# PIC LABS

# C·170

# User Manual

# C-170 SYSTEM

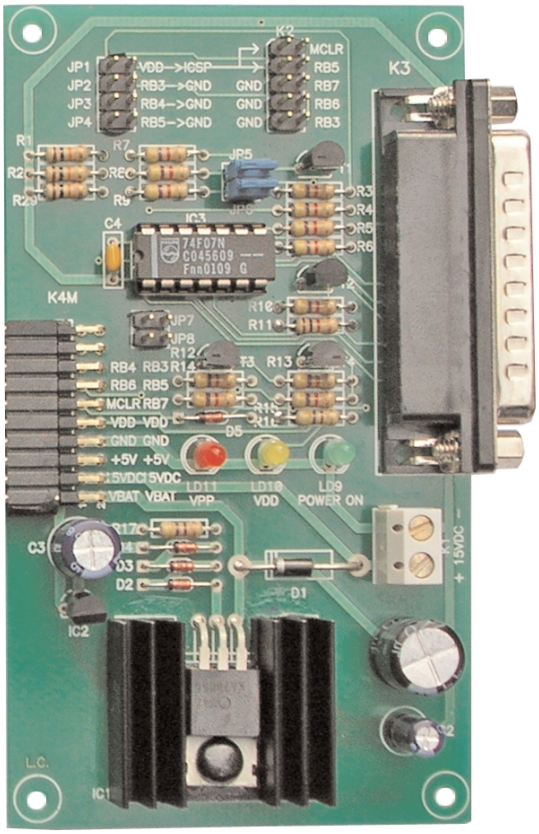
The C-170 system supports the PDIP socket MCU PIC in 40, 28, 18, 14 and 8 pins format. The programming of microcontrollers could be execute through the parallel port of a PC and the enclosure software or connecting C-170 system with Microchip In-Circuit Debugger ICD2. After the programming it's possible utilize the RUN condition to test and debug the correct operation of downloaded MCU firmware. The C-170 system disposes of a 20 MHz quartz and offer the possibility to insert another quartz of wishes frequency. On C-170 system there are also more 5 connectors correspondent at all microcontroller's I/O ports. It's also available a Reset button.
It's possible to decompose the system in two sections:
- programming section: C-202 board
- socket section: C-203 board
Parallel cable to linking at PC parallel port not included.
For the correct use of C-170 board you need: 17Vdc/Minimum 300 mA power generator; a IBM compatible PC with Windows(TM) 95/98/ME/NT/2000/XP, CD-ROM reader and a free parallel port.

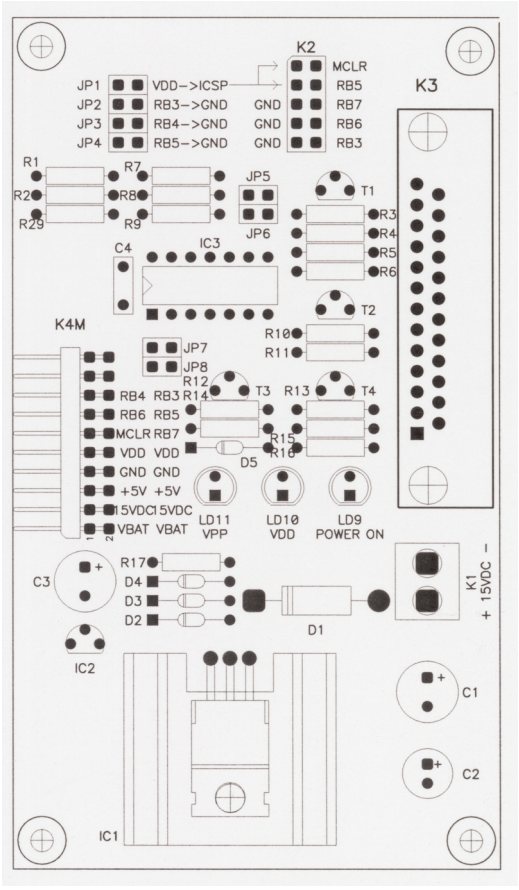
# SUMMARY

# PROGRAMMING SECTION

## Introduction

C-202 board contains the electronic to programming the PIC microcontrollers.
The system use two lines for Clock and Data and needs the application of a voltage of 13.2 V at MCLR pin.
For the MCU PIC that contain LVP (Low Voltage Programming) programming modality too it's necessary apply a level logic low at PGM pin.
The circuit needs a voltage power of 17 Vdc.
It's necessary apply the voltage power at K1 terminal in accordance with the polarity.
Diode D1 protects the board from polarity inversion.
Chip IC2 (7805) supply a +5 V regulated voltage, the LD9 green led show the presence in the circuit of this voltage.
Chip IC1 (78L12) supply a +13,2 V regulated voltage, necessary in the programming phase.



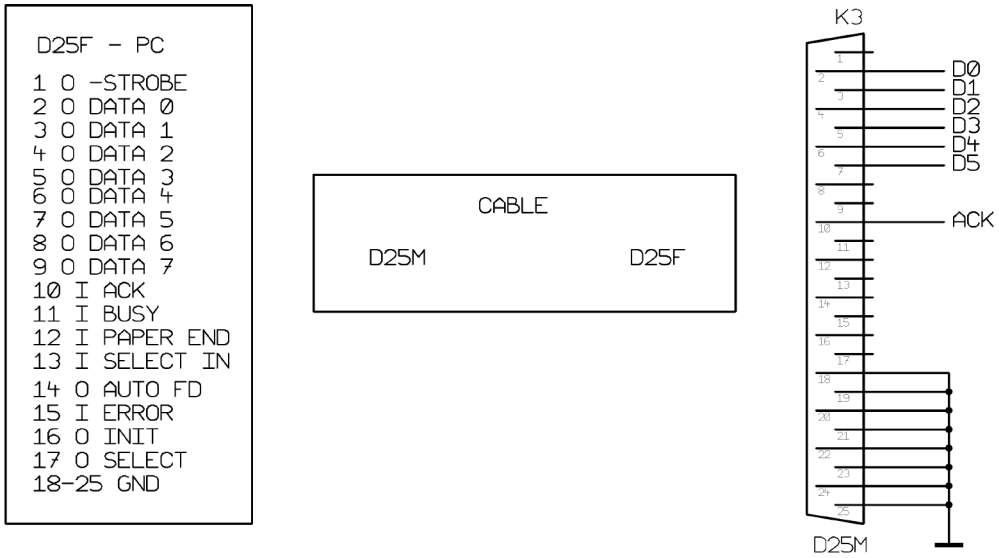



## Connectiong to the PC

It's necessary connecting the board at a parallel port of a IBM compatible PC equipped with IC-Prog software. For the linking between PC and board use a 25 pins parallel cable with a D25 male connector (PC side) and a D25 female connector (C-202 board side).
Signals DATA from number 0 to number 5 and signal ACK of parallel port are used in the following mode:

- **D0** DATA signal from PC to PIC microcontroller (apply at **RB7** pin)
- **ACK** DATA signal from PIC microcontroller to PC (pick up from **RB7** pin)
- **D1** CLK signal from PC to PIC microcontroller (apply at **RB6** pin)
- **D2** controls Vdd voltage alimentation of PIC microcontroller (coincide with **VDD** pin)
- **D3**, **D4** and **D5** control PIC microcontroller voltage programming (coincide with **MCLR** pin)

All these signals are isolated from chip IC3 parallel port; in this way it's never possible damage PC parallel port.




## Signallings

C-202 board contains three signalling leds:

**LD9** green led it's on on presence of **+5V** voltage

**LD10** yellow led it's on on presence of **VDD** voltage

**LD11** red led it's on on presence of **VPP** voltage

## Programming lines jumpers

Writing / reading of microcontrollers PIC it's execute serially through **PGD** (Program Data), **PGC** (Program Clock) pins. **VDD** (Positive Power), **MCLR** (Memory Clear) and, if available, **PGM** (Program Mode) pins are also used in the writing operations.

**PGD** (Program Data) coincide with **RB7** port of microcontroller PIC.

**PGC** (Program Clock) coincide with **RB6** port of microcontroller PIC.

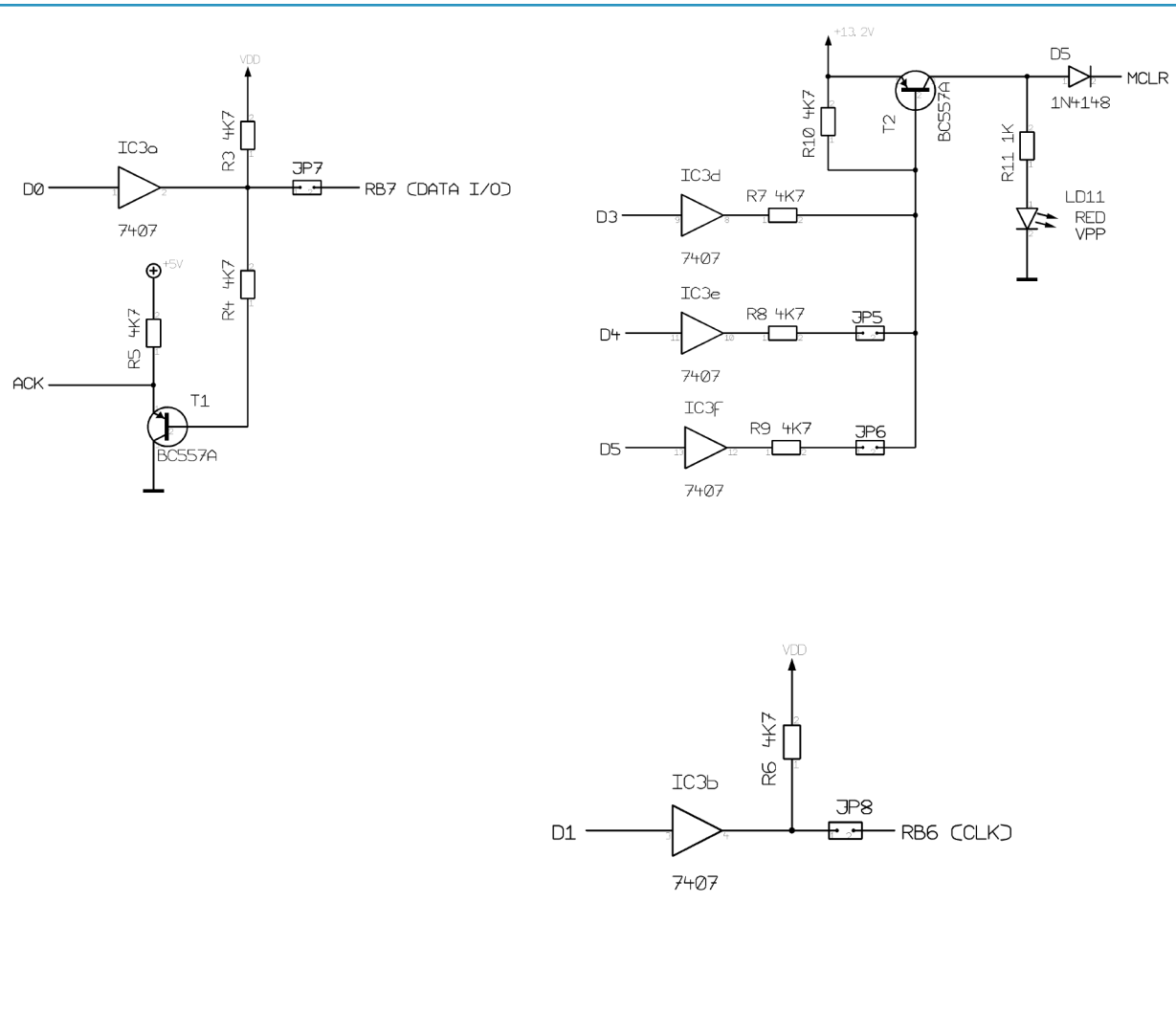
**VPP** (Voltage Programming) coincide with **MCLR** line of microcontroller PIC.

**PGM** (Program Mode) coincide with **RB3**, **RB4** or **RB5** port of microcontroller PIC.

The two serial lines PGD and PGC could be interrupted through JP7 and JP8 jumpers.

**JP7** interrupt **PGD** line.

**JP8** interrupt **PGC** line.



## PGM Jumper

C-202 board have 3 jumpers that could be used to connect microcontrollers PIC RB3, RB4 and RB5 ports at ground, through a 10 KOhm resistor.

Closing **JP2** the MCU **RB3** port is connected at ground through the 10 KOhm resistor R1.

Closing **JP3** the MCU **RB4** port is connected at ground through the 10 KOhm resistor R29.

Closing **JP4** the MCU **RB5** port is connected at ground through the 10 KOhm resistor R2.

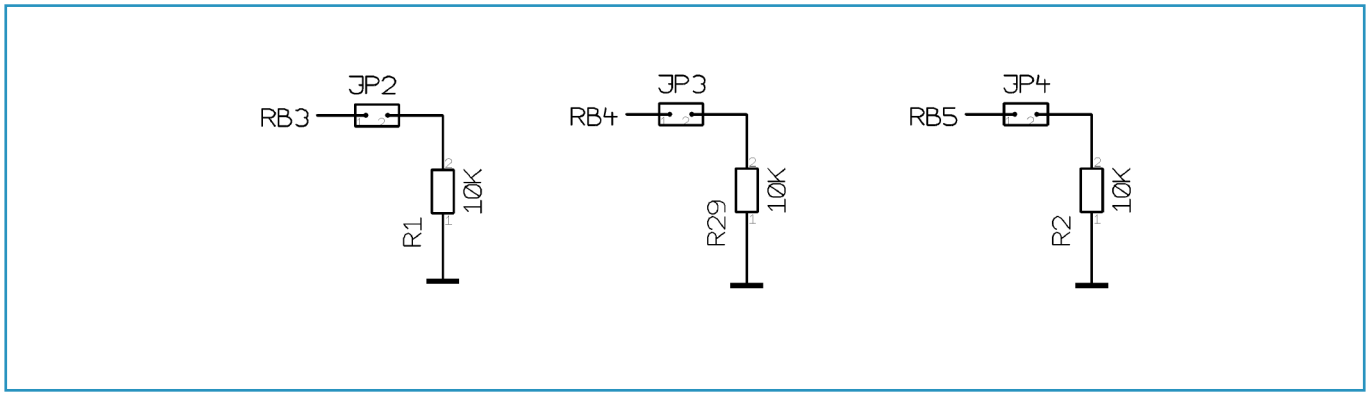
If microcontroller PIC to programming got PGM line, it's necessary closing the correct jumper before starting the programming phase. To know if PGM line is available, please refer at relative microcontroller PIC datasheet.

Generally apply following rules:

**PIC16F87X** family:
PGM line is available on port **RB3**; so in programming phase it's necessary closing the JP2 jumper.

**PIC16F62X** family:
PGM line is available on port **RB4**; so in programming phase it's necessary closing the JP3 jumper.

**PIC18FXXX** family:
PGM line is available on port **RB5**; so in programming phase it's necessary closing the JP4 jumper.

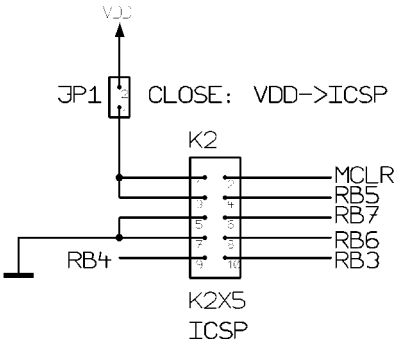


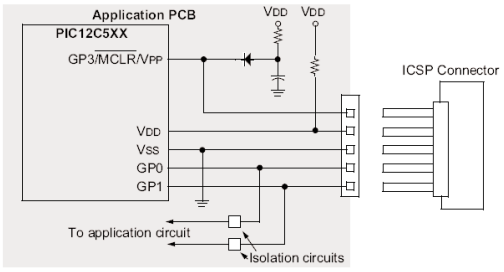
## ICSP connector (In-Circuit Serial Programming)

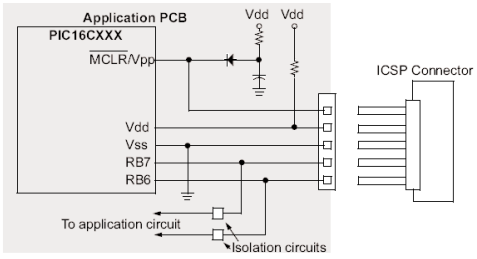
All the lines that are necessary in the writing/reading operations of microcontrollers PIC, are available on the 10 pins connector **K2**.
It's so possible programming the microcontroller PIC installed on the user's board without the necessity to remove it from the user's board.
Jumper **JP1** interrupt the presence of voltage **VDD** on connector K2.

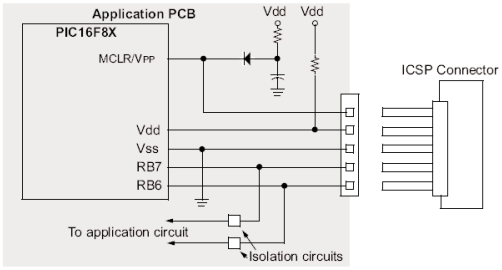
For the In-Circuit Serial Programming it's necessary foreseen a specific hardware on the user's board. Please, for more informations, refer to **ICSP Guide** documents, available on the Microchip web site.

Following are reported the electric plans of connector K2 and the tipically In-Circuit Serial Programming needed circuit.

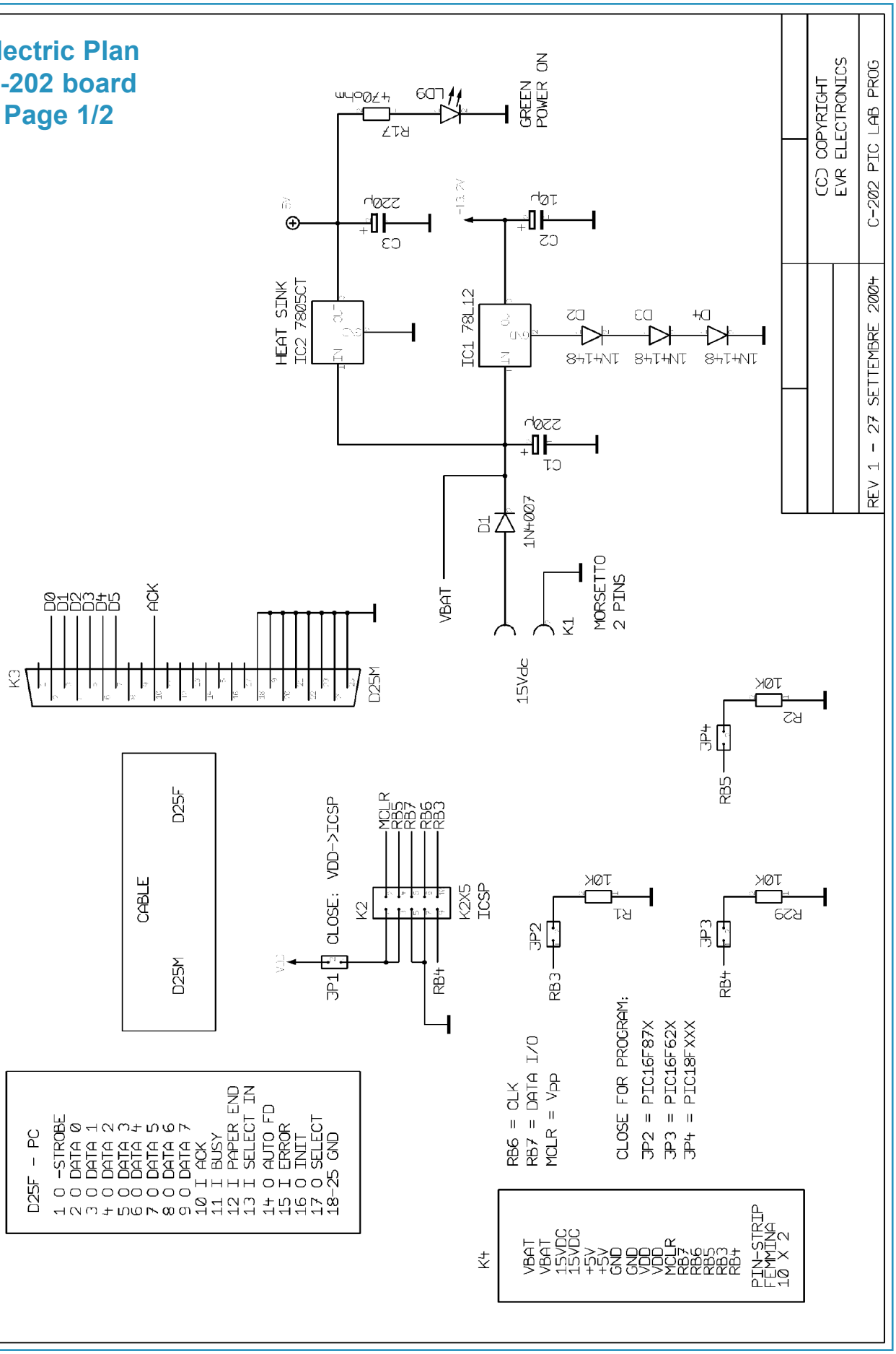


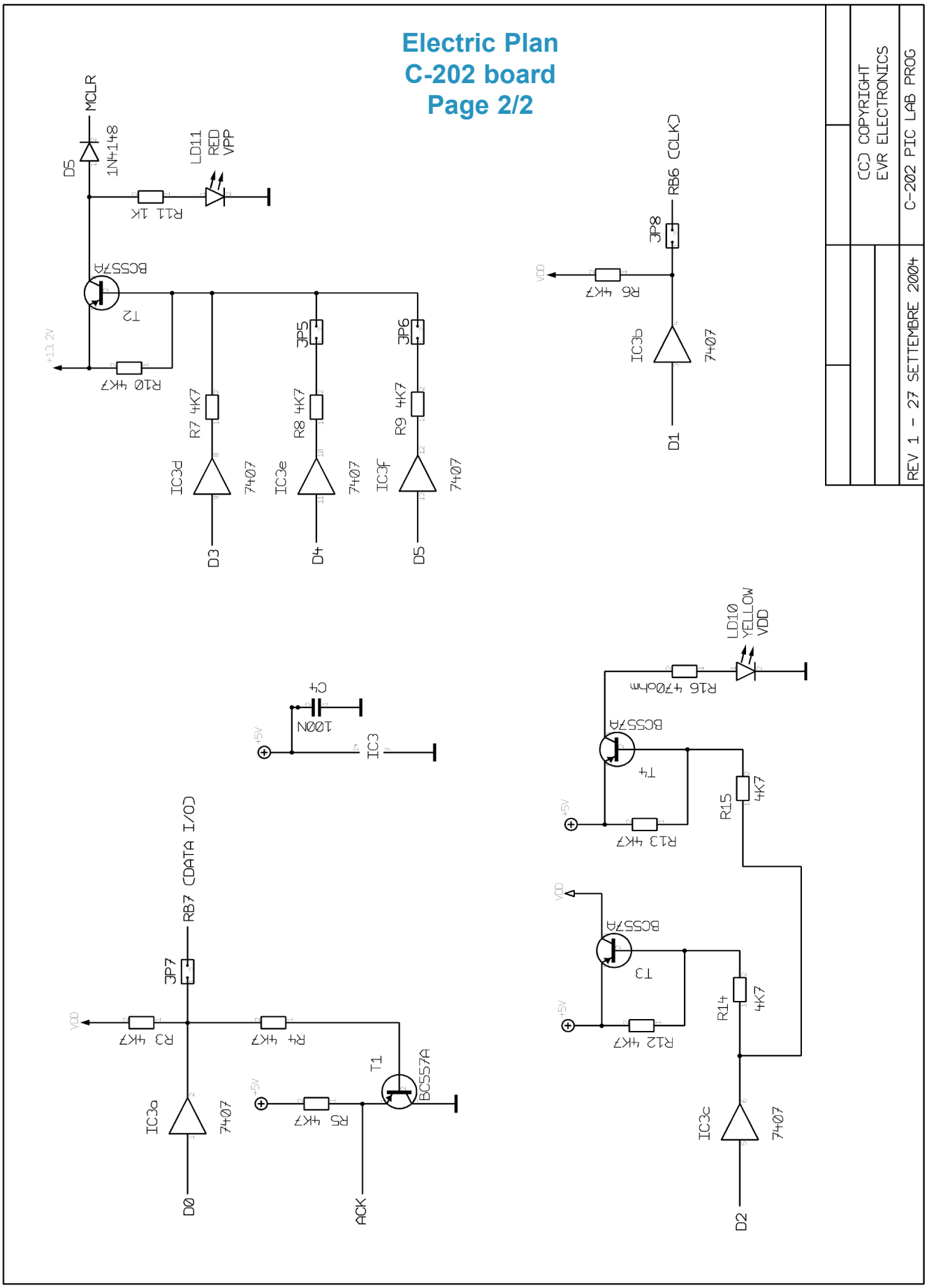



# Electric Plan
# C-202 board
# Page 1/2



# Electric Plan
# C-202 board
# Page 2/2



**PART LIST C-202**

| | | |
|---|---|---|
| **R1**: | 10 Kohm | ¼ W |
| **R2**: | 10 Kohm | ¼ W |
| **R3**: | 4,7 Kohm | ¼ W |
| **R4**: | 4,7 Kohm | ¼ W |
| **R5**: | 4,7 Kohm | ¼ W |
| **R6**: | 4,7 Kohm | ¼ W |
| **R7**: | 4,7 Kohm | ¼ W |
| **R8**: | 4,7 Kohm | ¼ W |
| **R9**: | 4,7 Kohm | ¼ W |
| **R10**: | 4,7 Kohm | ¼ W |
| **R11**: | 1 Kohm | ¼ W |
| **R12**: | 4,7 Kohm | ¼ W |
| **R13**: | 4,7 Kohm | ¼ W |
| **R14**: | 4,7 Kohm | ¼ W |
| **R15**: | 4,7 Kohm | ¼ W |
| **R16**: | 470 ohm | ¼ W |
| **R17**: | 470 ohm | ¼ W |
| **R29**: | 10 Kohm | ¼ W |
| **D1**: | 1N4007 | |
| **D2**: | 1N4148 | |
| **D3**: | 1N4148 | |
| **D4**: | 1N4148 | |
| **D5**: | 1N4148 | |
| **LD9**: | LED GREEN 3 MM | |
| **LD10**: | LED YELLOW 3 MM | |
| **LD11**: | LED RED 3 MM | |

| | | |
|---|---|---|
| **C1**: | 220 □F | 35 V |
| **C2**: | 10 □F | 50 V |
| **C3**: | 220 □F | 35 V |
| **C4**: | 100 nF | |
| **IC1**: | 78L12 | |
| **IC2**: | 7805 | |
| **IC3**: | 7407 | |
| **T1**: | BC557A | |
| **T2**: | BC557A | |
| **T3**: | BC557A | |
| **T4**: | BC557A | |
| **K1**: | MORSETTO 2 PINS | |
| **K2**: | PIN-STRIP M 5 + 5 PINS | |
| **K3**: | D25M PINS 90 GRADI | |
| **K4M**: | PIN-STRIP M 10 + 10 90° | |
| **JP1**: | PIN-STRIP 2 PINS | |
| **JP2** | PIN-STRIP 2 PINS | |
| **JP3** | PIN-STRIP 2 PINS | |
| **JP4** | PIN-STRIP 2 PINS | |
| **JP5** | PIN-STRIP 2 PINS | |
| **JP6** | PIN-STRIP 2 PINS | |
| **JP7** | PIN-STRIP 2 PINS | |
| **JP8** | PIN-STRIP 2 PINS | |
| **-** | SOCKET 7 + 7 PINS | |
| **-** | DISSIP 25 X 29 MM | |
| **-** | N. 6 JUMPERS F | |
| **-** | CS C202-1 | |

# SOCKET SECTION

## Introduction

C-203 board supports the PDIP socket MCU PIC in 40, 28, 18, 14 and 8 pins format.

It's available a 40 pins socket that could be used for every PDIP40 case MCU, as the 16F74, 16F77, 16F871, 16F874, 16F877, 18F4320, 18F442, 18F452, 18F448, 18F458, etc. PICs.

It's available a 28 pins socket that could be used for every PDIP28 case MCU, as the 16F72, 16F73, 16F76, 16F870, 16F872, 16F873, 16F876, 18F2020, 18F2320, 18F242, 18F252, 18F248, 18F258, etc. PICs.

It's available a 18 pins socket that could be used for every PDIP18 case MCU, as the 16C620, 16C622, 16C710, 16C711, 16C712, 16C715, 16C716, 16F627, 16F628, 16F84, 16F818, 1dF819, etc. PICs.

It's available a 14 pins socket that could be used for every PDIP14 case MCU, as the 16F684, 16F630, 16F688, 16F676, 16F636, 16F505, etc. PICs.

It's available a 8 pins socket that could be used for every PDIP8 case MCU, as the 12C508, 12C509, 12C518, 12C519, 12C671, 12C672, 12C673, 12C674, 12F629, 12F675, etc. PICs.

If a not-PDIP case MCU is used, it's necessary use a proper adapter.

All the Input/Output lines are available on handly 2.54 mm connectors:
K5 for PortA, K6 for PortB, K7 for PortC, K8 for PortD and K9 for PortE.

Every I/O line is bringing from sockets to K5, K6, K7, K8 and K9 connectors and at K11M, K12M and K13M output connectors. All the I/O lines of every socket and every connector could be immediatly identify through the board's serigrafy.

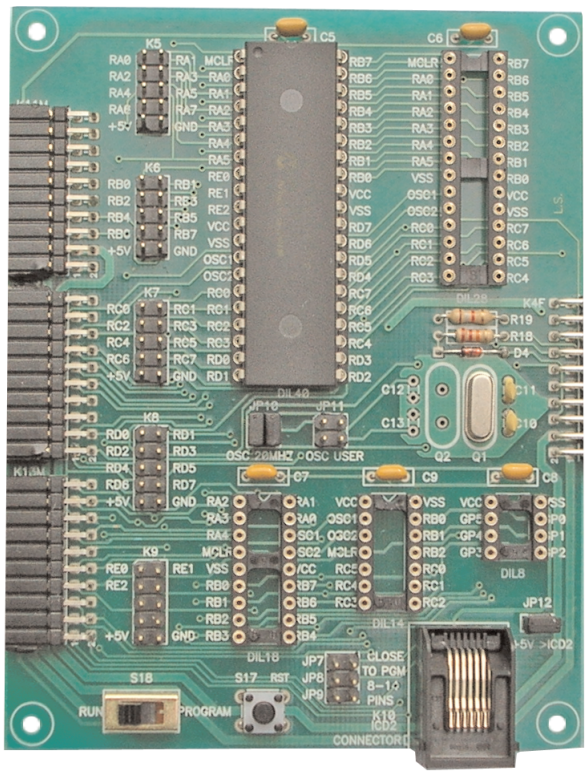
**Concerning the 8 pins socket (DIL8), the GP lines are connected at PortB lines:**
**GP0=RB0; GP1=RB1; GP2=RB2; GP3=RB3; GP4=RB4; GP5=RB5.**

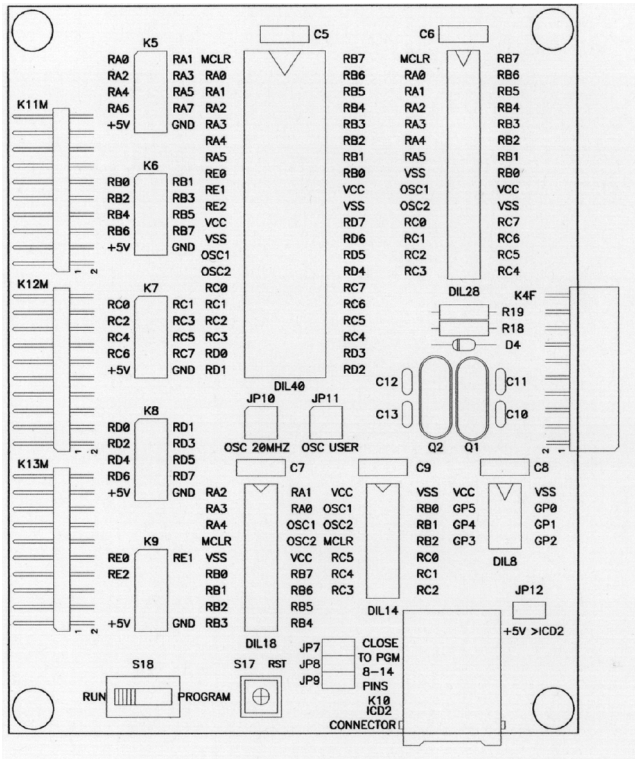
**Concerning the 14 pins socket (DIL14), the pins number 13, 12 and 11 are connected at lines RB0, RB1 and RB2. Pin number 4 is used as Master Clear (MCLR), while pins number 2 and 3 are used as OSC1 and OSC2.**

All the lines OSC1 of every socket (except DIL8) are connected together and are bringed at JP10 and JP11 jumpers; the same is done for the lines OSC2. Closing JP10 (insert two bridges in vertical position) the on-board 20 MHz quartz it's selected; closing JP11 (insert two bridges in vertical position) the (not-mounted) Q2 quartz and (not-mounted) capacitors C12 and C13 are selected.

Voltage power pins of every socket (VCC pins) are connected together and are bringed on S18 switch: if S18 is in PROGRAM position, voltages VCC are connected together with programming board VDD signal; if S18 is in RUN position, voltages VCC are connected together with voltage +5 V supplied by programming board.

## Notes





## MCU PIC 40 pins format socket

Pins of PDIP40 microcontrollers PIC.

The serially programming use the following lines:

**PGD** (Program Data): RB7 **pin number 40**.

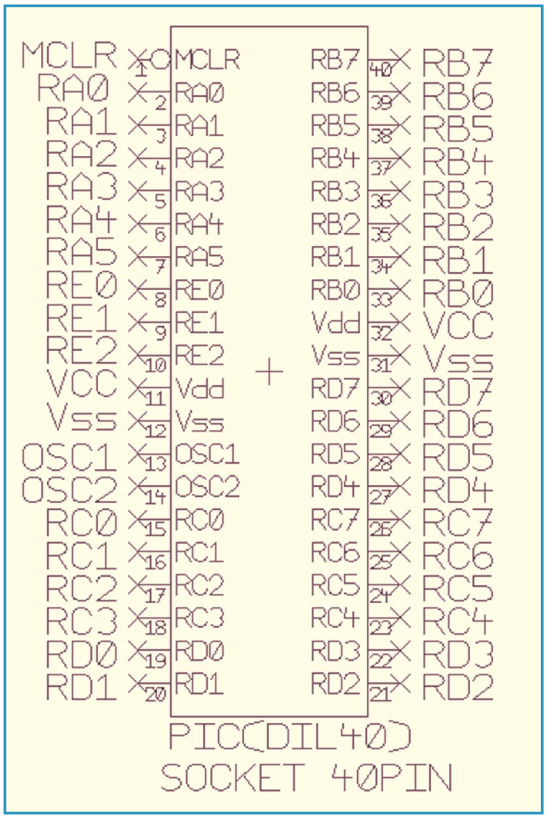
**PGC** (Program Clock): RB6 **pin number 39**

**PGM** (Program Mode): RB3 **pin number 36**

**VPP** (Programming Voltage): MCLR **pin number 1**

**VDD** (Positive Power Voltage): Vdd **pins number 11 and 32**

**GND** (Ground): Vss **pins number 12 and 31**




## MCU PIC 28 pins format socket

Pins of PDIP28 microcontrollers PIC.

The serially programming use the following lines:

**PGD** (Program Data): RB7 **pin number 28**

**PGC** (Program Clock): RB6 **pin number 27**

**PGM** (Program Mode): RB3 **pin number 24**

**VPP** (Programming Voltage): MCLR **pin number 1**

**VDD** (Positive Power Voltage): Vdd **pin number 20**

**GND** (Ground): Vss **pins number 8 and 19**

## MCU PIC 18 pins format socket

Pins of PDIP18 microcontrollers PIC.

The serially programming use the following lines:

**PGD** (Program Data): RB7 **pin number 13**

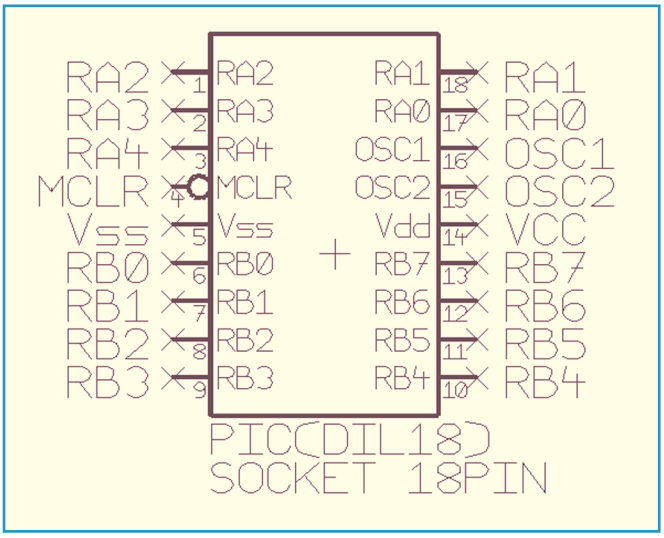
**PGC** (Program Clock): RB6 **pin number 12**

**PGM** (Program Mode): RB4 **pin number 10**

**VPP** (Programming Voltage): MCLR **pin number 4**

**VDD** (Positive Power Voltage): Vdd **pin number 14**

**GND** (Ground): Vss **pin number 5**




## MCU PIC 14 pins format socket

Pins of PDIP14 microcontrollers PIC.

The serially programming use the following lines:

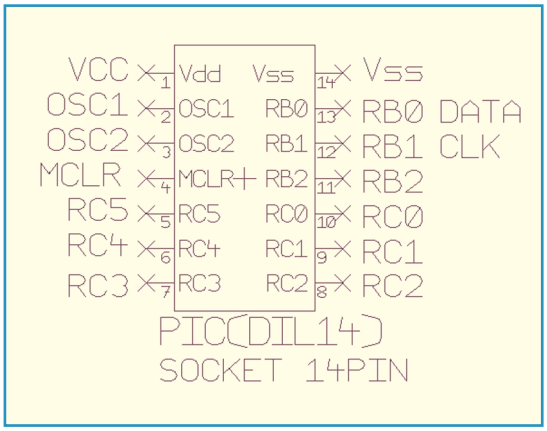
**PGD** (Program Data): RB0 **pin number 13**

**PGC** (Program Clock): RB1 **pin number 12**

**VPP** (Programming Voltage): MCLR **pin number 4**

**VDD** (Positive Power Voltage): Vdd **pin number 1**

**GND** (Ground): Vss **pin number 14**




**Pins number 13, 12, 11 are connected at lines RB0, RB1 and RB2.**
**Pin number 4 is used as Master Clear (MCLR); pins number 2 and 3 are used as OSC1 and OSC2.**

## MCU PIC 8 pins format socket

Pins of PDIP8 microcontrollers PIC.

The serially programming use the following lines:

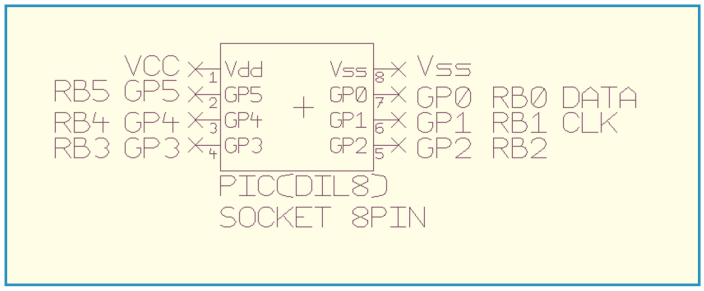
**PGD** (Program Data): GP0 **pin number 7**

**PGC** (Program Clock): GP1 **pin number 6**

**VPP** (Programming Voltage): GP3 **pin number 4**

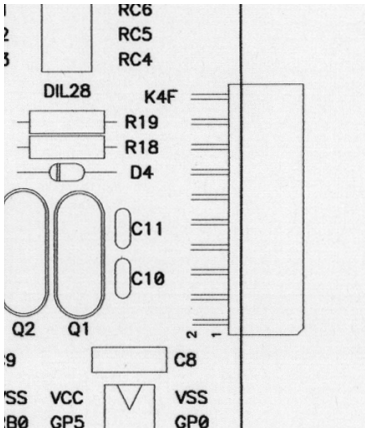
**VDD** (Positive Power Voltage): Vdd **pin number 1**

**GND** (Ground): Vss **pin number 8**



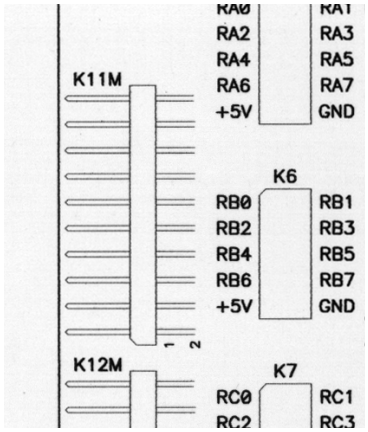

**GP lines are connected at PortB lines:**
**GP0=RB0; GP1=RB1; GP2=RB2;**
**GP3=RB3; GP4=RB4; GP5=RB5.**

## Input connector K4F



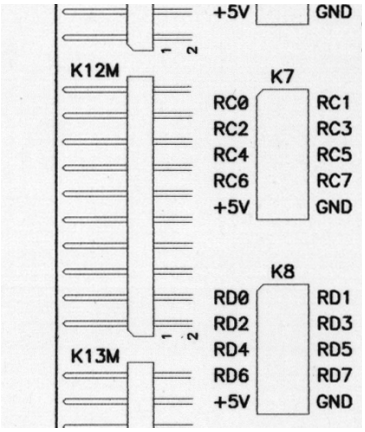

| | | | |
|---|---|---|---|
| | **20** | **19** | |
| | **18** | **17** | |
| RB3 | **16** | **15** | RB4 |
| RB5 | **14** | **13** | RB6 |
| RB7 | **12** | **11** | MCLR |
| VDD | **10** | **9** | VDD |
| GND | **8** | **7** | GND |
| +5V | **6** | **5** | +5V |
| 15VDC | **4** | **3** | 15VDC |
| VBAT | **2** | **1** | VBAT |

## Output connetor K11M



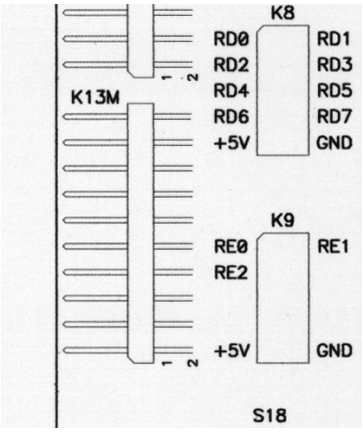

| | | | |
|---|---|---|---|
| | **19** | **20** | |
| | **17** | **18** | |
| | **15** | **16** | |
| | **13** | **14** | |
| | **11** | **12** | |
| | **9** | **10** | |
| GND | **7** | **8** | GND |
| +5V | **5** | **6** | +5V |
| 15VDC | **3** | **4** | 15VDC |
| VBAT | **1** | **2** | VBAT |

## Output connetor K12M



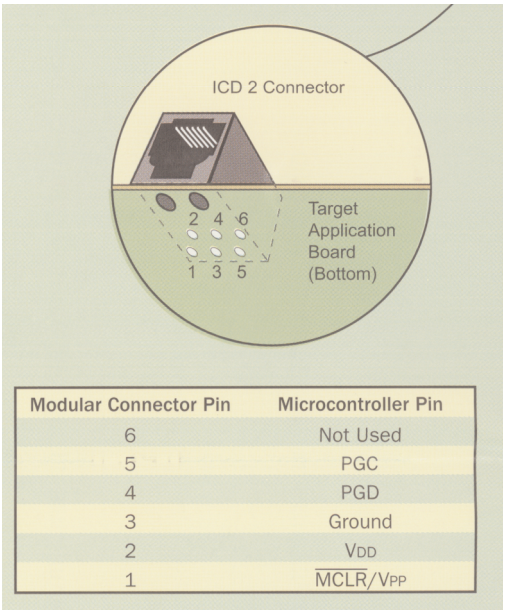

| | | | |
|---|---|---|---|
| | **19** | **20** | |
| | **17** | **18** | |
| RB6 | **15** | **16** | RB7 |
| RB4 | **13** | **14** | RB5 |
| RB2 | **11** | **12** | RB3 |
| RB0 | **9** | **10** | RB1 |
| RA6 | **7** | **8** | RA7 |
| RA4 | **5** | **6** | RA5 |
| RA2 | **3** | **4** | RA3 |
| RA0 | **1** | **2** | RA1 |

## Output connetor K13M

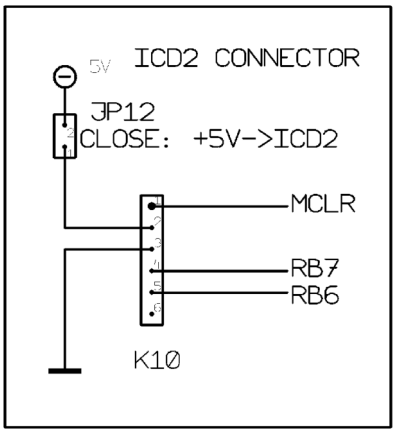



| | | | |
|---|---|---|---|
| RE2 | **19** | **20** | |
| RE0 | **17** | **18** | RE1 |
| RD6 | **15** | **16** | RD7 |
| RD4 | **13** | **14** | RD5 |
| RD2 | **11** | **12** | RD3 |
| RD0 | **9** | **10** | RD1 |
| RC6 | **7** | **8** | RC7 |
| RC4 | **5** | **6** | RC5 |
| RC2 | **3** | **4** | RC3 |
| RC0 | **1** | **2** | RC1 |

## ICD2 connector




| Modular Connector Pin | Microcontroller Pin |
|---|---|
| 6 | Not Used |
| 5 | PGC |
| 4 | PGD |
| 3 | Ground |
| 2 | VDD |
| 1 | $\overline{\text{MCLR}}$/VPP |



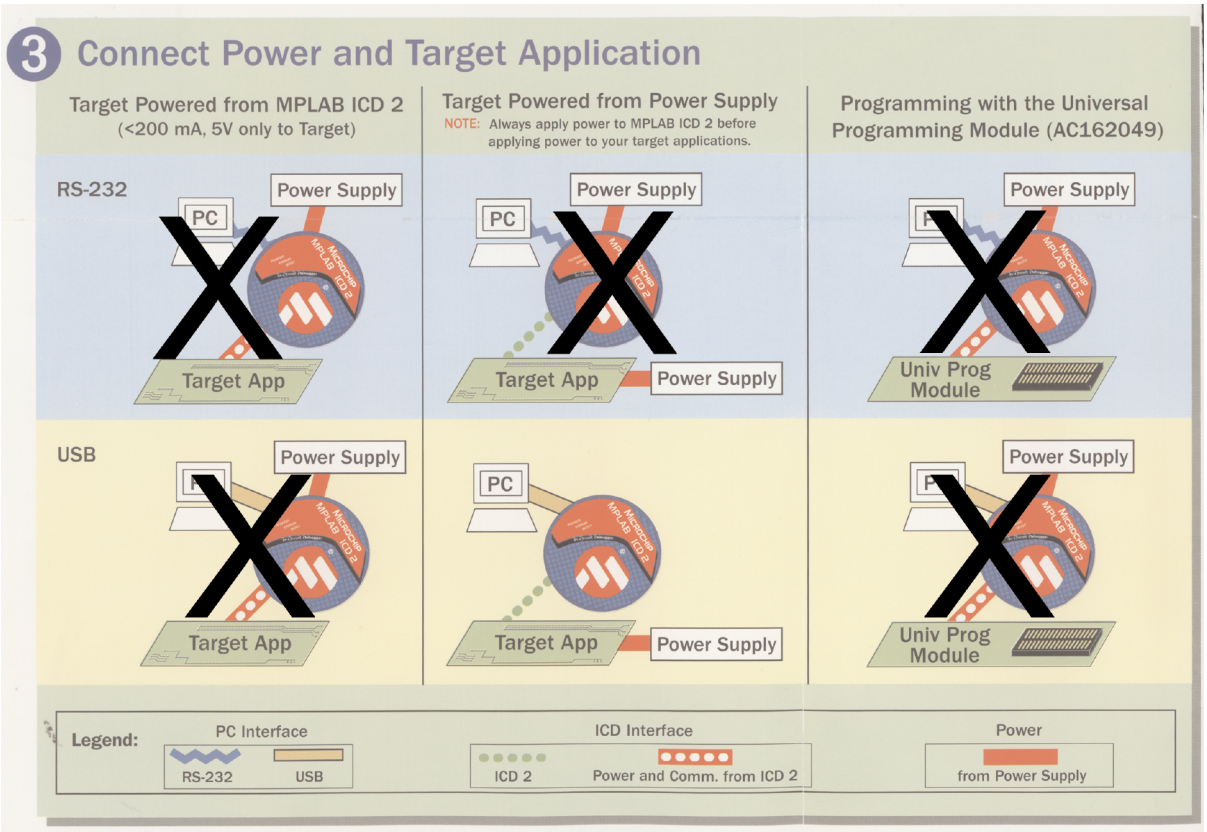
## Use of ICD2

C-203 board could be managed by Microchip In-Circuit Debugger/Programmer ICD2.

Before using ICD2 apply the following configurations:
**-Disconnect parallel cable that connect PC parallel port at C-202 programming board** and verify that red led (VPP) it's off. In this way the programming voltage (managed by ICD2) will never clashed with voltage VPP that could be unintentionally generated by C-202 programming board.
- **Open JP7 jumper.** In this way PGD (Program Data) signal it's disconnected by C-202 programming board.
- **Open JP8 jumper.** In this way PGC (Program Clock) signal it's disconnected by C-202 programming board.
- **Always** (during both programming and debug operations) **keep S18 switch** (C-203 Socket board) **on RUN position.** In this way pins VCC of every socket are always connected at voltage +5 V.

The system it's planned to interface with ICD2 using a **USB communication** and with **Target Power On**; then it's necessary set up the system as follow:

- **ICD2 connected at PC through included USB cable;**
- **ICD2 Power OFF;**;
- **C-202 programming board Power ON;**
- **JP12 jumper of C-203 Socket board closed**. This operation allow to bring the voltage +5 V at ICD2 drivers. To install MPLAB IDE, the configuration of MPLAB IDE with ICD2 and the installation of USB drivers please refer at ICD2 manual.




**Attention:**
**when system ICD2 is used it's necessary disconnect parallel cable.**

## Example about the use of ICD2

To help you on how using ICD2, we create a project file named T18F4320.MCP available into ICD2 folder. This testing program sequentially and intervalled by 2 seconds, turn on MCU lines. Now let's see how programming a microcontroller PIC18F4320-I/P starting from available project file and using ICD2.

**All these operations must be executed after the installation of software MPLAB IDE and USB configuration ICD2 with power taken from Target (refer at ICD2 manual) and after that the C-202 and C-203 boards are configurated how explained in "USE OF ICD2" paragraph.**

- Launch MPLAB IDE;

- Select Project, then Open and choose the file T18F4320.MCP;

- In the Project window choose Source Files, click the right button of mouse, choose Add Files, choose T18F4320.ASM (this file is available in the ICD2 folder);

- Choose Configure, choose Select Device, choose PIC18F4320;

- Choose Configure, choose Configuration Bits and set up the following parameters: Oscillator=HS; Watchdog Timer=Disable; Low Voltage Program=Disable; all other parameters could be leaved in the default condition.

- Choose Project, choose Build All: then file T18F4320.ASM will be compilated;

- Verify that C-202 programming board and C-203 Socket board are configurated how explained in "USE OF ICD2" paragraph;

- **Verify that parallel cable is disconneted**;

- Insert a MCU PIC18F4320 into 40 pins socket;

- Close C-202 programming board JP2 jumper; in this way microcontroller PGM pin is connected at ground through a 10 KOhm resistor;

- Power On C-202 programming board with a stabilizzated and direct voltage of 17 Vdc;

- Insert ICD2 plug into K10 connector available on C-203 Socket board;

- Choose Programmer, choose Connect;

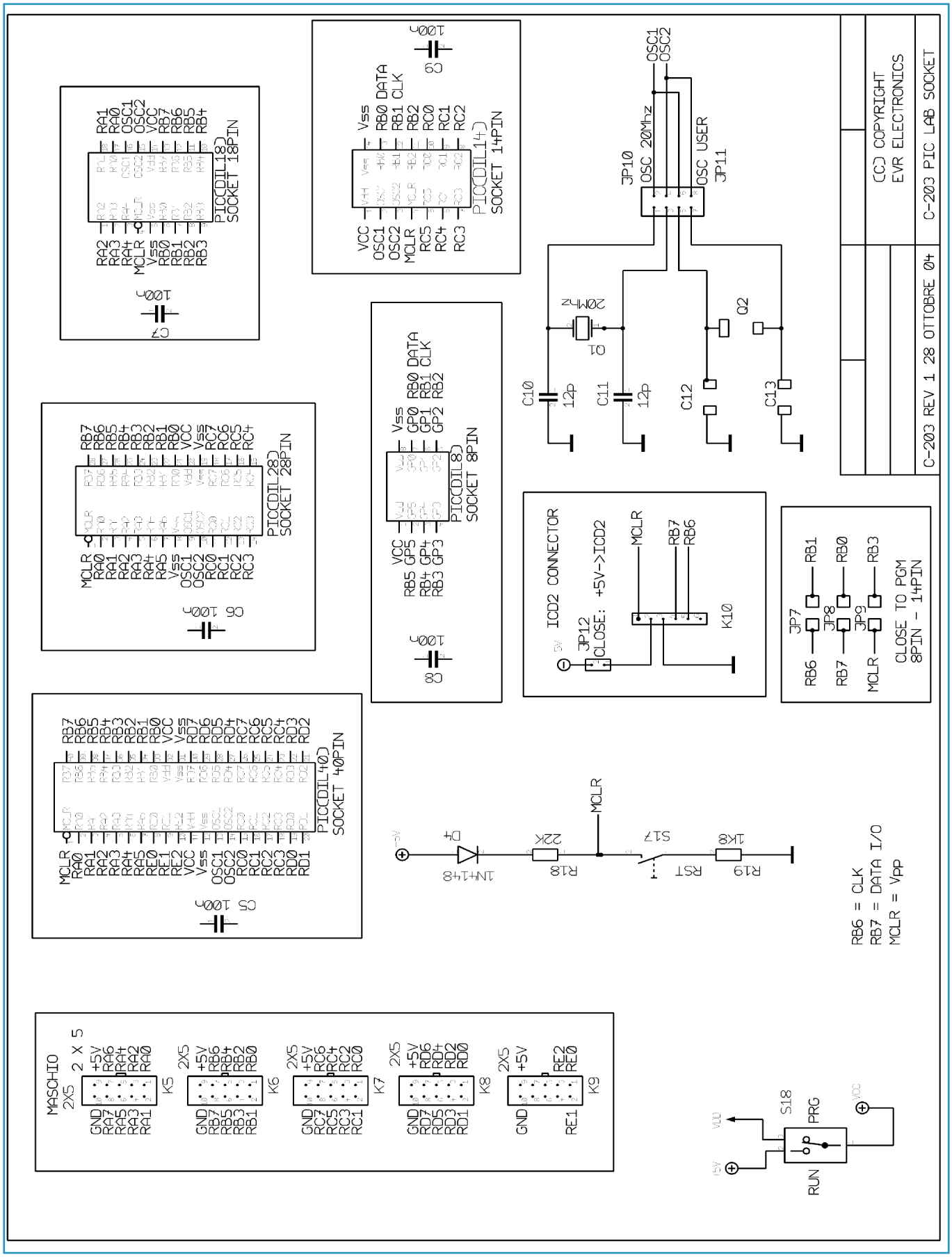
- Choose Programmer, choose Program; file T18F4320 will be writing into microcontroller program memory, MPLAB Output window shows the result of every operation;

- Open C-202 programming board JP2 jumper;

- Remove ICD2 plug from K10 connector available on C-203 Socket board;

- Verify the correctness of program writed into MCU memory.

# Electric plan Socket C-203 board



**PART LIST C-203**

**R18**: 22 Kohm
**R19**: 1,8 Kohm
**D4**: 1N4148
**C5**: 100 nF
**C6**: 100 nF
**C7**: 100 nF
**C8**: 100 nF
**C9**: 100 nF
**C10**: 12 pF
**C11**: 12 pF
**C12**: not mounted
**C13**: not mounted
**Q1**: 20 MHz
**Q2**: not mounted
**S17**: pushbutton cs
**S18**: shunter cs
**DIL8**: socket 8 pins
**DIL14**: socket 14 pins
**DIL18**: socket 18 pins
**DIL28**: socket 28 pins
**DIL40**: socket 40 pins
**K4F**: pin-strip F 90° 10+10 pins
**K5**: pin-strip M 5+5 pins
**K6**: pin-strip M 5+5 pins
**K7**: pin-strip M 5+5 pins
**K8**: pin-strip M 5+5 pins
**K9**: pin-strip M 5+5 pins
**K10**: connector FCC 6/6
**K11M**: pin-strip M 90° 10+10 pins
**K12M**: pin-strip M 90° 10+10 pins
**K13M**: pin-strip M 90° 10+10 pins
**JP7, JP8, JP9**: pin-strip M 3+3 pins
**JP10**: pin-strip M 2+2 pins
**JP11**: pin-strip M 2+2 pins
**JP12**: pin-strip M 2 pins
**-** CS C203-2

# PROGRAMMING SOFTWARE

## Introduction

IC-Prog is a software really versatile and simply to use and it could program almost the totality of Microchip microcontroller PICs. For more informations about IC-Prog performances please refer at the program help file.
IC-Prog needs a IBM PC compatible and works in Windows 95, 98, ME, NT, 2000, XP operative systems.

## Installing

IC-Prog software is available as compressed folder; there are two files:
icprog105.zip and icprog_driver.zip.

To install the software proceed as following:
- Create on your own PC, into Program folder, a new folder named IC-Prog;
- Unzip icprog105.zip file into IC-Prog folder of your own PC;
- Unzip icprog_driver.zip into IC-Prog folder of your own PC;

Right-click the file icprog.exe contents into IC-Prog folder, choose Property and then Compatiblity; with mouse choose "Run the program in modality compatiblity for" and choose one of following modality (Windows 95, Windows 98/Windows ME, Windows NT, Windows 2000) in accordance with the version of your own operative system.

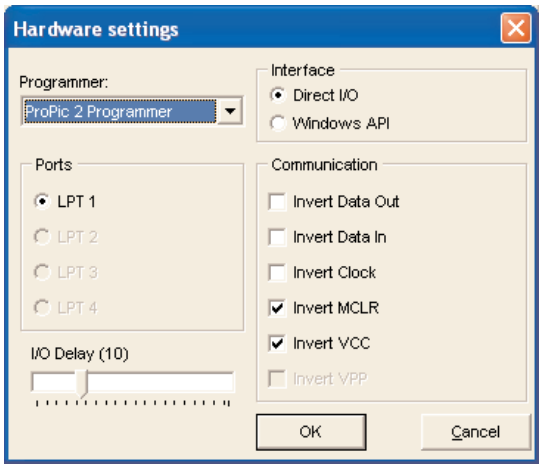
## Launch

To start the program, double-click the file icprog.exe characterized by an integrated-circuit icone; then the main window will be show. The main window of IC-Prog shows all the informations about the selected device needed, in our case choosing every device PIC, the main window could be ever subdivided in 4 sections:

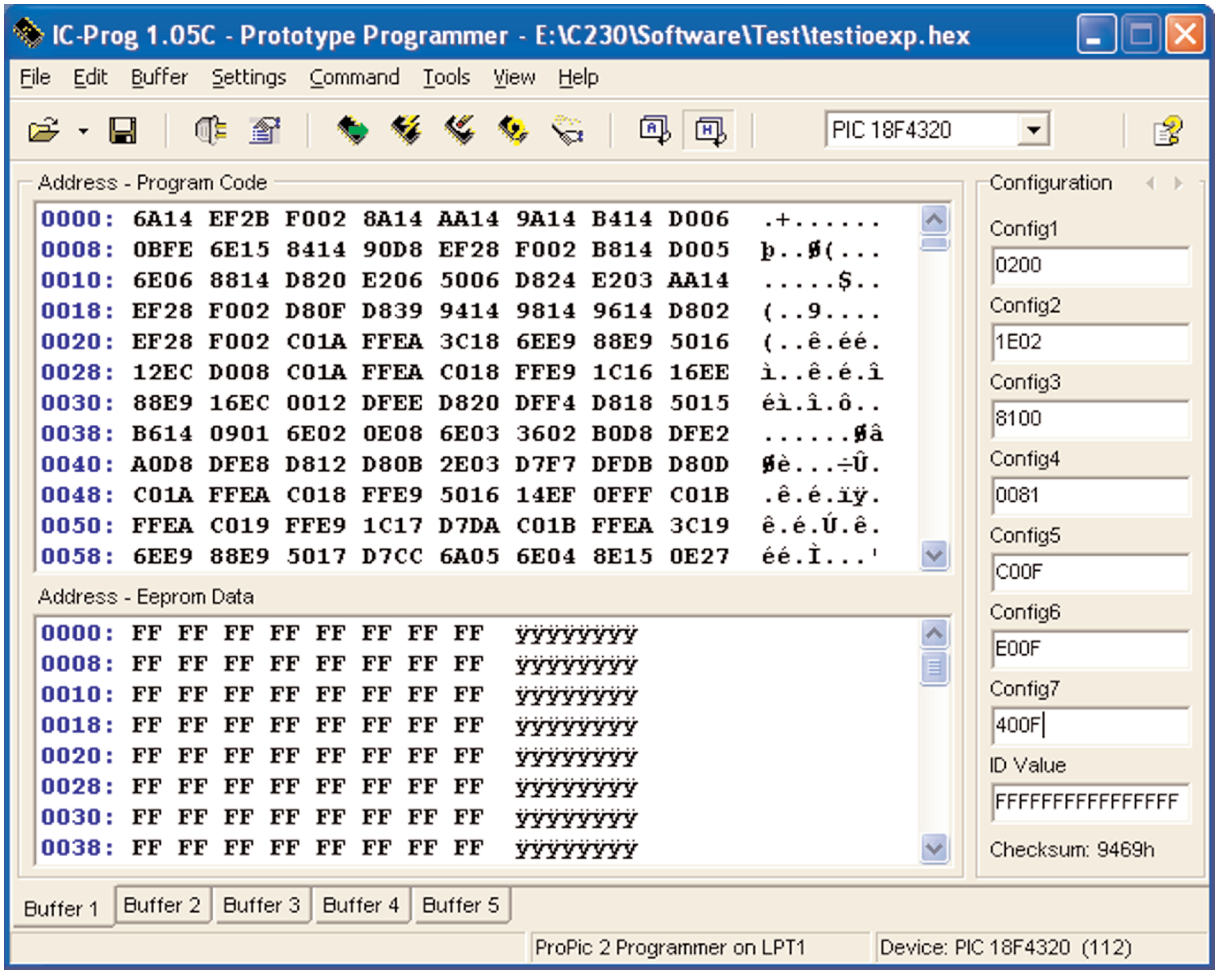
- commands bar;
- Program Area (Program Code) where the code of program that will be writed into microcontroller PIC or that it's been readed from microcontroller PIC is shown. The informations are shown on more lines, depending by the dimension of selected device program memory. Every line shows on the first column the hardware address of the first byte of program memory shows in the line; then there are the contents of 8 bytes of program memory express in hexadecimal notation; in the last column there is the content of the same 8 bytes of program memory but express in ASCII code. The content of every byte is express by a 16-bit word; to express 16 bits IC-Prog use exadecimal words between 0000 and FFFF. Not all microcontroller PIC devices use 16 bit words, but for example 14, 12 and 8 bits for word. In this case IC-Prog use the same a 4 numeral hexadecimal notation, showing in the code area 3FFF, 0FFF and 00FF for 14, 12 and 8 bits.
- Data Area (Eeprom Data) where the contents of microcontroller PIC Data Memory EEPROM is shown. In this case too, the informations are shown on more lines, depending by the dimension of selected device EEPROM memory. Every line shows on the first column the hardware address of the first byte of EEPROM memory shows in the line; then there are the contents of 8 bytes of EEPROM memory express in hexadecimal notation; in the last column there is the content of the same 8 bytes of EEPROM memory but express in ASCII code. The content of every byte is express by a 8-bit byte express in hexadecimal notation, then using 2 numeral words between 00 and FF.
- Configuration of device Area (Configuration) that depending by the microcontroller PIC selected.

## Configuration

Since IC-Prog software is compatible with a lot of hardware resources, it's necessary configure it for the C-202 programming board.
Then from the Menù Bar choose Setting and then Hardware: the Hardware Configuration window will be shown.
Choose the ProPic 2 as programmer (Programmer=ProPic 2 Programmer).
Choose between LPT1, LPT2, LPT3, LPT4, the parallel port connected at C-202 programmer board.
Set the switching input/output delay on 10; I/O Delay (10); this value should be compatible for all computers, last generations too. If during programming phase problems are rediscounted, then increase this delay value.
Select direct interface: Interface=Direct I/O.
Set up communication values choosing Invert MCLR and Invert Vcc.
Click the OK button, to accept the settings: then all the settings will be saved by IC-Prog. In this way in the future will not be necessary to set up the settings again.



## IC-Prog Main Window



# PROGRAMMING WITH IC-PROG

## Preliminary operations

The C-170 board supports the PDIP socket MCU PIC in 40, 28, 18, 14 and 8 pins format.
If a different PDIP socket MCU PIC is used, then the use of a proper adapter will be necessary.
Connect the C-202 programming board at PC parallel port.
Get a power generator of **stabilized voltage 17 Vdc and minimum current of 300 mA**.
Remove resource section (for example C-230 or C-180) from the rest of the board.

Verify that jumpers are configurated as follows:

**C-202 Programming Section:**

**JP1** - VDD->ICSP = OPEN
**JP2** - RB3->GND = depends by the MCU that will be programmed (refer at following pags)
**JP3** - RB4->GND = depends by the MCU that will be programmed (refer at following pags)
**JP4** - RB5->GND = depends by the MCU that will be programmed (refer at following pags)
**JP5** - LPT D4 = CLOSE
**JP6** - LPT D5 = CLOSE
**JP7** - PGD = CLOSE
**JP8** - PGC = CLOSE

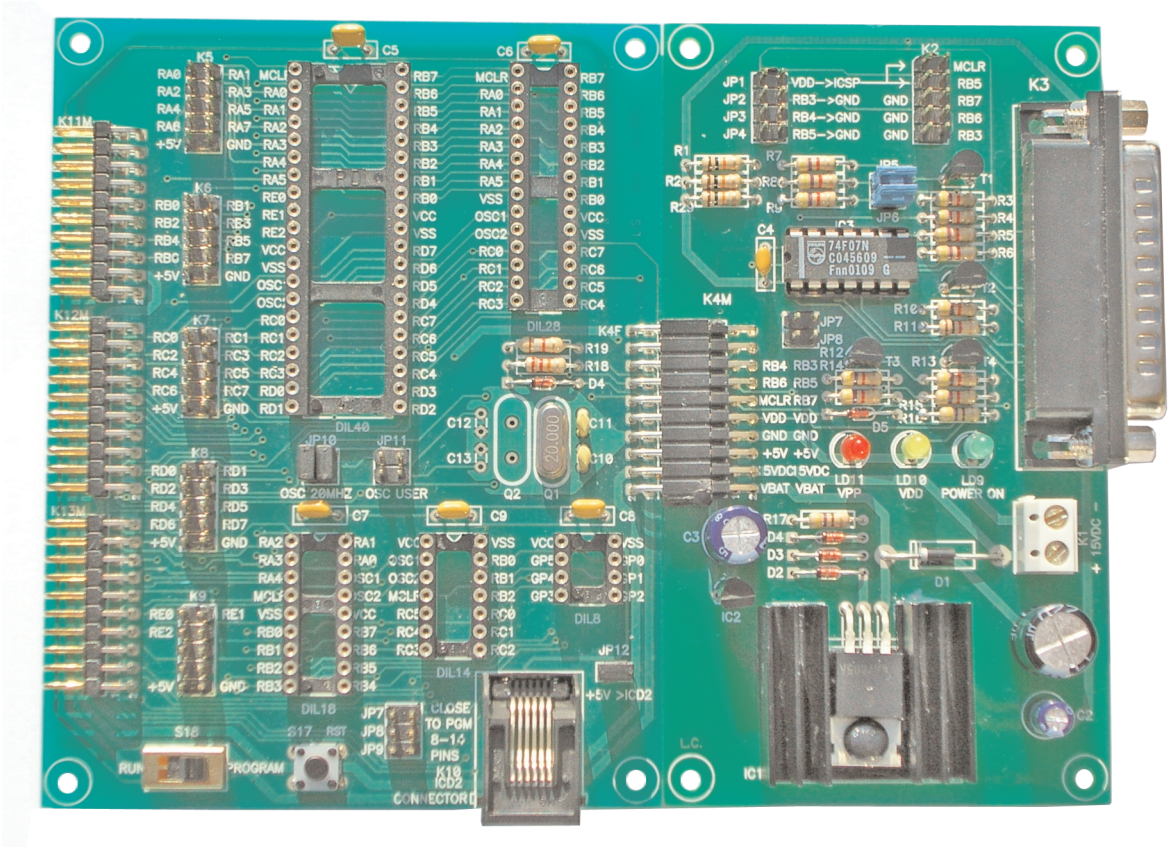
**C-203 Socket Section:**

**JP7** - RB6->RB1 = OPEN for MCU 40-28-18 pins format / CLOSE for MCU 14-8 pins format
**JP8** - RB7->RB0 = OPEN for MCU 40-28-18 pins format / CLOSE for MCU 14-8 pins format
**JP9** - MCLR->RB3 = OPEN for MCU 40-28-18 pins format / CLOSE for MCU 14-8 pins format
**JP10** - OSC 20MHz = CLOSE close with 2 jumpers in vertical position
**JP11** - OSC USER = OPEN

**S18** = PROGRAM

There is a testing program for every PDIP socket choosing one of more diffuse microcontroller PIC.
The program will set up all the microcontroller PIC I/O lines as output and sequentially turn on (level logic high) the MCU lines. Code, jumper and IC-Prog configuration are reported in the following pages.

## Notes

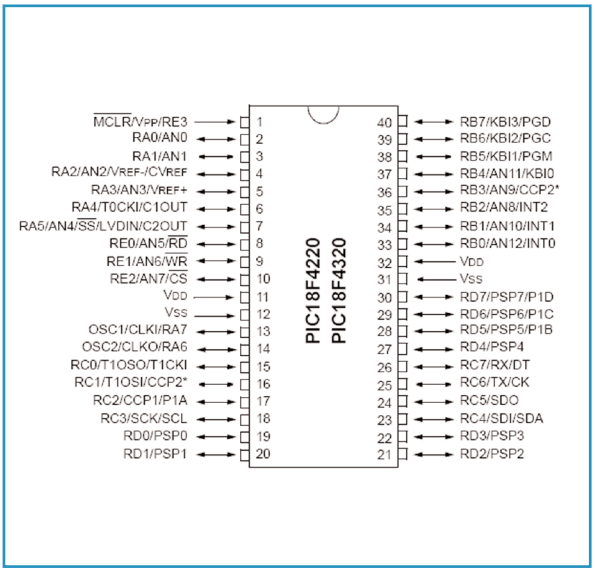


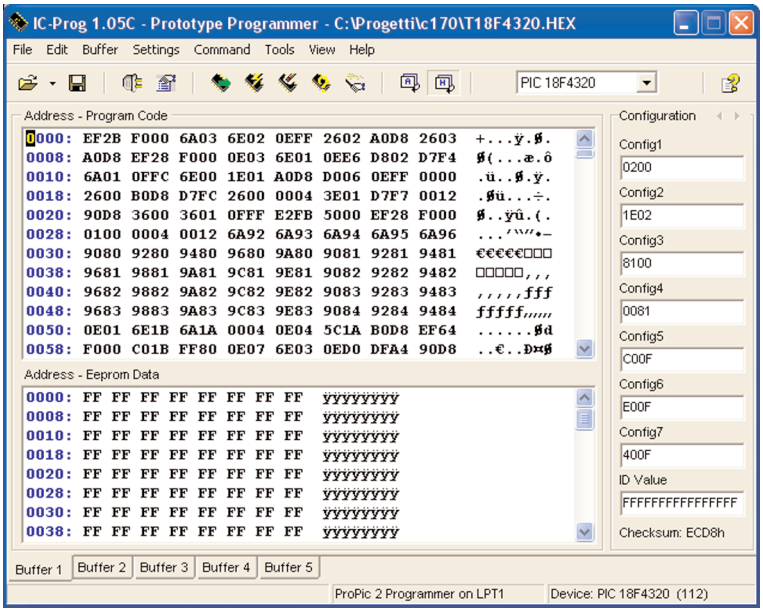
## MCU PIC 40 pins format (PIC18F4320)

In this section the operations necessary to write or read a microcontroller PIC 40 pins format are shown.
The operations remains the same for every MCU PIC in 40 pins format, but for semplicity we will use **PIC18F4320-I/P**, one of more diffuse microcontroller PIC in 40 pins format.
There is a testing program, available on CD whether source code or executable format, named **T18F4320** writed using PicBasic Pro language.

Let's see now the phases necessary to writing the demo file T18F4320.HEX into the microcontroller's memory.

- Launch IC-Prog software and choose **PIC18F4320** device
- Choose, from Menù Bar, Menù File
- Choose Open File command and then **T18F4320.HEX** file
- Set up the Config bits as shown in the follow figure;
- Insert a **PIC18F4320-I/P** device into 40 pins DIL40 format socket respecting the polarity
- Power on the board
- Verify that **JP7** and **JP8** jumpers of the C-202 board are closed
- Close **JP4** jumper of C-202 programming board to connect microcontroller PIC PGM pin at ground
- Switch **S18** switching in **PROGRAM** position
- Choose **Program All** command: then the sign "Do you really want to program the devices" will be shown
- Choose **Yes** option
- Then the programming phase will be started; this phase is composed by 3 sub-phase, everyone marked by a Progression Bar: Programming Code; Programming Data and Programming Config
- Then the verify of writed data's phase will be started; this phase is marked by a Progression Bar too
- If verify phase results ok, then the sign "Device Successfully Verified" will be shown and this means that microcontroller PIC is been correctly programmed
- Remove **JP4, JP7** and **JP8** jumpers from C-202 board and switch **S18** switching in **RUN** position
- Verify with an oscilloscope or with a tester that the microcontroller PIC's lines are sequentially turn at ON (logic level high).




Note: RA4 is an open collector port; this means that it needs a pull-up external resistor and, for this reason, it's excluded from our test.

# T18F4320.BAS

```
'*****************************************************************
'*  File Name: T18F4320
'*  Notice  : Copyright (c) 2005 EVR electronics
'*          : All Rights Reserved
'*  Date    : 26/01/2005
'*****************************************************************

    DEFINE  OSC 20

    TRISA   =   %00000000   'PORTA OUT
    TRISB   =   %00000000   'PORTB OUT
    TRISC   =   %00000000   'PORTC OUT
    TRISD   =   %00000000   'PORTD OUT
    TRISE   =   %00000000   'PORTE OUT

    TEMP    VAR BYTE
    CONTA   VAR BYTE

    PORTA.0 =   0
    PORTA.1 =   0
    PORTA.2 =   0
    PORTA.3 =   0
    PORTA.5 =   0

    PORTB.0 =   0
    PORTB.1 =   0
    PORTB.2 =   0
    PORTB.3 =   0
    PORTB.4 =   0
    PORTB.5 =   0
    PORTB.6 =   0
    PORTB.7 =   0

    PORTC.0 =   0
    PORTC.1 =   0
    PORTC.2 =   0
    PORTC.3 =   0
    PORTC.4 =   0
    PORTC.5 =   0
    PORTC.6 =   0
    PORTC.7 =   0

    PORTD.0 =   0
    PORTD.1 =   0
    PORTD.2 =   0
    PORTD.3 =   0
    PORTD.4 =   0
    PORTD.5 =   0
    PORTD.6 =   0
    PORTD.7 =   0

    PORTE.0 =   0
    PORTE.1 =   0
    PORTE.2 =   0

LOOP:
    TEMP = 1
    for CONTA = 0 TO 3
        PORTA =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT CONTa
    PORTA.3 = 0
    PORTA.5 = 1
    PAUSE 2000
    PORTA.5 = 0
    PAUSE 2000

    TEMP = 1
    for CONTA = 0 TO 7
        PORTB =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT conta
    PORTB.7 = 0
    pause 2000

    TEMP = 1
    for CONTA = 0 TO 7
        PORTC =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT conta
    PORTC.7 = 0
    pause 2000

    TEMP = 1
    for CONTA = 0 TO 7
        PORTD =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT conta
    PORTD.7 = 0
    pause 2000

    TEMP = 1
    for CONTA = 0 TO 2
        PORTE =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT conta
    PORTE.2 = 0
    pause 2000

    Goto LOOP
```

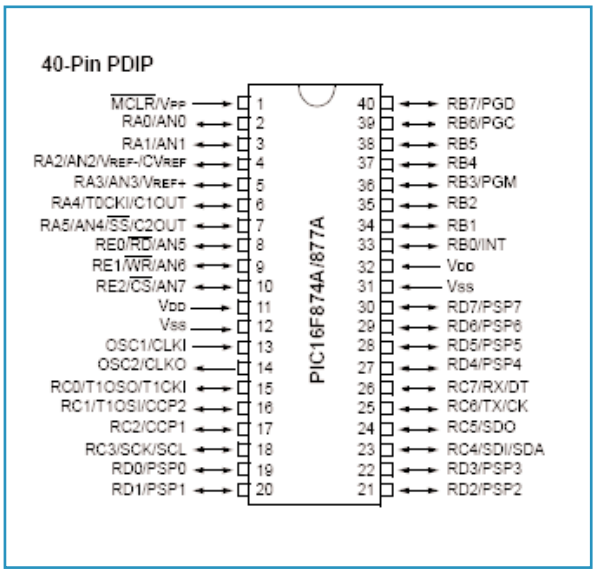
## MCU PIC 40 pins format (PIC16F877A)

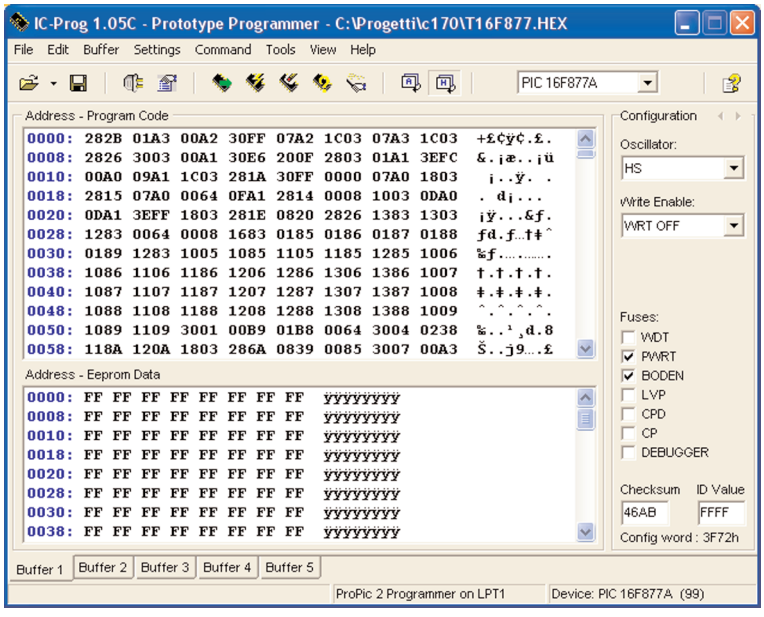
In this section the operations necessary to write or read a microcontroller PIC 40 pins format are shown.
The operations remains the same for every MCU PIC in 40 pins format, but for semplicity we will use **PIC16F877A-I/P**, one of more diffuse microcontroller PIC in 40 pins format.
There is a testing program, available on CD whether source code or executable format, named **T16F877** writed using PicBasic language.

Let's see now the phases necessary to writing the demo file T16F877.HEX into the microcontroller's memory.

- Launch IC-Prog software and choose **PIC16F877A** device
- Choose, from Menù Bar, Menù File
- Choose Open File command and then **T16F877.HEX** file
- Set up the oscillator config as HS **(Oscillator = HS)**
- Disable the writing in memory **(Write Enable = WRT OFF)**
- Select **PWRT** and **BODEN** and verify that all the others options of Fuses group are disabled
- Insert a **PIC16F877A** device into 40 pins DIL40 format socket respecting the polarity
- Power on the board
- Verify that **JP7** and **JP8** jumpers of the C-202 board are closed
- Close **JP2** jumper of C-202 programming board to connect microcontroller PIC PGM pin at ground
- Switch **S18** switching in **PROGRAM** position
- Choose **Program All** command: then the sign "Do you really want to program the devices" will be shown
- Choose **Yes** option
- Then the programming phase will be started; this phase is composed by 3 sub-phase, everyone marked by a Progression Bar: Programming Code; Programming Data and Programming Config
- Then the verify of writed data's phase will be started; this phase is marked by a Progression Bar too
- If verify phase results ok, then the sign "Device Successfully Verified" will be shown and this means that microcontroller PIC is been correctly programmed
- Remove **JP2, JP7** and **JP8** jumpers from C-202 board and switch **S18** switching in **RUN** position
- Verify with an oscilloscope or with a tester that the microcontroller PIC's lines are sequentially turn at ON (logic level high).

Note: RA4 is an open collector port; this means that it needs a pull-up external resistor and, for this reason, it's excluded from our test.



# T16F877.BAS

```
'*****************************************************************
'*  File Name: T16F877
'*  Notice  : Copyright (c) 2005 EVR electronics
'*          : All Rights Reserved
'*  Date    : 26/01/2005
'*****************************************************************

   DEFINE  OSC 20

   TRISA   =   %00000000   'PORTA OUT
   TRISB   =   %00000000   'PORTB OUT
   TRISC   =   %00000000   'PORTC OUT
   TRISD   =   %00000000   'PORTD OUT
   TRISE   =   %00000000   'PORTE OUT

   TEMP    VAR BYTE
   CONTA   VAR BYTE

   PORTA.0 =   0
   PORTA.1 =   0
   PORTA.2 =   0
   PORTA.3 =   0
   PORTA.5 =   0

   PORTB.0 =   0
   PORTB.1 =   0
   PORTB.2 =   0
   PORTB.3 =   0
   PORTB.4 =   0
   PORTB.5 =   0
   PORTB.6 =   0
   PORTB.7 =   0

   PORTC.0 =   0
   PORTC.1 =   0
   PORTC.2 =   0
   PORTC.3 =   0
   PORTC.4 =   0
   PORTC.5 =   0
   PORTC.6 =   0
   PORTC.7 =   0

   PORTD.0 =   0
   PORTD.1 =   0
   PORTD.2 =   0
   PORTD.3 =   0
   PORTD.4 =   0
   PORTD.5 =   0
   PORTD.6 =   0
   PORTD.7 =   0

   PORTE.0 =   0
   PORTE.1 =   0
   PORTE.2 =   0

LOOP:
   TEMP = 1
   for CONTA = 0 TO 3
      PORTA =   TEMP
      PAUSE   2000
      TEMP = TEMP << 1
   NEXT CONTa
   PORTA.3 = 0
   PORTA.5 = 1
   PAUSE 2000
   PORTA.5 = 0
   PAUSE 2000

   TEMP = 1
   for CONTA = 0 TO 7
      PORTB =   TEMP
      PAUSE   2000
      TEMP = TEMP << 1
   NEXT conta
   PORTB.7 = 0
   pause 2000

   TEMP = 1
   for CONTA = 0 TO 7
      PORTC =   TEMP
      PAUSE   2000
      TEMP = TEMP << 1
   NEXT conta
   PORTC.7 = 0
   pause 2000

   TEMP = 1
   for CONTA = 0 TO 7
      PORTD =   TEMP
      PAUSE   2000
      TEMP = TEMP << 1
   NEXT conta
   PORTD.7 = 0
   pause 2000

   TEMP = 1
   for CONTA = 0 TO 2
      PORTE =   TEMP
      PAUSE   2000
      TEMP = TEMP << 1
   NEXT conta
   PORTE.2 = 0
   pause 2000

   Goto LOOP
```

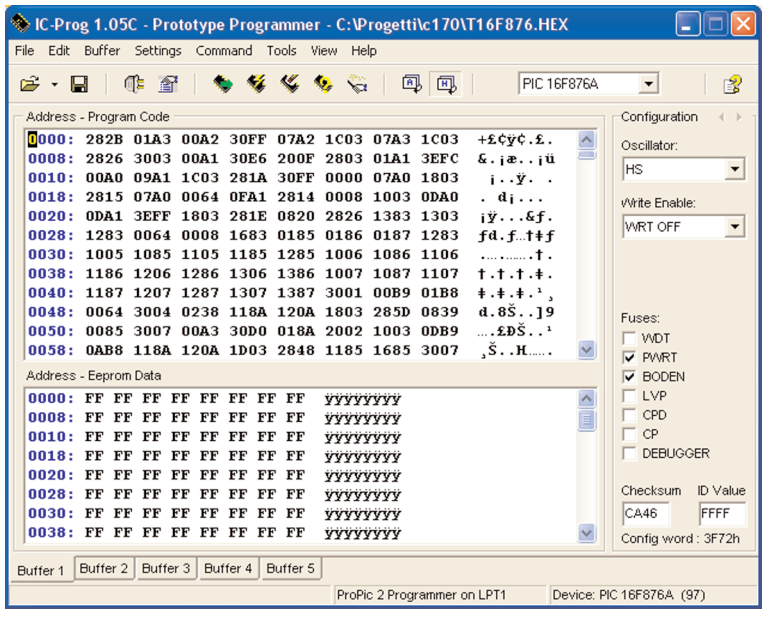
## MCU PIC 28 pins format (PIC16F876A)

In this section the operations necessary to write or read a microcontroller PIC 28 pins format are shown.
The operations remains the same for every MCU PIC in 28 pins format, but for semplicity we will use **PIC16F876A-I/SP**, one of more diffuse microcontroller PIC in 28 pins format.
There is a testing program, available on CD whether source code or executable format, named **T16F876** writed using PicBasic language.

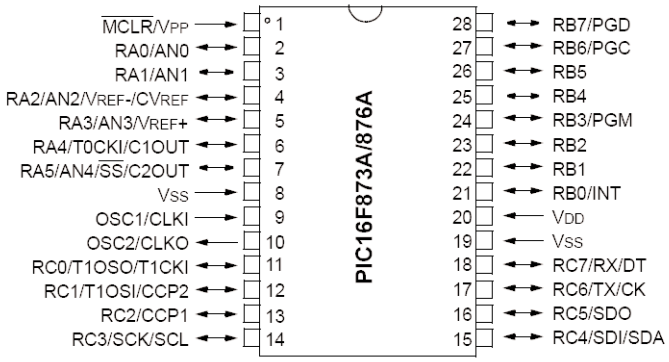
Let's see now the phases necessary to writing the demo file T16F876.HEX into the microcontroller's memory.

- Launch IC-Prog software and choose **PIC16F876A** device
- Choose, from Menù Bar, Menù File
- Choose Open File command and then **T16F876.HEX** file
- Set up the oscillator config as HS **(Oscillator = HS)**
- Disable the writing in memory **(Write Enable = WRT OFF)**
- Select **PWRT** and **BODEN** and verify that all the others options of Fuses group are disabled
- Insert a **PIC16F876A** device into 28 pins DIL28 format socket respecting the polarity
- Power on the board
- Verify that **JP7** and **JP8** jumpers of the C-202 board are closed
- Close **JP2** jumper of C-202 programming board to connect microcontroller PIC PGM pin at ground
- Switch **S18** switching in **PROGRAM** position
- Choose **Program All** command: then the sign "Do you really want to program the devices" will be shown
- Choose **Yes** option
- Then the programming phase will be started; this phase is composed by 3 sub-phase, everyone marked by a Progression Bar: Programming Code; Programming Data and Programming Config
- Then the verify of writed data's phase will be started; this phase is marked by a Progression Bar too
- If verify phase results ok, then the sign "Device Successfully Verified" will be shown and this means that microcontroller PIC is been correctly programmed
- Remove **JP2, JP7** and **JP8** jumpers from C-202 board and switch **S18** switching in **RUN** position
- Verify with an oscilloscope or with a tester that the microcontroller PIC's lines are sequentially turn at ON (logic level high).

Note: RA4 is an open collector port; this means that it needs a pull-up external resistor and, for this reason, it's excluded from our test.



**28-Pin PDIP, SOIC, SSOP**




## T16F876.BAS

```
'*****************************************************************
'*  File Name: T16F876
'*  Notice  : Copyright (c) 2005 EVR electronics
'*          : All Rights Reserved
'*  Date    : 26/01/2005
'*****************************************************************

    DEFINE  OSC 20

    TRISA   =   %00000000   'PORTA OUT
    TRISB   =   %00000000   'PORTB OUT
    TRISC   =   %00000000   'PORTC OUT

    TEMP    VAR BYTE
    CONTA   VAR BYTE

    PORTA.0 =   0
    PORTA.1 =   0
    PORTA.2 =   0
    PORTA.3 =   0
    PORTA.5 =   0

    PORTB.0 =   0
    PORTB.1 =   0
    PORTB.2 =   0
    PORTB.3 =   0
    PORTB.4 =   0
    PORTB.5 =   0
    PORTB.6 =   0
    PORTB.7 =   0

    PORTC.0 =   0
    PORTC.1 =   0
    PORTC.2 =   0
    PORTC.3 =   0
    PORTC.4 =   0
    PORTC.5 =   0
    PORTC.6 =   0
    PORTC.7 =   0


LOOP:
    TEMP = 1
    for CONTA = 0 TO 3
        PORTA =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT CONTa
    PORTA.3 = 0
    PORTA.5 = 1
    PAUSE 2000
    PORTA.5 = 0
    PAUSE 2000

    TEMP = 1
    for CONTA = 0 TO 7
        PORTB =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT conta
    PORTB.7 = 0
    pause 2000

    TEMP = 1
    for CONTA = 0 TO 7
        PORTC =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT conta
    PORTC.7 = 0
    pause 2000

    Goto LOOP
```

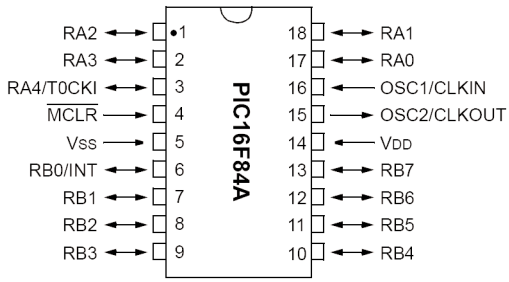
## MCU PIC 18 pins format (PIC16F84A)

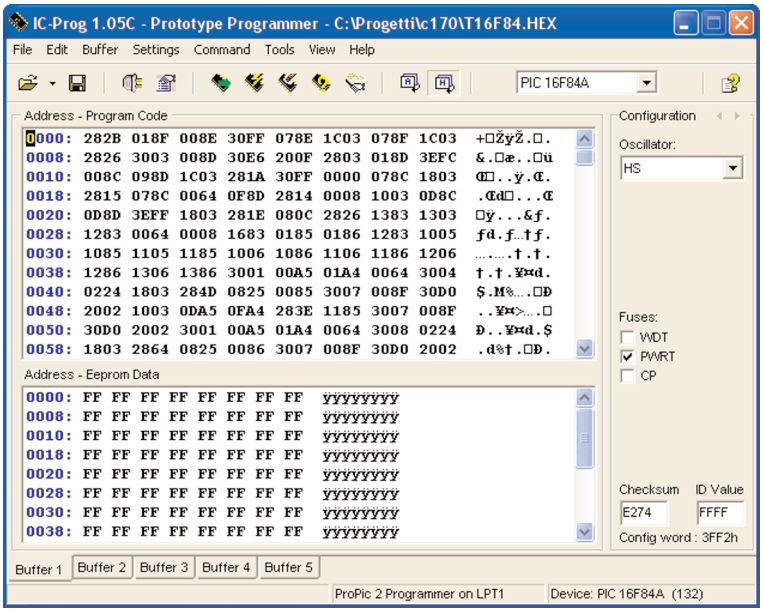
In this section the operations necessary to write or read a microcontroller PIC 18 pins format are shown.
The operations remains the same for every MCU PIC in 18 pins format, but for semplicity we will use **PIC16F84A-20I/P**, one of more diffuse microcontroller PIC in 18 pins format.
There is a testing program, available on CD whether source code or executable format, named **T16F84** writed using PicBasic language.

Let's see now the phases necessary to writing the demo file T16F84.HEX into the microcontroller's memory.

- Launch IC-Prog software and choose **PIC16F84A** device
- Choose, from Menù Bar, Menù File
- Choose Open File command and then **T16F84.HEX** file
- Set up the oscillator config as HS **(Oscillator = HS)**
- Disable the writing in memory **(Write Enable = WRT OFF)**
- Select **PWRT** and verify that all the others options of Fuses group are disabled
- Insert a **PIC16F84A** device into 18 pins DIL18 format socket respecting the polarity
- Power on the board
- Verify that **JP7** and **JP8** jumpers of the C-202 board are closed
- Switch **S18** switching in **PROGRAM** position
- Choose **Program All** command: then the sign "Do you really want to program the devices" will be shown
- Choose **Yes** option
- Then the programming phase will be started; this phase is composed by 3 sub-phase, everyone marked by a Progression Bar: Programming Code; Programming Data and Programming Config
- Then the verify of writed data's phase will be started; this phase is marked by a Progression Bar too
- If verify phase results ok, then the sign "Device Successfully Verified" will be shown and this means that microcontroller PIC is been correctly programmed
- Remove **JP7** and **JP8** jumpers from C-202 board and switch **S18** switching in **RUN** position
- Verify with an oscilloscope or with a tester that the microcontroller PIC's lines are sequentially turn at ON (logic level high).

Note: RA4 is an open collector port; this means that it needs a pull-up external resistor and, for this reason, it's excluded from our test.



## T16F84.BAS

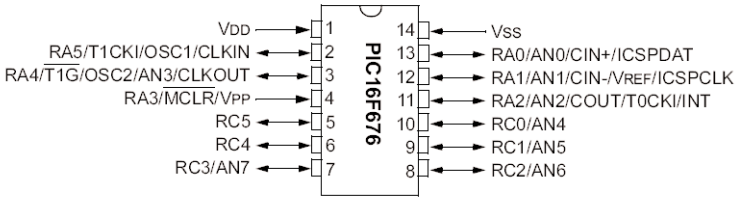
```
'*****************************************************************
'*  File Name: T16F84
'*  Notice  : Copyright (c) 2005 EVR electronics
'*          : All Rights Reserved
'*  Date    : 26/01/2005
'*****************************************************************

    DEFINE  OSC 20

    TRISA   =   %00000000   'PORTA OUT
    TRISB   =   %00000000   'PORTB OUT

    TEMP    VAR BYTE
    CONTA   VAR BYTE

    PORTA.0 =   0
    PORTA.1 =   0
    PORTA.2 =   0
    PORTA.3 =   0

    PORTB.0 =   0
    PORTB.1 =   0
    PORTB.2 =   0
    PORTB.3 =   0
    PORTB.4 =   0
    PORTB.5 =   0
    PORTB.6 =   0
    PORTB.7 =   0


LOOP:
    TEMP = 1
    for CONTA = 0 TO 3
        PORTA =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT CONTa
    PORTA.3 = 0
    pause 2000
    TEMP = 1
    for CONTA = 0 TO 7
        PORTB =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT conta
    PORTB.7 = 0
    pause 2000
    Goto LOOP
```

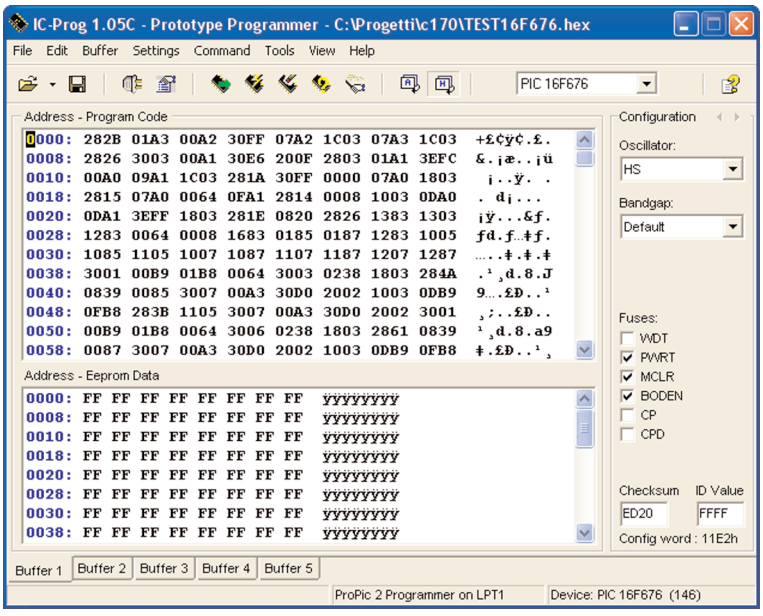
## MCU PIC 14 pins format (PIC16F676)

In this section the operations necessary to write or read a microcontroller PIC 14 pins format are shown.
The operations remains the same for every MCU PIC in 14 pins format, but for semplicity we will use **PIC16F676-I/P**, one of more diffuse microcontroller PIC in 14 pins format.
There is a testing program, available on CD whether source code or executable format, named **T16F676** writed using PicBasic language.

Let's see now the phases necessary to writing the demo file T16F676.HEX into the microcontroller's memory.

- Launch IC-Prog software and choose **PIC16F676** device
- Choose, from Menù Bar, Menù File
- Choose Open File command and then **T16F676.HEX** file
- Set up the oscillator config as HS **(Oscillator = HS)**
- Set up **Bandgap = Default**
- Select **MCLR, BODEN** and **PWRT** and verify that all the others options of Fuses group are disabled
- Insert a **PIC16F676** device into 14 pins DIL14 format socket respecting the polarity
- Power on the board
- Verify that **JP7** and **JP8** jumpers of the C-202 programming board are closed
- Close **JP7**, **JP8** and **JP9** jumpers of the C-203 socket board
- Switch **S18** switching in **PROGRAM** position
- Choose **Program All** command: then the sign "Do you really want to program the devices" will be shown
- Choose **Yes** option
- Then the sign "Oscillator calibration value = xxxx.Do you want to use value from file instead?" will be shown
- Choose **No** option to keep the calibration value writted by constructor into last program memory location
- Then the programming phase will be started; this phase is composed by 3 sub-phase, everyone marked by a Progression Bar: Programming Code; Programming Data and Programming Config
- Then the verify of writed data's phase will be started; this phase is marked by a Progression Bar too
- If verify phase results ok, then the sign "Device Successfully Verified" will be shown and this means that microcontroller PIC is been correctly programmed
- Remove **JP7**, **JP8** and **JP9** jumpers from C-203 socket board
- Remove **JP7** and **JP8** jumpers from C-202 programming board
- Switch **S18** switching in **RUN** position
- Verify with an oscilloscope or with a tester that the microcontroller PIC's lines are sequentially turn at ON (logic level high).



**Note:**
**Pins number 13, 12 and 11 are respectly connected at microcontroller PIC lines RB0, RB1 and RB2. Pin number 4 is used as Master Clear (MCLR), while pins number 2 and 3 are used as OSC1 and OSC2.**

**T16F676.BAS**

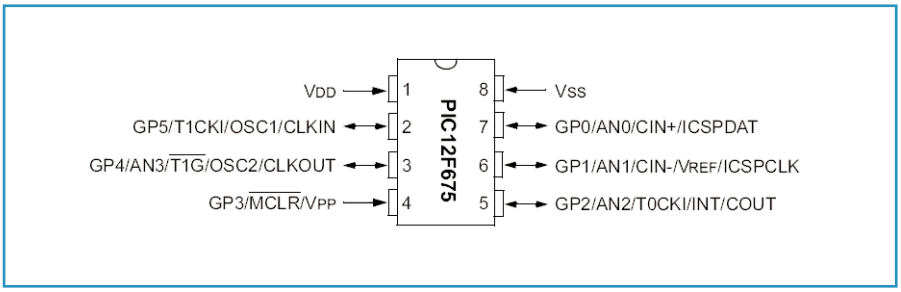
```
'*****************************************************************
'*  Name    : T16F676.BAS
'*  Notice  : Copyright (c) 2005 EVR electronics
'*          : All Rights Reserved
'*  Date    : 26/01/2005
'*****************************************************************

    DEFINE  OSC 20

    TRISA   =   %00000000   'PORTA OUT
    TRISC   =   %00000000   'PORTC OUT

    TEMP    VAR BYTE
    CONTA   VAR BYTE

    PORTA.0 =   0
    PORTA.1 =   0
    PORTA.2 =   0

    PORTC.0 =   0
    PORTC.1 =   0
    PORTC.2 =   0
    PORTC.3 =   0
    PORTC.4 =   0
    PORTC.5 =   0

LOOP:
    TEMP = 1
    for CONTA = 0 TO 2
        PORTA =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT CONTa
    PORTA.2 = 0
    pause 2000
    TEMP = 1
    for CONTA = 0 TO 5
        PORTC =   TEMP
        PAUSE   2000
        TEMP = TEMP << 1
    NEXT conta
    PORTC.5 = 0
    pause 2000
    Goto LOOP
```

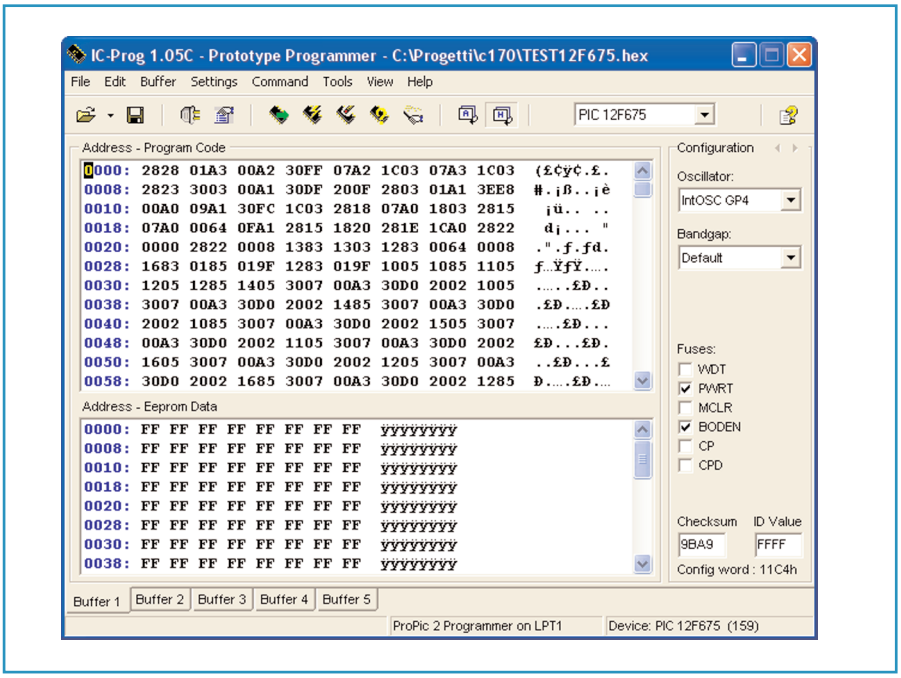
## MCU PIC 8 pins format (PIC12F675)

In this section the operations necessary to write or read a microcontroller PIC 8 pins format are shown.
The operations remains the same for every MCU PIC in 8 pins format, but for semplicity we will use **PIC12F675-I/P**, one of more diffuse microcontroller PIC in 8 pins format.
There is a testing program, available on CD whether source code or executable format, named **T12F675** writed using PicBasic language.

Let's see now the phases necessary to writing the demo file T12F675.HEX into the microcontroller's memory.

- Launch IC-Prog software and choose **PIC12F675** device
- Choose, from Menù Bar, Menù File
- Choose Open File command and then **T12F675.HEX** file
- Set up the internal oscillator **(Oscillator = INTOSC)**
- Set up **Bandgap = Default**
- Select **BODEN** and **PWRT** and verify that all the others options of Fuses group are disabled
- Insert a **PIC12F675** device into 8 pins DIL8 format socket respecting the polarity
- Power on the board
- Verify that **JP7** and **JP8** jumpers of the C-202 programming board are closed
- Close **JP7**, **JP8** and **JP9** jumpers of the C-203 socket board
- Switch **S18** switching in **PROGRAM** position
- Choose **Program All** command: then the sign "Do you really want to program the devices" will be shown
- Choose **Yes** option
- Then the sign "Oscillator calibration value = xxxx.Do you want to use value from file instead?" will be shown
- Choose **No** option to keep the calibration value writted by constructor into last program memory location
- Then the programming phase will be started; this phase is composed by 3 sub-phase, everyone marked by a Progression Bar: Programming Code; Programming Data and Programming Config
- Then the verify of writed data's phase will be started; this phase is marked by a Progression Bar too
- If verify phase results ok, then the sign "Device Successfully Verified" will be shown and this means that microcontroller PIC is been correctly programmed
- Remove **JP7**, **JP8** and **JP9** jumpers from C-203 socket board
- Remove **JP7** and **JP8** jumpers from C-202 programming board
- Switch **S18** switching in **RUN** position
- Verify with an oscilloscope or with a tester that the microcontroller PIC's lines are sequentially turn at ON (logic level high).



**Note:**
**GP lines are connectted at Microcontroller PIC PortB lines in the following mode:**
**GP0 = RB0; GP1 = RB1; GP2 = RB2; GP3 = RB3; GP4 = RB4; GP5 = RB5.**

Note:
GP3 line is exluse from our test because this line could work only as input.

```
'*****************************************************************
'*  Name    : T12F675.BAS
'*  Notice  : Copyright (c) 2005 EVR electronics
'*          : All Rights Reserved
'*  Date    : 26/01/2005
'*****************************************************************

    TRISIO  =  0      'All output

    ANSEL   =  0
    ADCON0  =  0

    GPIO.0  =  0
    GPIO.1  =  0
    GPIO.2  =  0
    GPIO.4  =  0
    GPIO.5  =  0

LOOP:

    GPIO.0  =  1
    PAUSE   2000
    GPIO.0  =  0
    PAUSE   2000
    GPIO.1  =  1
    PAUSE   2000
    GPIO.1  =  0
    PAUSE   2000
    GPIO.2  =  1
    PAUSE   2000
    GPIO.2  =  0
    PAUSE   2000
    GPIO.4  =  1
    PAUSE   2000
    GPIO.4  =  0
    PAUSE   2000
    GPIO.5  =  1
    PAUSE   2000
    GPIO.5  =  0
    PAUSE   2000

    GOTO    LOOP
```

**T12F675.BAS**

## Technique features

Power voltage: stabilizzated and direct voltage beetwen 12 Vdc and 17 Vdc
- 12Vdc (only normal operation)
- 17Vdc (normal operation and MCU programming)

Power current: minimum 300 mA
Operation temperature range: beetwen 0°C and 40°C
Maximum umidity: <RH80%
C-202 programming board dimensions: 65 x 112 mm
C-203 socket board dimensions: 89 x 112 mm
C-202 programming board weight: 70 gr.
C-203 socket board weight: 60 gr

## Warranty

This product is guaranteed without fault concerning components and/or construction process as specificated in legal terms since 1 year from sold date. The warranty is guaranteed only if the user have one original copy of purchase's evidence as for example invoice or fiscal ticket.
Constructor responsibility is limitated at fault repair or, if necessary, at faulty components replacement or repair. Costs and risks concerning product transport, removal or reposition, and every other costs directly or in-directly concerned the component repair, could not be attributed at constructor.
Constructor is not responsable of every damage caused by a product bad operation.

## Security instructions

- This product is been planned and realizzed to be used in an electronic laboratory and in an environment provided with protections against electrostatic discharge.
- All cables used to connect the product with other devices or personal computers must have special system to limit the electromagnetic emission
- Do not disassembly or remove product parts or components
- Handly the product with care
- Do not exceed the maximum and minimum values indicates in technique features
- Do not wash the product
- Verify that product is enough breezy
- Product not destined at childreens

## Technique support

For technique support concerning product and for product's repair, please contact us at our email:
**support@evr-electronics.com**

PIC is a Registered trademark of Microchip company.
All other trademarks are the property of their respective owners.

Constructor got the possibility to change product's feature or to cease the product's production without notice and constructor must no include or provide new functions or new instructions in already sold products. Constructor could not be responsable for damage, direct or indirect, that could be regarding the product's use. The products could not be used into vitality systems or systems that could create dangerous situations.

Modello: C-170
File: C-170 User Manual Rev1.qxp
Data: 20 Gennaio 2005