Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

ST7MDTS1-EMU2B
HDS2 Series Emulator
User Manual

Release 1.0

August 2001

Ref: DOC-ST7MDTS1-EMU2B

INSTRUCTIONS FOR USE—WARNING

This product is conform to the 89/336/EEC Directive. It complies with the ITE EN55022 standard for EMC emissions and generic 50082-1 (1992 edition) immunity standards.

This product is an FCC Class-A apparatus. In a residential environment, it may cause radioelectrical disturbances.

In addition, some parts of this emulator are not contained in an outer casing; consequently, it cannot be immune against electrostatic discharges (ESD). It should therefore be handled only in static safe working areas. Please refer to *Appendix A: EMC Conformity and Safety Requirements* on page 53 for relevant safety information.

USE IN LIFE SUPPORT DEVICES OR SYSTEMS MUST BE EXPRESSLY AUTHORIZED.

STMicroelectronics PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF STMicroelectronics. As used herein:

1. Life support devices or systems are those which (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided with the product, can be reasonably expected to result in significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can reasonably be expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Table of Contents

Table of Contents

1 INTRODUCTION

Thanks for choosing ST7! This manual will help you get started with the ST7MDTS1-EMU2B.

The ST7MDTS1-EMU2B package is a development tool designed for emulation of the following microcontrollers of the ST72SCR family:

Supported Devices
ST72SCR1R4T1
ST72FSCR1R4T1
ST72SCR1F4M1
ST72FSCR1F4M1

The ST7MDTS1-EMU2B package will assist you in debugging your application hardware as well as your software. The ST7MDTS1-EMU2B kit comes with the ST7 Visual Debug software package which contains all of the necessary resources to help you design, develop and debug ST7 application software running in a real environment.

Tip: *If you come across any terms or abbreviations you do not understand, you can check their meaning in the Appendix C: Glossary on page 59.*

Note: *Please refer to Section 4.7: Emulation functional limitations and discrepancies on page 52 for description of the limitations on the current release of this emulator kit.*

First off, check that the ST7 MCU that you have picked for your application is in the list of devices (see table above) supported by this version of the ST7MDTS1-EMU2B emulator.

The emulator package is made up of two main parts:

- The Hardware Development System (**ST7-HDS2**), which is the common mainframe to all ST7 emulators.
- The **ST7MDTS1-Active Probe**, dedicated to the family, which constitutes the physical link between the emulator and your application.

Note: *When receiving the ST7MDTS1-EMU2B development tool, please refer to the Delivery checklist on page 11 to confirm that all of the contents of the package are present.*

The emulator performs two main functions:

- It replaces the microcontroller in the application, by means of an emulation probe that is plugged into the application in place of the emulated MCU.
- It controls the internal data bus of the emulated microcontroller, providing arbitration and tracing capabilities on all accesses to either of the following resources:
  - ST7-HDS2 resource,
  - ST7MDTS1-Active Probe resources,
  - Application resources.

Therefore, you can have the emulator running your software in the application as the emulated microcontroller unit (MCU) would do, associated with extensive tracing capabilities (keeping a trace of what the MCU did) and control capabilities (ability to react specifically upon defined conditions).

In this way, it is possible to obtain a full emulation of the microcontroller resources.

1.1 Emulator configuration

Figure 1 shows a general configuration for the ST7MDTS1-EMU2B emulator kit. The main ST7-HDS2 box is connected to your PC via the parallel port. Two flat cables connect the ST7-HDS2 box to the ST7MDTS1-Active Probe, to which a device adapter can be fixed via an extender or flex cable so that you can connect the emulator to your application board.

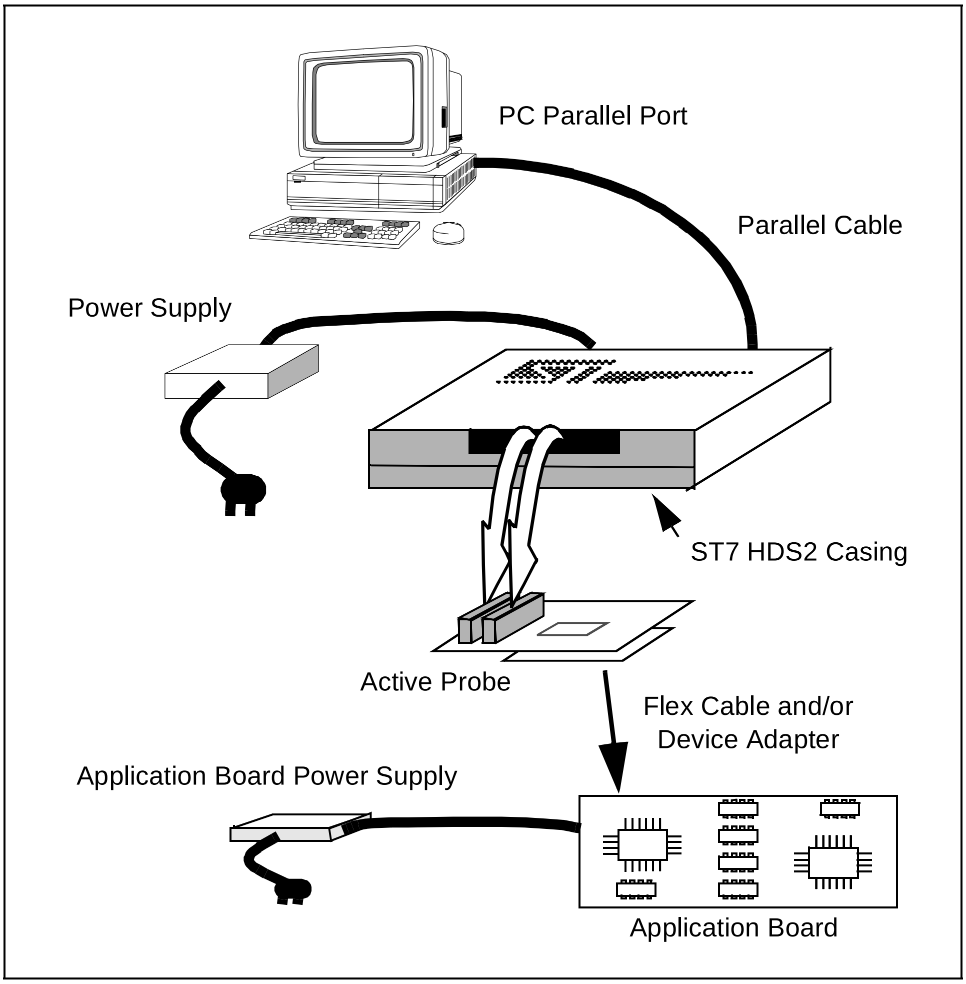


Figure 1: ST7MDTS1-EMU2B general configuration

1.2 Emulator operation

A symbolic debugger, **ST7 Visual Debug**, (also referred to as **STVD7**), is provided to control the emulator.

ST7 Visual Debug can be run on a PC under the Windows environment, and is common to all ST7 devices. ST7 Visual Debug uses a window menu-driven interface, and enables you to configure the emulator.

Chapter 3: STVD7 on page 19, explains how to install ST7 Visual Debug on your PC, and set up the emulator configuration so that you can begin your debugging session.

Once assembled and linked, the application software is ready to be downloaded into the ST7 emulator. The development station performs a real-time emulation of the target device, thus allowing high performance testing and debugging of both application hardware and software.

When the program is fully debugged, the ST7 EPROM programming board (ref.: ST7MDTS1-EPB — not provided with this emulator kit) can be used to program the emulation device with the Motorola S Record format file produced by the OBSEND formatter.

1.3 Software and documentation for the emulator kit

The “MCU on CD” CD-ROM contains:

- ST7 Tools, comprising the following software:
  - The source-level graphic debugger, STVD7, that operates with ST7-HDS2 Emulators and ST7 Development Kits or as a standalone ST7 simulator.
  - The ST7 Assembly chain, composed of an assembler, linker, librarian and formatter.
  - The ST7 Windows Epromer to program your MCU target devices.
- Third-party C compiler and toolchain demos (Hiware and Cosmic).
- ST7 application notes (with sources), training slides and exercises, and other useful reference documents in PDF format, such as:
  - Datasheets for the ST7 MCU family
  - *ST7 Programming Manual*
  - *ST7 Assembler-Linker User Manual*
  - *STVD7 User Manual*

1.4 About this manual....

Detailed instructions on how to install your emulator configuration is described in *Chapter 2: Getting Started* on page 11.

How to start debugging your application using your emulator and STVD7 is described in *Chapter 3: STVD7* on page 19.

The emulator kit's hardware features are described in *Chapter 4: Emulator Features* on page 43.

The following conventions are used in this manual:

Bold text highlights key terms, phrases and is used when referring to names of dialog boxes, windows and tabs within windows.

Bold italic text denotes menu commands (or sequence of commands), options, buttons or checkboxes which you must click in order to perform an action.

Italicized text highlights document names, variable strings, column names and field names.

`Code font` designates file names, programming commands, path names and any text you must type.

The **>** symbol is used in a sequence of commands to mean "then". For example, to open an application in Windows, we would write: "Click ***Start>Programs>ST7 Tool Chain***>....".

1.5 Getting assistance

For more information, application notes, FAQs and software updates on all the ST microcontroller families, check out the CD-ROM or our website:

http://mcu.st.com

For assistance on all ST microcontroller subjects, or if you need help with using your emulator, use the contact list provided in *Contact list* on page 61. We'll be glad to help you!

2 GETTING STARTED

2.1 Your system requirements

The ST7MDTS1-EMU2B HDS2 Emulator (both hardware and software components) has been designed to work with PCs meeting the following requirements:

- One of the following operating systems: Microsoft® Windows® 95, 98 or NT®.
- Intel® Pentium (or compatible) processor with minimum speed of 100 MHz.
- Minimum RAM of 32 MB.
- 21 MB of free hard disk space to install all of the ST7 tools.

2.2 Delivery checklist

The emulator unit, (ref.: ST7MDTS1-EMU2B), is delivered with the following (refer to *Figure 2*):

1 One emulator box containing the ST7-HDS2 main board (ref.: MB176) with two 50-wire flat cables to connect the board to the emulation probe.

2 One emulator power supply and cable.

3 Three EMC suppress ferrites.

4 One generic emulation probe—the ST7 Generic Probe (ref.: DB467).

5 One rainbow-colored logic probe cable for the logical analyser.

6 Two SMB to BNC cables for use with the output triggers.

7 An ST7MDTS1-Target Emulation Board (TEB) (ref.: DB484).

8 One parallel cable.

9 A 20 cm TQFP64 flex cable with two 40-pin connectors (for connection between the emulation probe and your application board) for the TQFP64 package (ref.: DB513A). Used with (*10*).

10 A TQFP64-0.8 socket (ref.: TET 064SA) and cover, for soldering to your application board).

11 A 20 cm SDIP32 flex cable with one 40-pin connector (for connection between the emulation probe and your application board) for the SDIP32 package (ref.: DB510B). Can be combined with (*12*).

12 A SDIP32/SO24 adapter (ref.: DB514A) comprising two parts that fit together. The topmost part fits onto the SDIP32 flex cable (*11*) while the bottom-most part can be removed to be easily soldered to your application board.

13 This manual. (Not shown.)

14 The “MCU on CD” CD-ROM containing ST7 information and software, including ST7 Visual Debug. (Not shown.)

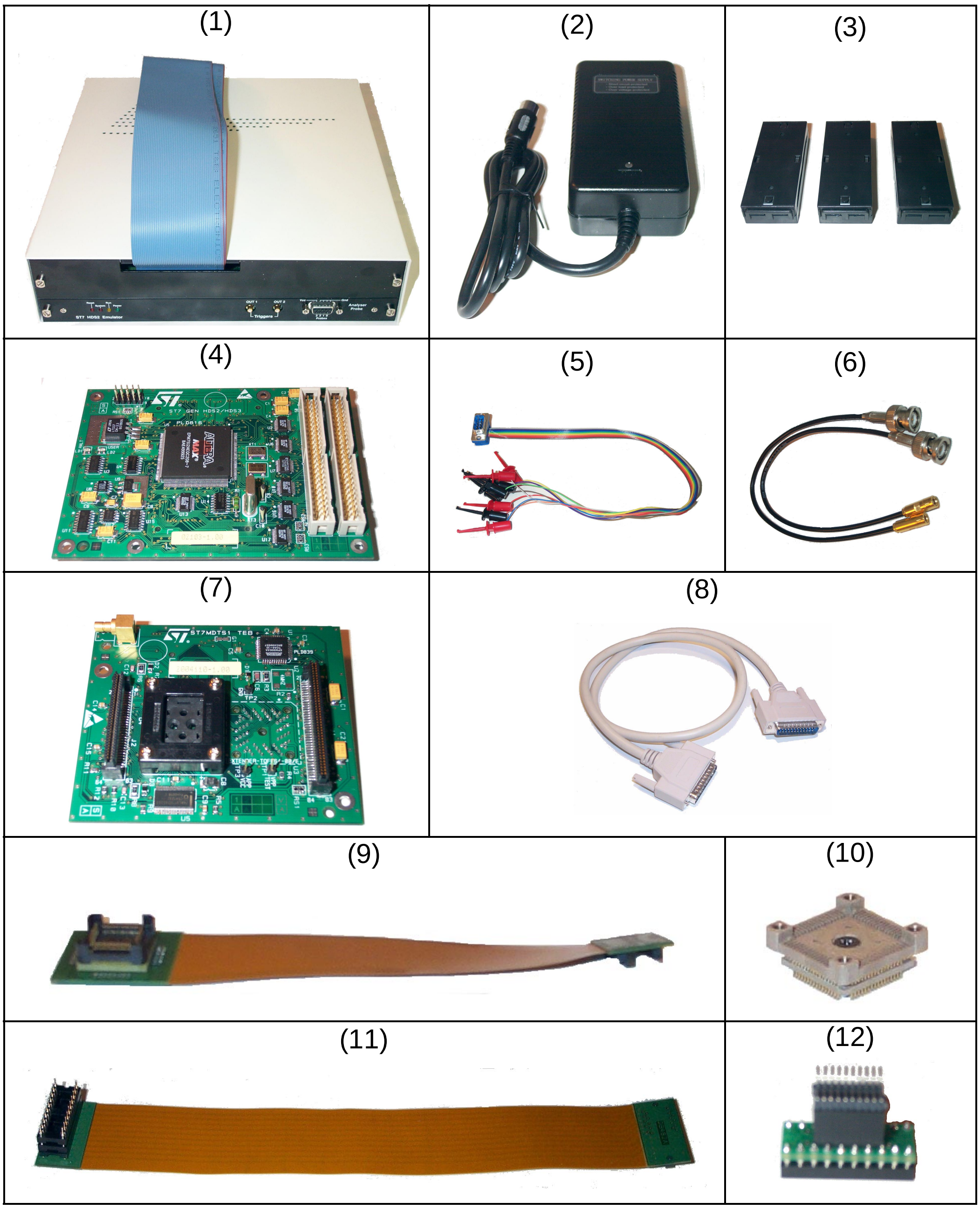


Figure 2: Main components of ST7MDTS1-EMU2B emulator kit

2.3 Installing the hardware

The ST7-HDS2 emulator is connected through the parallel port to a PC computer which runs the control software (ST7 Visual Debug) as explained later. To connect your ST7 HDS2 emulator, you will have to follow these general steps:

1. Connect the ST7-HDS2 to your PC using the parallel cable provided.
2. Connect the two flat cables of your ST7-HDS2 emulator to the emulation probe connectors.
3. Connect the emulation probe to your application board using the appropriate flex cable and/or socket adapter.
4. Connect the power supply cable between the power supply block and the power connector located on the rear panel of your ST7-HDS2 emulator.
5. Power up the emulator and then connect your application board power supply.

A connection flow diagram is shown in *Figure 3*. Step details are discussed in the following sections.

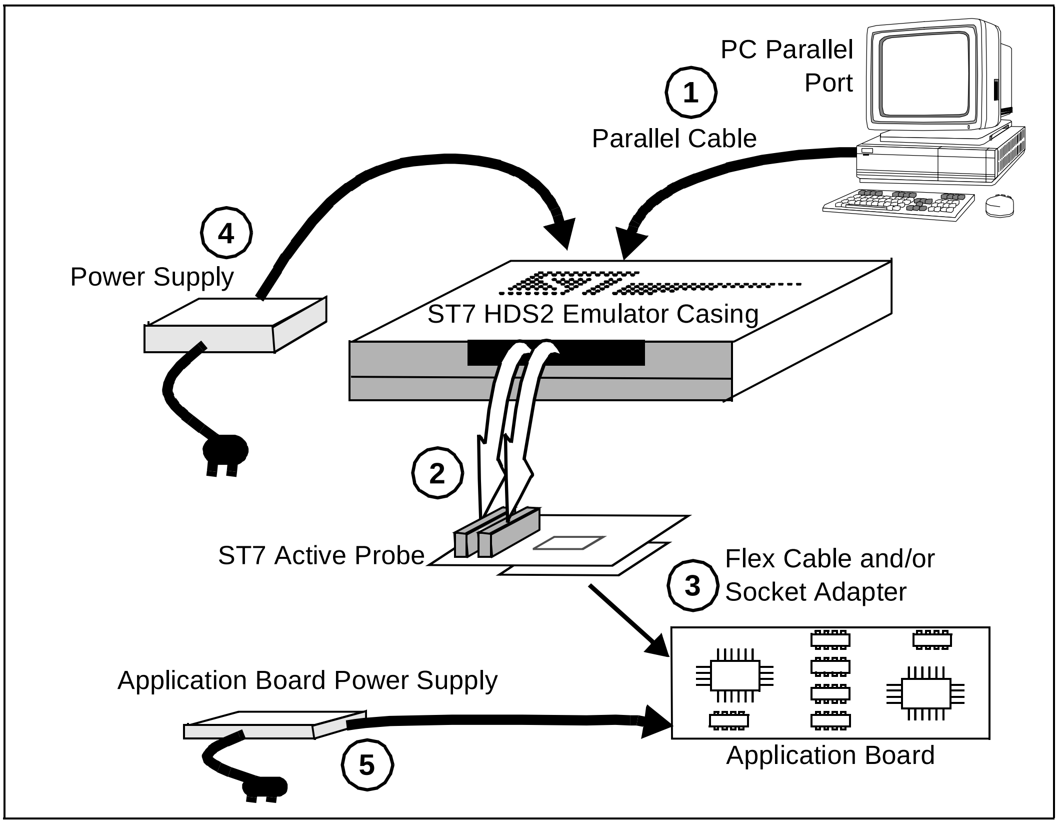


Figure 3: Connection flow

2.3.1 Step 1: Connecting the emulator to your PC

1 Shutdown and power-off the PC that is to be connected to the emulator.

2 Connect one end of the parallel cable to the emulator's rear panel 25-pin SUB-D connector and the other end to one of the PC's parallel ports (LPT1 to LPT2)—refer to *Figure 4*.

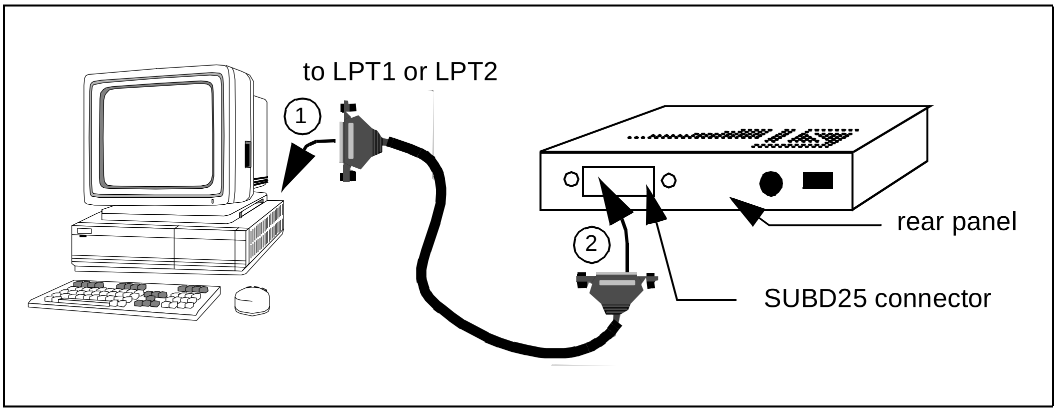


Figure 4: Connecting the emulator to the PC

2.3.2 Step 2: Connecting the probe

1 Ensure that the application and the emulator are **powered-off**.

2 Plug the two 50-wire flat cables into the J1 and J2 connections on the probe, as described below (refer also to *Figure 5*):

ST7 HDS2 emulator (ref.: MB176)	ST7 generic probe (ref.: DB467)
Upper cable	J2
Lower cable	J1

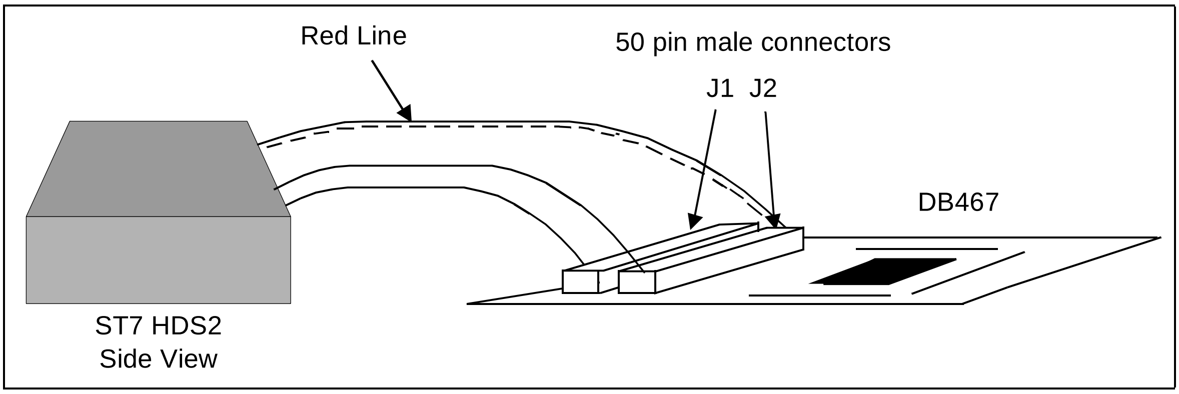


Figure 5: Connecting the emulator to the generic probe

3 **EMC-Compliant Probes:** In order to work under an EMC-compliant environment, you will have to clip two or three EMC-ferrites on the 50-wire flat cables linking the probe to the emulator box. Place these ferrites as close to the emulator window as possible. Three ferrites are provided in the package. See *Figure 6* for an illustration of where to attach the ferrites.

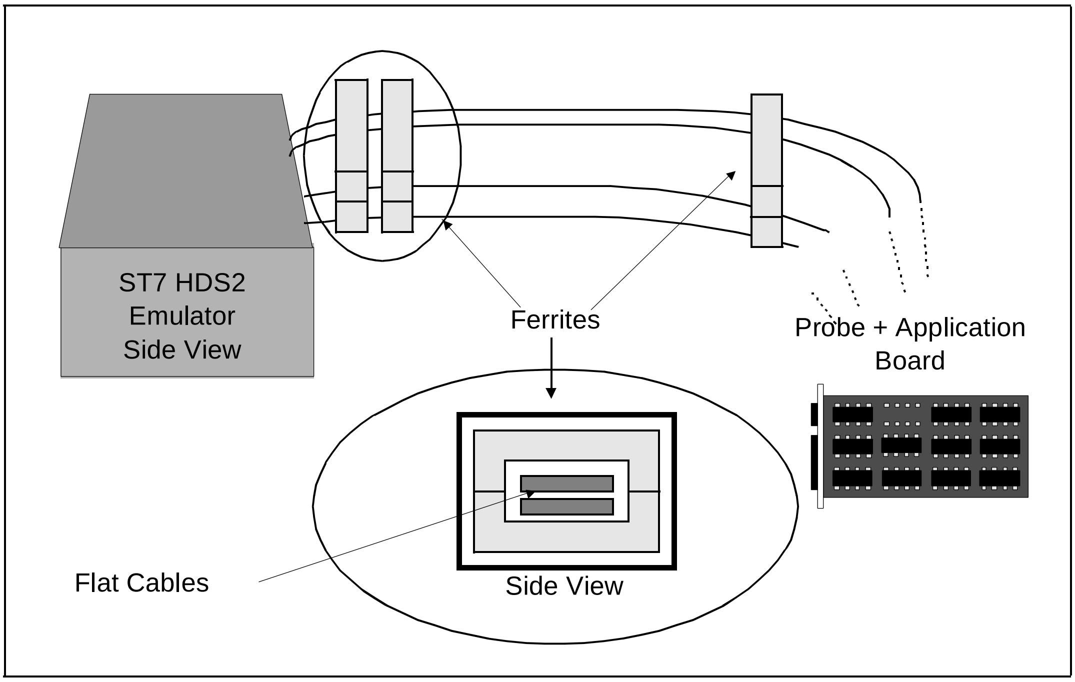


Figure 6: Making your emulation probe EMC-compliant

2.3.3 Step 3: Connecting the probe to your application board

Emulated devices of the ST7MDTS1 family are available in two packages, each package has its own connection procedure, found on the page cited in the following table:

Packages	Devices	Page No.
TQFP64	ST72SCR 1R4T1 ST72FSCR 1R4T1	16
SO24	ST72SCR 1F4M1 ST72FSCR 1F4M1	17

If you are using the TQFP64 package, proceed as follows (see *Figure 7*):

To connect the ST7MDTS1-Active Probe to your application board, a TQFP64 flex cable and a TQFP64-0.8 socket base have been provided.

1 Solder the TQFP64-0.8 socket base (ref.: TET 064SA) onto your application board. Do not use the socket cover (set it aside for future use with an actual MCU).

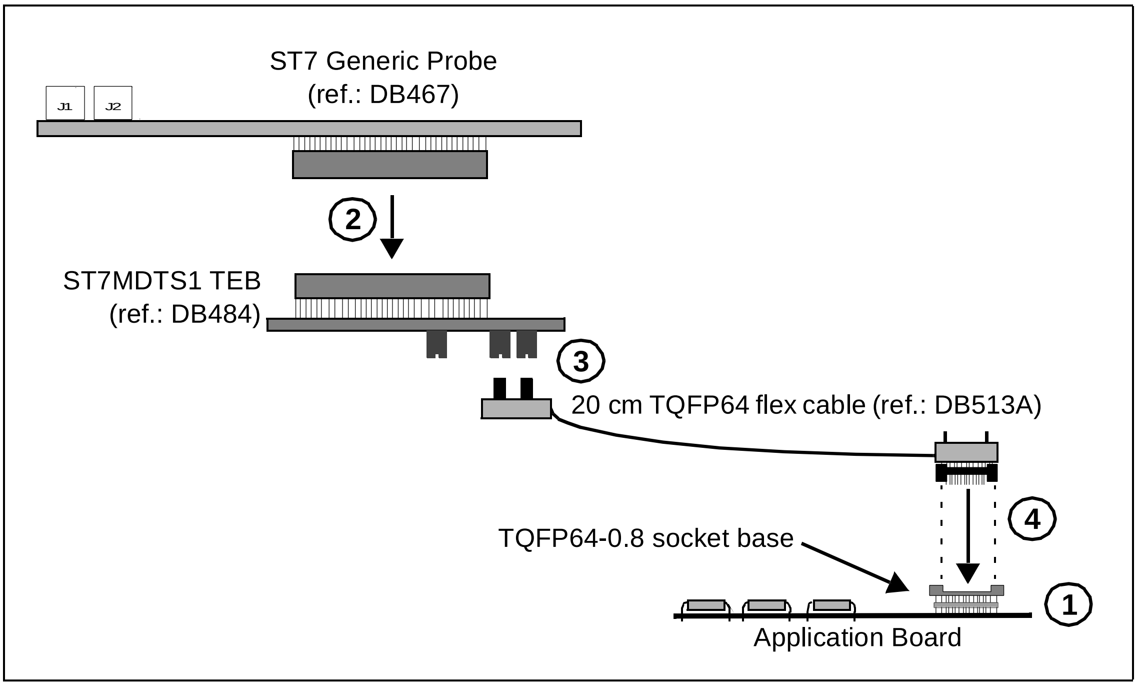


Figure 7: TQFP64 MCU package connections

2 Connect the ST7 Generic Probe (ref.: DB467) to the ST7MDTS1 TEB (ref.: DB484).

3 Connect the two 40-pin connectors of the 20 cm TQFP64 flex cable (ref.: DB513A)) to the two 40-pin connectors (labelled QFP64 FLEX ADAPTER) on the underside of the ST7MDTS1 TEB (ref.: DB484), making sure to align the pin1's of the flex cable connectors with the pin 1's of the TEB connectors.

4 Connect the TQFP64 connector on the end of the flex cable to the TQFP64 socket base, making sure to align pin 1 on the TQFP64 flex connector with pin 1 on the socket base. (Pin 1 is indicated by a chamfer on the TQFP flex connector and by a little arrow or chamfer on the socket base.)

5 Now use the four screws supplied to fasten this entire assembly together through the holes located on the TQFP flex cable connector as indicated in *Figure 7*.

If you are using the SO24 packages, proceed as follows (see *Figure 8*):

To connect the ST7MDTS1-Active Probe to your application board, a SDIP32 flex cable and a SDIP32/SO24 adapter have been provided. The SDIP32/SO24 adapter (ref.: DB514A) is comprised of two parts that fit together. The topmost part fits onto the SDIP32 flex cable (*11*) while the bottom-most part can be removed to be easily soldered to your application board.

1 Solder the bottom part of the SDIP32/SO24 adapter (ref.: DB514A) onto your application board.

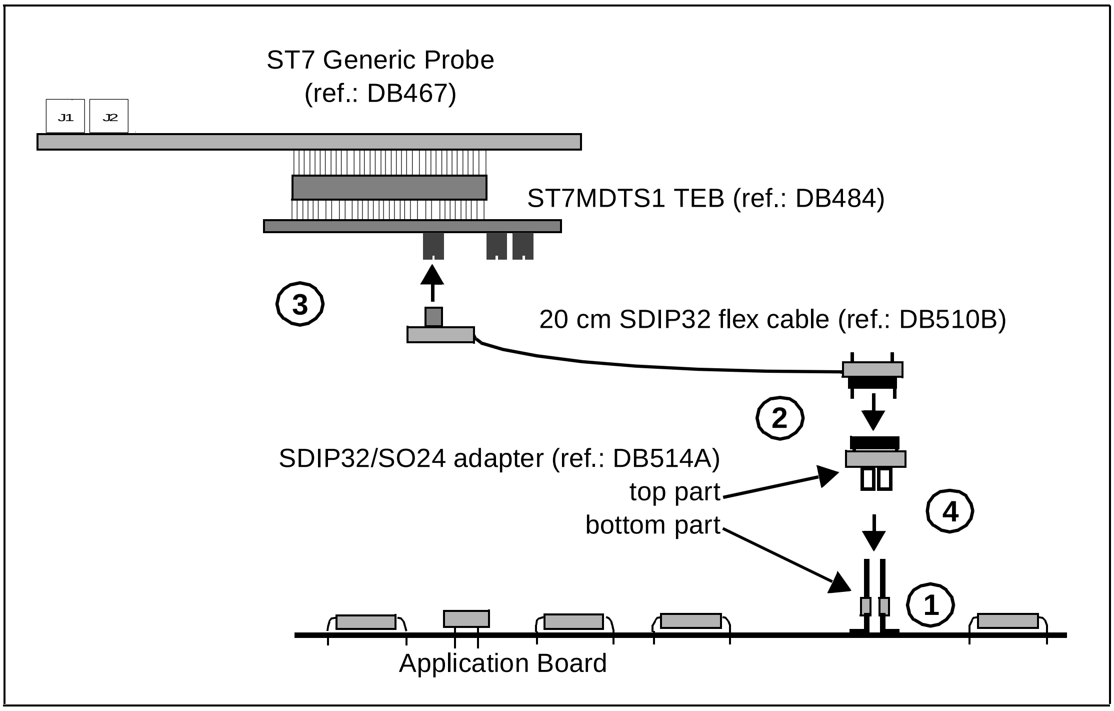

ST7 Generic Probe
(ref.: DB467)
J1
J2
ST7MDTS1 TEB (ref.: DB484)
3
20 cm SDIP32 flex cable (ref.: DB510B)
2
SDIP32/SO24 adapter (ref.: DB514A)
top part
bottom part
4
1
Application Board

Figure 8: SO24 MCU package connections

2 Connect the top part of the SDIP32/SO24 adapter (ref.: DB514A) to the SDIP32 flex cable (ref.: DB510B).

3 Connect the single 40-pin connector on the SDIP32 flex cable (ref.: DB510B) to the single 40-pin connector (labelled SO24 FLEX ADAPTER) on the ST7MDTS1-TEB (ref.: DB484).

4 Connect the top and bottom parts of the to the SDIP32/SO24 adapter (ref.: DB514A).

2.3.4 Step 4: Connecting the power supply

Caution: *Make sure that both the ST7MDTS1-EMU2B and the application board are powered OFF before making any connections.*

1 Connect the external power supply provided with the emulator to the rear panel of the mainframe using a 5-pin DIN connector.

2 Plug the power supply into the mains using the supply cable provided.

Mains Voltage Specifications

AC Voltage	100 V to 240 V
Frequency	50 Hz to 60 Hz

2.3.5 Step 5: Powering up

1 Check the ST7-HDS2 operating voltage (110 V/220 V), indicated on the label on the power supply. Contact your dealer if this voltage does not correspond to your mains' power supply.

2 Power up the ST7 HDS2 emulator from the ON/OFF switch located on the rear panel. The LED labelled ***Power*** on the front panel should then light up.

3 Power up your application board.

Warning: **Always power on your ST7 HDS2 emulator first, then power up your application board. When powering off, always power off your application board first, then power off your emulator.**
NEVER have your application board under power while the emulator is powered off—this will cause serious damage to the emulator.

2.4 Debuggers supporting the ST7 HDS2 emulator

ST7 Visual Debug (also known as STVD7) is the STMicroelectronics debugger currently supporting the ST7 HDS2 emulator.

Third party debuggers supporting the ST7 HDS2 emulator include the following:

- ST7-Hiwave by HIWARE
- Zap by COSMIC
- Hitop by HITEX.

ST7 Visual Debug is free software. It is available on the STMicroelectronics website. See *Product Support* on page 61 for more information.

3 STVD7

STVD7 is an integrated development environment that allows you to edit, debug and rebuild your application all from within STVD7.

The following sections tell you:

- *Section 3.1*—how to install the STVD7 software,
- *Section 3.2*—how to launch STVD7,
- *Section 3.3*—a little about STVD7's debugging features,
- *Section 3.4*—what a workspace is,
- *Section 3.5*—what toolchains and executable files are supported by STVD7,
- *Section 3.6*—how to create a STVD7 workspace,
- *Section 3.7*—how to open existing workspaces,
- *Section 3.8*—how to open binary files,
- *Section 3.10*—how to change your project settings,
- *Section 3.11*—how to save workspaces,
- *Section 3.12*—how to switch from the build context to the debug context,
- *Section 3.13*—how to configure the target MCU in order to debug more accurately and efficiently.

3.1 Installing STVD7

Your emulator comes with the "MCU on CD" CD-ROM which contains a number of ST7 software tools. These tools run under the Windows® 95, 98, 2000 and Windows® NT® operating systems.

*Note: To install the software on "*MCU on CD*", Windows® 2000 and NT® users must have administrator privileges.*

To install and setup the ST7 software tools, follow these steps:

1 Close all other open applications on your Windows desktop.

2 Insert the "MCU on CD" into your CD-ROM drive. The CD-ROM's autorun feature will open up a welcome screen on your PC. If the autorun feature does not work, use Windows® Explorer to browse to the CD-ROM' s root folder, and double-click on `Welcome.exe`.

3 Select ***Install Your Development Tools*** from the list of options. A new screen will appear listing the different families of STMicroelectronics MCUs.

4 Use your mouse to place the cursor over the ***ST7 Tools*** option. Choose ***ST Tools***, then ***ST7 Toolchain*** from the lists that appear.

5 The install wizard will be launched. Follow the instructions that appear on the screen.

You can choose to install the complete toolchain (i.e. the appropriate version of STVD7, the Windows Epromer and the Assembler-Linker) for each type of development tool (development kit, HDS2 or EMU3 emulators or simulator), or perform a customized installation.

If you choose a customized installation, you can choose to install any or all of the STVD7 versions, and/or the Windows Epromer and/or the Assembler-Linker. **As a minimum, in order to emulate your application with your emulator, you must install STVD7 for HDS2.**

If you also install the ST7 Assembly Toolchain, you will be able to use the ST7 Assembly Toolchain as part of STVD7's integrated development environment.

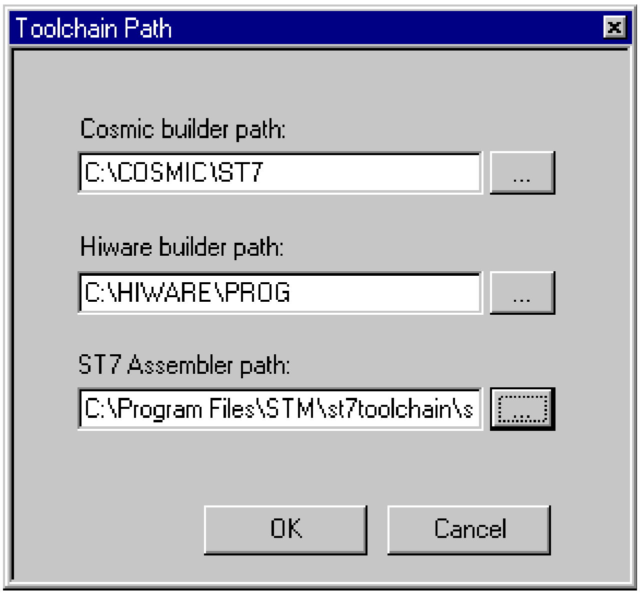
The installation is now complete. You will be prompted to reboot your computer. You should do so before launching STVD7.

3.2 Launching STVD7

1 From your Windows desktop, select ***Start>Programs>ST7 Tool Chain>Development Tools>STVD7 HDS2 emulator.***

2 The first time you open a version of STVD7 you will be prompted to enter the toolchain paths to be used by STVD7's integrated development environment.

Enter the paths for the toolchains that you use (i.e. any or all of the Hiware, Cosmic or ST7 ASM toolchains) and click ***OK***. (The default paths for each toolchain are shown below.)



Toolchain Path

Cosmic builder path: C:\COSMIC\ST7

Hiware builder path: C:\HIWARE\PROG

ST7 Assembler path: C:\Program Files\STM\st7toolchain\s

3 If you choose ***Cancel,*** you will be prompted again to enter the toolchain paths the next time you launch STVD7.

Note: You may modify the toolchain path at any time from within STVD7—simply select ***Project>Toolchain Paths*** *from the main menu to access the dialog box above.*

3.3 About STVD7 debugging features

A number of advanced features are included in the STVD7 software:

- **Data Breakpoints** on the occurrence of a memory access via a read operation or a write operation, or both.
- **Instruction Breakpoints** on the occurrence of an opcode fetch.
- A **Logical Analyser** that allows you to control either the recording of the trace buffer, or a break in the execution of the application using a series of specific conditions (events).
- A **Trace window** to view the contents of the **trace buffer**, which permanently records in real time on 32-bits:
  - Address and data bus information.
  - Flag status and 4 external signal values.

  You can record up to 1024 executed cycles. Using trace filtering, you can filter out only those cycles you wish to record in the trace buffer. You can equally control which of the recorded cycles are displayed in the Trace window using line filtering. Addresses, data, control/status bits and 4 user signals are displayed using mnemonic and user symbols.
- Internal synchronization signals can be output to either of two **Trigger Outlets** on the front panel of the emulator (OUT1 or OUT2). This feature enables you to count events using an external equipment, when optimizing software for example, or to synchronize an oscilloscope when debugging hardware.
- **Hardware Events** can be used to control the sending of signals to the trigger outputs.
- You can choose the output that you wish the signal to be sent to (i.e. either OUT1 or OUT2).
- A **Hardware Test** function that allows you to perform a number of hardware tests on the Development Board, at your choosing. Refer to *Running the Hardware Test* for more information.
- A powerful **online help** facility can be invoked at any time to give additional information about the commands, the processor or the emulator kit.

3.4 Workspaces

STVD7 organizes project development and debugging into workspaces. Workspaces allow you to store application and project settings and save them as a `*.wsp` file, so that each time you wish to work on the project, you will find all of the settings exactly as you left them.

Creating a workspace is the first thing that you need to do when using STVD7 for the first time or when starting any new project. You must have an open workspace to work with STVD7. How to create a new workspace is described in detail in *Section 3.6* on page 26. Sample workspaces for each supported toolchain are provided so that you can familiarize yourself with STVD7 (for a listing of sample workspaces, see *Table 1* on page 24).

Each workspace is comprised of three information sets: the project settings, the visual environment and the debugging context.

- The **project settings** consists of the information necessary for a successful build of an application (commands to run, makefile file etc....). Your workspace’s project settings include the definition of your application toolchain (see *Section 3.5* on page 23).
- The **visual environment** consists of the open windows elements along with their current layout, bookmarks and other features. The visual environment is composed of two environments, one in the **Build context** and one in the **Debug context** (see *Section 3.12* on page 36).
- The **debugging information** includes information on breakpoints, memory mapping, advanced breakpoints programs, trace etc..

3.5 Toolchains and application files

A quick summary of development toolchains and application file types supported by STVD7 will help you in setting up your workspace.

Three different development toolchains are currently supported by the STVD7. Each type of toolchain has its own application and executable file types, project environment and building tools (i.e. linkers and convertors):

- The **ST7 macroassembler toolchain** from STMicroelectronics, which generates either `.s19` or `.hex` executable files with various intermediate files, such as `.map` or `.lst` files.
- The **Hiware C or Assembler toolchain**, which generates `.abs` executable files with various intermediate files, such as `.o` or `.dbg` files.
- The **Cosmic C or Assembler toolchain** which generates `.elf` executable files with various intermediate files, such as `.o` or `.st7` files.

When you set up a workspace, you will need to define the following project settings:

- **The toolchain to be used—**Hiware, Cosmic or ST7 macroassembler.
- **The executable file** (`*.abs`, `*.elf`, `*.s19` or `*.hex` depending on toolchain—refer to *Table 2* on page 25).
- **The maker program** for the toolchain. The maker program can be a part of the toolchain software (such as Hiware's `maker.exe`) or you can choose to use a generic maker such as `Nmake.exe` or `Gmake.exe` (which is provided with the STVD7).
- **The maker batch file** (`*.mak` or `*.bat`). This is a file which you create for each application which spawns the compilation and/or link step each time you wish to build or rebuild. In it, you define the conditions for recompiling, re-linking or both.
  Default `*.mak` or `*.bat` files are often included with the toolchains—for example, `maker.mak` is included with the Hiware toolchain and simply recompiles your application if it detects that the file has been saved since the start of your debugging session. The STVD7 software includes sample `*.mak` and/or `*.bat` files for each toolchain—these are listed in *Table 1*.

Table 1: Sample files included with STVD7

Toolchain	Sample Workspace (with default path	Sample Make and/or Batch files (with default path[1])	Description of Make/Batch File
ST Macro assembler	`.../realtim/realtim.wsp`	`.../realtim/tim_rtc.bat`	Batch file that forces a recompile of the application.
	`.../spim11/spim11.wsp`	`.../spim11/spim11.bat`	Batch file that forces a recompile of the application.
Cosmic	`.../c/cosmic/sample.wsp`	`.../c/cosmic/sample.mak`	Recompiles only if one (or more) of the application files has been resaved.
		`.../c/cosmic/sample.bat`	Batch file that forces a recompile of the application.
Hiware	`.../c/hiware/sample.wsp`	`.../c/hiware/build.mak`	Recompiles only if one (or more) of the application files has been resaved.
		`.../c/hiware/rebuild.mak`	Forces a recompile of the application.

[1] The full default path is: `C:/Program Files/Stm/st7toolchain/stvd7/hds/sample/...`

3.5.1 About executable files

The user should verify that the options to include debug information were active during creation of the project files. *Table 2* on page 25 summarizes the way each toolchain functions and lists the different file types (source files, intermediate files and executable files) used and produced by the toolchain. The **executable file types** and **intermediate file types** necessary to exploit fully the STVD7 capabilities are listed.

Table 2: Toolchain steps and their output files

Toolchain:	ST Macroassembler	Hiware	Cosmic
Compile or Assemble Step:			
Source File Types	`.asm`	`.c, .asm`	`.c, .s`
Required Options	`asm -li macrost7.asm`	<NONE>	`+debug`
Resulting File Types	`.obj, .lst`	`.o, .dbg`	`.o`
Linker Step:			

Required Options	`lyn macrost7.obj, macrost7`
	`asm macrost7.asm  -sym -fi=macrost7.map`
		<NONE>	<NONE>

Resulting File Types	`.map, .lst`	`.abs`	`.st7`

Converter Step:	`obsend macrost7, f, macrost7.s19, srec`
	or
	`obsend macrost7, f, macrost7.hex, intel`
		not applicable	`cvdwarf`

Resulting executable file:	`.s19 or .hex`	`.abs, .elf`	`.elf`
Necessary Intermediate Files:	`.map, .lst`	`.o, .dbg`	<NONE>

The **executable file(s)**, **source files** and any necessary **intermediate files** (these are listed above and contain debug information necessary to the STVD) should be located in the same project directory. You do this when you define your workspace.

Note: *It is always preferable to have access to all of the files generated by the development toolchain. However, you can load* `*.s19` *or* `*.hex` *binary files directly and have limited debugging capabilities (refer to Section 3.8 on page 30).*