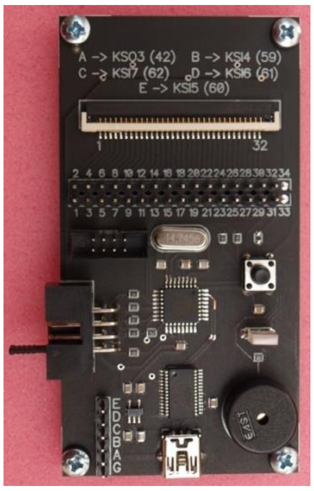
Brief characteristics of possibilities

1. **Reading and writing of multi-controllers `KB9010, KB9012, KB9016, KB9022.`**



By joint soldering of 6 wires (the lace is in the kit)
With the help of the flexible trail without any soldering
(there are trails with 24/28/30 wires in the kit)

2. **Reading and writing of `EEPROM LCD panels – EDID`**

(You can buy a made lace or you can make it yourself)

3. **The check-up of notebooks' keyboards in the program shell**

(the step of output of the keyboard trail is `0.5mm 0.8mm 1mm`)

4. **The autonomous mode of notebooks' keyboards check-up**

(the sound alarm (lights) of pressed buttons and mistakes)

The instructions version is of the 18th of December, 2014

The current version of the program shell 4.1 (both for the version 1 and version 2)

The controller's firmware of programmer 4.1 (the firmware/broaching is identical for both system board 1 and system board 2.

→ The preparation for work

For the working with the device it is necessary to install the package of drivers made by the `FTDI company.`

Start the program *`CDM v2.08.30 WHQL Certified.exe`*.

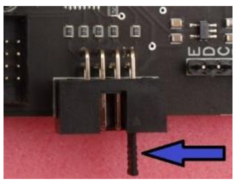
After that connect the device to the computer with the help of cable `MINI_USB. After a while the drivers will be installed and you need to make sure that the driver for the virtual COM` input-output port has been installed from the package of the `FTDI company.`

`Start the program shell and begin working.`

`The program works 32-64 bits in all Windows.`

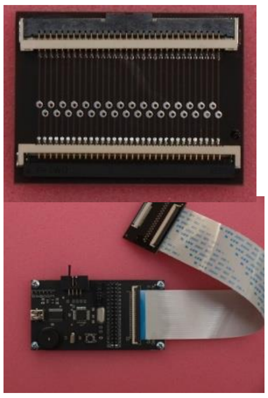
→ **The** autonomous mode of check-up of note-books' keyboards.

For the work of the programmer in the autonomous mode it is necessary to istall the jumper on 1-2 contacts of the output and connect it to a computer, a phone, an I-Pad, etc. with the help of the `MINI_USB with the USB connector at the output.`



`Doing that we just power the device.`

(**We need the jumper ONLY for turning on the autonomous mode! In all the other occassions you need to remove it**).



Then you need to connect the programmer with the adaptor card by means of trail with 32 outputs. We are connecting the keyboard which we are checking now to the adaptor card. You can change the keyboard without switching off the programmer's supply. After you have connected the keyboard I would recommend you to press the resetting button on the programmer. The majority of keyboards are the structure which consists of rows and columns, and if the circuit between them happens it is considered as the pressing of this or that button. The jumper for one or even two contacts is already present on some keyboards. This fact in the autonomous mode is also taken into consideration. Meanwhile they don't take into consideration more than one separate jumper, though personally I have not seen such keyboards. After pressing the resetting button the lighted green light-emitting diode will mean the presence of such a jumper.
Pressing the button on the working keyboard leads to the circuit of just two lines: row and column. Putting it in simpler words pressing the button is the result of the circuit of the two outputs on the trail. In this case we will hear *tilim* sound from the buzzer. If the circuit of more than two outputs on the trail will be discovered then you will hear the sound of mistake *tadaam*. In lots of spilt on keyboards there are already short circuits of tracks which is equal to stopped-up (stiff) buttons. In this case you will constantly hear the sound of a mistake *tadaaam*. For receiving more detailed information about such "stopped-up" buttons you will need to take off the jumper, press the button and start the utility of the keyboards' chek-up which is in the shell. As it is shown in practice the usage of the autonomous mode allows to make a conclusion about the efficiency of the keyboard in most occassions. It will be either each separate button working or it will be a few buttons stopped-up previously or none of the buttons are working.

→ Reading and writing of **EEPROM LCD panels – EDID**.

For broaching EEPROM matrixes (EDID) you will need a string with two outputs on their edges 30 and 40 pins.
30 pins matrix is mostly a matrix with a lamp highlighting.
40 pins matrix is a LED matrix. You can purchase the string together with the programmer or you can make it yourself. If u make it yourself u may use LVDS cables. The responding part (output) of the programmer is called IDC-08F.

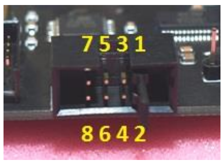
The soldering of string for 30 pins/40 pins/ and the programmer:

PROG 6 or 8 output -> 10 output of 40 pin on LCD -> 1 or 10 output of 30 pin on LCD (GND)

PROG 7 output -> 4 output of 40 pin on LCD -> 4 output of 30 pin on LCD (VCC)

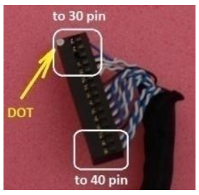
PROG 3 output -> 7 output of 40 pin on LCD -> 7 output of 30 pin on LCD (SDA)

PROG 5 output -> 6 output 40 pin on LCD -> 6 output of 30 pin on LCD (SCL)

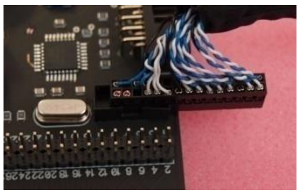


There is a triangle on the output connector of the programmer which indicates the first lead, there is number two below it and so on and so on ..it goes to the left side increasing. The upper line goes from left to right 7,5,3,1. Under them accordingly – 8,6,4,2.

At the beginning for the testing the BR24C02 was taken as the slowest and the common distributed one. The frequency of its clock is maximum 100kHz, so with this frequency we also work on EEPROM (for the comfortable recalculation the real frequency is about 80kHz). I used to choose it on the programmers all the time and there were not any problems with that. That is why it is assumed as the basis. All the matrixes which I had were recognised in 100% cases, only those which had WP = 1 (7 output EEPROM) could not be read. You would not be able to write only in case if on EEPROM the 7th output will be pulled up towards the supply. It is meant by default that the capacity of EEPROM is 256 bits, I have never seen another capacity for EDID. In the process of reading and writing the control of answering from EEPROM – ACK takes place. If you are not able to connect the matrix, or you will connect it in a wrong way or the EEPROM microcircuit will not work at all, the program will notify you about some problem in all these occasions.



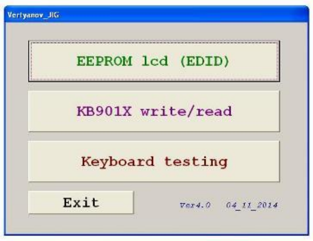
The connecting of the ready-made lace (string) is happening the little bit another way. From one side of the EDID string we have two outputs for the matrix and from another one there is a long block with fifteen pins by two lines. That side with the white dot on its block is pulled apart into 30 pins socket of matrix, the opposite edge – into 40 pins. They are not drawn into parallel between each other .And you should turn them over if you alternate the work of 30 pins matrixes with 40 pins matrixes.

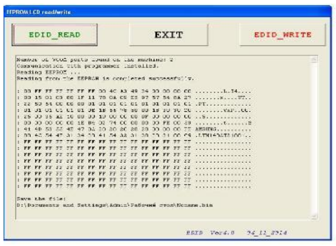


As shown in this figure, for example, the connecting for the broach of the matrix is done through 30 pins connector. If to turn around into 180 degrees – through 40 pins one.

After everything has been connected you start the program shell and choose work with EDID.

In a new window you choose read or write EEPROM. Automatically it will be suggested to name the file for writing or the file for reading. If during the reading or writing it will not be any mistakes then you will hear the *tilim* sound. If there are some problems you will hear *tadaaam*.





```
Number of COM ports found on the machine: 2
Communication with programmer installed.
Reading EEPROM ...
Reading from the EEPROM is completed successfully.

: 00 FF FF FF FF FF FF 00 4C A3 49 34 00 00 00 00 ........L.I4....
: 00 15 01 03 80 1F 11 78 0A E5 85 97 57 54 8A 27 .......x....WT..
: 22 50 54 00 00 00 01 01 01 01 01 01 01 01 01 01 .PT.............
: 01 01 01 01 01 01 9E 1B 56 [illegible] 50 00 15 30 30 20 ........V.P..00.
: 25 00 35 AE 10 00 00 19 00 00 00 0F 00 00 00 00 ..5.............
: 00 00 00 00 00 1E 86 02 74 00 00 00 00 FE 00 53 ........t......S
: 41 4D 53 55 4E 47 0A 20 30 8C 80 80 00 00 00 FE AMSUNG..........
: 00 4C 54 4E 31 34 30 41 54 32 31 30 30 11 00 C9 .LTN140AT2100...
: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ................
: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ................
: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ................
: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ................
: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ................
: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ................
: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ................
: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ................

Save the file:
D:\Documents and Settings\Admin\Рабочий стол\Noname.bin
```

→ **Reading and record of multi-controllers KB9010, KB9012, KB9016, KB9022.**

There are two ways of these multi-controllers broaching with the help of the programmer :

- The way of soldering the wires towards the motherboard (system board)
- The way of connecting to the system board with the help of flexible trails

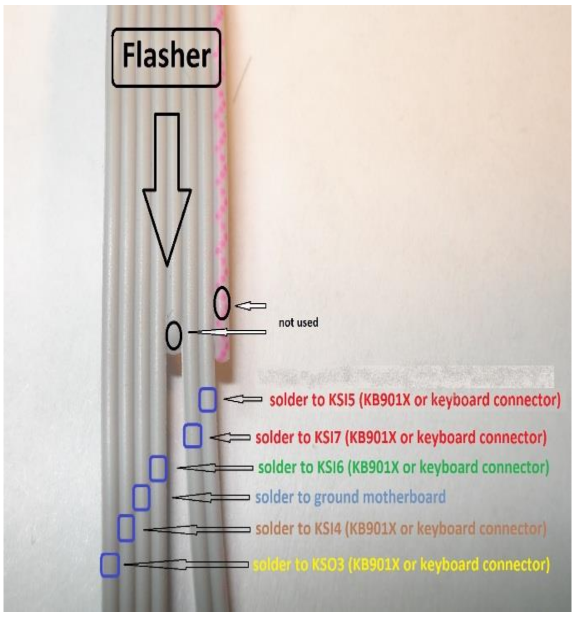
The set consists of the string for soldering and three trails for 24/28/30 pins with 1mm step and the length of the trail is 20 cm.

For the wires soldering we need to define on which leads of the keyboard socket KSO3/KSI4/KSI5/KSI6/KSI7 signals come to. In this case the scheme of the motherboard can help or you can figure it out yourself if you know the multi-controller decoding. It is more convenient to solder the trail wires towards just the keyboard connector because the pitch is bigger between the leads (wires) than between the multi-controller's legs.

Trail's wire № from the programmer / lead's № is KB9012

/ the name of the signal
1 wire is not used Reset for ATMEGA88
2 wire **60 output** EDI_CLK / KSI5
3 wire **62 output** EDI_DO / KSI7
4 wire is not used output 3v3 / 100mA
5 wire **61 output** EDI_DIN / KSI 6
6 wire **GND**
7 wire **59 output** EDI_CS / KSI4
8 wire **42 output** KSO3

You can use this scheme →→→


Flasher
not used
solder to KSI5 (KB901X or keyboard connector)
solder to KSI7 (KB901X or keyboard connector)
solder to KSI6 (KB901X or keyboard connector)
solder to ground motherboard
solder to KSI4 (KB901X or keyboard connector)
solder to KSO3 (KB901X or keyboard connector)

After the spots of soldering to the keyboard connector are defined it is necessary to solder the trail.

So after the drivers are there and the *tail* has been soldered we are trying it in practice.

I normally do the following way:

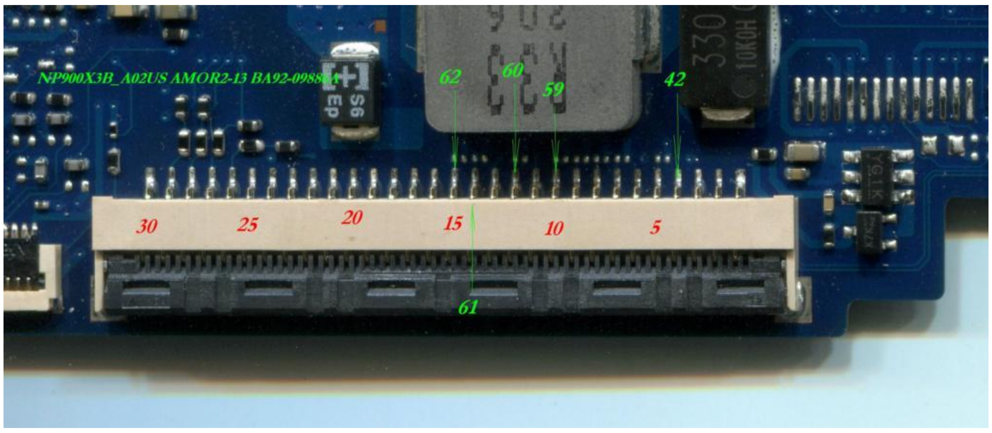
1. The motherboard is disconnected (powered off).
2. I connect the soldered *tail* to the broacher.
3. Take off the jumper from the programmer if it is installed!
4. Connect the MINIUSB cable to the PC where the broaching program is installed.
5. Connect the powering from the power supply unit into the motherboard! Not from the accumulator! Take off the accumulator by all means! You just connect the power plug but do not press the start button on your note-book!
6. Start broaching the program itself on the PC. By this time the virtual COM port in the system has been installed.

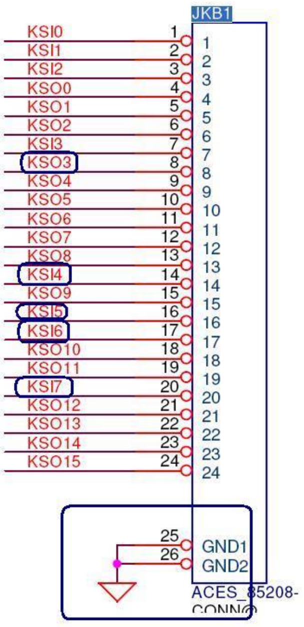
7. Choose the work with KB90XX and then press read or write.

I will not stop at the simple things in details but will go through them really quickly.

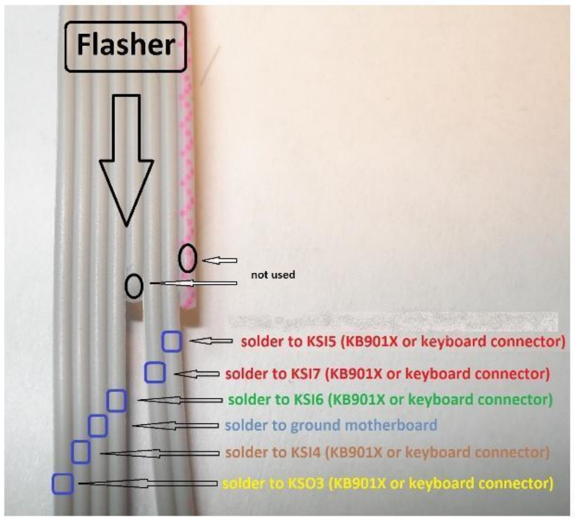
If the connection with the multi-controller has been established after you press the button <READ>, it will be suggested to you to point out the way for saving the FLASH of memory KB90XX contents. The process of reading will start.
The <WRITE> button works the similar way. It asks you to point out the file for memory fill. The connection check-up with the multi-controller takes place in similar way as if during the writing.
The deleting of FLASH memory KB90XX before the writing happens automatically. After the writing process the data verification starts. Reading lasts around 9 seconds, writing and data verification do around 28 seconds. You will see everything in the shell. After we have read or written the information we should cut power off notebook's system board from KB90XX. Then you should pull the "tail" out of the programmer (remove the programming mode KB90XX)and power the note-book again. Now you can power it from both: accumulator and a power supply unit. Start your note-book pressing the button and check what is done.
I must notify that KB90XX will turn into the programming mode only if to put its 42th leg on the ground connector. In my version it will happen only if to connect the "tail" to the programmer. So after the broaching has been finished without wires soldering you can already check the work of the motherboard by plugging out the "tail" and twisting the supply of the note-book's motherboard.

There are a couple of pictures with some different ways of soldering of the keyboards' connectors.



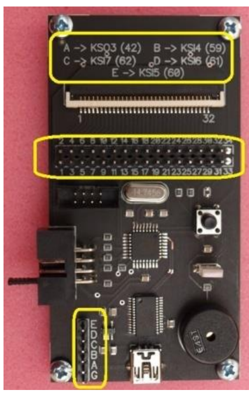


<--- LA-8941P keyboard


Flasher
not used
solder to KSI5 (KB901X or keyboard connector)
solder to KSI7 (KB901X or keyboard connector)
solder to KSI6 (KB901X or keyboard connector)
solder to ground motherboard
solder to KSI4 (KB901X or keyboard connector)
solder to KSO3 (KB901X or keyboard connector)

How to connect the motherboard with the help of flexible trails.
This method is a little more difficult , but after you do it once you will not have any difficulties further. After we have defined on which outputs there are `KSO3/KSI4/KSI5/KSI6/KSI7` signals on the keyboard's connector we have to *enter* the right combination on the programmer.
For example it will look this way on some following motherboards:

```
LA-7983P KSO3=18 KSI4=5 KSI7=2 KSI6=3 KSI5=6

LA-9912P KSO3=18 KSI4=5 KSI7=2 KSI6=3 KSI5=6

LA-9535P KSO3=4 KSI4=23 KSI7=26 KSI6=25 KSI5=24

LA-7912P KSO3=4 KSI4=23 KSI7=26 KSI6=25 KSI5=24

LA-8331P KSO3=4 KSI4=23 KSI7=26 KSI6=25 KSI5=24

NP900X3B AMOR2-13 KSO3=4 KSI4=10 KSI7=15 KSI6=14 KSI5=12

DA0U82MB6D0 KSO3=15 KSI4=28 KSI7=31 KSI6=30 KSI5=27

NP530U3B LOTUS13 KSO3=4 KSI4=10 KSI7=15 KSI6=14 KSI5=12

LA-B511P KSO3=4 KSI4=23 KSI7=26 KSI6=25 KSI5=24
```



And so on and so on. I guess it is all clear at this stage now and no explanations needed. The numbers on the keyboard's connector mean the number of contact.

Now according to the received data let us enter for

LA-8331P KSO3=4 KSI4=23 KSI7=26 KSI6=25 KSI5=24.

We are taking the multi-colored cable with USB connector at its edge.

Connect the USB socket to the motherboard. By this way we connect the programmer's and board's grounds. We plug the cable into the programmer in that way to let the wire which connects the grounds to get into the contact with the symbol "G" (ground).

Wire "E" (KSI5) connect with pin 24,

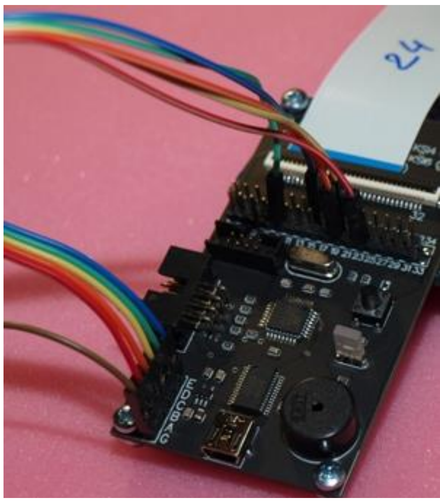
Wire "D" (KSI6) with pin 25,

Wire "C" (KSI7) with pin 26,

Wire "B" (KSI4) with pin 23,

Wire "A" (KSO3) with pin 4.

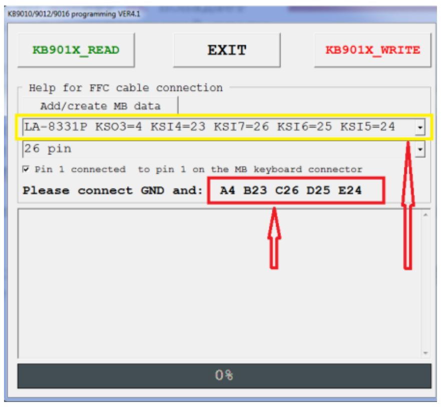
That is all. The field is done and you will see something similar to this →→→



Now you have to choose the trail of the necessary "width" to let all important signals for the multi-controller broaching reach the keyboard's connector. For LA-8331P where the biggest contact number is 26 you need the trail not less than 26.

The next condition is – one contact of the programmer on the trail has to get into one contact of the keyboard's connector of the motherboard. To fulfill this condition you can use a straight or an inverse trail. It will be a little bit more difficult situation when ideally you need an inverse trail but you just have the straight one available and vice versa… There is an exit from this situation. There is a recalculation module KSO3/KSI4/KSI5/KSI6/KSI7 in the shell in the window of work with KB90XX just in case if one programmer's contact on the trail doesn't get into one contact with the keyboard's connector of the motherboard.

For example,using the trail with 24 contacts, the last contact of the trail gets into the 1st output of the programmer's connector and along the trail comes to the 24th output of the keyboard's connector on MB. You should always insert the trail into the 1st contact both on the programmer and on the motherboard! With the thirty contacts' trail one programmer gets into 30 MB, etc. For this purpose we use the recalculation module.


KB9010/9012/9016 programming VER4.1
KB901X_READ
EXIT
KB901X_WRITE
Help for FFC cable connection
Add/create MB data

LA-8331P KSO3=4 KSI4=23 KSI7=26 KSI6=25 KSI5=24

26 pin

☑ Pin 1 connected to pin 1 on the MB keyboard connector

Please connect GND and: A4 B23 C26 D25 E24

0%

The file which contains platforms prompts is called "MB connection KB901X.txt". It has to be kept in the disc root C:. You may fulfill it yourself editing if necessary in the text editing program. Exactly from this file the lines will be reflected in the coming up upper list in the program's window. Underneath – how many contacts we use the trail for. We check in the box after the 1st programmer on the trail gets into the 1st MB.

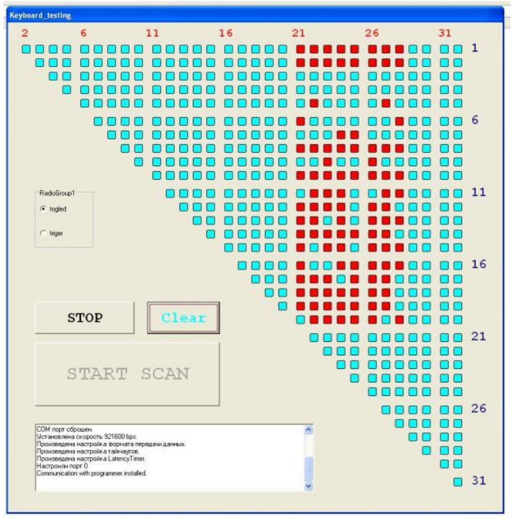
I would like to stress that this recalculation module *does not affect* the process of reading or writing of KB90XX *in any way.* It is created exceptionally for the comfort of quick keying in the field for multi-controlers' broaching with the help of the flexible trails.

After we have connected everything in case of wires soldering you should take the further actions which are described above.

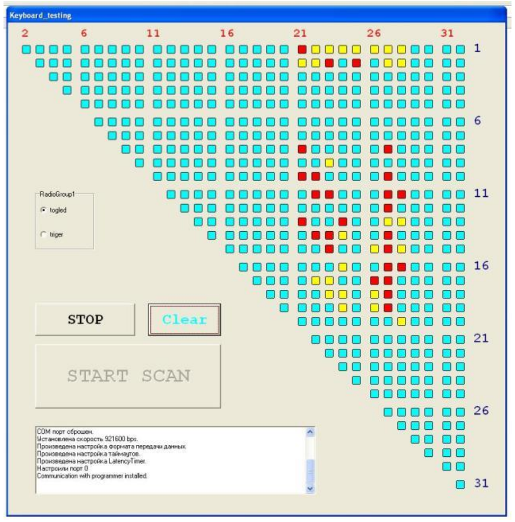
The "MB connection KB901X.txt" file is gotten renewed from time to time and is kept HERE. You will not have any difficulties when renewing or changing it yourself. All the information n material on it is kept there too.

→ Checking up note-books' keyboards in the program's shell.

The scanning happens when we put log "0" on 1 of the wire of the programmer output (the line on the right) and go through the output from 2 to 32. Then we put log on 2 (the line on the right) and catch already from 3 to 32. And so on and so on. That is how we explain why the field shape is "triangle" in the shell. There is no necessity to show the circuit for example 2 from 32 and 32 from 2. One button (the crossing) is one small square. While some single-phase circuit changes the square color into red, the repeated circuit changes the color into yellow. And so forth along the circle: red, yellow, red, yellow and so on and on.



```
COM порт сброшен.
Установлена скорость 921600 bps.
Произведена настройка формата передачи данных.
Произведена настройка таймаутов.
Произведена настройка LatencyTimer.
Настроили порт 0
Communication with programmer installed.
```



```
COM порт сброшен.
Установлена скорость 921600 bps.
Произведена настройка формата передачи данных.
Произведена настройка таймаутов.
Произведена настройка LatencyTimer.
Настроили порт 0
Communication with programmer installed.
```