

# HC908GP32-Starter Kit

## User's Manual

Brno

October 1 2001





# Content

Introduction-----	5
Who is Beta Control-----	5
GP32EVB features -----	6
Software and Motorola CDMCU/D-----	7
Quick start instructions -----	8
MC68HC908 security feature-----	9
MON08 mode troubleshooting -----	10
GP32EVB hardware installation -----	12
Schematic description -----	13
GP32EVB schematic -----	14
GP32EVB part list-----	15
Parts location diagram and jumpers setting -----	16
Jumper setting for DBG08 kit-----	18



# Introduction

The Motorola 68HC08 is a very strong and useful product family. The excellent FLASH memory, various peripheral subsystems, large variety of packages, programming and debugging on-chip support brings designer's dreams reality.

The Beta Control's Low Cost Starter Kit represents very good entry gate to design in. Various Target Boards could be used as a stand alone board or together with the universal debugger module. Some instant on board demo peripheral components and associated tutorial programs enable users to start immediately.

This is ideal tool for any distributors seminars, for the teachers at schools, users clubs and also for anybody who is interested in application with modern microcontrollers.

Jiri Gutman  
Motorola consultant

## Who is Beta Control

Beta Control Ltd. is an innovation based company. Its basic strategy consists in utilizing its own know how of modern technologies in the field of electronic control and information systems. More then sixty percent of employees are focused on design and development. As the most products of Beta Controls are based on Motorola technology and products, the Beta Control became a cooperation partner in promoting Motorola 68HC08 family.

**Contact:**  
Beta Control Ltd.  
Cerneho 58/60  
CZ-635 00 Brno  
Czech Republic

Phone: (+420 5) 46 22 34 91  
Fax: (+420 5) 46 22 34 70  
[www.betacontrol.cz](http://www.betacontrol.cz)



ISO 9001  
Beta Control Ltd.  
CZ-141/2000



## GP32EVB features

- MCU 68HC908GP32CP
  - 32 Kbytes of on-chip FLASH memory with in-circuit programming capabilities
  - 512 bytes of on-chip RAM
  - SCI, SPI
  - 8-channel, 8-bit ADC
  - two 16-bit TIM channels
  - up to 33 general-purpose I/O pins with selectable pull-ups
  - 40-pin plastic dual-in-line package (PDIP)
- in-circuit debugging
- clock generator module with 32 kHz crystal PLL
- low-cost development tools
- compatible with Motorola MON08 interface and ICS software tools
- power-on reset switch
- simple applications examples
  - SCI
  - 2 LEDs
  - 2 push-buttons
  - temperature sensor LM35
- wire wrap area to easy build your application
- power supply range 8–12 V DC or direct 3 V or 5 V ( $\pm 10\%$ )
- dimensions 104x119 mm

---

## Software and Motorola CDMCU/D

The HC908 Starter Kit is a hardware product only but you can't start your development work without software of course.

You can use various software products with HC908 Starter Kit.

There is a free Assembler in MCUez Package on the Motorola Microcontrollers double CD CDMCU/D enclosed in this package. You can also find some other software on the web but the best and highly recommended is a complex software package from P&E company. This Integrated Development Environment has been used for years for the Motorola 68HC05 Family and now it represents the best tool for 68HC08 Family also.

You can find this software at an internet address <http://www.pemicro.com> altogether with manuals and it is recommended to go there often to check the last versions of software and documentation. But you can find older versions also on the Motorola Microcontrollers double CD in the link `..\DEVTOOLS\0508sw.htm`

Motorola Microcontrollers double CD CDMCU/D is an excellent source of information. On this CD there is a lot of Data Books, Reference Manuals and Application Notes (including source code). The HC08 Family is fully upward object code compatible with the 68HC05 Family and you can use all application notes for HC05. You can use also the very good teaching book SMLMCUR2.PDF from the TUTORIAL folder.

If there will be some missing member of 68HC08 Family on CD in future, you can find information at the Motorola web site [www.mot-sps.com](http://www.mot-sps.com) under link Products.

## Quick start instructions

For users experienced in installing Motorola or other development tools, the following steps provide a quick-start installation procedure for the GP32EVB hardware and software. If problems occur with the quick start procedures, refer to “Technical Reference and Troubleshooting” for troubleshooting instructions.

- Assemble and connect the hardware pod (see “Hardware Installation GP32EVB” and Electrostatic Discharge Caution). Make certain that power is off before performing this step.
- Install the ICS08GP software package. To start the software installation, run the program on CD ROM-Path: DEVTOOLS\SOFTWARE\DEVSW\ics08gpz\_version\_XXX.exe. During installation, follow the instructions in the installation wizard.
- Start the WinIDE editor and open a project file. Start the editor either from the Windows Start menu or by double clicking its icon. From the WinIDE Environment menu, choose the Open Project option, and choose a project file from the Specify project file to open dialog. Note that the environment is pre-configured. If no project file exists, choose the New option from the File menu to create a new project file. The desktop and environment settings made in the Environment Settings dialog are stored in the WIMDE.INI file and read each time the WinIDE editor is started. The project-specific desktop and environment settings can also be saved in a project file (\*.PPF), which is read when the project is opened, allowing it to be saved and used as a general environment as well as custom environments for individual projects. To create the project file:
  - Specify the project-specific desktop and environment settings in the WinIDE editor.
  - Choose the Save Project As option from the WinIDE Environment menu to name and save the project to a directory folder.
- Assemble the code. Press the ASSEMBLE COMPILE FILE button (Hotkey F4) on the WinIDE toolbar to assemble the source code in the active WinIDE window.
- Run the ICS08 simulator. With a project or source file open in the WinIDE main window, click the In-Circuit Simulator button (Hotkey F6) on the WinIDE toolbar to start the ICS08 debugger and debug the contents of the active source window. If communications are not established with the pod, it may be necessary to select the proper port (COM1 or COM2) and baud rate 9600Bd. Select Class III – Custom board with MON08 serial port circuitry built in. When communications are established, the DBG08 LED will not light. You can also use Simulator in stand-alone mode (Hotkey F9) without connected GP32EVB.
- Run the PROG08SW programmer. Press the PROGRAM button (Hotkey F7) on the WinIDE toolbar to start the programmer. Select Class III – Custom board with MON08 serial port circuitry built in and baud rate 9600 Bd.

- Run the ICD08SW real-time debugger. Press the REAL-TIME DEBUGGER button (Hotkey F8) on the WinIDE toolbar to start the ICD08SW in-circuit debugger and emulator. Select Class III – Custom board with MON08 serial port circuitry built in and baud rate 9600 Bd.

*ESD CAUTION: Ordinary amounts of static electricity from clothing or the work environment, can damage or degrade electronic devices and equipment. For example, the electronic components installed on printed circuit boards are extremely sensitive to electrostatic discharge (ESD). Wear a grounding wrist strap whenever handling any printed circuit board. This strap provides a conductive path for safely discharging static electricity to ground.*

## MC68HC908 security feature

The MC68HC908GP32 MCUs contain a security feature based on information programmed into the part. Security bytes are specified in addresses \$FFF6 – \$FFFD. The PROG08SW software continually records any changes to these security bytes and stores them in the file SECURITY.INI.

The information in this file is also shared with the ICS08 in-circuit simulator and the ICD08SW in-circuit debugger software. This allows the user to reset the device and still have access to the monitor mode. The ICS08 software automatically attempts to access the part by trying the default (nothing written) and up to the last 10 sequences of bytes that have been written to the part.

If after trying each of the sequence of bytes stored in the SECURITY.INI file, the ICS08 software is unable to access the part, a security dialog box is displayed. This dialog allows the user to enter the security bytes manually or to read the information from the S19 file from which the MCU was last programmed.

# MON08 mode troubleshooting

These instances include in-circuit simulation/emulation and FLASH memory programming. If difficulties are experienced when quick-start the kit using the procedure outlined in Quick Start Instructions, follow these steps.

- Disconnect any target cables from the board. These troubleshooting steps assume that no target system connections are present.
- Make sure that the MCU is installed correctly.
- Make sure that the DBG08 board is installed correctly.
- Make sure the board is getting power:
  - Check the power at the output of the adapter. First disconnect the pod from the power supply, then measure the power at the wall adapter's output connector to confirm that it produces appropriate dc voltages. If there is no power at the connector, verify that the adapter is getting power from the AC power outlet.
  - Check the power at the GP32EVB. First remove the DBG08 board from the GP32EVB, then plug the adapter's output connector into the GP32EVB. The system power LED VL1 should light. Check for 5V DC at the connector X8 on GP32EVB's. The Configuration header JP1 must be closed if power is from X1!
  - Check the pod system power with the DBG08 board attached. Disconnect the GP32EVB from the power supply. Configure the DBG08 board to the MON08 mode and attach the DBG08 board to the GP32EVB. Reconnect the power supply to the GP32EVB. The system power LED VL1 should light. If the LED does not light, may be there is a problem with the DBG08 causing too much of a drain on the 5V DC supply.
  - Check the DBG08 board's VTST output when the GP32EVB board is installed. It should be approximately 8.5V DC at JP2 pin 3. If the voltage is not present when the system power LED is lit, there may be a problem with the DBG08's internal step-up power supply.
- Make sure that the host PC can communicate with the MCU. If communication is correct, DBG08 VL1 LED will not light.
  - The MCU's PTA0 pin is used for host communications. DDRA bit 0 should never be set to 1 as this interrupts monitor-mode communications. The target connector PTA0 pins (X3 and X5 pin 16) are never connected to the MCU's PTA0 pin, when DBG08 is installed.
  - Make sure that the serial cable is correctly attached to the pod and to the correct serial port on the host computer.
  - Make sure that the cable is a straight-through cable supporting all nine pins of the serial port connection.
  - Make sure that no hardware security key or other devices are attached to the serial port or cable.

- Make sure that the host PC supports the minimum speed requirements of the ICS08 software. Select Class III – Custom board with MON08 serial port circuitry built in and baud rate 9600 Bd.
  - Make sure to use the correct security code to access the MCU. If the security bytes have been programmed previously, the part will not unlock and enter monitor mode unless the correct security code is sent to the MCU.
  - Check for data at the pod end of the serial cable. TXD on pin 3 of this connector carries RS-232 data into the pod; RXD pin 2 carries RS-232 data out of the pod. DTR signal on pin 4 controls the MCU reset. Pin 5 is ground. While the ICS08 software is trying to establish communications, pins 3 and 4 should both toggle between +8 V DC and –8 V DC (or +12 V DC and –12 V DC). If these signals are not seen at the cable end, the problem is on the PC and cable side of the system. You can test it with ICS08 software because it detects hardware loop-back on startup PROG08SZ.
  - Make sure the serial data is getting to the MCU’s PTA0 pin.
- Make sure that the MCU has a good clock source. Use an oscilloscope to check the OSC input at the MCU. The clock frequency is a 9.83 MHz for a 9600-baud communications rate. If the clock signal is not present, check to see that the jumpers are installed (JP1 on DBG08 and JP3 on GP32EVB ).
  - Make sure that the MCU can enter and remain in monitor mode. For this to happen, the following conditions must occur:
    - IRQ must be at VTST (from 7.5 to 9 V DC) at the rising edge of RST. Using a dual-trace oscilloscope, trigger channel 1 on the rising edge of RST and read the IRQ pin with channel 2. Start the ICS08 software as described in Quick Start Instructions and verify that the signal is approximately 8.5 V DC when RST rises. If IRQ is not at 8.5 V DC, there may be a problem with the DBG08 board’s VTST circuit. Check JP2 position.
    - At the rising edge of RST, PTA0, PTA7, PTC0, PTC1 and PTC3 must be held at definite logic values to enter into MON08 mode. With DBG08 installed, the MCU’s PTA0 pin is not connected to the target pins, as it is used for host communication.
    - PTA7 must be held low for at least 24 bus cycles after RST goes high. The counter circuit comprised of DD6 keeps the PTA7 pin near 0V for 13 ms. PTA7 is reconnected to the target connector pins when this delay expires.
    - Either RST or IRQ must remain at 8.5V DC to hold the MCU in monitor mode. The DBG08 board has an interrupt lockout feature to keep IRQ at 8.5 V DC when the RST signal is asserted
  - Make sure that external circuitry does not interfere with the monitor-mode communications. This ensures that the target system will not interfere with the communications and setup of the MCU’s monitor mode by allowing the pod to disconnect some target system components during monitor mode entry.

# GP32EVB hardware installation

This chapter explains how to:

- Configure the GP32EVB development board
- Assemble the GP32EVB board and the DBG08 serial interface to form the hardware pod and connect the pod to a host PC
- Connect the pod to a target system

In interactive mode, the GP32EVB is connected to the serial port of a host PC. The actual inputs and outputs of a target system can be used during simulation of source code. In stand-alone mode, the GP32EVB is not connected to the PC. The ICS08GPW software can be used as a stand-alone simulator.

## Configuring the GP32EVB

The configuration headers provide for jumper-selectable hardware options. Tables on page 16 describe these settings.

### Assembling the target board with DBG08 for the MON08 mode

To assemble the target and prepare it for use with a host PC:

- Check the configuration headers on the target and DBG08 kits
- Install the DBG08 board onto the target MON08 connectors X2, X4. Make certain that power is off before performing this step.
- Connect the DBG08 to the host PC. Using the cable provided, connect it to a serial COM port on the host PC.
- Apply power to the target. Connect the correct power supply to the round connector on the GP32EVB. The green system power LED on the target should light.

*NOTE: The GP32EVB's emulation of the on-board MCU is limited. Port A bit 0 (PTA0) is used for host-to-MCU communication. The port bit is not available for connection to a target system. Setting DDRA bit 0 to 1 will stop communications with the simulation or debugger software and will require a system reset to regain communication with the MCU. Port bits PTA7, PTC0, PTC1 and PTC3 are temporarily disconnected from the target system during reset. Emulation of the MC68HC908GP's RST signal is also limited in that the signal is not a bi-directional, open-drain signal. It is emulated as an output when using the target connectors.*

## Schematic description

The basic component on the GP32EVB Kit board is a 68HC908GP32 microcontroller (DD1) from Motorola.

The power supply is connected via the connector X1 and operating voltage range is from 8 V to 12 V DC. This input is protected against reversing of polarity by diode VD1. You can also connect 3 V or 5 V DC directly via connector X8. You can take off the JP1 to prevent LED VL1 and NR1 from effecting current consumption e.g. when you test special application supplied from 3 V DC. In this case the kit is protected by the NT1 but you have to use additional external fuse. It protects against the overvoltage above 6.8 and reversing of polarity. Use the switch SA3 to generate POR if the kit is supplied via X1. VL1 signalizes the supply voltage for the kit.

To avoid problems whenever the voltage is below operating level the MCU contains low voltage inhibit module. This module monitors the voltage on the VDD pin and can force a reset when the voltage is below LVI trip falling voltage, VTRIP. The LVI50R bit in the CONFIG1 register in the MCU selects the value of VTRIP.

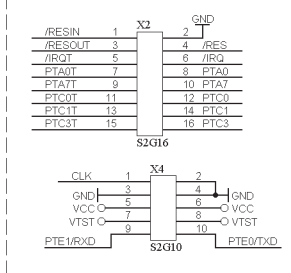
The MCU generates its clocks from a 32.768 kHz crystal connected to the OSC1 and OSC2 pins and CGMC module implemented in the MCU. To generate definite clock frequency for your application you must program the CGMC that is used here. The Kit has special jumper JP3 for clock selection. You must set the JP3 correctly for the normal operation (position labeled OSC1) or special monitor mode (EXT CLK).

All controller signals (except CGMXFC) are available via two double row header connectors X3 and X5. The connector X5 is not mounted as factory default to allow you to solder them up- or downward, depending on your application. Some signals are used by MCU to enter into the special monitor mode. You must short pins on connector X2 with jumper or install DBG08 development kit if your application needs them. The pins are: /RESOUT-/RES, IRQT/--/IRQ, PTA0T-PTA0, PTA7T-PTA7, PTC0T-PTC0, PTC2T-PTC2, PTC3T-PTC3. The kit contains some simple applications and you can connect them with the MCU pins. You can connect analog input PTB0 via JP2 with the temperature sensor LM35. It converts temperature into voltage with constant step 10 mV/°C. There are two LEDs and push buttons on the kit. The LEDs light when MCU sets value 0 on pins PTD4 and PTD5, when the buttons are pushed down there are 0 on the MCU pins PTA2 and PTA3. The connector X7 is prepared especially for the RS232. The connector X8 is for the potentiometer that you can connect with analog input PTB1. The I/O pins of the MCU have internal pull-up resistors, so there shouldn't be problems leaving these pins unterminated.

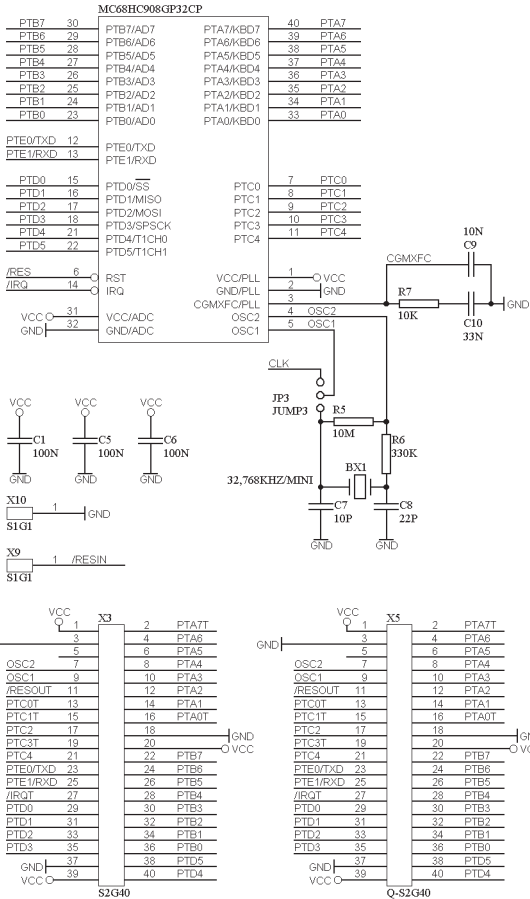
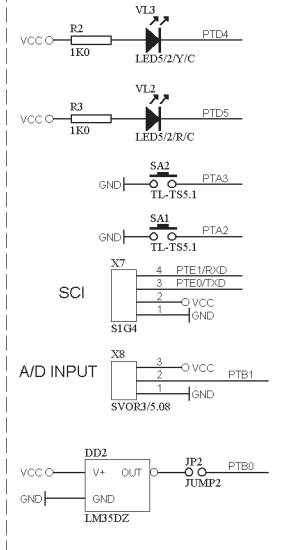
The connector X2 is a Motorola MON-08 compliant 16 pin header for the debug interface of the HC908. Some signals for special monitor mode (CLK, VTST...) are on connector X4. You may use them to connect a DBG08 kit pod while debugging a program or for programming the FLASH memory. To enter into monitor mode you must set JP3 into the position labeled EXT OSC.

# GP32EVB schematic diagram

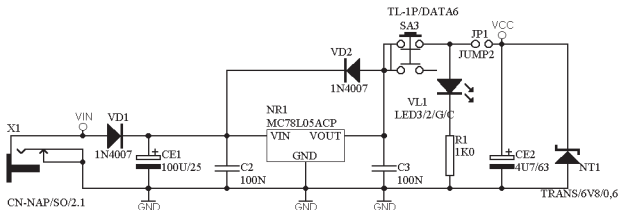
## DEBUG CONNECTORS



## APPLICATION EXAMPLES



NOT MOUNTED



# GP32EVB part list

Part No.	Qty	Value	Typ
<b>PCB</b>			
1	PWB1	1	DPS DK2411 BOARD
<b>IC</b>			
2	DD1	1	MC68HC908GP32CP
3	DD2	1	LM35DZ
4	NR1	1	MC78L05ACP
<b>Optical</b>			
5	VL1	1	LED3/2/G/C L-934 LGD
6	VL2	1	LED5/2/R/C L-53 LID
7	VL3	1	LED5/2/Y/C L-53 LYD
<b>Diode</b>			
8	VD1, VD2	2	1N4007
9	NT1	1	TRANS/6V8/0,6 P6KE6.8ARL
<b>Quartz</b>			
10	BX1	1	32,768 KHz MINI Q32,768KHz
<b>Resistors</b>			
11	R1	1	680R 491-0
12	R2, R3	2	330R 491-0
13	R5	1	10M 491-0
14	R6	1	330K 491-0
15	R7	1	10K 491-0
<b>Capacitors</b>			
16	C1 - C3, C5, C6	5	100N C320C104K5R5CA7303

Part No.	Qty	Value	Typ
<b>Capacitors</b>			
17	C7	1	10P EGPU 100V N150
18	C8	1	22P EGPU 100V N150
19	C9	1	10N KDPU Z5URM2,5
20	C10	1	33N KDPU Z5URM2,5
21	CE1	1	100U/25 THGS
22	CE2	1	4U7/63 THGS
<b>Connectors</b>			
23	X1	1	CN-NAP/SO/2.1 SCD-016
24	X2	0,2	S2G16 WWD 80 G
25	X3	0,5	S2G40 WWD 80 G
26	X4	0,125	S2G10 WWD 80 G
27	X5	0,5	Q-S2G40 WWD 80 G
28	X6	1	CN-ZAS/SO/2.1 SCP-2009B
29	X7	0,1	S1G4 WWS 40 G
30	X8	1	SVOR3/5.08 MV 253/5,08
31	X9, X10	0,05	S1G1 WWS 40 G
<b>Others</b>			
32	JP1, JP2	0,2	JUMP2 WWS 40 G
33	JP3	0,075	JUMP3 WWS 40 G
34	JS1 - JS10	10	JUMPER JUMPER SCHWARZ
35	PN1 - PN4	4	PNOZ-11/3,2 SJ-5201
36	SA 1, SA2	2	TL-TS5.1 TS 062
37	SA3	1	TL-1P/DATA6 DATA MOM

**Connectors X3 and X5**

PTA7T	PTA6T	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0T	GND	VCC	PTB7
2	4	6	8	10	12	14	16	18	20	22
1	3	5	7	9	11	13	15	17	19	21
VCC	GND	NC	OSC2	OSC1	/RESOUT	PTC0T	PTC1T	PTC2	PTC3T	PTC4

**Connector X8**

1	2	3
GND	PTB1	VCC

PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0	PTD5	PTD4
24	26	28	30	32	34	36	38	40
23	25	27	29	31	33	35	37	39
PTE0/TXD	PTE1/RXD	/IRQT	PTD0	PTD1	PTD2	PTD3	GND	VCC

**Connector X7**

1	2	3	4
GND	VCC	PTE0/TXD	PTE1/RXD

**Connector X2**

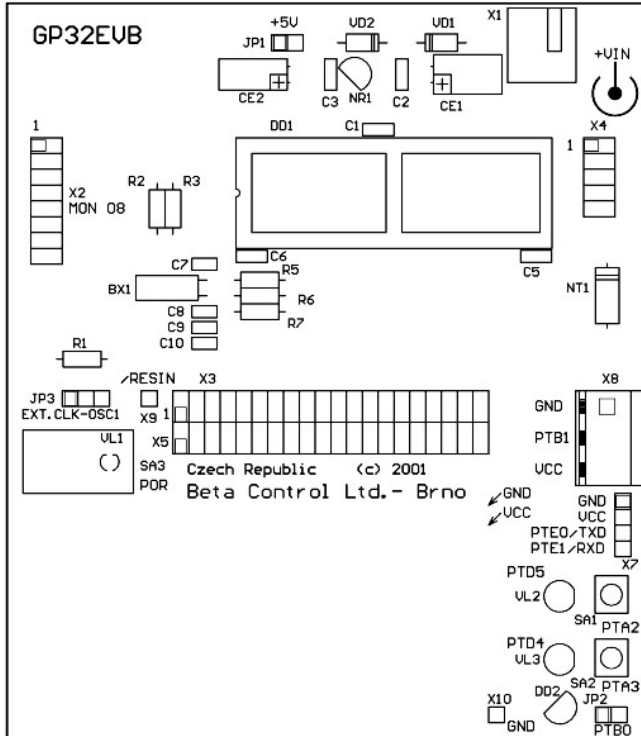
GND	/RES	/RQ	PTA0	PTA7	PTC0	PTC1	PTC3
2	4	6	8	10	12	14	16
1	3	5	7	9	11	13	15
/RESIN	/RESOUT	/IRQT	PTA0T	PTA7T	PTC0T	PTC1T	PTC3T

**Connector X4**

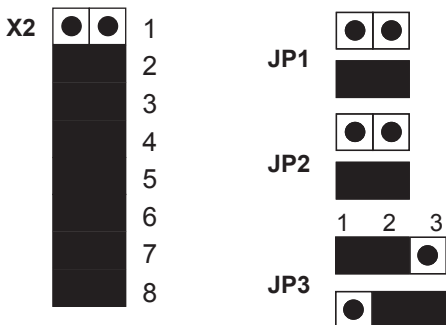
GND	GND	VCC	VTST	PTE0/TXD
2	4	6	8	10
1	3	5	7	9
CLK	GND	VCC	VTST	PTE1/RXD

# Parts location diagram and jumpers setting

Version. 1.1



Present jumpers if use GP32EVB alone

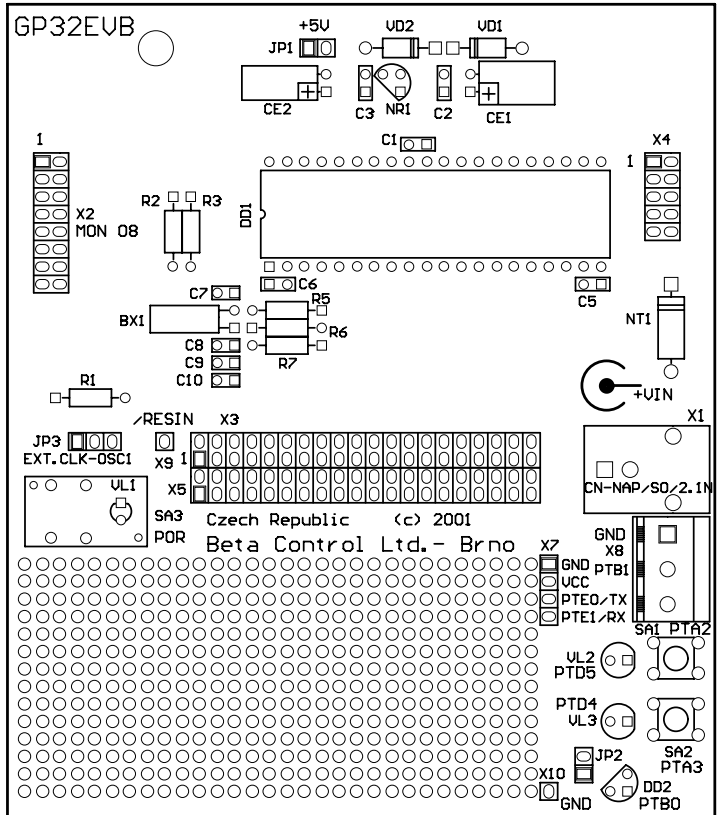


Power supply from X8 (5V= or 3V=)  
 Power supply from X1 (8V= to 12V=)

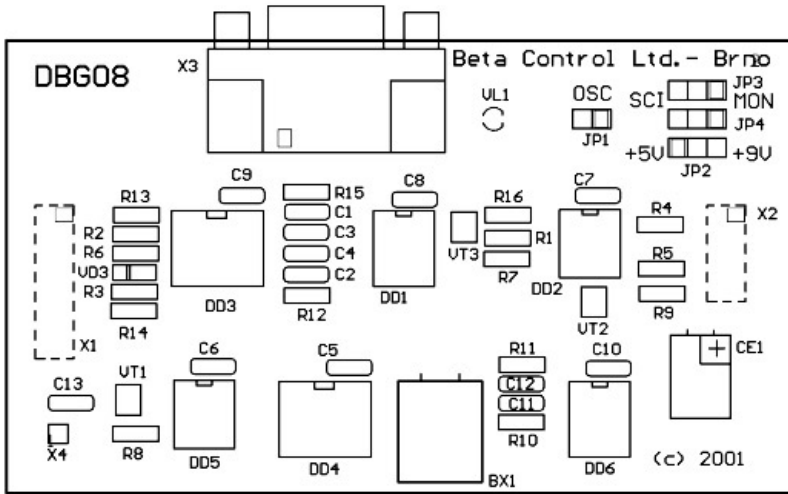
Port PTB0 is open – general purpose  
 Port PTB0 is connected to thermal sensor

External CLK from X4  
 Internal quartz oscilator

Version. 1.2



# Jumper setting for DBG08 kit



**JP1 - Configuration Header  
– Oscillator Setting**

State	Signal Name	Description
ON	OSCOUT	Allows using 9,8304MHz DBG08 oscillator output
OFF	OSCOUT	Disconnect 9,8304MHz DBG08 oscillator output

**JP2 - Configuration Header  
– VTST select**

Pins	Signal Name	Description
1 – 2	+5V	After reset the target MCU runs the user's application program
2 – 3	+9V	After reset the target MCU starts into MON08 mode

**JP3 - Configuration Header  
– RX select (RS232)**

Pins	Signal Name	Description
1 – 2	COMIN	MON08 mode (position labeled MON)
2 – 3	RXD	User's application uses RS232 interface (position labeled SCI)

**JP4 - Configuration Header  
– TX select (RS232)**

Pins	Signal Name	Description
1 – 2	COMM	MON08 mode (position labeled MON)
2 – 3	TXD	User's application uses RS232 interface (position labeled SCI)



