

User's manual for **MEC** brand, **Program-mate** Series

UNP2

Universal 40-pin drive Programmer with ISP capability

MCU2

MCS51 Series and Atmel AVR Microcontrollers Programmer with ISP capability

PIC2

Microchip PICmicro Programmer with ISP capability

SEP2

Serial EEPROM Programmer



**COPYRIGHT © 2004
Mobicon**

This document is copyrighted by MOBICON. All rights reserved. This document or any part of it may not be copied, reproduced or translated in any form or in any way without the prior written permission of MOBICON.

The control program is copyright MOBICON. The control program or any part of it may not be analysed, disassembled or modified in any form, on any medium, for any purpose.

Information provided in this manual is intended to be accurate at the moment of release, but we continuously improve our products. Please consult manual on www.meceselectronic.com.

MOBICON assumes no responsibility for misuse of this manual.

MOBICON reserves the right to make changes or improvements to the product described in this manual at any time without notice. This manual contains names of companies, software products, etc., which may be trademarks of their respective owners. MOBICON respects those trademarks.

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 Users Manual.

Dear customer,

thank you for purchasing one of the **MEC Program-mate** programmer.

Note: *Since this User's manual is common for MEC Program-mate programmers, read section(s) for respective programmer, that you have bought, please.*

This manual contains two main sections:

Quick Start

Read this section if you are an experienced user. You will find only specific information regarding installation of the control program and use of your programmer. For more detailed instructions you may read the **Detailed description** section or the **Troubleshooting** section for the respective programmer.

Detailed description

Read this section for the respective programmer if you are a less experienced user or if you need detailed information. You may find some less relevant features of programmer described here, but all programmer features are described in this section along with details regarding installation of the control program. Read this section to explore all of the features provided by your programmer.

Please, download actual version of manual from MEC WEB site (www.mecelctronic.com), if current one will be out of date.



*Program
mate*



Table of contents

How to use this manual	3
Introduction	7
Products configuration	10
PC requirements	11
Quick Start	13
Detailed description	15
UNP2	17
Introduction	18
UNP2 elements	20
Connecting UNP2 to PC	21
In-system serial programming by UNP2	22
Selftest and calibration	24
Technical specification	25
MCU2	29
Introduction	30
Connecting MCU2 programmer to PC	32
In-System serial programming by MCU2	33
Selftest and calibration	35
Technical specification	36
PIC2	39
Introduction	40
Connecting PIC2 programmer to PC	42
In-System serial programming by PIC2	43
Selftest and calibration	45
Technical specification	46
SEP2	49
Introduction	50
Connecting SEP2 programmer to PC	51
Technical specifications	53
Software	55
The programmer software	56
File	60
Device	64
Buffer	80
Options	86
Diagnostics	92
Help	93
Common notes	95
Software	96
Hardware	98
ISP (In-System Programming)	99
Other	104
Troubleshooting	105



Conventions used in the manual

References to the control program functions are in bold, e.g. **Load, File, Device**, etc. References to control keys are written in brackets <>, e.g. <F1>.

Terminology used in the manual:

- Device** any kind of programmable integrated circuits or programmable devices
- ZIF socket** Zero Insertion Force socket used for insertion of target device
- Buffer** part of memory or disk, used for temporary data storage
- Printer port** type of port of PC (parallel), which is primarily dedicated for printer connection.
- HEX data format** - format of data file, which may be read with standard text viewers; e.g. byte 5AH is stored as characters '5' and 'A', which mean bytes 35H and 41H. One line of this HEX file (one record) contains start address and data bytes. All records are secured with checksum.

Introduction



This user's manual covers MEC Program-mate programmers:
UNP2, MCU2, PIC2 and SEP2.

UNP2 is a small, fast and powerful programmer of all kinds of programmable devices. Using build-in in-circuit serial programming (ISP) connector the programmer is able to program ISP capable chips in-circuit. It has design, which allows easily add new devices to the device list. Nice "value for money" in this class.

MCU2 is little, powerful and very fast portable programmer for MCS51 series and Atmel AVR microcontrollers with ISP capability. MCU2 enables also programming serial EEPROM with interface types IIC (24Cxx), Microwire (93Cxx) and SPI (25Cxx).

PIC2 is little, very fast and powerful portable programmer for PICmicro® family microcontrollers and serial EEPROM with IIC (24Cxx), Microwire (93Cxx) and SPI (25Cxx) interface types. Using build-in in-circuit serial programming (ISP) connector programmer is able to program PICmicro® family microcontrollers using serial algorithms.

SEP2 is universal programmer of all serial EEPROM in 8 pin DIL package. SEP2 programs EEPROM with interface IIC, SPI and Microwire, and also specialty as for example digital thermometers. The programmer supports LV (3.3V) devices too.

All programmers of our works with almost any IBM compatible PC, AT or higher, portable or desktop personal computers. No special interface card is required to connect to the PC, since programmers use the parallel (printer) port.

All programmers flawlessly work under WIN95/98/Me/NT/2000/XP.

All programmers are driven by an **easy-to-use, control program** with pull-down menus, hot keys and online help. Control program is common for MEC's Program-mate programmers (UNP2, MCU2, PIC2 and SEP2).

Free additional services:

- free technical support (phone/fax/e-mail).

Free software updates are available from our Internet address www.mecelectronic.com.

Note: *We don't recommend use programmers SEP2 for In-circuit programming.*



Products configuration

Before installing and using your programmer, please carefully check that your package includes all next mentioned parts. If you find any discrepancy with respective parts list and/or if any of these items are damaged, please contact your distributor immediately.

UNP2, MCU2 and PIC2 programmer configuration

- programmer
- cable with two 25 pin, D-type connectors for connecting the programmer to the PC
- external power supply (suitable for respective programmer)
- diagnostic POD for selftest of programmer
- cap for ZIF socket (anti-dust cover)
- CD with the control program, user's manual and additional files
- shipping case

SEP2 programmer configuration

- programmer
- cable with two D-type connectors for connecting the programmer to the PC
- external power supply (suitable for respective programmer)
- CD with the control program, user's manual and additional files
- shipping case

PC requirements

Minimal PC requirements

- PC 486
- 16MB RAM
- one CD drive
- HDD, 30 MB free space
- operating system WIN95/98/Me/NT/2000/XP
- one free printer port with nothing attached

Recommended PC requirements

- Pentium PC 100MHz or higher
- 32 MB free RAM
- one CD drive
- HDD with minimum 30 MB free space
- operating system: WIN95/98/Me/NT/2000/XP
- one free bi-directional printer port with nothing attached
- for UNP2, MCU2 and PIC2 free parallel (printer) port on PCI bus, IEEE 1284 compatible (ECP/EPP)

Note: *For convenience, we suggest that you use a supplementary multi I/O card to provide an additional printer port (LPT2 for example), in order to avoid sharing the same LPT port between printer and programmer.*

Quick Start



Installing programmer hardware

- switch off the PC and programmer
- connect the communication port of programmer to a printer port of PC using cable supplied
- switch on the PC
- connect the connector of the power supply adapter to the programmer

Installing the programmer software

Run the installation program from the CD (Setup.exe) and follow the on-screen instructions. Please, for latest information about the programmer hardware and software see www.mecelectronic.com.

Using programmer software

Launch PG4UW.EXE to enter the control program. The menu **Device** contains the device manipulation commands. The menu **File** contains commands for files and directories. The menu **Buffer** is to be used for buffer manipulation.

Programming a device - the shortest way

Use the hot key **<Alt+F5>** to input the device name and/or manufacturer to select the desired type of target device. If you want to copy an existing device, insert it into the ZIF socket of the programmer and then press key **<F7>**. If you want to program a target device with data from a disk press key **<F3>** and read the appropriate file into the buffer. Then insert your target device into the ZIF socket. To check if the device is blank - press key **<F6>**. Now you can program the device by pressing key **<F9>**. After programming you may perform additional verification by pressing key **<F8>**.

Detailed description

UNP2





Introduction

UNP2 is a new generation of WIN95/98/Me/NT/2000/XP based MEC Program-mate universal programmers. Programmer is built to meet the demands of the development labs and field engineers to universal, but portable programmer.

UNP2 is a small, fast and powerful programmer of all kinds of programmable devices. Using build-in in-circuit serial programming (ISP) connector the programmer is able to program ISP capable chips in-circuit.

Provides very competitive price but excellent hardware design for reliable programming. Nice "value for money" in this class.

Very fast programming due to high-speed FPGA driven hardware and support of IEEE1284 (ECP/EPP) high-speed parallel port. Surely faster than competitors in this category.

UNP2 interfaces with the IBM PC, AT or above, portable or desktop personal computers through any standard parallel (printer) port (no special interface card needed). Therefore you can take programmer and move it to another PC without assembly/disassembly of PC.

UNP2 has 40 powerful TTL pindrivers 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. Pin drivers operate down to 1.8V so you'll be ready to program the full range of today's advanced low-voltage devices.

The programmer performs **device insertion test** (wrong device position in socket) and contact check (poor contact pin-to-socket) before it programs each device. These capabilities, supported by signature-byte check help prevent chip damage due to operator error.

Programmer's hardware offers enough resources for **selftest**, that control program is any time be able to check pindrivers, present and correct level of all voltages, check the timing and communication between programmer and PC.

UNP2 programmer performs programming **verification** at the marginal level of supply voltage, which, obviously, improves programming yield, and guarantees long data retention.

UNP2 is driven by an easy-to-use control program with pull-down menus, 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 enhanced by some test functions (insertion test, signature-byte check), and some special functions (autoincrement). All known data formats are supported. Automatic file format detection and conversion during load of file.

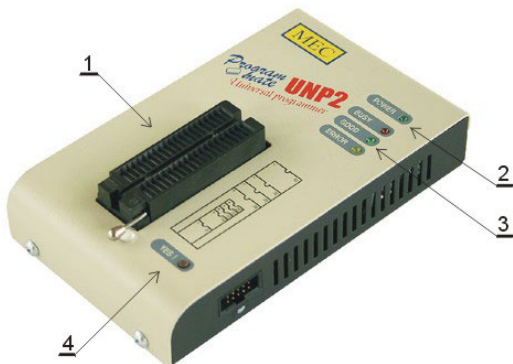
The rich-featured **autoincrement function** enables to assign individual serial numbers to each programmed device - or simply increments a serial number, or the function enables to read serial numbers or any programmed device identification signatures from a file.

Various socket converters are available to handle device in PLCC, SOIC and other packages.

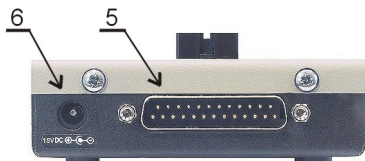


UNP2 elements

- ① 40 pin ZIF socket
- ② LED power/sleep
- ③ LED, which indicate work result
- ④ YES! button



- ⑤ Connector for PC ↔ UNP2 communication cable
- ⑥ Power supply connector



- ⑦ Connector for ISP



Note: Due to low power consumption of UNP2 in inactive state, it doesn't require power switch. When the power LED indicator glows with a low intensity the UNP2 is in inactive mode.

Connecting UNP2 to PC

Switch off the PC and programmer. Insert the connection cable, included in the UNP2 programmer delivery, to the free printer port of 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 mainly for the connector to programmer. Though replacing the printer cable by the programmer cable is uncomfortable, it is not recommended to operate the UNP2 programmer through a mechanical printer switch. Use of an electronic printer switch isn't possible.

Connect the mains connector of the power supply (or wall-plug power supply self) to a mains plug, connect the 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 UNP2 is ready to run.

Next switch on the PC and run the control program.

Caution! *If you don't want to switch off your PC when connecting the UNP2, 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**.

Problems related to the UNP2 ⇔ PC interconnection, and their removing

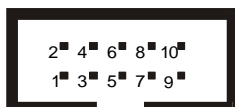
If you have any problems with UNP2 ⇔ PC interconnection, see section **Common notes** please.



In-system serial programming by UNP2

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

Description of UNP2 ISP connector



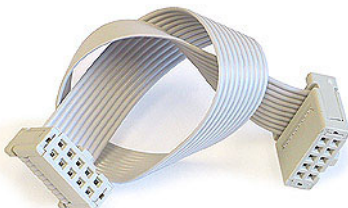
Front view at ISP connector of programmer.

Pin	Capability of ISP connector pins Description
1	VCCP for target device, with sense
2, 10	H/L/read, GND, VCCP, VPP
3	H/L/read, GND, VPP
4	H/L/, read, VPP
5	NC
6	H/L/read, GND, VCCP
7,9	GND
8	H/L/read, GND

Specification of ISP connector pins depends on the device, which you want to program. You can find it in the control SW for programmer (PG4UW), 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.

These specifications correspond with application notes published of device manufacturers. Used application notes you may find on www.meceselectronic.com.

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



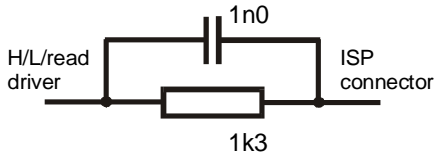
UNP2 ISP cable

Warnings:

- **When** you use UNP2 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.

- **UNP2 can supply** programmed device only, but target system **cannot supply UNP2**.
- UNP2 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.

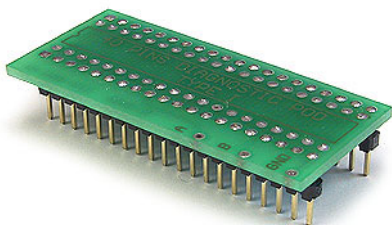
Note: H/L/read UNP2 driver





Selftest and calibration

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. For optimal results with programmer we recommend you undertake every 6 months, an extended test and to check the calibration. See instructions for selftest in the **Diagnostics** menu of PG4UW.



Manipulation with the programmed device

After selection of desired device for your work, you can insert into the open ZIF socket (the lever is up) and close socket (the lever is down). The correct orientation of the programmed device in ZIF socket is shown on the picture near ZIF socket on the programmer's cover. The programmed device is necessary to insert into the socket also to remove from the socket when LED BUSY light off.

Technical specification

HARDWARE

Programmer

- two D/A converters for VCCP and VPP, controllable rise and fall time
- VCCP range 0..7V/350mA
- VPP range 0..25V/200mA
- FPGA based IEEE 1284 slave printer port, up to 1MB/s transfer rate
- autocalibration
- selftest capability

ZIF socket, pindriver

- 40-pin DIL ZIF (Zero Insertion Force) socket accepts both 300/600 mil devices up to 40-pins
- pindriver: 40 TTL pindrivers, universal GND/VCC/VPP pindriver
- FPGA based TTL driver provides H, L, CLK, pull-up, pull-down on all pindriver pins, level H selectable from 1.8 V up to 5V
- in-circuit serial programming (ISP) capability included
- continuity test: each pin is tested before every programming operation

DEVICE SUPPORT

Programmer

- EPROM: NMOS/CMOS, 2708 (*2), 27xxx and 27Cxxx series, with 8/16 bit data width, full support of LV series (*1)
- EEPROM: NMOS/CMOS, 28xxx, 28Cxxx, 27EExxx series, with 8/16 bit data width, full support of LV series (*1)
- Flash EPROM: 28Fxxx, 29Cxxx, 29Fxxx, 29BVxxx, 29LVxxx, 29Wxxx, 49Fxxx series, with 8/16 bit data width, full support of LV series (*1)
- Serial E(E)PROM: 17Cxxx, 24Cxxx, 24Fxxx, 25Cxxx, 59Cxxx, 85xxx, 93Cxxx, series, full support of LV series
- Configuration PROM: 17xxx, LV series including
- NV RAM: Dallas DSxxx, SGS/Inmos MKxxx, SIMTEK STKxxx, XICOR 2xxx, ZMD U63x series
- PLD: series: Atmel, AMD-Vantis, Cypress, ICT, Lattice, NS (*1)
- microcontrollers MCS51 series: 87Cxxx, 87LVxx, 89Cxxx, 89Sxxx, 89LVxxx series from Atmel, Atmel W&M, Intel, Philips, SST, Winbond, (*1*2), parallel and serial (ISP) mode



- microcontrollers Atmel AVR: ATtiny, AT90Sxxx, ATmega series, (*1*2), parallel and serial (ISP) mode
- Microcontrollers Microchip PICmicro: PIC12Cxxx, PIC16C5x, PIC16Cxxx, PIC17Cxxx, PIC18Cxxx series, 8-40 pins (*1*2), parallel and serial (ISP) mode
- Microcontrollers Scenix (Ubicom): SX18xxx, SX20xxx, SX28xxx series

Notes:

- (*1) - suitable adapters are available for non-DIL packages
- (*2) - there exist only few adapters for devices with more than 40 pins. Therefore think please about more powerful programmer (UNP8...), if you need to program devices with more than 40 pins

Look please at list of all supported devices.

I.C. Tester

- Static RAM: 6116 .. 624000

Programming speed

Note. These times strongly depend on PC speed, LPT port type and operating system free resources. Therefore values of two different PC configurations are given for comparison.

Device	Operation	Mode	Time A	Time B
27C010	programming and verify	in ZIF	28 sec	24 sec
AT29C040A	programming and verify	in ZIF	38 sec	28 sec
AM29F040	programming and verify	in ZIF	102 sec	87 sec
PIC16C67	programming and verify	in ZIF	13 sec	11 sec
PIC18F452	programming and verify	in ZIF	11 sec	9 sec
AT89C52	programming and verify	in ZIF	18.5 sec	16.5 sec
PIC16F873A	programming and verify	ISP	7 sec	6 sec
PIC12C508	programming and verify	ISP	11 sec	9 sec

Time A conditions: Pentium MMX, 250 MHz, ECP/EPP, WIN98.

Time B conditions: Athlon, 750 MHz, ECP/EPP on PCI bus, WIN98.

SOFTWARE

- **Algorithms:** only manufacturer approved or certified algorithms are used. Custom algorithms are available at additional cost.
- **Algorithm updates:** software updates are available approx. every 2 weeks, free of charge.
- **Main features:** revision history, session logging, on-line help, device and algorithm information

Device operations

- **standard:**
 - intelligent device selection by device type, manufacturer or typed fragment of part name
 - blank check, read, verify
 - program
 - erase
 - configuration and security bit program
 - illegal bit test
 - checksum
- **security**
 - insertion test
 - contact check
 - ID byte check
- **special**
 - auto device serial number increment
 - 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, JEDEC

PC system requirements

See section *Introduction/ PC requirements*

GENERAL

- operating voltage 15..20V DC, max. 500mA
- power consumption max. 6W active, 1.4W inactive
- dimensions 160x95x35 [mm] (6.3x3.8x1.4 [inch])
- weight (without external power adapter) ca. 500g
- temperature 5°..40°C
- humidity 20%..80%, non condensing



Package includes

- UNP2 programmer
- connection cable PC-programmer
- ISP cable
- diagnostic POD for selftest
- anti-dust cover for ZIF socket
- wall plug adapter 15V DC/500mA, unstabilized
- CD contains control program software, user's manual and additional files

Additional services

- free technical support (hot line)

MCU2





Introduction

MCU2 is a new generation of WIN95/98/Me/NT/2000/XP based MEC Program-mate specialized programmers. Programmer is capable to support all today available microcontrollers of MCS51 series (up to 40 pins) and AVR microcontrollers (8-40 pins) by parallel and serial way. MCU2 has been developed in close cooperation with Atmel W&M, therefore programmer's hardware is focused to support all current and future microcontrollers of Atmel W&M MCS51 family.

MCU2 is little, very fast and powerful portable programmer for MCS51 series and Atmel AVR microcontrollers. MCU2 enables also programming of serial EEPROM with IIC (24Cxx), Microwire (93Cxx) and SPI (25Cxx) interface types. Using build-in in-circuit serial programming (ISP) connector programmer is capable to program MCS51 family microcontrollers and Atmel AVR microcontrollers in serial way.

Provides very competitive price but excellent hardware design for reliable programming. Nice "value for money" in this class.

Very fast programming due to high-speed FPGA driven hardware and support of IEEE1284 (ECP/EPP) high-speed parallel port. Surely faster than competitors in this category.

MCU2 interfaces with the IBM PC, AT or higher, portable or desktop personal computers. A programmer allows you to directly connect to your PC through any standard parallel (printer) port - no special interface card is needed.

MCU2 has 40 powerful TTL pindrivers 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 programming without overshoot or ground bounce for all supported devices. Pin drivers operate down to 1.8V so you'll be ready to program the full range of today's advanced low-voltage devices.

The programmer 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 signature-byte check help prevent chip damage due to operator error.

Programmer's hardware offers enough resources for **selftest**, that control program is any time be able to check pindrivers, present and correct level of all voltages, check the timing and communication between programmer and PC.

MCU2 performs programming **verification** at the marginal level of supply voltage, which, obviously, improves programming yield, and guarantees long data retention

MCU2 is driven by an easy-to-use comfortable software with pull-down menu, hot keys and on-line help. You can select device by classes, by manufacturers or simply type device vendor's name and part number. The standard device operation functions (read, blank check, program, verify) have been completed with some test functions. The program facilitates the use of the buffer and files, including automatic file format detection and conversion.

The control program provides an **auto-increment function** that enables you to assign individual serial numbers to each programmed device, this function simply increments a serial number in the buffer each time a new device is inserted in the socket. Furthermore, the function enables the operator to read serial numbers and/or any programmed device identification signatures from a file.

For MCU2 are available DIL to PLCC and SOIC socket converters.



Connecting MCU2 programmer to PC

Switch off the PC and programmer. Insert the connection cable, included in the MCU2 programmer delivery, to the free printer port of 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 mainly for the connector to programmer. Though replacing the printer cable by the programmer cable is uncomfortable, it is not recommended to operate the MCU2 programmer through a mechanical printer switch. Use of an electronic printer switch isn't possible.

Connect the mains connector of the power supply (or wall-plug power supply self) to a mains plug, connect the connector to the appropriate programmer's connector. Then, on the programmer lights up LED POWER and the programmer MCU2 is ready to run. Next switch on the PC and run the control program.

Caution! *If you don't want to switch off your PC when connecting the MCU2, 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**.

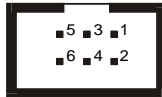
Problems related to the MCU2 ⇔ PC interconnection, and their removing

If you have any problems with MCU2 ⇔ PC interconnection, see section **Common notes** please.

In-System serial programming by MCU2

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

Description of MCU2 ISP connector



Front view at ISP connector of programmer.

Capability of ISP connector pins

Pin	Description
1,3,4,5	H/L/read
2	target VCC sense
6	GND

Specification of ISP connector pins depends on the device, which you want to program. You can find it in the control SW for programmer (PG4UW), 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.

These specifications correspond with Atmel application note AVR910: In-System Programming. Used application note you may find on www.meceselectronic.com.



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

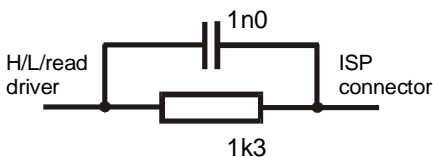
ISP cable of MCU2



Warnings:

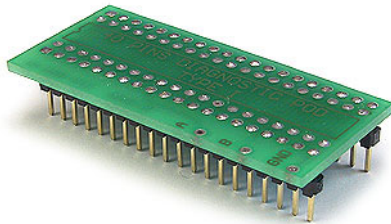
- **When** you use MCU2 as **ISP programmer**, **don't insert** device to **ZIF socket**.
- **When** you program devices in **ZIF socket**, **don't insert** ISP cable to **ISP connector**.
- Use only **attached ISP cable**. When you use other ISP cable (other material, length...), programming may occur unreliable.
- **MCU2 cannot** supply target system and **target system cannot supply MCU2**. Before action with target device MCU2 check power supply of target system. If this power supply is different as expected, no action with device will be executed.

Note: H/L/read MCU2 driver.



Selftest and calibration

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. For optimal results with programmer we recommend you undertake every 6 months, an extended test and to check the calibration. See instructions for selftest in the **Diagnostics** menu of PG4UW.



Manipulation with the programmed device

After selection of desired device for your work, you can insert into the open ZIF socket (the lever is up) and close socket (the lever is down). The correct orientation of the programmed device in ZIF socket is shown on the picture near ZIF socket on the programmer's cover. The programmed device is necessary to insert into the socket also to remove from the socket when LED BUSY light off.

Warning! *MCU2 programmer hasn't protection devices, which protect the content of programmed device against critical situations, for example power failures and PC failure (interrupted cable...). Moreover, a device is usually destroyed in the programming mode due to forced interruption of the control program run (Reset or switching the computer off) due to removing the connecting cable, or unplugging the programmed device from the ZIF socket. Incorrectly placed device in the ZIF socket can cause its damage or destruction.*



Technical specification

Socket, pin drivers and DACs

- FPGA based IEEE 1284 slave printer port, up to 1MB/s transfer rate
- 40-pin DIL ZIF (Zero Insertion Force) socket accepts both 300/600 mil devices up to 40-pin
- two D/A converters for VCCP and VPP, controllable rise and fall time
- VCCP range 0 – 6.5V / 150mA
- VPP range 0 – 15V / 100mA
- special GND/VCCP/VPP pindriver for MCS51 and AVR devices
- spare GND, VCCP and VPP driver, which add additional made-by-wire GND/VCCP/VPP pin capability for future devices
- FPGA based TTL driver provides H, L, CLK, pull-up, pull-down on and read for all pindriver pins,
- level H selectable from 1.8 V up to 5V
- in-circuit serial programming (ISP) capability included

Device support

- microcontrollers 51 series: 87Cxxx, 87LVxx, 89Cxxx, 89Sxxx, 89LVxxx series from Atmel, Atmel W&M, Intel, Philips, ... 8/40 pins
- microcontrollers Atmel AVR: ATtiny, AT90Sxxx, ATmega series (parallel and serial mode), 8-40 pins
- serial E(E)PROM: 24Cxxx, 24Fxxx, 25Cxxx, 59Cxxx, 85xxx, 93Cxxx series

Programming speed

Note. These times strongly depend on PC speed, LPT port type and operating system free resources. Therefore are given values of two different PC configurations for comparison.

Device	Operation	Time A	Time B
AT89C52	programming and verify	17,5 sec	15,5 sec
T87C5111	programming and verify	45 sec	25 sec

Time A conditions: Pentium MMX, 250 MHz, ECP/EPP, WIN98.

Time B conditions: Athlon, 750 MHz, ECP/EPP on PCI bus, WIN98.

Device operations

- **standard:**
 - intelligent device selection by device type, manufacturer or typed fragment of part name
 - blank check
 - read
 - program
 - verify
 - erase
 - configuration and security bit program
- **security:**
 - insertion test, reverse insertion check
 - contact check
 - ID byte check
- **special:**
 - statistic
 - count-down mode
 - auto device serial number increment

Buffer operations

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

Supported file formats

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

PC system requirements

See section *Introduction/ PC requirements*

General

- operating voltage 12..15V DC, max. 500mA
- power consumption 5W max.
- dimensions 132x66x40 [mm] (5.2x2.6x1.6 [inch])
- weight (without external adapter) ca. 200g
- temperature 5° ÷ 40°C
- humidity 20%..80%, non condensing



Package includes

- MCU2 programmer
- connection cable PC-programmer
- ISP cable
- diagnostic POD for selftest
- anti-dust cover for ZIF socket
- wall plug adapter, 12V DC/500mA, unstabilized
- CD contains control program software, user's manual and additional files

Additional services

- free technical support (phone/fax/e-mail).

PIC2





Introduction

PIC2 is a new generation of WIN 95/98/ME/NT/2000/XP based MEC Program-mate specialized programmers. Programmer is capable to support all currently available Microchip™ PICmicro® series microcontrollers (8-40 pins) using parallel and serial algorithms. The PIC2 programmer isn't development grade programmer, but meets all Microchip's requirements to manufacturing grade of programmers. The PIC2 has been developed in close cooperation with Microchip™ Company, therefore programmer's hardware is focused to support all current and future PICmicro® family microcontrollers.

PIC2 is a small, very fast and powerful portable programmer for PICmicro® family microcontrollers and serial EEPROM with IIC (24Cxx), Microwire (93Cxx) and SPI (25Cxx) interface types. Using build-in in-circuit serial programming (ISP) connector programmer is able to program PICmicro® family microcontrollers using serial algorithms.

Provides very competitive price but excellent hardware design for reliable programming. Nice "value for money" in this class.

Very fast programming due to high-speed FPGA driven hardware and support of IEEE1284 (ECP/EPP) high-speed parallel port. Surely faster than competitors in this category.

PIC2 interfaces with the IBM PC, AT or above, portable or desktop personal computers through any standard parallel (printer) port (no special interface card needed). Therefore you can take programmer and move it to another PC without assembly/disassembly of PC.

PIC2 has 40 powerful TTL pindrivers 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. Pin drivers operate down to 1.8V so you'll be ready to program the full range of today's advanced low-voltage devices.

The programmer performs **device insertion test** (wrong device position in socket) and contact check (poor contact pin-to-socket) before it programs each device. These capabilities, supported by signature-byte check help prevent chip damage due to operator error.

PIC2 programmer performs programming **verification** at the marginal level of supply voltage, which, obviously, improves programming yield, and guarantees long data retention.

PIC2 is driven by an easy-to-use control program with pull-down menus, 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 enhanced by some test functions (insertion test, signature-byte check), and some special functions (autoincrement). 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.

Various socket converters are available to handle device in PLCC, SOIC and other packages.



Connecting PIC2 programmer to PC

Switch off the PC and programmer. Insert the connection cable, included in the PIC2 programmer delivery, to the free printer port of 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 mainly for the connector to programmer. Though replacing the printer cable by the programmer cable is uncomfortable, it is not recommended to operate the PIC2 programmer through a mechanical printer switch. Use of an electronic printer switch isn't possible.

Connect the mains connector of the power supply (or wall-plug power supply self) to a mains plug, connect the connector to the appropriate programmer's connector. Then, on the programmer lights up LED POWER and the programmer PIC2 is ready to run. Next switch on the PC and run the control program.

Caution! *If you don't want to switch off your PC when connecting the PIC2, 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**.

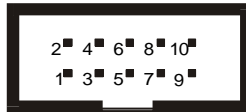
Problems related to the PIC2 ⇔ PC interconnection, and their removing

If you have any problems with PIC2 ⇔ PC interconnection, see section **Common notes** please.

In-System serial programming by PIC2

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

Description of PIC2 ISP connector



Front view at ISP connector of programmer.

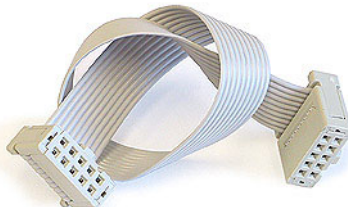
Capability of ISP connector pins

Pin	Description
2,3,4,6,8	H/L/read
1	VCCP for target device
7,9	GND
10	H/L/read, VPP on MCLR\
5	NC

Specification of ISP connector pins depends on the device, which you want to program. You can find it in the control SW for programmer (PG4UW), 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.

These specifications correspond with Microchip application note: In-Circuit Serial Programming™(ICSP™) Guide. Used application note you may find on www.mecelronic.com.

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



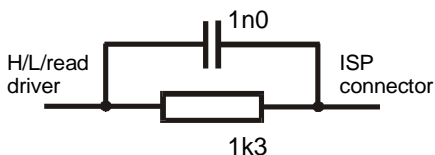
ISP cable of PIC2



Warnings:

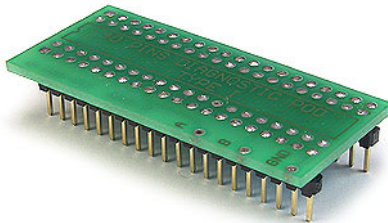
- **When** you use PIC2 as **ISP programmer**, **don't** insert device to **ZIF socket**.
- **When** you program devices in **ZIF socket**, **don't** insert ISP cable to **ISP connector**.
- Use only **attached ISP cable**. When you use other ISP cable (other material, length...), programming may occur unreliable.
- **PIC2 can supply** programmed device only, but target system **cannot supply PIC2**.
- PIC2 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 PIC2 driver



Selftest and calibration

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. For optimal results with programmer we recommend you undertake every 6 months, an extended test and to check the calibration. See instructions for selftest in the **Diagnostics** menu of PG4UW.



Manipulation with the programmed device

After selection of desired device for your work, you can insert into the open ZIF socket (the lever is up) and close socket (the lever is down). The correct orientation of the programmed device in ZIF socket is shown on the picture near ZIF socket on the programmer's cover. The programmed device is necessary to insert into the socket also to remove from the socket when LED BUSY light off.

Warning! *PIC2 programmer hasn't protection devices, which protect the content of programmed device against critical situations, for example power failures and PC failure (interrupted cable...). Moreover, a device is usually destroyed in the programming mode due to forced interruption of the control program run (Reset or switching the computer off) due to removing the connecting cable, or unplugging the programmed device from the ZIF socket. Incorrectly placed device in the ZIF socket can cause its damage or destruction.*



Technical specification

Socket, pin drivers and DACs

- FPGA based IEEE 1284 slave printer port, up to 1MB/s transfer rate
- 40-pin DIL ZIF (Zero Insertion Force) socket accepts both 300/600 mil devices up to 40-pins
- two D/A converters for VCCP and VPP, controllable rise and fall time
- VCCP range 0..7V/250mA
- VPP range 0..15V/150mA
- pindriver: 40 TTL pindrivers, special GND/VCC/VPP pindriver for PICmicro® devices
- spare GND, VCC and VPP driver, which add additional made-by-wire GND/VCC/VPP pin capability for future devices
- FPGA based TTL driver provides H, L, CLK, pull-up, pull-down on all pindriver pins
- level H selectable from 1.8 V up to 5V
- in-circuit serial programming (ISP) capability included
- continuity test: each pin is tested before every programming operation
- selftest capability
- autocalibration

Device support

- microcontrollers Microchip™ PICmicro®: 12xxx, 14xxx, 16xxx, 17xxx and 18xxx series, 8 to 40 pin (*1), parallel and serial mode
- Serial E(E)PROM: 24Cxxx, 24Fxxx, 25Cxxx, 59Cxxx, 85xxx, 93Cxxx series
(*1) - suitable adapters are available for non-DIL packages and devices with more than 40 pins

Programming speed

Note. These times strongly depend on PC speed, LPT port type and operating system free resources. Therefore are given values of two different PC configurations for comparison.

Device	Operation	Mode	Time A	Time B
PIC16C67	programming and verify	in ZIF	13 sec	11 sec
PIC18F452	programming and verify	in ZIF	11 sec	9 sec
PIC16F873A	programming and verify	ISP	7 sec	6 sec
PIC12C508	programming and verify	ISP	3 sec	2.5 sec

Time A conditions: Pentium MMX, 250 MHz, ECP/EPP, WIN98.

Time B conditions: *Athlon, 750 MHz, ECP/EPP on PCI bus, WIN98.*

Device operations

- **standard:**
 - intelligent device selection by device type, manufacturer or typed fragment of part name
 - blank check, read, verify
 - program
 - erase
 - configuration and security bit program
 - illegal bit test
 - checksum
- **security**
 - insertion test
 - contact check
 - ID byte check
- **special**
 - auto device serial number increment
 - 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

PC system requirements

See section *Introduction/ PC requirements*

General:

- operating voltage 12..15V DC, max. 500mA
- power consumption max. 5W active
- dimensions 132x66x40 [mm] (5.2x2.6x1.6 [inch])
- weight (without external power adapter) ca. 200g
- temperature 5°..40°C



-
- humidity 20%..80%, non condensing

Package includes

- PIC2 programmer
- connection cable PC-programmer
- ISP cable
- diagnostic POD for selftest
- anti-dust cover for ZIF socket
- wall plug adapter, 12V DC/500mA, unstabilized
- CD contains control program software, user's manual and additional files

Additional services

- free technical support (phone/fax/e-mail).

SEP2





Introduction

SEP2 is universal programmer of all serial EEPROM in 8-pin DIL package. SEP2 programs EEPROM with interface IIC, SPI and Microwire, and also specialty as for example digital thermometers. The programmer supports LV (3.3V) devices too.

SEP2 interfaces with the IBM PC, AT or higher, portable or desktop personal computers. Programmer allows you to directly connect to your PC through any standard parallel (printer) port - no special interface card is needed.

SEP2 is driven by an easy-to-use comfortable software with pull-down menu, hot keys and on-line help. You can select device by classes, by manufacturers or simply type device vendor's name and part number. The standard device operation functions (read, blank check, program, verify) have been completed with some test functions. The program facilitates the use of the buffer and files, including automatic file format detection and conversion.

For **SEP2** programmer are available DIL to SOIC socket converters.

Connecting SEP2 programmer to PC

Switch off the PC and programmer. Insert the connection cable, included in the SEP2 programmer delivery, to the free printer port of 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 mainly for the connector to programmer. Though replacing the printer cable by the programmer cable is uncomfortable, it is not recommended to operate the SEP2 programmer through a mechanical printer switch. Use of an electronic printer switch isn't possible.

Connect the mains connector of the power supply (or wall-plug power supply self) to a mains plug, connect the connector to the appropriate programmer's connector. Then, on the programmer lights up LED POWER and the programmer SEP2 is ready to run. Next switch on the PC and run the control program.

Caution! *If you don't want to switch off your PC when connecting the SEP2, 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**.

Problems related to the SEP2 ↔ PC interconnection, and their removing

If you have any problems with SEP2 ↔ PC interconnection, see section **Common notes** please.

Manipulation with the programmed device

After selection of desired device for your work, you can insert into the open ZIF socket (the lever is up) and close socket (the lever is down). The correct orientation of the programmed device in ZIF socket is shown on the picture near ZIF socket on the programmer's. The programmed device is necessary to insert into the socket also to remove from the socket when LED BUSY light off.

Warning! *SEP2 programmer hasn't protection devices, which protect the content of programmed device against critical*



situations, for example power failures and PC failure (interrupted cable...). Moreover, a device is usually destroyed in the programming mode due to forced interruption of the control program run (Reset or switching the computer off) due to removing the connecting cable, or unplugging the programmed device from the ZIF socket. Incorrectly placed device in the ZIF socket can cause its damage or destruction.

Technical specifications

Socket and control of pins:

- DIL/ZIF socket (300mil)
- each pin is possible to set in position Low and Pull-up
- from each pin is possible to read
- support of Low Voltage devices

For works with serial EEPROM are used nearest 8 pins to lever, unused pins are not connected.

Supported devices:

- EEPROM IIC (24Cxxx)
- EEPROM Microwire (93Cxxx)
- EEPROM SPI (25Cxxx)
- specially devices (digital thermometers, ...)

Device operations:

- **standard:**
 - blank check
 - read
 - program
 - verify
 - checksum

Buffer operations

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

File format conversion

- binary
- HEX: Intel, Intel EXT, Motorola S, MOS, Exormax, ASCII - space - HEX

PC systems requirements

See section *Introduction/ PC requirements*

General:

- power supply: 8V...15V, max.50mA
- dimensions: 132x66x40 [mm]
- mass: ca 150g
- temperature range: 5..40°C



Package includes

- SEP2 programmer
- connection cable PC to programmer
- wall plug adapter, 12V DC/500mA, unstabilized
- CD contains control program software, user's manual and additional files

Additional services

- free technical support (phone/fax/e-mail).

Software



The programmer software

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 MEC's Program-mate programmers work. Differences and modifications to this manual (if they exist) may be found in www.mecelectronic.com web site.

Installing of programmer software

Installing the programmer software is very easy. Insert delivered CD to your CD drive and install program starts automatically. Install program (setup.exe), which will guide you through the installation process and which will do all the necessary steps before you can first run the control program.

If you use **DOS** operating system, run **Install** program from CD.

Program PG4UW.EXE is common control programs for MEC's Program-mate programmers. We guarantee running of these programs under all of above mentioned operating systems without any problems. Also background operation under Windows is error-free.

New versions of programmer software

In order to exploit all the capabilities of programmer we recommend using the latest version of PG4UW. You may download the latest version of programmer software (file PG4UWARC.EXE) from our Internet site www.mecelectronic.com. You may also obtain CD with this file by snail-mail (a mailing charge will apply).

Upgrading the programmer software

Copy PG4UWARC.EXE to a temporary directory then launch it. After extraction you will see all available files needed for the installation process. Then redo a standard installation (run the Setup program). You may delete all files from the temporary folder after the installation process is complete.

Using the programmer software

The control program delivered by MEC, included on the CD in your package, is granted to be free from any viruses at the moment of delivery. To increase their safety our programs include a special algorithm for detecting possible virus infections.

Run the control program

in Windows environment: double click to icon PG4UW.

After start, control program PG4UW automatically scan all existing ports and search for the connected any MEC's Program-mate programmer. Program PG4UW is common for MEC's Program-mate programmers, hence program try to find all supported (UNP2, MCU2, PIC2 and SEP2) programmers.

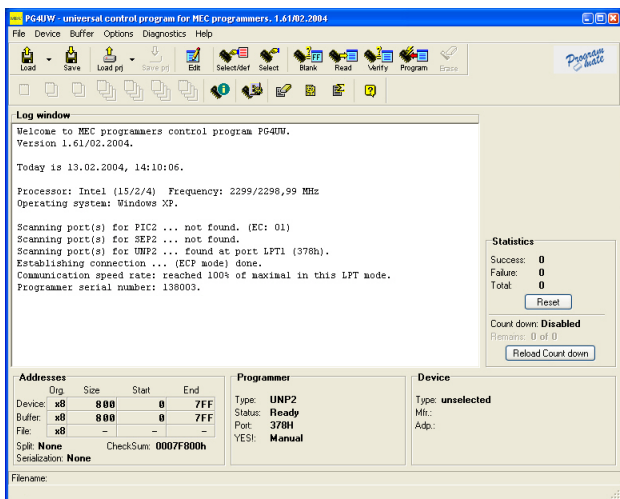
Notes: *When the PG4UW program is started, program is checked for its integrity. Than the program display a standard user menu and waits for your instructions.*

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.



Description of the user screen

Windows program PG4UW



Header bar

the name, copyright statement and version of the PG4UW control program

Menu bar

list of basic functions

Filename

information on the currently loaded file in buffer

Programmer window

information about the status of the programmer

Addresses window

organization, size, start and end addresses of the target device, buffer and file

Device window

all relevant information about the current target device

Help bar

a brief description of selected command

Menu selection is carried out in the normal GUI fashion - either by cursor moving plus pressing **<Enter>**, or by typing the highlighted letter in the wanted menu or - of course - by mouse. Hot-keys are available for even quicker selection of intensely used commands.

Note: Data entered through keyboard is in HEX format, excepting ASCII blocks in Buffer/View/Edit command.

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 PG4UW
<Alt+X>	Exit and save	Terminates the PG4UW 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

This submenu 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**. When Binary file format is selected, there can be specified Buffer start value. Buffer start value is buffer address from which data read from file will be written to buffer.

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.

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)

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 last 10 selected devices to disk and returns back to the operation system.



Device

The 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 last 10 selected devices including its 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 (max. 10 devices). This buffer is accessible with **Device / Select from default devices** command.

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.

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 (max. 10 devices). This buffer is accessible with **Device / Select from default devices** command.

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), 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 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 (max. 10 devices). This buffer is accessible with **Device / Select from default devices** command.

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.

Select device ... / Only selected manufacturer

This window allows selecting the desired device type by manufacturer. First select a required manufacturer in Manufacturer box using mouse or cursor keys. It will cause a list of selected manufacturer 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 (max. 10 devices). This buffer is accessible with **Device / Select from default devices** command.

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.

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.*

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)
Buffer start address	(default 0)
Split (default none)	this option allows to set special mode of buffer when programming or reading device. Following table describes buffer to device and device to buffer data transfer

- group Insertion test:

insertion test	(default ENABLE)
check ID bytes	(default 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)

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

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.

Note: *Serialization can work with control program's main buffer only. It means the serialization can be used for device areas placed inside control program's main buffer. Device special areas placed outside the program's main buffer could not use serialization feature.*

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
- Bin

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
- Hexadecimal.

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 below as example number 2.

Examples:

1. Write serial numbers to AT29C040 devices at address 7FFF00H, 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: 7FFF00H
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 00 00 00 16
The 2nd device
Address Data
007FFF0 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 02 00 00 16
etc.

"xx" mean user data programmed to device

Serial numbers are written to device from address 7FFF00H to address 7FFFFFH because serial number size is 4 bytes.

2. Following example shows usage of serial number 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 W11	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 us 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.

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 and written to buffer on address specified in input file.

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.

From file serialization file format

From file serialization input 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:

[] - 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).*

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 \$FFFFFF.

Note: *Address for Serialization is always assigned to actual device organisation and buffer organisation that control program is using for current device. If the buffer organisation is byte org. (x8), the Serialization Address will be byte address. If the buffer organisation is wider than*



byte, e.g. 16 bit words (x16), the Serialization Address will be word address.

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.

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 reading whole device or its part into the buffer. The control program reports a finish of this action by write 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.

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 / 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 and other general information about current device too.

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



Buffer

This submenu 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 to 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 start 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.*

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 MS 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

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 MS 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 in current buffer block. This block must start 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).

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 will be written only low byte of checksum value.

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**.



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:

1. Prompt before reloading file
2. Reload automatically
3. Ignore change scanning of current file

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 analyse, 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 MS 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 programmers speaker after each device operation for indicating device operation result – good or bad result.

Other

Page Other allows user to manage other program settings. 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.



Save options

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

Don't save options - don't save options during quitting program and don't ask for saving options.

Auto save options - save options during quitting program without asking for saving options.

Prompt for save options - program asks user for saving options before quitting program. User can select to save or not to save options.

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 / Display errors

This option allows you set a form of errors displaying as a result of programmed data verifying. Errors can be displayed to the **screen** (max. 45 differences), **saved to VERIFY.ERR** file on the disk in current directory or it will **not displayed**. In case the displaying errors are turned off, the control program reports a warning message in INFO window only.

This setting can be saved to disk by command **Options / Save options**. Default form is set to a screen displaying.

Options / Find programmer

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.

Note: *Items Establish communication and Speed are available only for SEP2.*

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**.

Options / Handler

In dialog **Handler** a Handler type and Handler communication parameters can be set. Handler is an external device for special control of device operations in control program. When None Handler is selected, this means default state of control program, i.e. device operations are controlled directly by user otherwise control program is in special mode, when device operations are controlled automatically with co-operation with Handler.

Dialog **Handler** contains following items:

Selected Handler select wished Handler type.

Search at port select a COM port, which will be scanned for a requested Handler.

Pressing key <Enter> or button **OK** initiates scanning for Handler by set parameters. If selected Handler type is **None**, no Handler scanning will be processed. Current Handler settings are saved to configuration file by command **Options / Save options** or when control program is closed.



Handler is not available for sale.

Options / Automatic YES!

This command is used for setting **Automatic YES!** mode. In this mode isn't necessary to press "YES!" labeled button to repeat last activity. You just put a device into ZIF socket and program automatically detects an insertion of a new device and last operation will be repeated automatically. 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. This feature is available for UNP8 programmer only.

Note: *During waiting for an insertion a new device into ZIF socket, the LED BUSY on the programmer is blinking.*

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 possible to set a time interval within must be detected device in ZIF socket to accept an insertion of a new device. Default is set standard interval. If socket adapter is used then is recommended to set an elongated interval.

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

This list is erased if a new device is selected by **Device / Select default** or **Device / Select device...**

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

Options / Log file

This options associates with using of **Log window**. All reports for this window can be written into the **Log file** too. The Log file name as **REPORT.REP** and the control program creates this file in current directory.

Sets **New** caused deleting old Log file if exist and creating a new file for reports. Sets **Append** adds all reports into existing Log file. If file not exist, the new file will be created. Settings are applied only at program start.

This setting can be saved to disk by command **Options / Save options**. Default form is set to a using Log window without Log file i.e. all reports will be displayed to a Log window only.

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.

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, which 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).

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.



Diagnostics

This command includes selftest for programmers and IC test.

Diagnostics / Selftest

Command executes a selftest of current programmer without diagnostic POD. We recommend execute also **Diagnostics / Selftest plus** of programmer.

Diagnostics / 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.

Diagnostics / IC test

This command activates a test section for ICs separated by compatibility 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.

This feature is available for UNP8 programmer only.

Diagnostics / Create diagnostic report

Command Create Diagnostic 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. Diagnostic 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 diagnostic report. Diagnostic report can help manufacturer to localise the reason of error and resolve it sooner.

Help

Pressing the <F1> key accesses the Help. When you selecting menu item and press <F1>, you access context-sensitive help. If PG4UW 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.mecelectronic.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.

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 its 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 (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.

About

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

Common notes



Software

PG4UW is common control program for all of the MEC programmers. Thus, during work with him, it is possible to find some items, those refer not to current selected programmer.

Some special devices (e.g. Philips Coolrunner family) require external DAT files, that aren't present in standard PG4UW SW delivery on CD. If you need to program these devices, look at www.mecelronic.com.

You can start control program with different **command line parameters**.

Basic rules for using of executive command line parameters:

1. command line parameters are not case sensitive
2. command line parameters can be used when first starting of program or when program is already running
3. if program is already running, then any of command line operation is processed only when program was not busy (no operation was currently executing in program). Program must be in basic state, i.e. main program window focused, no modal dialogs displayed, no menu commands opened or executed.
4. order of processing command line parameters when using more parameters together is defined firmly as following:
 1. Load file (/Loadfile:...)
 2. Load project (/Prj:...)
 3. Program device (/Program[:switch])
 4. Close of control program (/Close only together with parameter /Program)

Available command line parameters:

- /Axxx check programmer present on LPT port with address xxx only
example: /A3bc
- /SPP force PC <-> programmer communication in unidirectional mode

Available executive command line parameters:

- /Prj:<file_name> forces project load when program is starting or even if program is already running, <file_name> means full or relative project file path and name
- /Loadfile:<file_name> forces file load when program is starting or even if program is already running, <file_name> means full or relative path to file that has to be

	loaded, file format is detected automatically
/Program[:switch]	forces start of "Program device" operation automatically when program is starting, or even if program is already running, also one of following optional switches can be used:
switch 'noquest'	forces start of device programming without question
switch 'noanyquest'	forces start of device programming without question and after operation on device is completed, program doesn't show "Repeat" operation dialog and goes directly into main program window

Examples:

1. /Program
2. /Program:noquest
3. /Program:noanyquest

/Close	this parameter has sense together with /Program parameter only, and makes program to close automatically after device programming is finished (no matter if operation was successful or no)
--------	---



Hardware

Due a large variety of parallel port types, a case may occur when the programmer cannot "get concerted" with the PC. This problem may be shown as none communication between the PC and the programmer, or by unreliable communication. If this behaviour occur, try to connect your programmer to some other PCs or other parallel ports near you.

If you find none solution, please document the situation, i.e., provide us an accurate description of your PC configuration, including some other circumstances bearing on the problem in question, and advise the manufacturer of your problem. Don't forget please enter of PC type, manufacturer, speed, operation system, resident programs; your parallel port I/O manufacturer and type.

ISP (In-System Programming)

Definition

In-system programming allows programming and reprogramming of device positioned inside the end system. Using a simple interface, the ISP programmer communicates serially with the device, reprogramming nonvolatile memories on the chip. In-system programming eliminates the physical removal of chips from the system. This will save time and money, both during development in the lab, and when updating the software or parameters in the field.

Target device is the device (microcontroller, PLD, etc...), which is to be in-system programmed.

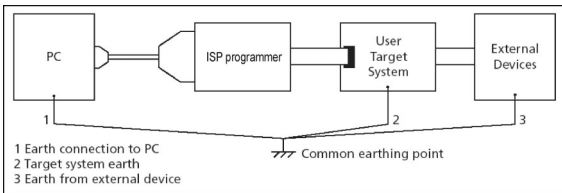
Target system is the physical Printed Circuit Board (PCB), which contains the device to be in-system programmed.

ISP programmer is programmer, which has in-system programming capability (for example UNP2, MCU2, PIC2...).

General rules for in-system programming

We recommended respect following rules to avoid damage PC, ISP programmer, and target device or target system:

- Ensure common earth point for target system, ISP programmer and PC.
- For laptop or other PC that is not connected to common earth point: make hard - wired connection from laptop to common earth point (for example use LPT or COM port D – connector).
- Any devices connected to target system must be connected to common earth point too.



Direction of connect MEC Program-mate ISP programmer to target system:

During in-system programming you connect two electrical devices – ISP programmer and target system. Unqualified connection can damage these devices.



Note: When you don't keep below directions and you damage programmer during in-system programming, it is damage of programmer by unqualified manipulation and is out of warranty.

1. Turn off both devices – ISP programmer and target device.
2. Assign same GND potential for all devices, e.g. connect GND of all devices by wire.
3. Insert one connector of ISP cable to ISP programmer, turn on programmer and control program.
4. In control program select target device and operation options.
5. Start action on target device (read, program).
6. After direction of control program, connect other ISP cable connector to target system and turn on it.
7. After direction of control program, disconnect other ISP cable connector from target system and turn off it.
8. If you need another action on target device, you continue with step 5.

The recommendation for design of target system with ISP programmed device

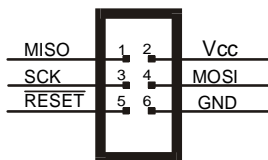
The target system must be designed to allow all signals, which are use for In-system programming to be directly connected to ISP programmer via ISP connector. If target system use these signals for other function, is necessary isolated these signals. Target system mustn't affect these signals during In-system programming.

For in-system programmable devices manufacturers publish application notes. Design of MEC Program-mate programmers together with respect of these application notes allows proper In-system programming. Condition is exactly respect these application notes. Applications notes, which MEC use in ISP programmers, are published in www.mecelectronic.com.

Example of application note

Microcontrollers Atmel AVR and AT89Sxxx series

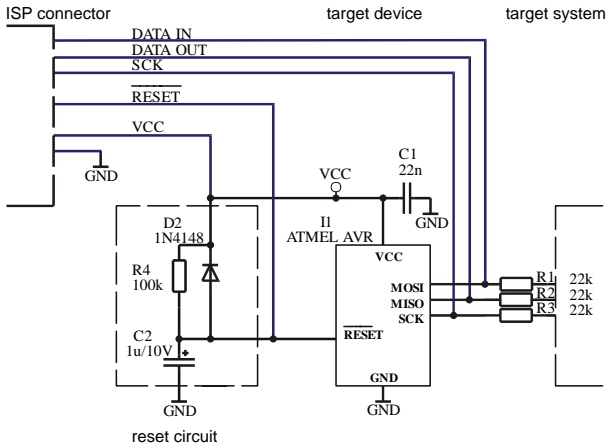
This application note is used in MCU2. This interface corresponds with Atmel application note AVR910: In-System Programming. This application note describes the recommended ISP interface connector layout in target system (top view).



Description of required pins for in-system programming by AVR910.

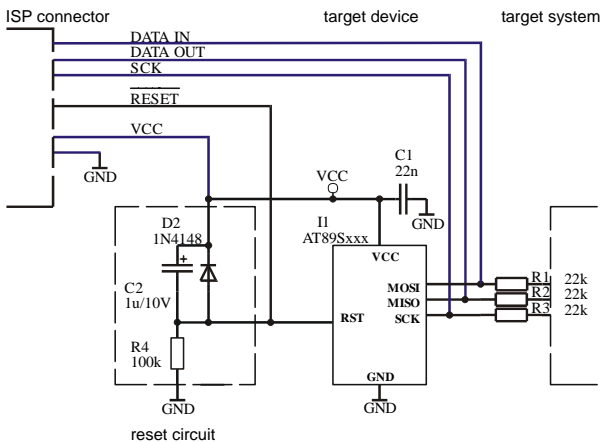
Pin	Name	Comment
SCK	Serial Clock	Programming clock, generated by the In-System programmer (master).
MOSI	Master Out – Slave In	Communication line from In-System programmer (master) to target MCU being programmed (slave).
MISO	Master In – Slave Out	Communication line from target MCU (slave) to In-System programmer (master).
GND	Common Ground	The two systems must share the same common ground.
RESET	Target MCU Reset	To enable In-System programming, the target MCU Reset must be kept active. To simplify this, the In-System programmer should control the target MCU Reset
Vcc	Target Power	To allow simple programming of targets operating at any voltage, the In-System programmer can draw power from the target. Alternatively, the target can have power supplied through the In-System programming connector for the duration of the programming cycle

MEC's recommended circuit for ATMEL AVR:





MEC's recommended circuit for AT89Sxxx:



PICmicro[®] microcontrollers

This application note is used in PIC2. This interface corresponds with Microchip application notes TB013, TB017, TB016: How to Implement ICSP™ Using PIC16CXXX OTP (PIC12C5XX OTP) (PIC16F8X Flash) MCUs. These application notes describes requirement for target system with In-system programming device and ISP programmer.

Following signals are use for In-system programming of PICmicro[®] microcontrollers.

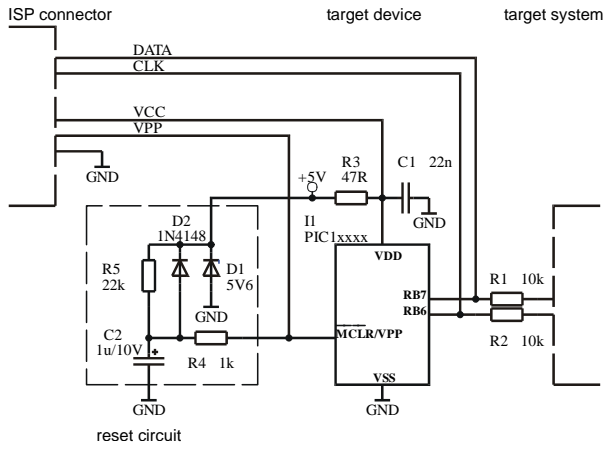
MCLR\ / VPP	reset / switch to programming mode
RB6 (GP1)	clock
RB7 (GP0)	data input / output
VDD	power supply
GND	ground

When PICmicro[®] device is programmed, pin MCLR\ / VPP is driven to approximately 12 V. Therefore, the target system must be isolated from this voltage provided by programmer.

RB6 and RB7 signals are used by the PICmicro[®] for In-system programming, therefore target system mustn't affect these signals during In-system programming to avoid programming errors.

Marginal verify is used after programming. Programmer must verify the program memory contents at both minimal and maximal power supply, therefore VDD pin of PICmicro[®] must be isolated from rest of target system during programming.

MEC's recommended circuit for PICmicro:



Note: External reset circuit is necessary only if VDD power-up slope is too slow.



Other

Attention to multitasking OS's (WIN95/98/Me/NT/2000/XP). There is needful for regular running of control program for any MEC Program-mate programmer that printer port, on which is programmer connected, must be reserved for this programmer only. Otherwise, any other program must not simultaneously to use (or any way to modify) this printer port.

PG4UW SW can handle all modes of LPT port (full IEEE 1284 support), thus you don't need to configure LPT port for connection of MEC programmers.

WIN98 have bug (or wittingly) in the MSDOS.SYS file. Initial setting of variable DoubleBuffer is 1, therefore DOS applications run slowly. Write please DoubleBuffer=0.

In case of WIN software, please don't move any window during BUSY LED is on - watching circuit can be activate to switch the programmer in safe status as in case communication PC-programmer error.

Troubleshooting



Troubleshooting

We really want you to enjoy our product. Nevertheless, problems can occur. In such cases please follow the instructions below.

- It might be your mistake in properly operating the programmer or its control program PG4UW.
 - Please read carefully all the enclosed documentation again. Probably you will find the needed answer right away.
 - Try to install programmer and PG4UW on another computer. If your system works normally on the other computer you might have a problem with the first one PC. Compare differences between both computers.
 - Ask your in-house guru (every office has one!).
 - Ask the person who already installed programmer.
- If the problem persists, please call the local dealer, from whom you purchased the programmer, or call MEC direct. Most problems can be solved by phone, e-mail or fax.

If you have an unsupported target device

If you need to operate on a target device not supported by the control program for programmer, please do not despair and follow the next:

Look in the device list of the latest version of the control program on our Internet site www.meceselectronic.com, (file corresponded to your programmer). Your new target device might already be included in this version! If yes, download the file PG4UWARC.EXE and install the new version of the control program.