2xxxVL, M4-xx/xx, M4LV-xx/xx, M4A3-xx/xx, M4A5-xx/xx, LC4xxxB/C/V/ZC
- Various PLD (also by JAM player/JTAG support):
Altera: MAX 3000A, MAX 7000A, MAX 7000B, MAX 7000S, MAX 9000, MAX

II

Xilinx: XC9500, XC9500XL, XC9500XV, CoolRunner XPLA3, CoolRunner-II

Notes:

- Devices marked * are obsolete, programming with additional module
- For all supported devices see actual Device list on <http://www.minato.co.jp/>

I.C. Tester

- TTL type: 54,74 S/LS/ALS/H/HC/HCT series
- CMOS type: 4000, 4500 series
- static RAM: 6116.. 624000
- user definable test pattern generation

Package support

- support all devices in DIP with default socket
- package support includes DIP, SDIP, PLCC, JLCC, SOIC, SOP, PSOP, SSOP, TSOP, TSOPII, TSSOP, QFP, PQFP, TQFP, VQFP, QFN (MLF), SON, BGA, EBGA, FBGA, VFBGA, UBGA, FTBGA, LAP, CSP, SCSP etc.
- support devices in non-DIP packages up to 48 pins with universal adapters
- programmer is compatible with third-party adapters for non-DIP support

Programming speed

Device	Size [bits]	Operation	Time
M50FW080 (parallel Flash)	100000Hx8 (8 Mega)	programming and verify	22 sec
MX28F640C3BT (parallel Flash)	400000Hx16 (64 Mega)	programming and verify	57 sec
K9F1G08U0M (parallel NAND Flash)	8400000Hx8 (1 Giga)	programming and verify	239 sec
AT45D081 (serial Flash)	108000Hx8 (16 Mega)	programming and verify	36 sec
AT89C51RD2 (microcontroller)	10000Hx8	programming and verify	15 sec
PIC18LF452 (microcontroller)	4000Hx16	programming and verify	4 sec

Conditions: P4, 2,4GHz, 512 MB RAM, USB 2.0 HS, Windows XP

Device operations

standard:

- intelligent device selection by device type, manufacturer or typed fragment of part name
- automatic ID-based selection of EPROM/Flash EPROM
- blank check, read, verify
- program
- erase
- configuration and security bit program
- illegal bit test
- checksum
- interpret the Jam Standard Test and Programming Language (STAPL), JEDEC standard JESD-71
- interpret the VME files compressed binary variation of SVF files

security

insertion test, reverse insertion check

contact check

ID byte check

special

production mode (automatic start immediately after device insertion)

lot of serialization modes (more type of incremental modes, from-file mode, custom generator mode)

statistic

count-down mode

Buffer operations

- view/edit, find/replace
- fill/copy, move, byte swap, word/dword split
- checksum (byte, word)
- print

File load/save

- no download time because programmer is PC controlled
- automatic file type identification

Supported file formats

- unformatted (raw) binary
- HEX: Intel, Intel EXT, Motorola S-record, MOS, Exormax, Tektronix,

ASCII-SPACE-HEX,, ASCII HEX

- Altera POF, JEDEC (ver. 3.0.A), eg. from ABEL, CUPL, PALASM, TANGO PLD, OrCAD PLD, PLD Designer ISDATA, etc.
- JAM (JEDEC STAPL Format), JBC (Jam STAPL Byte Code), STAPL (STAPL File) JEDEC standard JESD-71
- VME (ispVME file VME2.0/VME3.0)

GENERAL

- operating voltage 110-250V AC
- power consumption max. 20W active, about 2W sleep
- dimensions 197x140x56 mm (7.7x5.5x2.2 inch)
- weight 1.1kg (2.5 lb)
- temperature 5°C ÷ 40°C (41°F ÷ 104°F)
- humidity 20%..80%, non condensing

Installation

The programmer package contains a CD with the control program, useful utilities and additional information. The permission to freely copy the content of the CD is granted in order to demonstrate how MINATO ELECTRONICS INC.'s programmers work.

For programmers connected through USB (LPT) port, control program requires correctly installed USB driver

We recommend install software before connecting programmer to PC to avoid unwanted complication during installation.

Software setup

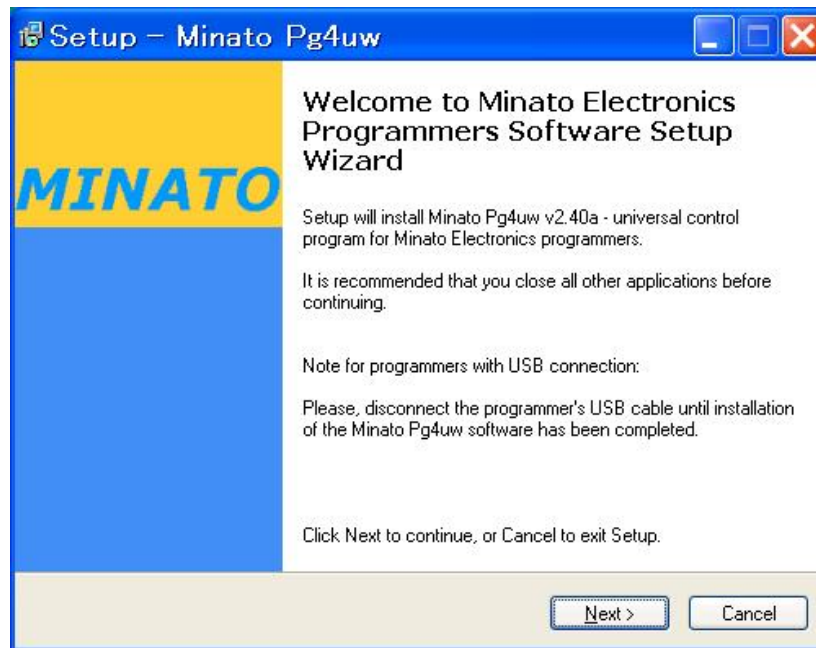
Insert delivered CD to your CD drive and install program starts automatically (if not, run setup.exe). Install program will guide you through the installation process and will do all the necessary steps before you can first run the control program.

Set up 1



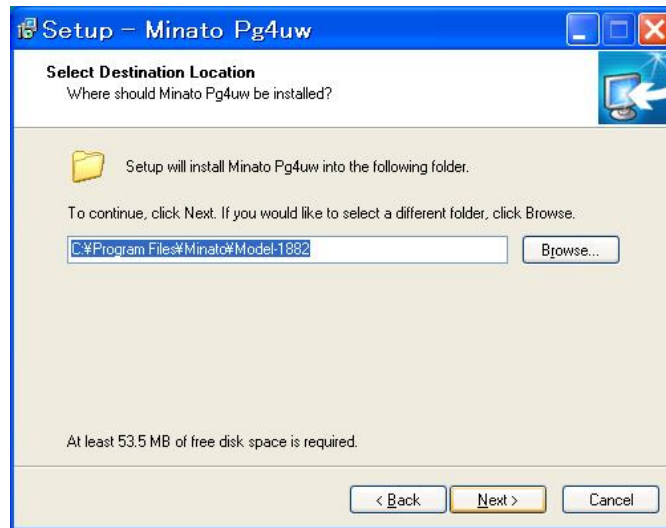
Click "Software installation PROGRAMMERS" button.

Set up 2



Click "Next" button.

Set up 3



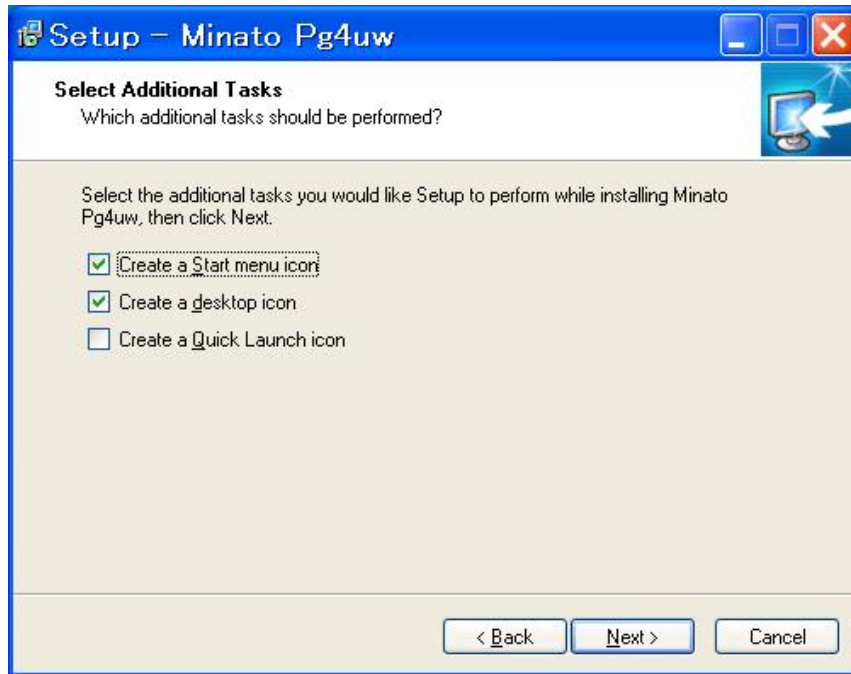
For change default folder click on “Browse” button, select the destination folder.
Then click on “Next” button

Set up 4



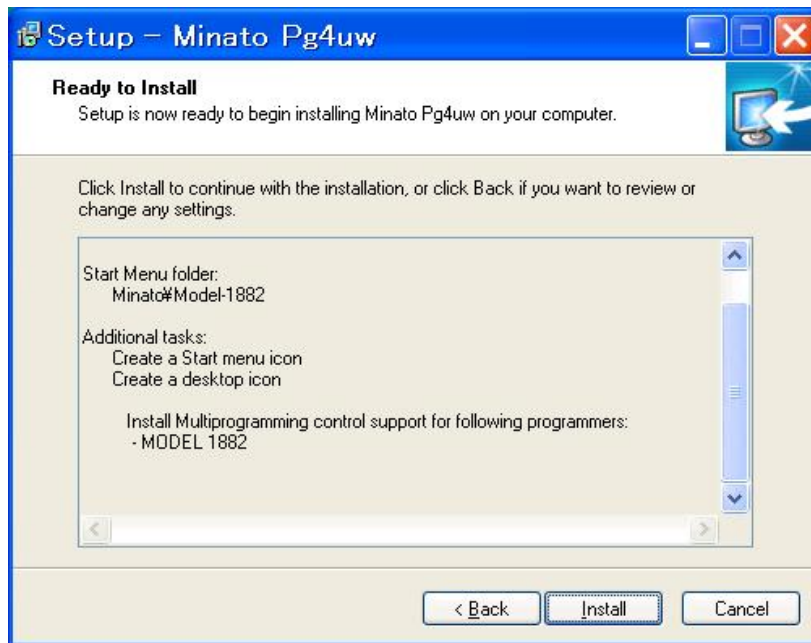
Select the destination folder.
Then click on “Next” button

Set up 5



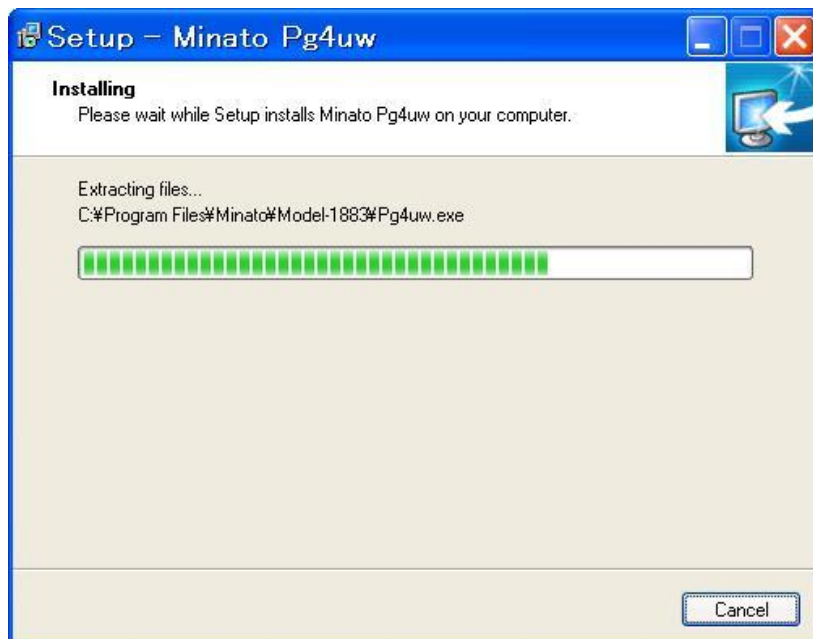
Check if “Install Multiprogramming control support” is selected. Change default setting, if you want. Then click on “Next” button

Set up 6



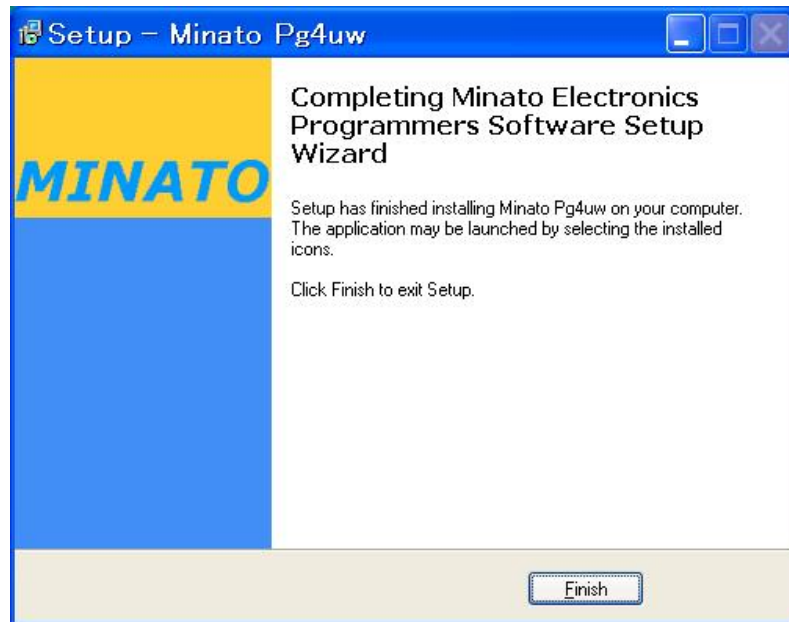
Check your setting and then click on “Install” button

Set up 7



Installation process will start.

Set up 8



If this message is expressed, Installation is finished.
Please click on Finish button.

Hardware setup

When the programmer is connected to USB port before control program was installed, Windows will detect new hardware and ask user to select driver installation method: automatically or manually. To detect programmer correctly, control program installation CD must be inserted to computer's CD-ROM drive and following steps have to be done:

Step 1.

Directly connect USB (LPT) cable to type B USB (LPT) port on programmer.

Step 2.

Directly connect USB (LPT) cable to type A USB2.0 (LPT) port on PC (high-speed recommended).

Step 3.

Connect connectors of power supply cable to appropriate connectors on programmer and wall plug.

Step 4.

Turn on programmer. At this time all 'work result' LEDs light up successive and then LEDs switch off.

For LPT connected programmer you may start work with your programmer now. For USB connected programmer continue with next step.

Step 5.

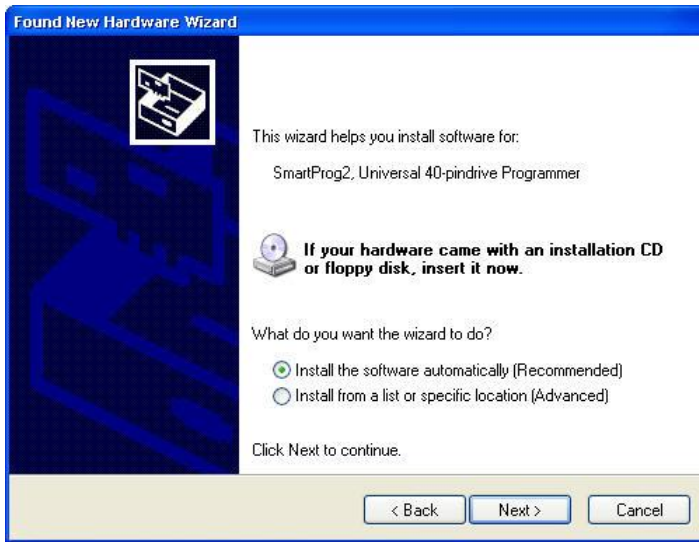
Windows will start with "Found new hardware wizard".

For Windows XP, Service Pack 2 users only:



Select "No, not this time" and then click on "Next" button.

For all:



Select “Install the software automatically” and then click on “Next” button.

Step 6.



Click on “Continue Anyway” button.

Step 7.



Click “Finish” button to finish setup.

Step 8.

“Found new hardware wizard” will launch for each programmer one time. Hardware setup will be continued with Step 5.

Note: If a different USB port on the PC is used for the next connection of programmer, “Found new hardware wizard” will launch again and install new USB drivers.

Chapter 2

M1882 Control Software Opetartion Manual

M1882 Control Software

Execute M1882 control software

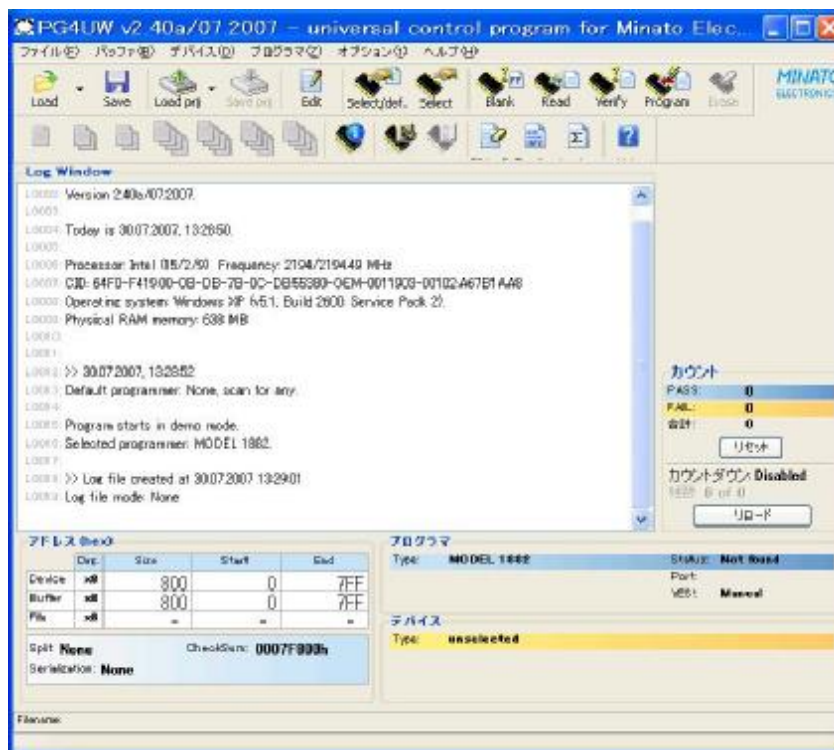
Click icon  on display.

After start, control program automatically scan ports and search for the connected M1882 programmer.

Notes: When it is started, program is checked for its integrity. Then the program display a standard user menu and waits for your instructions.

If the control program cannot communicate with the programmer, an error message appears on the screen, including error code and description of possible reasons (disconnected programmer, bad connection, power supply failure, incompatible printer port). Eliminate the error source and press any key. If error condition still exists, the program resumes its operation in the demo mode and access to the programmer is not possible. If you cannot find the cause of the error, follow the instructions in Troubleshooting section. In addition, the control program checks communication with programmer prior to any operation with the programmed device.

Explanation of main display



Toolbars

Under main menu are placed toolbars with button shortcuts of frequently used menu commands. Toolbars are optional and can be turned off by menu command Options / View.

Log window

Contains the flow-control progress information about almost every operation made in control software.

Operation can be:

- starting
- programmer search
- file/project load/save
- selection of device
- device operations (device read, blank check, programming, ...)
- remote control application connection and disconnection
- and other

Content of Log window can be saved to file concurrently while information is written to Log window. This option can be set by menu Options / General options (and tab Log file in dialog General options).

Panel Addresses

Panel Addresses contains information about actual address ranges of currently selected device, loaded file and buffer start-end address settings. Some devices allows modifying default device and buffer address ranges by menu command

Device / Device options / Operation options.

Panel Addresses also contains some advanced information about current status of Split, Serialization and buffer checksum. For more information about each of the options, please look at:

- Split - menu Device / Device options / Operation options
- Serialization - menu Device / Device options / Serialization
- Checksum - menu Buffer / Checksum at section Checksum displayed in main window

Panel Programmer

Contains information about currently selected programmer.

The information includes

- programmer type
- port via programmer is connected to computer
- programmer status, can be one of following
- Ready - programmer is connected, successfully found and ready to work
- Not found - programmer is not found
- Demo - when user selects option (button) Demo in dialog Find programmer
- YES! mode - some types of programmers allow to use special modes of starting next device operation in one of following ways -
- manually by control program dialog Repeat
- manually by button YES! placed directly on programmer
- automatically - programmer automatically detects device removing and insertion of new device

Panel Device

Contains information about currently selected device.

The information includes

- device name (type) and manufacturer
- device adapter needed to use with currently selected programmer
- reference to detailed Device info dialog, available also by menu Device / Device info
- reference to Advanced device options - this is available for some types of devices only

Panel Statistics

Contains statistics information about currently selected device.

The information includes

- number of successful, failure and total device operations
- count-down status indicating number of remaining devices

Statistics and count-down options are available by menu command Device / Device options / Statistics or by mouse right click on panel Statistics and select item Statistics from popup menu

List of hot keys

<F1>	Help	Calls Help
<F2>	Save	Save file
<F3>	Load	Load a file into the buffer
<F4>	Edit	Viewing/editing of buffer
<F5>	Select/default	Target-device selection from 10 last selected devices list
<Alt+F5>	Select/manual	Target-device selection by typing device/vendor name
<F6>	Blank	Blank check
<F7>	Read	Reads device's content into the buffer
<F8>	Verify	Compares contents of the target device with the buffer
<F9>	Program	Programs target device
<Alt+Q>	Exit without save	Terminates the M1882
<Alt+X>	Exit and save	Terminates the M1882 and saving settings too
<Ctrl+F1>		Displays additional information about current device
<Ctrl+F2>	Erase	Fill's the buffer with a given value
<Ctrl+Shift+F2>		Fill's the buffer with random values.

File

Menu File is used for source files manipulation, settings and viewing directory, changes drives, changes start and finish address of buffer for loading and saving files by binary, MOTOROLA, MOS Technology, Intel (extended) HEX, Tektronix, ASCII space, JEDEC, and POF format. The menu commands for loading and saving projects are located in this submenu too.

File / Load

Analyse file format and loads the data from specified file to the buffer. You can choose the format desired (binary, MOTOROLA, MOS Technology, Tektronix, Intel (extended) HEX, ASCII space, JEDEC and POF). The control program stores a last valid mask for file listing. You can save the mask into the config. file by command Options / Save options.

Checking the check box Automatic file format recognition tells program to detect file format automatically. When program can't detect file format from one of supported formats, the binary file format is assumed.

When the check box Automatic file format recognition is unchecked program allows user to manually select wished file format from list of available file formats on panel Selected file format. Default set is from Options / General options in panel Load file format at tab File options.

Attention: Program doesn't know recognize files in ASCII Hex format automatically, it recognizes them as binary. So download files in ASCII Hex format with disabled option for automatic file format recognition.

Checking the check box Buffer offset for loading tells the program to set buffer offset for all data addresses, which will be written to buffer. This feature is useful for binary and all HEX formats. Using this one-shot setting disables current setting of native offset in menu Options / General options in panel Negative offset for loading at tab Hex file options.

Checking the check box Erase buffer before loading tells the program to erase all buffer data using entered Erase value. Buffer erase is performed immediately before reading file content to buffer and it is functional for binary and all HEX file formats. Using this one-shot setting disables current setting of Erase buffer before loading option in menu Options / General options at tab Hex file options.

If the checkbox Swap bytes is displayed, the user can activate function of swapping bytes within 16bit words (or 2-byte words) during reading of file. This feature is useful especially when loading files with Motorola representation of byte order in file (big endian). Standard load file is using little endian byte order.

Note: Big-endian and little-endian are terms that describe the order in which a sequence of bytes are stored in computer memory. Big-endian is an order in which the "big end" (most significant value in the sequence) is stored first (at the lowest storage address). Little-endian is an order in which the "little end" (least significant value in the sequence) is stored first. For example, in a big-endian computer, the two bytes required for the hexadecimal number 4F52 would be stored as 4F52H in storage address 1000H as: 4FH is stored at storage address 1000H, and 52H will be at address 1001H. In a little-endian system, it would be stored as 524FH (52H at address 1000H, and 4FH at address 1001H).

Number 4F52H is stored in memory:

Address	Big endian system	Little endian system
1000H	4FH	52H
1001H	52H	4FH

The reserved key <F3> will bring out this menu from any menu and any time.

List of file format codes and error codes

There can occur some errors during file download in some of supported formats. The error is written to LOG window in face "Warning: error #xyy in line rrr", xx is file format code, y is error code and rrr is line number in decimal.

File format codes:

- #00y - binary
- #10y - ASCII Space
- #20y - Tektronix
- #30y - Extended tektronix
- #40y - Motorola
- #50y - MOS Technology
- #60y - Intel HEX

Load file error codes:

- #xx1 - bad first character - header
- #xx2 - bad character in current line
- #xx3 - bad CRC
- #xx4 - bad read address
- #xx5 - bad length of current line
- #xx6 - too big negative offset
- #xx7 - address is out of buffer range
- #xx8 - bad type of selected file format
- #xx9 - the file wasn't loaded all

File / Save

Saves data in the buffer, which has been created, modified, or read from a device onto a specified disk. The file format of saved file can be chosen from supported formats list box. There can be also entered the Buffer start and Buffer end addresses which exactly specify part of buffer to save to file. Supported file formats now are binary, MOTOROLA, MOS Technology, Tektronix, Intel (extended) HEX, ASCII space, JEDEC and POF.

If the checkbox Swap bytes is displayed, the user can activate function of swapping bytes within 16bit words (or 2-byte words) during writing to file. This feature is useful especially when saving files with Motorola representation of byte order in file (big endian). Standard save file operation is using little endian byte order.

The reserved key <F2> will bring out this menu from any menu and any time.

File / Load project

This option is used for loading project file, which contains device configuration buffer data saved and user interface configuration.

The standard dialog Load project contains additional window - Project description - placed at the bottom of dialog. This window is for displaying information about currently selected project file in dialog Load project.

Project information consists of:

- manufacturer and name of the first device selected in the project
- date and time of project creation
- user written description of project (it can be arbitrary text, usually author of project and some notes)

Note: for projects with serialization turned on

Serialization is read from project file by following procedure:

1. Serialization settings from project are accepted
2. Additional serialization file search is performed. If the file is found it will be read and serialization settings from the additional file will be accepted. Additional serialization file is always associated to the specific project file. When additional serialization file settings are accepted, project serialization settings are ignored.

Name of additional serialization file is derived from project file name by adding extension ".sn" to project file's name.

Additional serialization file is always placed to the directory "serialization¥" into the control program's directory.

Example:

Project file name: my_work.prj

Control program's directory: c:¥Program Files¥Programmer¥

The additional serialization file will be:

c:¥Program Files¥Programmer¥serialization¥my_work.prj.sn

Additional serialization file is created and refreshed after successful device program operation. The only requirement for creating additional serialization file is load project with serialization turned on.

Command File / Save project deletes additional serialization file, if the file exists, associated with currently saved project.

File / Save project

This option is used for saving project file, which contains settings of device configuration and buffer data saved. Data saved to project file can be restored anytime by menu command File / Load project.

The dialog Save project contains three additional windows in Project description panel placed at the bottom of dialog Save project. The windows are for displaying information about currently selected project file in dialog Save project and information about current project, which has to be saved. Dialog Save project contains also additional button with picture of key displayed. Clicking on this button password dialog appears which can be used to save project with password. Projects with password are special projects also called Protected mode projects. For more detailed information about project passwords see Options / Protected mode.

Project information consists of:

- manufacturer and name of the first device selected in the project
- date and time of project creation
- user written description of project (it can be arbitrary text, usually author of project and some notes)

The first (upper) window contains information about currently selected project file in dialog Save project.

The second (middle) windows displays information about actual program configuration including currently selected device, active programmer, date, time. These actual program settings are used for creation of project description header.

The third (bottom) window is user editable and contains project description (arbitrary text), which usually consists of project author and some notes.

File / Reload file

Choose this option to reload a recently used file.

When you use a file, it is added to the Reload file list. Files are listed in order depending on time of use of them. Lastly used files are listed before files used far off.

To Reload a file:

1. From the File menu, choose Reload file.
2. List of lastly used files is displayed. Click the file you want to reload.

Note: When reloading a file the file format is used, by which the file was lastly loaded/saved.

File / Reload project

Choose this option to reload a recently used project.

When you use a project, it is added to the Reload project list. Projects are listed in order depending on time of use of them. Lastly used projects are listed before projects used far off.

To Reload a project:

1. From the File menu, choose Reload project.
2. List of lastly used projects is displayed. Click the project you want to reload.

File / Project options

This option is used for display/edit project options of actually loaded project. Project options means basic description of project including following project data:

- device name and manufacturer
- project creation date
- user defined project description (arbitrary text), e.g. project author and other text data for more detailed project description

User can directly edit user defined project description only. Device name, manufacturer, project date and program version are generated automatically by program.

File / Load encryption table

This command loads the data from binary file from disk and it saves them into the part of memory, reserved for an encryption (security) table.

File / Save encryption table

This command writes the content of the memory's part, reserved for an encryption table, into the file on the disk as a binary data.

File / Exit without save

The command deallocates heap, cancels buffer on disk (if exists) and returns back to the operation system.

File / Exit and save

The command deallocates heap, cancels buffer on the disk (if exists), saves current setting of recently selected devices to disk and returns back to the operation system.

Buffer

Menu Buffer is used for buffer manipulation, block operation, filling a part of buffer with string, erasing, checksum and of course editing and viewing with other items (find and replace string, printing...).

Buffer / View/Edit

This command is used for view (view mode) or edit (edit mode) data in buffer (for viewing in DUMP mode only). Use arrow keys for select the object for edit. Edited data are signified by colour.

You can use <F4> hot key also.

View/Edit Buffer

- F1** display help of actual window
- F2** fill block causes filling selected block of buffer by requested hex (or ASCII) string. Sets start and end block for filling and requested hex or ASCII string.
- Ctrl+F2** erase buffer with specified blank value
- Ctrl+Shift+F2** fill buffer with random data
- F3** copy block is used to copy specified block of data in current buffer on new address. Target address needn't be out from source block addresses.
- F4** move block is used to move specified block of data in current buffer on new address. Target address needn't be out from source block addresses. Source address block (or part) will be filled by topical blank character.
- F5** swap bytes command swaps a high- and low- order of byte pairs in current buffer block. This block must started on even address and must have an even number of bytes. If these conditions do not fulfil, the program modifies addresses itself (start address is moved on lower even address and/or end address is moved on higher odd address).
- F6** print buffer
- F7** find string (max. length 16 ASCII characters)
- F8** find and replace string (max. 16 ASCII chars.)
- F9** change current address
- F10** change mode view / edit
- F11** switch the mode of buffer data view between 8 bit and 16 bit view. It can be also do by mouse clicking on the button to the right of View/Edit mode buffer indicator. This button indicates actual data view mode (8 bit or 16 bit), too.
- F12** checksum dialog allows to count checksum of selected block of buffer change mode view / edit
- Arrow keys** move cursor up, down, right and left
- Home/End** jump on start / end current line
- PgUp/PgDn** jump on previous / next page
- Ctrl+PgUp/PgDn** jump on start / end current page

Ctrl+Home/End	jump on start / end current device
Shift+Home/End	jump on start / end current buffer
Backspace	move cursor one position left (back)

Note: characters 20H - FFH (mode ASCII) and numbers 0..9, A..F (mode HEX) immediately changes content of edit area.

Warning: Editing of ASCII characters for word devices is disabled.

Print buffer

This command allows write selected part of buffer to printer or to file. Program uses at it an external text editor in which selected block of buffer is displayed and can be printed or saved to file, too. By default is set simple text editor Notepad.exe, which is standard part of all versions of Windows.

In Print buffer dialog are following options:

Block start

Defines start address of selected block in buffer.

Block end

Defines end address of selected block in buffer.

External editor

This item defines path and name of external program, which has to be used as text viewer for selected block of buffer. By default is set simple text editor Notepad.exe, which is standard part of all versions of Windows. User can define any text editor for example Wordpad.exe, which is able to work with large text files. In user defined text editor user can print or save to file selected block of buffer.

The external editor path and name is saved automatically to disk.

Find dialog box

Enter the search string to Find to text input box and choose <Find> to begin the search or choose <Cancel> to forget it.

Direction box specifies which way you want to search, starting from the current cursor position (In edit mode). Forward (from the current position or start of buffer to the end of the buffer) is the default. Backward searches toward the beginning. In view mode searches all buffer.

Origin specifies where the search should start.

Find & Replace dialog box

Enter the search string in the Text to find string input box and enter the replacement string in the Replace with input box.

In Options box you can select prompt on replace: if program finds instance you will be asked before program change it.

Origin specifies where the search should start.

Direction box specifies which way you want to search, starting from the current cursor position (In edit mode). Forward (from the current position or start of buffer to the end of the buffer) is the default. Backward searches toward the beginning. In view mode searches all buffer.

Press <Esc> or click Cancel button to close dialog window.

By pressing Replace button the dialog box is closed and a Question window is displayed. This window contains following choices:

Yes	replaces found item and finds next
No	finds next item without replacing current one
Replace All	replaces all found items
Abort search	aborts this command

View/Edit buffer for PLD

Ctrl+F2	erase buffer with specified blank value
Ctrl+Shift+F2	fill buffer with random data
F9	go to address...
F10	change mode view / edit
F11	switch the mode of buffer data view between 1 bit and 8 bit view. It can be also do by mouse clicking on the button to the right of View/Edit mode buffer indicator. This button indicates actual data view mode (1 bit or 8 bit), too.
Arrow keys	move cursor up, down, right and left
Home/End	jump on start / end current line
PgUp/PgDn	jump on previous / next page
Ctrl+PgUp/PgDn	jump on start / end current page
Ctrl+Home/End	jump on start / end edit area
Backspace	move cursor one position left (back)

Note: Characters 0 and 1 immediately changes content of edit area.

Buffer / Fill block

Selecting this command causes filling selected block of buffer by requested hex (or ASCII) string. Sets start and end block for filling and requested hex or ASCII string.

Buffer / Copy block

This command is used to copy specified block of data in current buffer on new address. Target address needn't be out from source block addresses.

Buffer / Move block

This command is used to move specified block of data in current buffer on new address. Target address needn't be out from source block addresses. Source address block (or part) will be filled by topical blank character.

Buffer / Swap block

This command swaps a high- and low- order of byte pairs, foursomes or nibbles inside bytes depending on swap mode selected by user. Swap operation is performed on buffer block specified by Start and End addresses. This block must start on even address and must have an even number of bytes. If the conditions do not fulfill, the program modifies addresses itself (start address is moved on lower even address and/or end address is moved on higher odd address).

Following swap modes are available, user can select from:

1. Swap 2-bytes inside 16-bit words swap of byte pairs inside 16-bit words.
2. Swap 4-bytes inside 32-bit words swap of byte foursomes inside 32-bit words.
3. Swap nibbles inside bytes swap of high- and low- nibbles inside each byte.

Examples of swap operation in buffer:

Swap bytes operation from Start address 0 to End address N modifies data in buffer by following tables:

Address	Original Data	Swap 2-bytes inside 16-bit words	Swap 4-bytes inside 32-bit words	Swap nibbles inside bytes
0000h	b0	b1	b3	b0m
0001h	b1	b0	b2	b1m
0002h	b2	b3	b1	b2m
0003h	b3	b2	b0	b3m
0004h	b4	b5	b7	b4m
0005h	b5	b4	b6	b5m
0006h	b6	b7	b5	b6m
0007h	b7	b6	b4	b7m

b0, b1, b2, ... means original buffer byte values from addresses 0, 1, 2, ...
b0m, b1m, b2m, ... means nibble-swapped original bytes b0, b1, b2, ... by following rules:

Original Byte bits	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Nibble-swapped Byte Bits	bit 3	bit 2	bit 1	bit 0	bit 7	bit 6	bit 5	bit 4

Buffer / Erase

If this command is selected, the content of the buffer will be filled with topical blank character.

The reserved key <Ctrl+F2> will bring out this menu from any menu and any time.

Buffer / Fill random data

If this command is selected, the content of the buffer will be filled with random data.

The reserved key <Shift+Ctrl+F2> will bring out this menu from any menu and any time.

Buffer / Duplicate buffer

This command performs duplicate buffer content in range of source EPROM to range of destination EPROM. This procedure is suitable if there is used for example 27C512 EPROM to 27C256 EPROM position.

Note: The procedure always uses buffer start address 00000h.

Buffer / Checksum

The checksum dialog is used for calculate checksums of selected block in buffer. The checksums are calculated by next way :

Byte sum by bytes to "word". CY flag is ignored
Word sum by words to "word". CY flag is ignored
Byte (CY) sum by bytes to "word". CY flag is added to result.
Word (CY) sum by words to "word". CY flag is added to result.
CRC-CCITT sum by bytes to "word" using $RESULT=PREVIOUS + (x^{16} + x^{12} + x^5 + 1)$
CRC-XModem sum by bytes to "word" using $RESULT=PREVIOUS + (x^{16} + x^{15} + x^2 + 1)$

Column marked as Neg. is a negation of checksum so, that Sum + Neg. = FFFFH.

Column marked as Suppl. is complement of checksum so, that Sum + Suppl. = 0 (+ carry).

Dialog checksum contains following items:

From address: This is a start address of block selected for calculating checksums in buffer. Address is defined as Byte address.

To address: This is an end address of block selected for calculating checksums in buffer. Address is defined as Byte address.

Insert checksum: This is special item used for select which kind of checksum will be written into the buffer when, the Calculate & insert was executed.

Insert at address: This is special item that specifies an address from the buffer where a result of chosen checksum will be written, when the Calculate & insert was executed. Address can not be specified inside the range <From address> to <To address>, from which will be checksum calculate. Address is defined as Byte address.

Size: This item is used for setting a size of chosen checksum result, which will be written into the buffer. A size of checksum result may be 8 (byte) or 16 (word) bits long. If word size was selected, whole checksum value will be written into the buffer. In other case only low byte of checksum value will be written into the buffer.

Note: If word size was selected, a low byte of checksum value will be written on address specified in box Insert address and a high byte will be written on address incremented by one.

Calculate: Click on the button Calculate starts calculating checksums for selected block in buffer. No writes into the buffer are executed.

Calculate & insert: Click on the button Calculate & insert starts calculating checksums for selected block in the buffer and writes the chosen checksum into the buffer on address specified by Insert address.

Checksum displayed in main window

Checksum value displayed in main program window in table "Addresses" shows sum of current data in main buffer.

The checksum is calculated by summing the contents of buffer data from address "Buffer Start" to address "Buffer End". "Buffer Start" and "Buffer End" addresses are displayed in table "Addresses" in the main program window.

The checksum value is displayed in 32-bit hexadecimal number format. Any carry bits exceeding 32-bits are neglected.

Buffer data are summed byte-by-byte irrespective of current buffer view mode (x8/x16/x1) organization.

Device

Menu Device includes functions for a work with selected programmable devices - device select, read data from device, device blank check, device program, device verify and device erase.

Device / Select from default devices

This window allows selecting the desired type of the device from list of default devices.

This one is a cyclic buffer in which are stored recently selected devices including their device options.

This list is saved to disk by command File / Exit and save.

If you wish display additional information about the current device, use an <Ctrl+F1> key.

This command provides a size of device, organization, programming algorithm and a list of programmers (including auxiliary modules) that supported this device.

You can find here package information and other general information about current device too.

Use a key for delete of current device from list of default devices.

There isn't possible to empty this list, if you repeat this access.

The last device stays in buffer and the key isn't accepted.

Device / Select device ...

This window allows selecting the desired type of the device from all devices supported by current programmer.

It is possible to choose device by name, by type or by manufacturer.

Selected device is automatically saved to buffer of default devices.

This buffer is accessible with Device / Select from default devices command.

In the Search mask field you can enter mask for filtering of whole device list by device name, manufacturer and/or programming adapter names.
The space as delimiter of filter items (fragments) has "OR" function.
If you want to enter exact filter string including spaces, use quotation mark character " .

Example:

We need to see the devices that need no adapter, and we know that such devices have following note string in Adapter column of device list: Note: in ZIF socket of programmer.

The suitable filter to show only wished devices is "in ZIF" (including quotation marks). The filter strings are not case sensitive, i.e. for example "ZIF" is the same as "zif".

If you wish display additional information about the current device, use button Device info or an <Ctrl+F1> key.

This command provides a size of device, organization, programming algorithm and a list of programmers (including auxiliary modules) that supported this device.

You can find here package information and other general information about current device too.

The currently displayed device list can be saved to text file by pressing button Save currently displayed list to file. Select device ... / All

This window allows selecting the desired type of the device from all devices supported by current programmer.

Supported devices are displayed in a list box.

Device can be select by double click on a line from list with desired manufacturer name and device number or by entering manufacturer name and/or device number in a search box (use a key <Space> as a separation character) and press <Enter> or click OK button.

Press a key <Esc> or click Cancel button at any time to cancel device selection without affecting the currently selected device.

Selected device is automatically saved to buffer of default devices.

This buffer is accessible with Device / Select from default devices command.

If you wish display additional information about the current device, use button Device info or an <Ctrl+F1> key.

This command provides a size of device, organization, programming algorithm and a list of programmers (including auxiliary modules), which supported this device. You can find here package information and other general information about current device too.

Select device ... / Only selected type

This window allows selecting the desired type of the device.

At the first - you must select a device type (e.g. EPROM) and device subtype (e.g. 64Kx8 (27512)), using mouse or cursor keys.

It will cause a list of manufacturers and devices will be displayed.

Device can be select by double click on a line from list with desired manufacturer name and device number or by entering device number in a search box (use a key <Space> as a separation character) and press <Enter> or click OK button.

Press a key <Esc> or click Cancel button at any time to cancel device selection without affecting the currently selected device.

Selected device is automatically saved to buffer of default devices.

This buffer is accessible with Device / Select from default devices command.

If you wish display additional information about the current device, use button Device info or an <Ctrl+F1> key.

This command provides a size of device, organization, programming algorithm and a list of programmers (including auxiliary modules) that supported this device.

You can find here package information and other general information about current device too.

Device / Select EPROM /Flash by ID

Use this command for autoselect an EPROM or Flash as active device by reading the device ID.

The programmer can automatically identify certain devices by the reading the manufacturer and the device-ID that are burnt into the chip.

This only applies to EPROM or Flash that supports this feature.

If the device does not support a chip ID and manufacturer's ID, a message will be displayed indicating this as an unknown or not supported device.

If more devices with identical chip ID and manufacturer's ID were detected, the list of these devices will be displayed.

A corresponding device can be chosen from this list by selecting its number (or manufacturer name) from list and press <Enter> (or click OK button).

Press a key <Esc> or click Cancel button at any time to cancel device selection without affecting the currently selected device.

Warning: The control program only support this time EPROM's and Flash with 28 and 32 pins. Any of programmers determines pins number automatically.

For other programmers you must enter this number manually.

The programmer applies a high voltage to the appropriate pins on the socket. This is necessary to enable the system to read the device ID.

Do not insert into the socket a device that is not an EPROM or Flash. It may be damaged when the programmer applies the high voltage.

We don't recommend apply this command to 2764 and 27128 EPROM types, because most of them ID not supports.

Device / Device options

All settings of this menu are used for programming process, serialization and associated file control.

Device / Device options / Operation options

All settings of this command are used for programming process control. This is a flexible environment, which content items associated with current device and programmer type. Items, which are valid for the current device but aren't supported by current programmer, are disabled. These settings are saving to disk along with associated device by File / Exit and save command.

The commonly used term are also explained in the user's manual to programmer. The special terms used here are exactly the terms used by manufacturer of respective chip. Please read the documentation to the chip you want to program for explanation of all used terms.

List of commonly used items:

group Addresses:

device start address	(default 0)
device end address	(default device size-1)
buffer start address	(default 0)
Split	(default none)

This option allows to set special mode of buffer when programming or reading device.

Using split options is particularly useful when using 8-bit data memory devices in 16-bit or 32-bit applications.

Following table describes buffer to device and device to buffer data transfer

Split type	Device	Buffer Address assignment
------------	--------	---------------------------

None	Device[ADDR]	Buffer[ADDR]
Even	Device[ADDR]	Buffer[2*ADDR]
Odd	Device[ADDR]	Buffer[1+(2*ADDR)]
1./4	Device[ADDR]	Buffer[4*ADDR]
2./4	Device[ADDR]	Buffer[1+(4*ADDR)]
3./4	Device[ADDR]	Buffer[2+(4*ADDR)]
4./4	Device[ADDR]	Buffer[3+(4*ADDR)]

Real addressing will be following: (all addresses are hexadecimal)

Split type	Device addresses	Buffer addresses
None	00 01 02 03 04 05	00 01 02 03 04 05
Even	00 01 02 03 04 05	00 02 04 06 08 0A
Odd	00 01 02 03 04 05	01 03 05 07 09 0B
1./4	00 01 02 03 04 05	00 04 08 0C 10 14
2./4	00 01 02 03 04 05	01 05 09 0D 11 15
3./4	00 01 02 03 04 05	02 06 0A 0E 12 16
4./4	00 01 02 03 04 05	03 07 0B 0F 13 17

Terms explanation:

Access to device address ADDR is written as Device[ADDR].

Access to buffer address ADDR is written as Buffer[ADDR].

ADDR value can be from zero to device size (in bytes).

All addresses are byte oriented addresses.

group Insertion test:

insertion test (default ENABLE)

If enabled, the programmer checks all pins of the programmed chip, if have proper connection to the ZIF socket (continuity test).

The programmer is able to identify the wrong contact, misinserted chip and also (partially) backinserted chip.

check ID bytes (default ENABLE)

If enabled, the programmer checks the electronic ID of the programmed chip.

Note 1: Some old chips don't carry electronic ID.

Note 2: In some special cases, several microcontrollers don't provide ID, if copy protection feature in the chip is set, even if device ID check setting in control program is set to "Enable".

group Command execution:

blank check before programming (default DISABLE)

erase before programming (default DISABLE)

verify after reading (default ENABLE)

verify (ONCE, TWICE)
verify options (nominal VCC +/-5%nominal VCC +/-10%VCCmin - VCCmax)

group ISP Target Supply Parameters

Enable target system power supply - enables supplying of target system from programmer.

Supply voltage for target system is switched on before action with programmed device and is switched off after action finished.

If Keep ISP signals at defined level after operation is enabled, then programmer will switch off supply voltage after pull-up/pull-down resistors are deactivated.

Voltage - supply voltage for target system.

Note: The voltage value given to target system depends also on current flowing to target system.

To reach exact voltage supply for target system, the proper Voltage and Max. current values has to be defined.

The Max. current value specified has to be as exact as possible equal to real current consumption of target system.

Max. current - maximum current consumption of powered target system.

Voltage rise time - determines skew rate of rising edge of target supply voltage (switch on supply voltage).

Target supply settle time - determines time, after which must be supply voltage in target system stabilized at set value and target system is ready to any action with programmed device.

Voltage fall time - determines skew rate of falling edge of target supply voltage (switch off supply voltage).

Power down time - determines time after switch off target system power supply within target system keeps residual supply voltage (e.g. from charged capacitor). After this time elapsed target system has to be without supply voltage and can be safely disconnected from programmer.

Group Target System Parameters

Oscillator frequency (in Hz) - oscillator's frequency of device (in target system). Control program sets programming speed by its, therefore is necessary set correct value.

Supply voltage (in mV) - supply voltage in target system. Control program checks or sets (it depends on programmer type) entered supply voltage in

target system before every action on device.

Disable test supply voltage - disables measure and checking supply voltage of programmed device, set in Supply voltage edit box, before action with device.

Delay after reset active - this parameter determine delay after Reset signal active to start action with device. This delay depends on values of used devices in reset circuit of device and can be chosen from these values: 10ms, 50ms, 100ms, 500ms or 1s.

Inactive level of ISP signals - this parameter determine level of ISP signals after finishing access to target device. Signals of ISP connector can be set to Pull-up (signals are tied through 22k resistors to supply voltage) or Pull-down (signals are tied through 22k resistors to ground).

Keep ISP signals at defined level after operation - enables keeping set level of ISP signals after access to target device finished. Control program indicates activated pull-up/pull-down resistors by displaying window with warning. After user close this window control program will deactivate resistors.

Device / Device options / Serialization

Serialization is special mode of program. When a serialization mode is activated, a specified value is automatically inserted on predefined address into buffer before programming each device. When more devices are programmed one by one, the serial number value is changed for each device automatically and inserted into buffer before programming device, so each device has unique serial number.

There are two types of serialization:

- Incremental mode
- From file mode
- Custom generator mode

If a new device is selected, the serialization function is set to a default state i.e. disabled.

Actual serialization settings for actually selected device are saving to disk along with associated device by File / Exit and save command.

When incremental mode is active following actual settings are saved to configuration file: address, size, serial value, incremental step and settings of modes ASCII / BIN, DEC / HEX, LS byte / MS Byte first.

When from-file mode is active following actual settings are saved to configuration file: name of input serialization file and actual label, which indicates the line with actual serial number in input file.

When program is in multiprogramming mode (multiple socket programmer is actually selected) the special section - Action on not programmed serial values due to error - is displayed in dialog Serialization. In this section two choices are available:

- Ignore not programmed serial values
- Add not programmed serial values to file

Ignore not programmed serial values means the not programmed serial values are ignored and no action is done with them.

Add not programmed serial values to file means the not programmed serial values are added to file. The file of not programmed serial values has the same text format as serialization file for "From-file" serialization mode. So there is possible to program the serial values later on by "From-file" serialization mode.

If device programming is stopped by user, program will not change the serial values ready for next batch of devices. The same situation is if device program is incomplete, e.g. for device insertion test error.

Ignoring or writing not programmed serial values is only used when at least one device from current batch of devices in multiple socket module programmer is completely programmed and verified without errors.

Serialization can work with control program's main buffer or extended buffers available for some types of devices, for example Microchip PIC16Fxxx devices with Data EEPROM Memory.

The selection which buffer has to used by serialization routine is available in dialog Serialization.

The extended buffer selection is ignored for From-file serialization in playlist file mode.

For more details about this limitation, see the From file mode serialization mode description please.

Device / Device options / Serialization / Incremental mode

The Incremental mode enables to assign individual serial numbers to each programmed device.

A starting number entered by user will be incremented by specified step for each device program operation and loaded in selected format to specified buffer address prior to programming of each device.

There are following options, that user can modify for incremental mode:

S / N size

S / N size option defines the number of bytes of serial value which will be written to buffer.

For Bin (binary) serialization modes values 1-4 are valid for S / N size and for

ASCII serialization modes values 1-8 are valid for S / N size.

Address

Address option specifies the buffer address, where serial value has to be written. Note that address range must be inside the device start and device end addresses.

Address must be correctly specified so the last (highest or lowest) byte of serial value must be inside device start and device end address range.

Start value

Start value option specifies the initial value, from which serialization will start. Generally, the max. value for serialization is \$1FFFFFFF in 32 bit long word.

When the actual serial value exceeds maximum value, three most significant bits of serial number are set to zero.

After this action the number is always inside 0..\$1FFFFFFF interval (this is basic style of overflow handling).

Step

Step options specify the increment step of serial value incrementation.

S / N mode

S / N mode option defines the form in which serial value has to be written to buffer. Two options are available:

ASCII - means the serial number is written to buffer as ASCII string. For example number \$0528CD is in ASCII mode written to buffer as 30h 35h 32h 38h 43h 44h ('0' '5' '2' '8' 'C' 'D'), i.e. six bytes.

Bin - means the serial number is written directly to buffer.

If the serial number has more than one byte length, it can be written in one of two possible byte orders.

The byte order can be changed in „Save to buffer“ item.

Style

Style option defines serial number base. There are two options:

Decimal numbers are entered and displayed using the characters '0' through '9'.

Hexadecimal numbers also use characters 'A' through 'F'.

The special case is Binary Dec, which means BCD number style.

BCD means the decimal number is stored in hexadecimal number, i.e. each nibble must have value from 0 to 9. Values A to F are not allowed as nibbles of BCD numbers.

Select the base in „Style“ options before entering numbers of serial start value and step.

Save to buffer

Save to buffer option specifies the serial value byte order to write to buffer.

This option is used for Bin S / N mode (for ASCII mode it has no effect).

Two options are available:

LSByte first (used by Intel processors) will place the Least Significant Byte of serial number to the lowest address in buffer.

MSByte first (used by Motorola processors) will place the Most Significant Byte first to the lowest address in buffer.

Split serial number at every N byte(s)

The option allows dividing serial number into individual bytes and placing the bytes at each Nth address of buffer.

This feature is particularly useful for example for Microchip PIC devices when the device serial number can be the part of program memory as group of RETLW instructions.

The example of using serial number split is listed in section Examples bellow as example number 2.

Example:

Example 1:

Write serial numbers to AT29C040 devices at address 7FFF0H, size of serial number is 4 bytes, start value is 16000000H, incremental step is 1, the serial number form is binary and least significant byte is placed at the lower address of serial number in device.

To make above described serialization following settings have to be set in Serialization dialog:

Mode: Incremental mode
S/N size: 4 bytes
S/N mode:: Bin
Style: Hex
Save to buffer: LS Byte first
Address: 7FFF0H
Start value: 16000000H
Step: 1

Following values will be written to device:

The 1st device

Address	Data
007FFF0	xx xx xx xx xx xx xx xx xx xx xx xx 00 00 00 16

The 2nd device

Address	Data
007FFF0	xx xx xx xx xx xx xx xx xx xx xx xx 01 00 00 16

The 3rd device

Address	Data
007FFF0	xx xx xx xx xx xx xx xx xx xx xx xx 02 00 00 16

etc.

"xx" mean user data programmed to device

Serial numbers are written to device from address 7FFFCH to address 7FFFFH because serial number size is 4 bytes.

Example 2:

Following example shows usage of SQTP serialization mode when serial number is split into RETLW instructions for Microchip PIC16F628 devices.

Device PIC16F628 has 14 bit wide instruction word. Instruction RETLW has 14-Bit Opcode:

Description	MSB 14-Bit word	LSB
RETLW	Return with literal in W	11 01xx kkkk kkkk

where xx can be replaced by 00 and k are data bits, i.e. serial number byte

Opcode of RETLW instruction is hexadecimal 34KKH where KK is data Byte (serial number byte)

Let's assume we want to write serial number 1234ABCDH as part of four RETLW instructions to device PIC. The highest Byte of serial number is the most significant Byte. We want to write the serial number to device program memory at address 40H. Serial number split is very useful in this situation. Serialization without serial number split will write the following number to buffer and device:

Address	Data
0000080	CD AB 34 12 xx xx xx xx xx xx xx xx xx xx xx

Note: address 80H is because buffer has byte organization and PIC has word organization so it has equivalent program memory address 40H. When buffer has word organization x16, the address will be 40H and number 1234ABCDH will be placed to buffer as following:

Address	Data
0000040	ABCD 1234 xxxx xxxx xxxx xxxx xxxx xxxx

We want to use RETLW instruction so buffer has to be:

Address	Data
0000040	34CD 34AB 3434 3412 xxxx xxxx xxxx xxxx

We can do this by following steps:

- A)** write four RETLW instructions at address 40H to main buffer (this can be done by hand editing buffer or by loading file with proper content). The bottom 8 bits of each RETLW instruction are not important now, because serialization will write correct serial number bytes at bottom 8 bits of

each RETLW instruction.

The buffer content before starting device program will look for example as following:

Address	Data
0000040	3400 3400 3400 3400 xxxx xxxx xxxx xxxx

8 bits of each RETLW instructions are zeros, they can have any value.

B) Set the serialization options as following:

S/N size: 4 Bytes

Address: 40H

Start value: 1234ABCDH

Step: 1

S/N mode: BIN

Style: HEX

Save to buffer: LS Byte first

Check the option "Split serial number at every N byte(s)" and split value N set to 2.

(It means split of serial number to buffer at every second Byte)

The correct serial number is set tightly before device programming operation starts.

The buffer content of serial number when programming the first device is:

Address	Data
0000040	34CD 34AB 3434 3412 xxxx xxxx xxxx xxxx

That's it.

Example 3:

Following example uses the same serialization options as Example number 2, instead the serial number split is set to 3 and 4.

When "Split serial number at every 3 byte(s)" is set, the buffer content will look as:

Byte buffer organization:

Address	Data
0000080	CD xx xx AB xx xx 34 xx xx 12 xx xx xx xx xx xx

Word16 buffer organization:

Address	Data
0000040	xxCD ABxx xxxx xx34 12xx xxxx xxxx xxxx

When "Split serial number at every 4 byte(s)" is set, the buffer content will look as:

Byte buffer organization:

Address	Data
0000080	CD xx xx xx AB xx xx xx 34 xx xx xx 12

Word16 buffer organization:

Address	Data
0000040	xxCD xxxx xxAB xxxx xx34 xxxx xx12 xxxx

Advice: When you are not sure about effects of serialization options, there is possible to test the real serial number, which will be written to buffer. The test can be made by following steps:

1. select wished serialization options in dialog Serialization and confirm these by OK button
2. in dialog Device operation options set Insertion test and Device ID check (if available) to Disabled
3. check there is no device inserted to programmer's ZIF socket
4. run Device Program operation (for some types of devices it is necessary to select programming options before programming will start)
5. after completing programming operation (mostly with some errors because device is not present) look at the main buffer (View/Edit buffer) at address where serial number should be placed

Note: Address for Serialization is always assigned to actual device organization and buffer organization that control program is using for current device. If the buffer organization is byte org. (x8), the Serialization Address will be byte address. If the buffer organization is wider than byte, e.g. 16 bit words (x16), the Serialization Address will be word address.

Device / Device options / Serialization / From file mode

Using the From-file method, serial values are read from the user specified input file(s) and written serialization data to buffer on specified addresses.

There are two basic kinds of From-file serialization depending on format of serialization file used.

1. "Classic" From-file mode - the serialization file has serial values directly included. Serialization data are then read directly from serialization file to buffer on address specified in the file. Classic From-file mode is indicated in main window and info window of M1882 control program on panel "Serialization" as "From-file" serialization. Description of "classic" From-file serialization file is listed in "Classic From-file serialization file format" chapter.
2. From-file mode from "playlist" file - the serialization file has not serial values directly included. The file contains name list of external files that contain serialization data. Serialization data are then read from these external data files, each file means one serialization step (one device programmed). Playlist

From-file mode is indicated in main window and info window of M1882 control program on panel "Serialization" as "From-file-pl" serialization. Description of "playlist" serialization file is listed in "Playlist From-file serialization file format" chapter.

There are two user options: File name and Start label.

File name

File name option specifies the file name from which serial addresses and values will be read. The input file for from file serialization must have special format, which is described in from file serialization file format below.

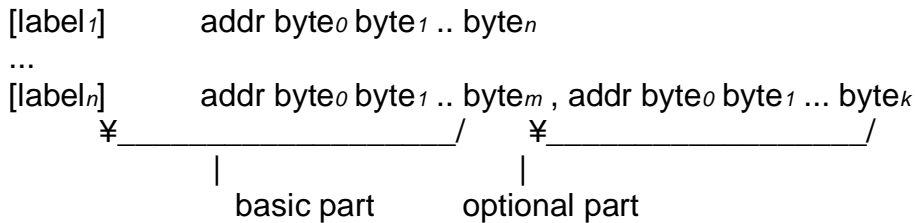
Start label

Start label defines the start label in input file. The reading of serial values from file starts from defined start label.

Size of serialization file is limited by free disk space. Recommended maximal number of serial records (items) in one serialization file is 10000 records. More records may cause slower operation when reading serial number before each device programming cycle.

CLASSIC FROM FILE SERIALIZATION FILE FORMAT

Classic From-file serialization input file has text format. The file includes addresses and arrays of bytes defining buffer addresses and data to write to buffer. Input file has text type format, which structure is:



; Comment

meaning is:

basic part

Basic part defines buffer address and array of bytes to write to buffer. Basic part must be always defined after label in line.

optional part

Optional part defines the second array of bytes and buffer address to write to buffer. One optional part can be defined after basic part of data.

label₁, label_n - labels

Labels are identifiers for each line of input file. They are used for addressing

each line of file. The labels should be unique. Addressing lines of file means, the required start label entered by user defines line in input file from which serial values reading starts.

addr -

Addr defines buffer address to write data following the address.

byte0..byten, byte0..bytem, byte0..bytek -

Bytes arrays *byte0..byten*, *byte0..bytem* and *byte0..bytek* are defining data, which are assigned to write to buffer. Maximum count of bytes in one data field following the address is 64 bytes. Data bytes are written to buffer from address *addr* to *addr+n*.

The process of writing particular bytes to buffer is:

byte0 to addr
byte1 to addr + 1
byte2 to addr + 2
....
byten to addr + n

Optional part is delimited from the first data part by character “ , “ (comma) and its structure is the same as in the first data part, i.e. address and following array of data bytes.

Characters with special use:

[] - labels must be defined inside square brackets

' ; ' – character which delimiters basic part and optional part of data

‘ ; ’ - the semicolon character means the beginning of a comment.

All characters from ‘ ; ; ’ to the end of line are ignored.

Comment can be on individual line or in the end of definition line.

Note:

Label names can contain all characters except '[' and ']’.

The label names are analysed as non case sensitive,

i.e. character ‘a’ is same as ‘A’, ‘b’ is same as ‘B’ etc..

All address and byte number values in input file are hexadecimal.

Allowed address value size is from 1 to 4 bytes.

Allowed size of data arrays in one line is in range from 1 to 64 bytes.

When there are two data arrays in one line, the sum of their size in bytes can be maximally 80 bytes.

Be careful to set correct addresses.

Address must be defined inside device start and device end address range.

In case of address out of range,

warning window appears and serialization is set to disabled (None).

Address for Serialization is always assigned to actual device organization and buffer organization that control program is using for current device.

If the buffer organization is byte org. (x8), the Serialization Address will be byte address.

If the buffer organization is wider than byte, e.g. 16 bit words (x16), the Serialization Address will be word address.

Example:

```
[nav1] A7890 78 89 56 02 AB CD ; comment1
```

```
[nav2] A7890 02 02 04 06 08 0A
```

```
[nav3] A7890 08 09 0A 0B A0 C0 ; comment2
```

```
[nav4] A7890 68 87 50 02 0B 8D
```

```
[nav5] A7890 A8 88 59 02 AB 7D
```

```
;next line contains also second definition
```

```
[nav6] A7890 18 29 36 42 5B 6D , FFFF6 44 11 22 33 99 88 77 66 55 16
```

```
; this is last line - end of file
```

In the example file six serial values with labels „nav1“, „nav2“, ...“nav6“ are defined. Each value is written to buffer on address \$A7890. All values have size 6 bytes. The line with „nav6“ label has also second value definition, which is written to buffer on address \$FFFF6 and has size 10 bytes, i.e. the last byte of this value will be written to address \$FFFFF.

Note: Address for Serialization is always assigned to actual device organization and buffer organization that control program is using for current device.

If the buffer organization is byte org. (x8), the Serialization Address will be byte address.

If the buffer organization is wider than byte, e.g. 16 bit words (x16), the Serialization Address will be word address.

PLAYLIST FROM FILE SERIALIZATION FILE FORMAT

From-file serialization playlist file includes list of file names which contain serialization data. The file format is similar to classic serialization file format. Following file format differences are for playlist files:

1. the playlist file must have special header at the first no empty line of file.
2. The header is text line in format
FILETYPE=M1882 SERIALIZATION PLAYLIST FILE

2. each serial data batch is represented by separate line in format

```
[label x] datafilename
```

labelx - represents label

Labels are identifiers for each no-empty line of input file. They are used for addressing each line of file. The labels should be unique within the file. Addressing lines of file means, that the required start label entered by user defines line in input file from which serial values reading starts.

datafilename - defines name of data file, which contains serialization data. When serialization requires new serial value, the data file will be loaded by standard M1882 "Load file" procedure to M1882 buffer. File format can be binary or Hex file (Intel Hex etc.). The auto-recognition system recognizes proper file format and forces load of file in the right file format. Data filename is relative to parent (playlist) serialization file.

Example of playlist serialization file:

```
;---- following file header is required -----  
FILETYPE=M1882 SERIALIZATION PLAYLIST FILE  
  
;----- references to serialization data files  
[nav1] file1.dat  
[nav2] file2.dat  
[nav3] file3.dat  
...  
  
[label n] filex.dat  
;----- end of file -----
```

For more detailed and fully functional example of serialization type From-file playlist, look the example files placed in the M1882 installation directory in Examples\subdirectory as following:

<M1882_inst_dir>\Examples\Serialization\fromfile_playlist_example\

The typical path can look like this:

C:\Program Files\MINATO ELECTRONICS

INC._sw\Programmer\Examples\Serialization\fromfile_playlist_example\

You can test the serialization by following steps:

1. start M1882
2. you need to have our programmer connected and correctly found in M1882
3. select wished device, the best are devices with erasable memory, (not OTP memory)
4. select dialog from menu Device | Device Options | Serialization
5. Set the From-file mode and in the panel From-file mode options select our example serialization file fromfile_playlist.ser
6. click the OK button to accept the new serialization settings
7. run "Program" device operation

You can see at the serialization indicating labels in the main window of M1882 and also in info progress window during device programming and repeating of programming.

Device / Device options / Serialization / Custom generator mode

Custom generator serialization mode provide maximum flexible serialization mode, because the user have serialization system fully in his hands.

When Custom generator mode of serialization is selected, serial numbers are generated by user made program “on-the-fly” before each device is programmed in M1882. Custom generator mode serialization allows user to generate unique sequence of serial numbers desired. Serial numbers can be incremented as a linear sequence or completely non-linear sequence. The user made serial number generator program details are described later in the following section Custom generator program.

Examples:

There are also example .exe and C/C++ source files available.

The files are placed in the M1882 installation directory in Examples¥ subdirectory as following:

<M1882_inst_dir>¥Examples¥Serialization¥customgenerator_example¥

The typical path can look like this:

C:¥Program Files¥MINATO ELECTRONICS
INC._sw¥Programmer¥Examples¥Serialization¥customgenerator_example¥

There are following options for Custom generator serialization in M1882 control software:

In dialog Serialization select in Mode panel option Custom generator mode.

The following options will be displayed:

Serialization data file

Specifies the path and name for the data file that will contain the current serial number. When device is to be programmed, the M1882 software calls user made serial number generator that updates the data file. The recommended extension of data file is .dat.

Note: The data file is completely and periodically overwritten during device programming with serialization. Be sure to enter the correct name of wished .dat file. Example: “c:¥serial_files¥serial.dat”

Serialization generator

Specifies the path and name for the executable file which will generate serialization data file.

First serial number

This option is required to specify the initial serial number that will be passed to

custom generator serialization program. The number is entered and displayed in hexadecimal format.

Last serial number

This option specifies the maximum value of serial number allowed. If the value is non-zero, it will be passed to serialization generator program. The generator is responsible for testing the value of last serial number and generate serial .dat file with appropriate error information in the serialization .dat file in case of current serial number greater than last serial number. If the value of Last serial number is zero, the value will not be passed to generator program.

Custom generator program

Custom generator program or serialization generator is program that will generate the unique sequence of serial numbers and write the serial data to serialization .dat file. This program is made by user. The path and name of the serialization program must be specified in the Serialization options dialog in Custom generator mode options.

The program will be called from M1882 every time the new serial data have to be generated. This is usually made before each device programming operation. M1882 control program passes command line parameters to serialization program and serialization program generates serialization .dat file which is read by M1882 control program. Following command line parameters are used:

-N<serial number> Specifies current serial number.

-E<serial number> Specifies ending (or last) serial number.

The parameter is only passed when value of Last serial number specified in dialog Serialization in M1882 software is no zero.

The serialization program should return error record T06 in the serialization .dat file, if the current serial number is greater than ending serial number.

For details look at section Serialization .dat file format.

Serialization .dat file format

Serialization .dat file generated by serialization generator must meet following text format.

Serialization .dat file consists of records and serial data section.

Record is line which begin with one of Txx prefixes as described bellow. Value of "xx" represents the record type code. Records are used to inform M1882 software about serialization status (current and last serial numbers, serialization data and data format, errors, etc.). Required records are records T01, T02, T03 and T04. Other records are optional.

T01:<serial number> Contains current serial number value passed to generator by command line parameter -N<serial number>.

T02:<serial number> Contains ending (or last) serial number value passed

to generator by command line parameter -E<serial number>.

T03:<data format code> Specifies the serialization data format.

Following formats are supported now:

T03:50 or T03:55 ASCII Space data format

T03:99 - Intel Hex data format

T04: - indicates the serialization data will follow from next line to the end of file. Serialization data are stored in one of standard ASCII data file formats, for example Intel Hex, ASCII Space and so on. The format used for data must be specified by record T03.

Example: Typical serialization data file:

```
T01:000005
T02:001006
T03:99
T04:
:0300000000096B89
:03000300000005F5
:02000C005A0197
:01003F004F71
:00000001FF
```

The file consists of following information:

line T01 - current serial number 000005h

line T02 - ending (last) serial number 001006h

line T03 - serialization data format after line T04 is Intel Hex

line T04 - serialization data, which will be loaded to buffer of M1882 before programming device,

data are represented in Intel Hex format

Optional records are:

T05:<message> Warning or error message. This record causes the serialization is stopped and warning or error message is displayed in M1882 software.

T06: Current serial number greater than limit
This record causes the serialization is stopped and warning or error message is displayed in M1882 software. The reason of turning serialization off is the current serial number is greater than allowed maximum ending serial number. This record can be used when -E command line parameter is specified, it means no zero Last serial value in dialog Serialization is specified.

T11:<message> Less important warning or message.
The serialization will not be interrupted.

Device / Device options / Statistics

Statistics gives the information about actual count of device operations, which were proceeded on selected type device. If one device is corresponding to one device operation, e.g. programming, the number of device operations will be equal to number of programmed devices.

The next function of statistics is Count down. Count down allows checking the number of device operations, and then number of devices, on which device operations have to be done. After each successful device operation the value of count down counter is decremented. Count down has user defined start number of devices to do. When count down value reach zero, it means, specified number of devices is complete and user message about complete count down will be displayed.

Statistics dialog contains following options:

Check boxes Program, Verify, Blank, Erase and Read define operations, after which statistics values increment.

Check box Count down sets Count down activity (enable or disable). Edit box following the Count down check box defines initial number of count down counter, from which count down starts.

Statistics dialog can be also opened by pressing right mouse button on Statistics panel and clicking displayed item Statistics.

Actual statistics values are displaying in main window of control program in Statistics panel.

Statistics panel contains three statistics values – Success, Failure, Total and two Count down information values Count down and Remains.

Meaning of the values is:

Success	number of operations which where successfully completed
Failure	number of operations which where not successfully completed
Total	number of all operations
Count down	informs about Count down activity (Enabled or Disabled)
Remains	informs about remaining number of device operations to do

Successful operation means any device operation of these, which is completed without errors:

- program
- verify
- blank check
- erase
- read

If device operation is finished with error(s) it is not successful operation.

When new device type is selected, all statistics values are set to zero and Count down is set to Disabled.

Reset button in Statistics panel reset statistics values.

Reload Count down button in Statistics panel reloads initial value to Count down.

Device / Device options / Associated file

This command is used for setting associated file with current device.

This is a file, which can be automatic loaded to buffer after device is selected from default devices select list or by start control program.

You can edit the associated file name in file name box, put a full pathname.

The control program checks the present of this file on the disk.

Also is possible enabling or disabling automatic load of this file.

You can save both settings i.e. associated file and enabling of automatic load of this file to disk by command File / Exit and save.

Device / Device options / Special options

The special terms used here are exactly the terms used by manufacturer of respective chip. Please read the documentation to the chip you want to program for explanation of all used terms.

If the name of this menu item is starting by "View/Edit ...", then the Read device command will read the content of the chip configuration and it can be viewed and edited by this menu command.

Device / Blank check

This command allows to blank check of all devices or its part if possible. The control program reports a result of this action by a write of a warning message to INFO window.

The menu command Device / Device options / Operation options allows to set another working area as the standard.

Device / Read

This command allows to read all device or its part into the buffer. The read procedure can also read the content of the chip configuration (if it exists and is readable). The special device configuration areas can be viewed or edited in dialogs available by menu View / Edit buffer and menu Device / Device options /

Special options (Alt+S).

The control program reports a finish of Read action by writing a message to INFO window.

The menu command Device / Device options / Operation options allows to set another working area as the standard. Setting an option Verify data after reading in this menu command means a higher reliability for device reading.

Device / Verify

This command compares the programmed data of the all device or its part with data in buffer. The control program reports a result of this action by a write of an error message to INFO window.

The menu command Device / Device options / Operation options allows to set another working area as the standard.

By the setting in the menu Options / Display errors the command lets to write the found errors on the display or write the found errors to VERIFY.ERR file. In the Display errors mode to the screen can display the program max. 45 the first found differences, which are located by the address where they were caused.

Device / Program

This command allows to programming of the all device or its part by the data of the buffer. The control program reports a result of this action by a write of an error message to INFO window.

The menu command Device / Device options / Operation options allows to set another working area as the standard, and set other operation options for programming process control.

Device / Erase

This command allows erasing the whole programmable device. The program reports the end without error or end with the error by writes the warning report on the display.

The Blank check procedure is applied after Chip erase command for such chips, where doesn't exist other way how to check, the chip is really erased.

Device / Test

This command executes a test with device selected from list of supported devices (e.g. static RAM) on programmers, which support this test.

Device / IC test

This command activates a test section for ICs separated by type to any libraries (on distribution CD). First select an appropriate library, wished device and then a mode for test vectors run (LOOP, SINGLE STEP). Control sequence and test results are displayed to LOG WINDOW. In case of need is possible to define the test vectors directly by user. Detailed description syntax and methods of creation testing vectors is described in example_e.lib file, which is in programs installation folder. Note. Because the rising/falling edges of programmers are tuned for programming of chips, it may happen the test of some chips fails, although the chips aren't defective (counters for example).

Device / JAM/VME/...Player

Jam STAPL was created by Altera® engineers and is supported by a consortium of programmable logic device (PLD) manufacturers, programming equipment makers, and test equipment manufacturers.

The Jam™ Standard Test and Programming Language (STAPL), JEDEC standard JESD-71, is a standard file format for ISP (In-System Programming) purposes. Jam STAPL is a freely licensable open standard. It supports programming or configuration of programmable devices and testing of electronic systems, using the IEEE 1149.1 Joint Test Action Group (JTAG) interface. Device can be programmed or verified, but Jam STAPL does not generally allow other functions such as reading a device.

The Jam STAPL programming solution consists of two components: Jam Composer and Jam Player.

The Jam Composer is a program, generally written by a programmable logic vendor, that generates a Jam file (.jam) containing the user data and programming algorithm required to program a design into a device.

The Jam Player is a program that reads the Jam file and applies vectors for programming and testing of devices in a JTAG chain.

The devices can be programmed in ZIF socket of the programmer or in target system through ISP connector. It is indicated by [PLCC44](Jam) or (ISP-Jam) suffix after name of selected device in control program. Multiple devices are possible to program and test via JTAG chain: JTAG chain (ISP-Jam)

More information on the website: <http://www.altera.com>

In-System Programmability Guidelines

<http://www.altera.com/literature/an/an100.pdf>

Using Jam STAPL for ISP & ICR via an Embedded Processor

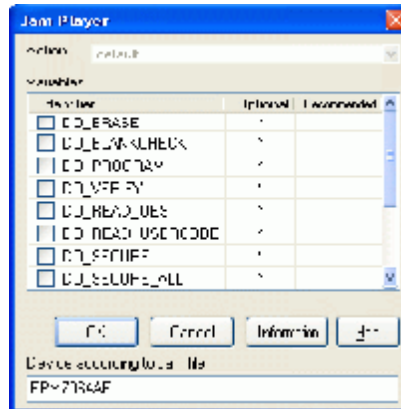
<http://www.altera.com/literature/an/an122.pdf>

Software tools:

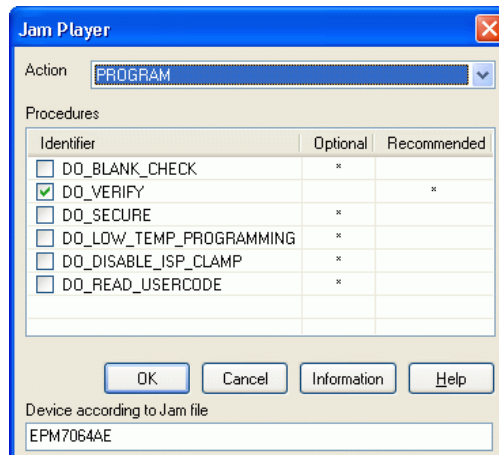
Altera: MAX+plus II, Quartus II, SVF2Jam utility (converts a serial vector file to a Jam file), LAT2Jam utility (converts an ispLSI3256A JEDEC file to a Jam file);

Xilinx: Xilinx ISE Webpack or Foundation software (generates STAPL file or SVF file for use by utility SVF2Jam);

JAM player dialog



Jam Player version 1 (see Action and Variables controls)



Jam Player version 2 (see Action and Procedures controls)

Action

Select desired action for executing.

Jam file of version 2 consists of actions. Action consists of calling of procedures which are executed.

Jam file of version 1 does not know statements 'action' and 'procedure', therefore choice Action is not accessible. Program flow starts to run instructions according to boolean variables with prefix DO_something. If you need some new boolean variables with prefix DO_something then contact us.

Procedures

Program flow executes statements from each procedure. Procedures may be optional and recommended. Recommended procedures are marked implicitly. You can enable or disable procedures according to your needs. Jam Player executes only marked procedures. Other procedures are ignored. Number of procedures is different, it depends on Jam file.

Variables

Jam file of version 1 does not know statements 'action' and 'procedure'.

Program flow starts to run instructions according to boolean variables with prefix

DO_something. Jam Player executes all marked DO_something cases in algorithm. Number of variables (procedures) is constant, it does not depend on Jam file. If you need some new boolean variables with prefix DO_something then contact us.

OK

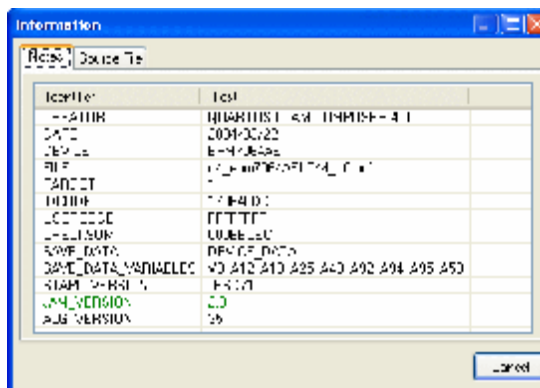
Accept selected action with appropriate procedures which are marked.

Information

Displays informations about Jam file. You can preview NOTES and source file in dialog.

Device according to Jam file

file is made for a specific device. Device name is found in Jam file in part NOTE identifier DEVICE. Device name must be identical with name of the device selected in dialog Select device. When devices are different, software will indicate this situation by warning message during start of the Jam Player. JAM file information dialog



Notes: statements are used to store information about the Jam file. The information stored in NOTE fields may include any type of documentation or attributes related to the particular Jam program.

Source file contains a program in Jam language. Jam program consists of a sequence of statements. Jam statement consists of a label, which is optional, an instruction, and arguments, and terminates with a semicolon (;). Arguments may be literal constants, variables, or expressions resulting in the desired data type (i.e., Boolean or integer). Each statement usually occupies one line of the Jam program, but this is not required. Line breaks are not significant to the Jam language syntax, except for terminating comments. An apostrophe character (') can be used to signify a comment, which is ignored by the interpreter. The language does not specify any limits for line length, statement length, or program size. More informations can be found on the website: <http://www.altera.com>

Jam file with extension .jbc is Jam STAPL Byte code format which is not visible. Converting JED file to Jam STAPL file for XILINX devices:

- install Xilinx Integrated Software Environment (ISE) 6.3i software
free download: WebPACK_63_fcfull_i.exe + 6_3_02i_pc.exe
(315MB or so)
- run Xilinx ISE 6/Accessories/iMPACT
- in dialog “Operation Mod Selection: What do you want to do first?”
choose: “Prepare Configuration Files”,
- in dialog “Prepare Configuration Files: I want create a:” choose:
“Boundary-Scan File”,
- in dialog “Prepare Boundary-Scan File: I want create a:” choose:
“STAPL File”,
- in dialog “Create a New STAPL File” write name of Jam file with
extension .stapl,
- in dialog “Add Device” select JED file with extension .jed,
- in the created jtag chain select device e.g.: XC2C32A (left mouse
button) and select sequence operation (e. g.: Erase, Blank, Program,
Verify; right mouse button),
- in menu select item “Output/Stapl file/Stop writing to Stapl file”
- run M1882, select device e.g.: Xilinx XC2x32A [QFG32](Jam), load
Jam file (Files of type: select STAPL File)
- choose “Device operation option Alt+O” press button “Jam
configuration”. Warning “Select device from menu "Select Devices"
and Jam file is probably different! Continue?” choose Yes. (Xilinx sw.
does not include line: NOTE "DEVICE" "XC2x32A"; in Jam file). In
dialog “Jam player” select action and procedures, finish dialogs,
press button “Play Jam” from toolbar and read Log window

The ispVM Virtual Machine

The ispVM Virtual Machine is a Virtual Machine that has been optimized specifically for programming devices which are compatible with the IEEE 1149.1 Standard for Boundary Scan Test. The ispVM EMBEDDED tool combines the power of Lattice's ispVM Virtual Machine™ with the industry-standard Serial Vector Format (SVF) language for Boundary Scan programming and test.

The ispVM System software generates VME files supporting both ispJTAG and non-Lattice JTAG files which are compliant to the IEEE 1149.1 standard and support SVF or IEEE 1532 formats. The VME file is a hex coded file that takes the chain information from the ispVM System window. The devices can be programmed in ZIF socket of the programmer or in target system through ISP connector. It is indicated by [PLCC44](VME) or (ISP-VME) suffix after name of selected device in control program. Multiple devices are possible to program and test via JTAG chain: JTAG chain (ISP-VME).

More information on the website:

<http://www.latticesemi.com/products/devtools/software/ispvmembed/index.cfm>
In-System Programmability Guidelines

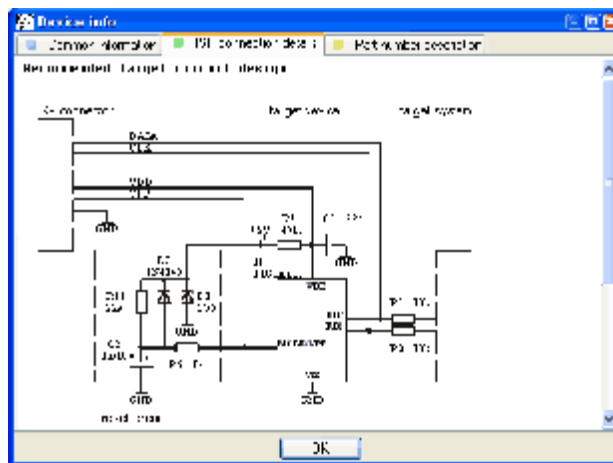
http://www.latticesemi.com/products/technology/isp_usage.cfm

Software tools:

Lattice: ispLEVER, ispVM System ISP Programming Software, PAC-Designer Software, svf2vme utility (converts a serial vector file to a VME file)

Device / Device info

The command provides additional information about the current device - size of device, organization, programming algorithm and a list of programmers (including auxiliary modules) that supported this device. You can find here package information, part number description and full information for ISP implementation. For example: description of ISP connector pins for currently selected chip, recommended target design around in-circuit programmed chip.



The reserved key **<Ctrl+F1>** will bring out this menu from any menu and any time immediately.

Programmer

Menu Programmer includes commands used for work with programmers.

Programmer / Find programmer

This item selects a new type of programmer and communication parameters. This command contains following items:

Programmer - sets a new type of programmer for find. If a Search all is selected, the control program finds all supported programmers.

Establish communication - allows manual or automatic establishing communication for a new programmer.

Speed - sets speed, if a manual establishing communication is selected, which PC sends data into the programmer. Speed is expressed as a

percent from a maximal speed.

The communication speed modification is important for PCs with "slow" LPT ports, which haven't sufficient driving power for a PC<->programmer cable (laptop, notebook, ...). Use this command, if you have any communication problems with connected programmer on the LPT port of your PC (e.g. control program reports a programmer absence, the communication with the programmer is unreliable, etc.).

If automatic establishing communication is selected, then control program sets a maximal communication speed.

Port - selects a LPT port, which will be scanned for a requested programmer. If all port is selected, the control program scans all LPT ports, which are available on standard addresses.

Address for special port - sets address of LPT port, if a Special port is selected.

Pressing key <Enter> or button OK initiates scanning for programmer by set parameters. There is same activity as at start the control program. The command clears a list of default devices without the current device, if the new selected programmer supports this one.

This setting is saved to disk by command Options / Save options.

Programmer / Refind programmer

This menu command is used to refind (reestablish communication with) currently selected programmer.

To select other type of programmer, programmer communication parameters and to establish communication with newly selected programmer use menu Programmer / Find programmer.

Programmer / Module options

This option is used for multiple socket programmers for defining MASTER socket and activity of each socket. MASTER socket group box allows user to set socket which is preferentially used for device reading operation. Enable/Disable socket checkbox array allows user to set enabling and disabling of each socket individually. Disabled sockets are ignored for any device operation.

Programmer / Automatic YES!

This command is used for setting Automatic YES! mode. In this mode you just take off the programmed device, then put new device into ZIF socket and a last operation will be repeated automatically. Program automatically detects an insertion of a new device and runs last executed operation

without pressing any key or button. An insertion of device into ZIF is displayed on the screen. Repeated operation executing will be cancelled by pressing key <ESC> during waiting for insert/remove a device to/from ZIF.

After an operation with a device is executed, one of the OK or ERROR (status) LEDs on the programmer will lights in dependence on the result of an operation and the BUSY LED will blinking.

If the program detects removal of a device, then status LED will switched off, but the BUSY LED will still blinking to indicate readiness of the program to repeat last operation with new device.

After the program indicates one or more pins of (new) device in the ZIF socket of the programmer, the BUSY LED will goes to light continually. From this the program will wait a requested time for insert the rest pins of new device. If a requested time (Device insertion complete time) overflows and a device is not correctly inserted, the program will lights the ERROR LED to indicate this state. After new device was inserted correctly, the program will switch off all status LEDs, except BUSY, and will start an operation with new device.

This mode may be enabled or disabled by item Automatic YES! mode. If a new programmer is selected Options / Find programmer, this mode will be disabled.

In Response time is interval between insertion of the chip into the ZIF socket and the start of selected device operation. If longer positioning of the chip in the ZIF socket is necessary select elongated response time.

In Pins with capacitors bar may be entered a list of a pins interconnected by capacitors (for example: if a converter, which have connected capacitor between VCC and GND, is used), which may makes problems at detecting insertion of a new device.

List of pins of device is in form:

pinA, pinB, pinC....

Example: 4,6,17

In Device removal hold off time is time period between you removed device from the ZIF socket and the time when software starts to check the socket for new device inserted. This interval is in seconds and must be from 1 to 120 (default value is 2 seconds).

In Device insertion complete time is possible to set a time within all pins of the device have to be properly inserted after a first pin(s) detected so that the program will not detects incorrectly inserted device. This interval is in seconds and must be from 1 to 120 (default value is 5 seconds).

The Suspend on error defines if the Automatic YES! function will be temporary disabled on error to see result of operation or will going on without suspension.

The options are set to defaults after new device is selected by Device / Select device

This setting is saved to disk by command Options / Save options.

Note: When using device socket adapters with some passive or active parts, for example capacitors for bypassing supply voltage, the Automatic YES! function may need to set these pins to Pins with capacitors list. This is necessary to make Automatic YES! function working properly. Otherwise Automatic YES! function will "think" the pins are still connected and it will not allow user to insert new device and start new programming.

Programmer / Selftest

Command executes a selftest of current programmer without diagnostic POD. We strongly recommend execute also Programmer / Selftest plus of programmer, because Selftest procedure without diagnostic POD is not able to check whole programmer and to discover (if exist) some special errors.

Programmer / Selftest plus

Command executes a selftest of current programmer using diagnostic POD, which is included in standard delivery of programmer.

For optimal results with programmer we recommend you undertake every 6 months.

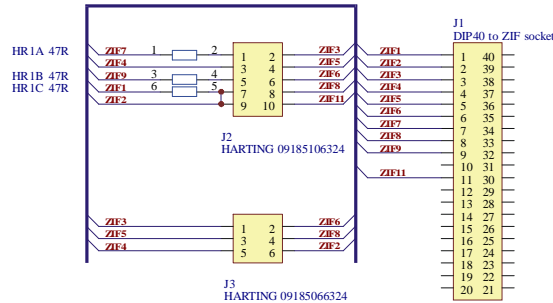
We recommend run this test as often as possible, e.g. once per month.

Programmer / Self test ISP connector

Command executes a selftest of ISP connector of current programmer using diagnostic POD for ISP connectors.

Diagnostic POD for ISP connectors is necessary to use for testing 6 and 10-pin ISP connectors of programmers. Diagnostic POD for ISP is available as optional accessory for ISP-capable programmers. The order number: 70-0208

Schematic of Diagnostic POD for ISP connector (if you are in hurry):



Sequence for testing 6 pins ISP connector:

1. Insert Diagnostic POD for ISP connectors into ZIF socket of the programmer. Diagnostic POD must be inserted as 40 pins device.
2. Interconnect 6 pins connector of Diagnostic POD with an ISP connector of the programmer with an ISP cable, included in programmer delivery package. Be sure that pins are interconnected properly (i.e. 1-1, 2-2, ..., 6-6).
3. Run selftest of ISP connector in M1882 (Programmer / Selftest ISP connector).

Sequence for testing 10 pins ISP connector:

1. Insert Diagnostic POD for ISP connectors into ZIF socket of the programmer. Diagnostic POD must be inserted as 40 pins device.
2. Interconnect 10 pins connector of Diagnostic POD with an ISP connector of the programmer with an ISP cable, included in delivery programmer package. Be sure that pins are interconnected properly (i.e. 1-1, 2-2, ..., 10-10).
3. Run selftest of ISP connector in M1882 (Programmer / Selftest ISP connector).

We recommend run this test every 6 months.

Options

The Options menu contains commands that let you view and change various default settings.

Options / General options

General options dialog allows user to control following options of program.

File options

File options page allows you to set file masks, auto-reload of current file and choose file format recognizing for loaded files.

File format masks is used for setting file-name masks to use as a filter for file listing in File / Save and File / Load file window for all file formats. Mask must contain one of wildcards at least and must be applied correctly by syntax.

Project file default extension is used for setting project files-extension used as default extension in File / Load project and File / Save project dialogs.

In group When current file is modified by another process can be set mode of reloading of actually loaded (current) file. There are three choices:

- Prompt before reloading file
- Reload automatically
- Ignore change scanning of current file

There are three situations when file modification is tested:

- switching to the control program from another application
- selecting the device operation Verify or Program
- when repeat of last device operation is selected in dialog "Repeat?"

Load file format allows to set mode of file format recognition for loading files. When automatic file format is selected, program analyses format of loading file and test file for each of supported formats that are available in program. If file format matches one of supported formats, the file is read to buffer in detected format.

Manual file format allows user to select explicitly wished file format from list of supported file formats. File may be loaded no completely or incorrectly, if file format does not match to user selected format.

Hex file options

This page contains several options for loading control by any of HEX formats. The first option sets erasing buffer (with desired value) automatically before the loading by any of HEX formats.

The second option sets a negative offset, which is used for data addresses modification by loading from any HEX file so, that data can be written to existing buffer addresses. Manual or Automatic negative offset mode can be set. We recommend automatic set of negative offset in special cases only. This option contain a heuristic analyze, which can treat some data in file incorrectly. There are especially critical files, which contain a fragmented addresses range and which exceeds a size of selected device - some block can be ignored. Automatic set of negative offset can be disabled by select of any special devices. No address range in files associated with special devices can be moved and no block can be removed from the file when reading the file. For special devices following negative offset options are available: Yes (negative offset is turned on) and No (negative offset is not used).

Example:

A file contents data by Motorola S - format. A data block started at address FFFF0H. It is a S2 format with length of address array of 3 bytes. For all data reading you can set a value of negative offset to FFFF0H. It means, that the offset will be subtracted from current real addresses and so data will be written from buffer address 0.

Warning: The value of negative offset is subtracted from real address and therefore a result of subtraction can be negative number.

Because take care of correct setting of this value.

Language

This page allows you to select another language for user interface such as menu, buttons, dialogs, information and messages. It also allows to select wished help file in another language. For another language support of user interface the language definition file is required.

Sound

Sound page allows user to select the sound mode of program. Program generates sounds after some activities, e.g. activities on device (programming, verifying, reading, etc.). Program generates sound also when warning or error message is displayed. User can now select sound from Windows system sound (required installed sound card), PC speaker or none sound.

In the panel Programmer internal speaker sound settings is possible to set sound options for some programmers with built-in internal speaker. Sound beeps are then generated from internal programmer speaker after each device operation for indicating device operation result – good or bad result.

Log file

This options associates with using of Log window. All reports for Log window can be written into the Log file too. The Log file name is "Report.rep" as default. The control program creates this file with name and directory specified in Log file name edit box.

Following Log file options are available:

- No default, content of Log window is not copied to Log file, i.e. all reports will be displayed to Log window only
- New deletes old Log file and creates new one during each start of control program
- Append adds Log window reports into existing Log file, If file does not exist, the new file will be created

Checkbox Add date information to Log file name allows user to set date information into Log file name specified by user in Log file name edit box. When the checkbox is checked, program automatically adds current date string into user specified Log file name by the following rules:

If user specified log file name has format:

<user_log_file_name>.<log_file_extension>

The name with added date will be:

<user_log_file_name><-yyyy-mmm-dd>.<log_file_extension>

The new part representing of date consists of yyyy - year, mmm - month and dd

- day.

Example: User specifies Log file name: c:\logs\myfile.log

The final log file name with added date will look like this (have a date November, 7th, 2006):

c:\logs\myfile-2006-nov-07.log

If do you wish to have log file name without any prefix before date information, you can specify the log file name as:

.<log_file_extension> - dot is the first in file name

Example: User specifies Log file name: c:\logs*.log

The final log file name with added date will look like this (have a date November, 7th, 2006):

c:\logs\2006-nov-07.log

Advanced options about Log file size limit are available too.

- option Use Log file text truncating when file size limit is reached - when checked, the Log file size limit is on. It means, that when Log file size reaches specified value, the part of text included in Log file will be truncated. When the option is unchecked, the size of Log file is unlimited, respectively is limited by free disk space only.
- option Maximum Log file size specifies the maximum size of Log file in kBytes.
- option Amount of truncated text specifies the percentage of Log file text, which will be truncated after Maximum Log file size is reached. The higher value means more text will be truncated (removed) from Log file.

The Log file settings can be saved to disk by command Options / Save options.

Display errors

This option allows to set a form of error displaying as a result of programmed data verifying. Errors can be displayed to the screen (max. 45 differences), saved to error file of differences on the disk or it will not be displayed. In case the displaying errors are turned off, the control program reports a warning message in INFO window only. The default error file name is "Verify.err". The file name and directory can be user specified in edit box Error file name.

Following Display errors settings are available:

- None does not display error values on screen nor to the file
- Screen default, displays errors to Log window
- File writes error reports to error file

The Display errors settings can be saved to disk by command Options / Save options.

Programmer

This option allows to set a form of error displaying as a result of programmed data verifying. Errors can be displayed to the screen (max. 45 differences), saved to error file of differences on the disk or it will not be displayed. In case the displaying errors are turned off, the control program reports a warning message in INFO window only. The default error file name is "Verify.err". The file name and directory can be user specified in edit box Error file name.

Remote control

Remote control of M1882 control program allows to control some functions of M1882 application by other application. This is very suitable feature for integrating device programmer to mass-production handler system or other useful application.

Remote application that controls M1882 acts as Server. Program M1882 acts as Client. Communication between M1882 and remote control program is made via TCP protocol - this allows the M1882 to be installed on one computer and remote control application to be installed on another computer, and these computers will be connected together via network.

Default TCP communication settings for remote control are:

Port: telnet Address: 127.0.0.1 or localhost

Address setting applies for M1882 (Client) only.

Port setting applies for M1882 (Client) and also for Server application.

Default settings allows to use remote control on one computer (address localhost). M1882 (Client) and remote control Server have to be installed on the same computer.

Note: If firewall is installed on system, firewall can display warning message when remote control Server or Client is starting. When firewall is showing warning with question asking to allow or deny network access for remote Server or Client, please select 'Allow' option, otherwise remote control will not work. Of course you can specify in firewall options more strict rights to allow remote Server/Client access on specified address and port only.

For more information about remote control of M1882 and demonstration remote control applications, please see the application note remotemanual.pdf placed in subdirectory ¥RemoteCtrl which is in the directory, where M1882 is installed. Manual for remote control is available also from Windows Start / Programs menu link to Remote manual, created during M1882 installation.

Save options

Page allows you to select the program options saving when exiting program. Three options are available here:

- Don't save don't save options during quitting program and don't ask for saving options
- Auto save save options during quitting program without asking for saving options
- Prompt for save program asks user for saving options before quitting program. User can select to save or not to save options

Other

Page Other allows user to manage other program settings.

Panel Application priority allows user to set the priority of the program. Priority settings can affect performance of programmer (device programming time), especially if there are running more demanding applications in the system. Please note that setting application priority level to Low can significantly slow down the program.

In the panel Tool buttons, hint display options on toolbar buttons in main program window can be modified. In the panel Start-up directory can be selected mode of selecting directory when program starts. Default start-up directory means directory, from which program is called. Directory in which program was lastly ended means the last current directory when program was lastly ended. This directory assumes the first directory from directory history list.

Options / View

Use the View menu commands to display or hide different elements of program environment such as toolbars.

Following toolbars are available now:

Options / View / Main toolbar

Choose this command to show or hide the Main toolbar.

Options / View / Additional toolbar

Choose this command to show or hide the Additional toolbar.

Options / View / Device options before device operation

Choose this command to enable/disable display of Device options before device operation is confirmed.

Options / Protected mode

Protected mode is special mode of program. When program is in Protected mode, there are disabled program operation and commands that can modify buffer or device settings. Protected mode is used for prevent operator from modify buffer or device settings due to insignificance. Protected mode is suitable for the programming of a large amount of the same type of devices.

Protected mode function is available independently in single programming control software M1882.

Protected mode in M1882

There are two ways how to switch program to Protected mode:

1. by using menu command Options / Protected mode.
This command displays password dialog.
User has to enter password twice to confirm the password is correct.
After password confirmation program switches to Protected mode.
The entered password is then used to switch off Protected mode.
2. by reading project, that was previously saved in Protected mode.
For details see File / Save project.

To switch program from Protected mode to normal mode, use the menu command Options / Normal mode. The "Password required" dialog appears. User has to enter the same password as the password entered during switch to Protected mode.

Other way to cancel Protected mode of program is closing of program, because program Protected mode is active until program is closed. The next program start will be to Normal (standard) mode (the only exception is case of project loaded by command line parameter name of project and the project was saved in Protected mode).

Protected mode in M1882

Program M1882 has Protected mode very similar to program M1882. The difference is, that Protected mode can be activated by menu command but cannot be activated by Project file. Another difference is, that Protected mode settings of M1882 are saved to configuration .ini file of M1882 while program M1882 is closed. During next start of application M1882 the recent Protected mode settings obtained from .ini file are used.

There is one menu command - Options / Protected mode - that allows to use Protected mode in application M1882. After selecting the menu Options / Protected mode, password dialog appears. User have to enter password twice to confirm the password is correct. After successful password confirmation program switches to Protected mode.

Protected mode settings are saved to configuration .ini file of M1882. During next start of program M1882 the Protected mode settings from .ini file are used.

There is also available one special option - Keep "Load project" operation allowed.
The option is set to disable at default - it means the Load project operation

button and menu will be disabled when Protected mode is active. If the option is enabled (checked), the Load project operation button and menu will be allowed in Protected mode.

To switch program from Protected mode back to Normal mode, use the menu command Options / Normal mode. The "Password required" dialog appears. User has to enter the same password as the password entered during switch to Protected mode.

When Protected mode is active, the label "Protected mode" is visible near the top of Log window of M1882 main window.

Note: Sometimes when Protected mode is switched from active state to inactive state (Normal mode), some commands (for example command "Load project") may remain disabled. This can be resolved by clicking on button Stop ALL.

Options / Save options

This command saves all settings that are currently supported for saving, even if auto-save is turned off. Following options are saved: options under the Options menu, ten last selected devices, file history, main program window position and size.

Help

Menu Help contains commands that let you view supported devices and programmers and information about program version too.

Pressing the <F1> key accesses the Help. When you are selecting menu item and press <F1>, you access context-sensitive help. If M1882 is executing an operation with the programmer <F1> generates no response.

The following Help items are highlighted:

- words describing the keys referred to by the current Help
- all other significant words
- current cross-references; click on this cross-reference to obtain further information.

Since the Help system is continuously updated together with the control program, it may contain information not included in this manual.

Detailed information on individual menu commands can be found in the integrated on-line Help.

Note: Information provided in this manual is intended to be accurate at the

moment of release, but we continuously improve all our products. Please consult manual on www.MINATO ELECTRONICS INC..com.

Help / Supported devices

This command displays list of all devices supported by at least one type of all supported programmers. It is useful especially when user wants to find any device supported by at least one type of programmers.

Prefix "g_" before name of device means the device is supported by multi-socket programmer.

Help / Supported programmers

This command displays information about programmers, where supported this program.

Help / Device list (current programmer)

This command makes a list of all devices supported by current programmer and saves it to DEV.txt text file and DEV.htm HTML file in the directory where control program is run from. Marks are replaced by abbreviated name of current programmer, the device list is generated for.

Help / Device list (all programmers)

This command makes device lists for all programmers and saves them to DEV.TXT text files and DEV.HTM HTML files in the directory where control program is running from. Characters are replaced by abbreviated name of programmers, the device lists are generated for.

Note: The control program loses all information about current device after this command is executed. Reselect wished device again by any of select methods in menu DEVICE.

Help / Device list (cross reference)

This command makes cross reference list of all devices supported by all programmers available on market and supported by this control program. The resulting list is in HTML format and consists of following files:

- one main HTML file TOP_DEV.htm with supported device manufacturers listed
- partial HTML files with list of supported devices for each device manufacturer

Main HTML file is placed to directory where this control program for programmers is located.

Partial HTML files are placed to subdirectory DEV_HTML placed to the directory where control program for programmers is located.

Programmer / Create problem report

Command Create problem report is used for writing more particular diagnostic information to Log window and consequently copy Log window content to clipboard. The Log window content can be placed from clipboard

to any text editor. Problem report is useful when error occurs in control program or programmer and kind of the error is, that user can not resolve it oneself and he must contact programmer manufacturer. In this case when customer send message to manufacturer about his problem it is good to send also problem report. Problem report can help manufacturer to localize the reason of error and resolve it sooner.

About

When you choose the Info command from the menu, a window appears, showing copyright and version information.

Warranty terms

The manufacturer, MINATO ELECTRONICS INC..gives a guarantee on failure-free operating of the programmer for one-year (M1882) from the date of purchase. If the product is diagnosed as defective, MINATO ELECTRONICS INC. or the authorized repair center will repair or replace defective parts at no charge. Parts used for replacement and/or whole programmer are warranted only for the remainder of the original warranty period.

For repair within the warranty period, the customer must prove the date of purchase.

This warranty terms are valid for customers, who purchase a programmer directly from MINATO ELECTRONICS INC. The warranty conditions of MINATO ELECTRONICS INC. sellers may differ depending on the target country law system or MINATO ELECTRONICS INC. seller's warranty policy.

The warranty does not apply to products that are of wear and tear or mechanically damaged. Equally, the warranty does not apply to products opened and/or repaired and/or altered by personnel not authorized by MINATO ELECTRONICS INC., or to products that have been misused, abused, accidentated or that were improperly installed.

For unwarrantable repairs you will be billed according to the costs of replacement materials, service time and freight. MINATO ELECTRONICS INC. or its distributors will determine whether the defective product should be repaired or replaced and judge whether or not the warranty applies.