



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



ST7MDT2-DVP2 Development Kit User Manual

Release 1.2

July 2001



Ref: DOC-ST7MDT2-DVP2



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, this development board is 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 67 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

Chapter 1:	Introduction	5
1.1	Development board functional configurations	6
1.2	Software and documentation for the development kit	8
1.3	About this manual....	8
1.4	Getting assistance	9
Chapter 2:	Getting Started	11
2.1	Your system requirements	11
2.2	Delivery checklist	11
2.3	Installing the hardware	12
2.4	Connecting the TQFP64 passive probe to your application board	14
Chapter 3:	STVD7	15
3.1	Installing STVD7	15
3.2	Launching STVD7	16
3.3	About STVD7 debugging features	17
3.4	Workspaces	18
3.5	Toolchains and application files	19
3.6	Creating a workspace	22
3.7	Opening an existing workspace	24
3.8	Opening binary files	26
3.9	Opening lone programmable files (*.s19 or *.hex)	27
3.10	Changing your project settings	28
3.11	Saving workspaces	30
3.12	Debug context and Build context	32
3.13	Configuring the MCU	33
3.14	Start debugging!	38
Chapter 4:	Programming ST7 Devices	39
4.1	Device programmer features	40
4.2	Programming methods	41
4.3	Device installation	42
4.4	Starting the Windows Epromer	43
4.5	Configuring the Epromer	44
Chapter 5:	Hardware Features	47
5.1	ST7MDT2-DVP2 development board layout	47

Table of Contents

5.2	ST7MDT2-DVP2 emulation architecture	48
5.3	Link to PC	48
5.4	Power supply	49
5.5	Jumper and solder point descriptions	50
5.6	CAN features	51
5.7	Pin descriptions and package footprints	53
5.8	Trigger/trace settings	60
5.9	Hardware events	62
5.10	On-chip peripherals	62
5.11	Emulation functional limitations and discrepancies	64
Appendix A: EMC Conformity and Safety Requirements		67
Appendix B: Troubleshooting		69
B.1	Identifying the problem	69
B.2	Changing the parallel port setup on your PC	70
5.12	Running the hardware test	70
Appendix C: Glossary		73
Product Support		77
	Getting prepared before you call.....	77
	Contact list	77
	Software updates	78
	Hardware spare parts	78
Index		79

1 INTRODUCTION

Thanks for choosing the ST7MDT2-DVP2 development kit! The ST7 DVP2 family of development kits offer the following new features:

- Delivered with the debugger software package — ST7 Visual Debug!
- Trace buffer recording, viewing and output.
- In Situ Programming (ISP) ability (for MCUs that support this feature).

This manual describes how to start and use the ST7MDT2-DVP2 development kit for the ST72334 MDT2 CAN-less family and the ST7511R9 MDT2 CAN family of MCUs, allowing you to get acquainted with the ST7 microcontroller world and become familiar with the methods for developing and debugging ST7-driven applications.

Note: If you come across any terms or abbreviations you do not understand, you can check their meaning in the Glossary on [page 73](#).

This manual also provides a guidance for programming a selection of Flash, Eprom and OTP (One Time Programmable) ST7 microcontrollers.

The ST7MDT2-DVP2 development kit contains all the necessary resources that will help you:

- design, develop and debug ST7 application software running in a real environment,
- program selected ST7 devices in a variety of modes (refer to [Table 3](#) on page 39).

First off, check that the ST7 MCU that you have picked for your application is in the list of devices supported by this version of the ST7MDT2-DVP2:

Supported Devices

ST72124J2/J4

ST72314J2/J4

ST72314N2/N4

ST72334J2/J4

ST72334N2/N4

ST72532R4

ST72311R6/R7/R9

ST72512R4

ST72511R6/R7/R9

The development kit can be used as a tool to emulate applications on the target MCU, or as a chip programming tool as summarized in the following sections.

1.1 Development board functional configurations

Figure 1 shows the development board of the ST7MDT2-DVP2 development kit in an ST7 MCU Emulator configuration.

Figure 2 shows the development board of the ST7MDT2-DVP2 development kit in an ST7 MCU Programming Board configuration.

Figure 3 shows how you can set up the Development Board to perform in situ programming of devices on an application board.

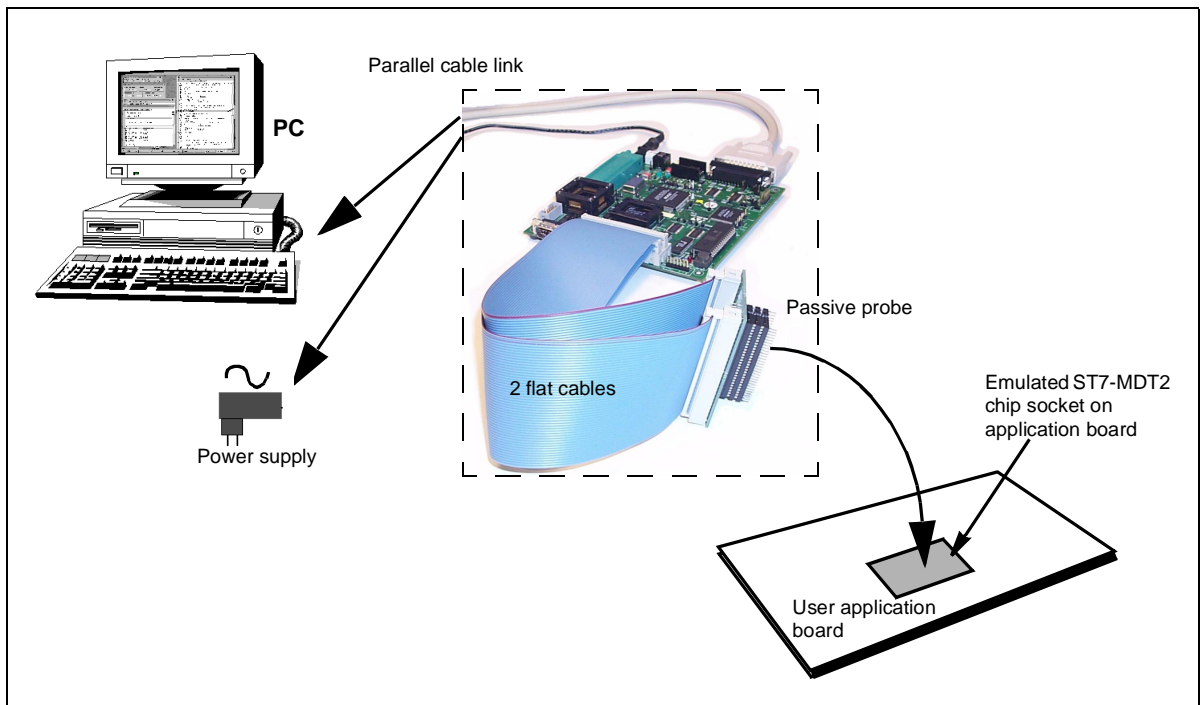


Figure 1: Using the development board as an ST7 MCU emulator

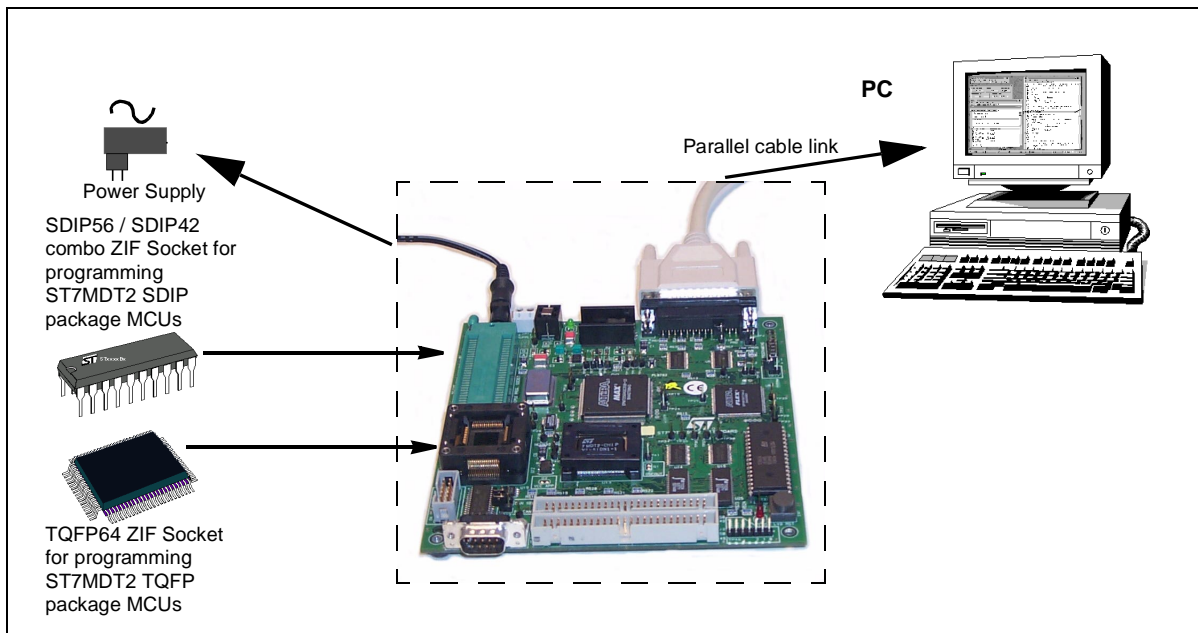


Figure 2: Using the development board as an ST7 MCU programming board

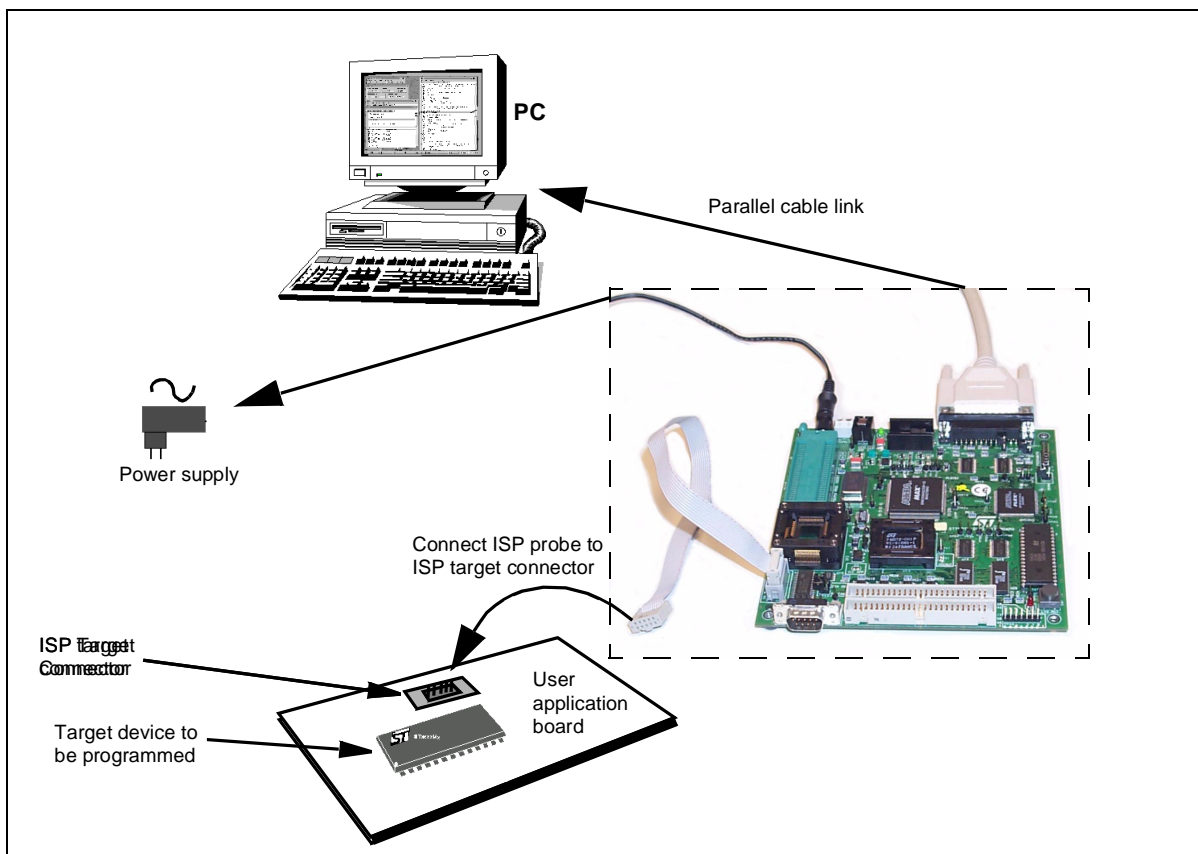


Figure 3: Using the development board for In Situ Programming (ISP)

1.2 Software and documentation for the development kit

The “MCU on CD” CD-ROM contains:

- ST7 Tools, comprising the following software:
 - The source-level graphic debugger, STVD7, that operates with ST7 development kits and ST7-HDS2 Emulators 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, this manual (in PDF version), 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*

1.3 About this manual....

Detailed instructions on how to install your development kit configuration is described in [Chapter 2: Getting Started](#) on page 11.

How to start debugging your application using your development kit and STVD7 is described in [Chapter 3: STVD7](#) on page 15.

How to program devices with the development kit is described in [Chapter 4: Programming ST7 Devices](#) on page 39.

The development kit's hardware features are described in [Chapter 5: Hardware Features](#) on page 47.

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 check boxes 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>....***”.

1.4 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 [Product Support](#) on page 77. We'll be glad to help you!

2 GETTING STARTED

2.1 Your system requirements

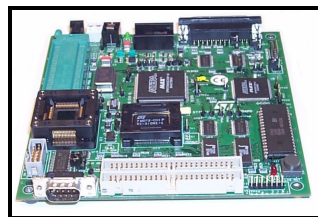
The ST7MDT2-DVP2 development kit (both hardware and software components) has been designed to work with PCs having the following configurations:

- One of the following operating systems: Microsoft® Windows® 95, 98, 2000 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.

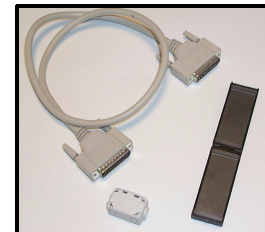
2.2 Delivery checklist

The ST7MDT2-DVP2 development kit contains:

- 1 One development board (Ref.: MB289).



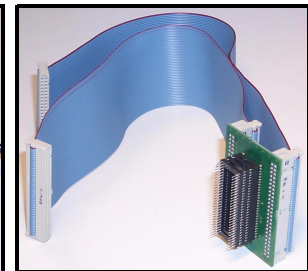
- 2 One parallel cable for PC connection and two EMC ferrites.



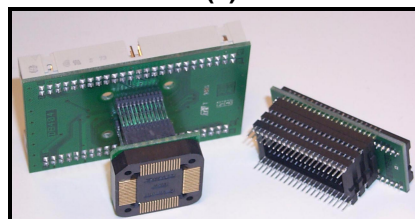
- 3 One 5 V external DC power supply with female connector cable (two possibilities exist).



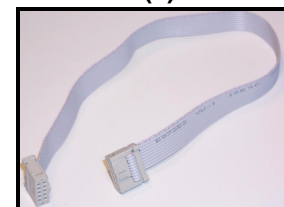
- 4 SDIP56 passive probe connector (Ref.: DB347) and two 50-pin flat cables.



- 5 One TQFP64 passive probe connector (Ref.: DB404) and one SDIP56 to SDIP42 (ref.: DB326) device adapter.



- 6 One ISP 10-pin flat cable for in situ programming.



- 7 One Yamaichi QFP64 socket, with its cover, screws and washers. (Not shown.)
- 8 One *MCU-on-CD* CD-ROM.(Not shown.)
- 9 This manual.(Not shown.)

Note: A *TQFP44* passive probe is available for the *ST7MDT2-DVP2* but must be ordered separately. Contact your nearest *STMicroelectronics* sales representative (see [page 77](#)).

2.3 Installing the hardware

To install the hardware, follow these steps:

- 1 Shut down and power-off the PC that is to be connected to the development board.
- 2 Connect one end of the supplied parallel cable to the parallel connector (P2) on the development board. Connect the other end of the parallel cable to the LPT1 or LPT2 parallel port on your PC.

Note: The supplied parallel cable has been tested in order to operate properly on most PCs. Do not use any other cable, especially if it is longer than the one provided in the kit—the board may not operate properly.

The cable should be connected directly to the DB-25 female connector of the PC parallel port. This connector is similar to the one installed on the board. Do not insert any additional cables or switchboxes between the PC and the board: a malfunctioning of the board may result. If a dongle is mounted on the PC parallel port, it should not interfere with the programming board. Should you notice that the board is dysfunctional, remove the dongle and restart the installation procedure.

- 3 Connect the two 50-pin passive probe cables to J1 and J2 on the development board.
- 4 To the other ends of the 50-pin cables, connect the passive probe and/or device adapter corresponding to the MCU package you wish to emulate:
 - SDIP56 passive probe alone for SDIP56 MCU packages.
 - TQFP64 passive probe alone for TQFP64 MCU packages.
 - SDIP56 passive probe with the SDIP42 device adapter for SDIP42 MCU packages.

Caution: Special precautions are required if you are using the **TQFP64** package. Because there is no Yamaichi socket available specifically for the **TQFP64** footprint (a QFP64 Yamaichi socket is furnished instead), there are special footprint precautions to be taken when designing your application board (see [Section 5.7.2: QFP64/TQFP64 footprint discrepancy issues](#) on page 59).

- 5 EMC-Compliant Probes (optional):** In order to work under an EMC-compliant environment, you will have to clip one EMC-ferrite on both 50-wire flat cables linking the application to the development kit board. Place this ferrite as close to the development kit board as possible. You also need to clip one EMC ferrite on the power supply wire coming from the power supply box. See [Figure 4](#) on page 13.

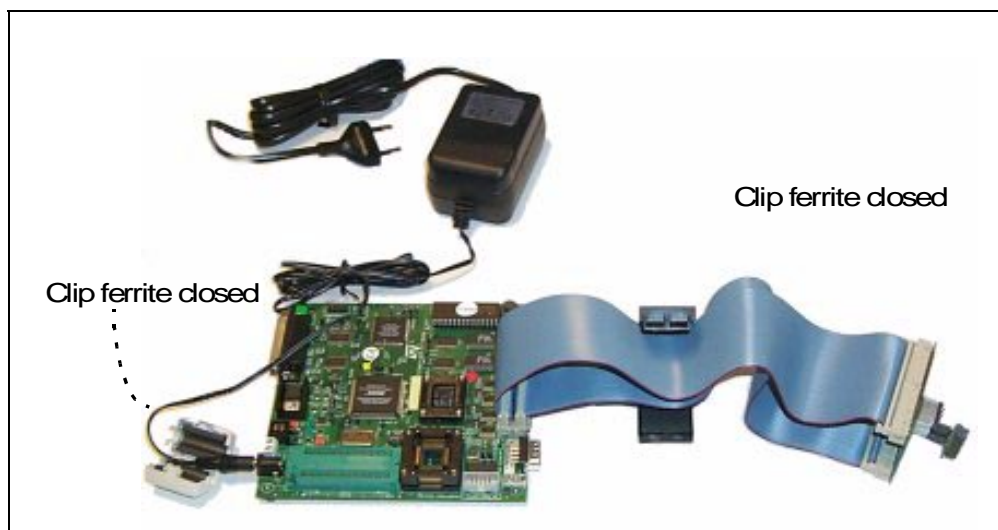


Figure 4: Connecting the ferrites

- 6** Connect the passive probe to the appropriate socket on your application board. For the **TQFP64 passive probe**, a Yamaichi QFP64 socket has been included. Follow the instructions in [Section 2.4](#) on page 14 to connect the TQFP64 passive probe to your application board.
- 7** Plug the DC power supply provided with the development kit into a power outlet. Connect the power cable to the development board. The green Power LED will light up.

Note: *The development board can also be fed via the JP1 connector by an external stabilized power supply (5 V \pm 0.25 V, 1 A) not provided with the Kit.
If the board is fed via the JP1 two-point connector make sure that the right feeders lead to the right polarities.*

- 8** Power on the PC and proceed with the installation of the software as described on [Section 3.1](#) on page 15).

Caution: *Do not use the jumper connections TP17, TP4 and TP5 — they are for factory testing only and modifications to them could cause your development board to malfunction. Refer to [Section 5.5](#) on page 50 for a description of all jumpers and solder connections on your development board.*

2.4 Connecting the TQFP64 passive probe to your application board

A Yamaichi QFP64 socket and its cover are provided in the package (Ref.: DB200). Before going through the procedure, make sure that you were properly delivered the socket, its cover and its screws and washers.

To connect the TQFP64 passive probe to your application board, proceed as follows (see following figure for flow order):

- 1 Solder the Yamaichi QFP64 socket base onto your application board (see [Section 5.7.2](#) on page 59 for footprint information). Do NOT screw the socket cover onto its base.
- 2 Place the end of the TQFP64 passive probe onto the Yamaichi socket base, taking care to align pin 1 of your application board with pin 1 of the passive probe. Pin 1 is indicated by a chamfer on the passive probe and by a little arrow or chamfer on the Yamaichi socket.
- 3 Once placed, you can screw the TQFP64 passive probe onto the Yamaichi socket using the four threaded holes located on the upper surface of the passive probe.

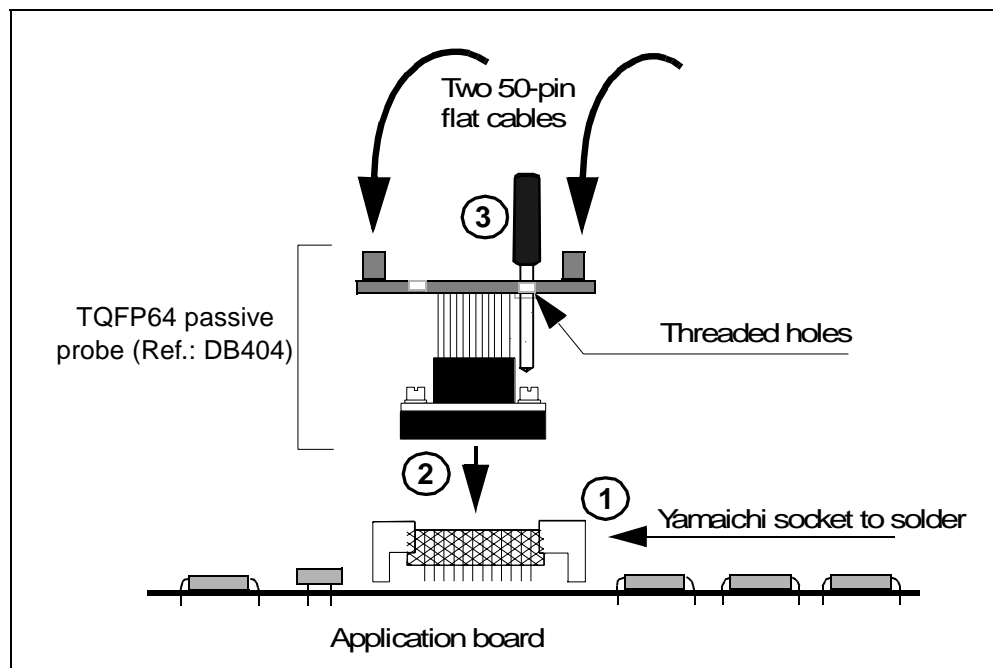


Figure 5: Connecting the TQFP64 probe to your application board

If you require supplementary sockets, the commercial references of Yamaichi QFP64 sockets are given below:

QFP64/TQFP64 Yamaichi socket WITH positioning pins	IC149-064-108-S5
QFP64/TQFP64 Yamaichi socket WITHOUT positioning pins	IC149-064-008-S5

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 development kit 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 and program your application with your DVP, you must install STVD7 for DVP and the Windows Epromer.**

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.

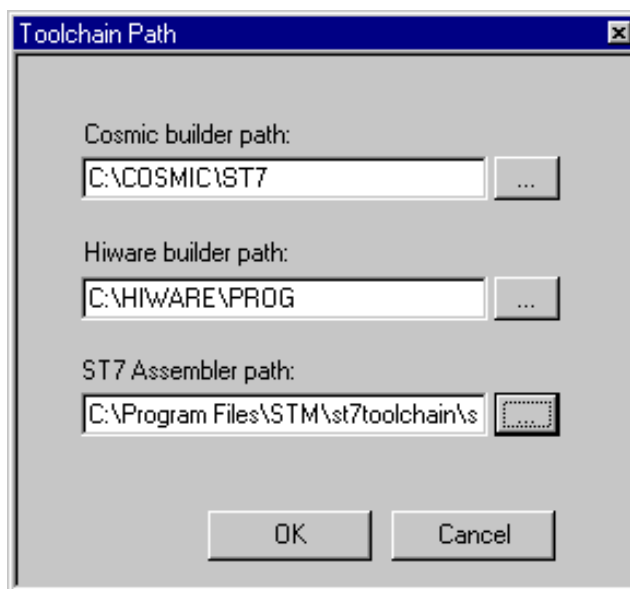
The installation is now complete. You will be prompted to reboot your computer. You should do so before launching STVD7.

3.2 Launching STVD7

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

- 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.)



- 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 **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 256 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.

- **Hardware Events** can be used to control the trace recording or the sending of signals to the trigger outputs.
- 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 22. 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 20).

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 19).
- 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 32).
- 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 21).
- **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/dvp/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 21 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

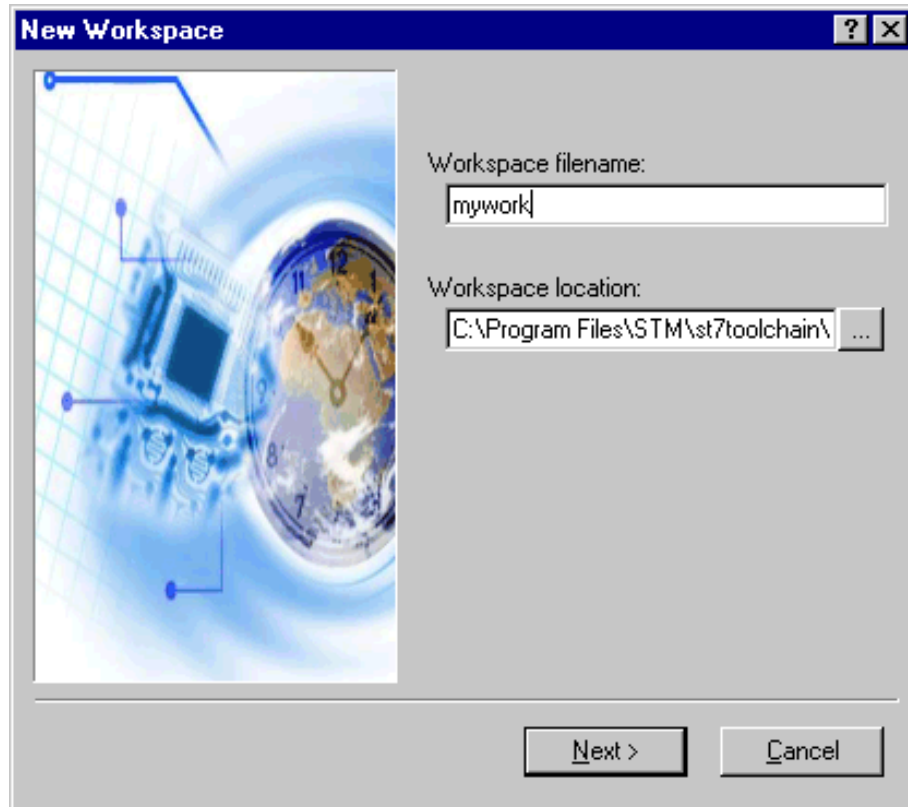
Toolchain:	ST Macroassembler	Hiware	Cosmic
Compile or Assemble Step:			
Source File Types	.asm	.c, .asm	.c, .s
Required Options	asm -li macrost7.asm	<NONE>	+debug
Resulting File Types	.obj, .lst	.o, .dbg	.o
Linker Step:			
Required Options	lyn macrost7.obj, macrost7 asm macrost7.asm -sym -fi=macrost7.map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

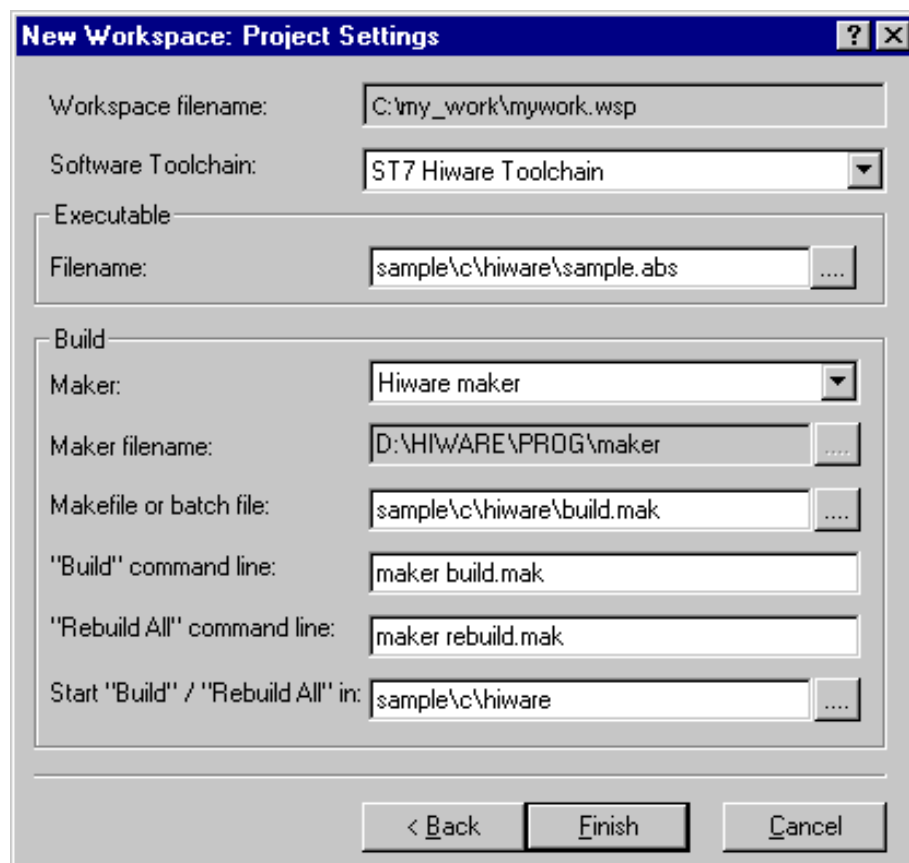
*Note: It is always preferable to have access to all of the files generated by the development toolchain. However, you can load *.s19 or *.hex binary files directly and have limited debugging capabilities (refer to [Section 3.8](#) on page 26).*

3.6 Creating a workspace


- 1 Select **File>New Workspace**. This command opens a new window where you define the name of your workspace and the directory in which you want to work.



- 2 Then, click **Next>**. The **New Workspace: Project Settings** dialog box appears:



Here you enter your software toolchain, your executable filename and your build parameters either by typing or using the drop boxes.

- 3 Select the toolchain and enter the name of your application's executable file. For example, if you wish to use the Hiware toolchain for ST7, your executable file will be of type *.abs (refer to [Table 2](#) on page 21)—click on the browse button  to browse to the folder where your executable file is saved and select it.
- 4 Next, choose the type of maker your application uses from the drop down list. In the example above, we have chosen the default Hiware maker, maker.exe. STVD7 will automatically look for this maker file in the folder you defined as the Hiware toolchain path.
- 5 Finally, you must define a make file or a batch file. There are several sample files provided with STVD7 (see [Table 1](#) on page 20). Here we have chosen

`build.mak` as the default make file, used when the **Build** command is issued, and `rebuild.mak` as the make file to use when the **Rebuild** command is issued.

- 6 After you have finished defining your project settings, click **Finish**.

Once the workspace is opened, the Workspace window displays its contents.

When you create a new workspace, the first time you switch to Debug context (see [Section 3.12](#) for an explanation of STVD7 contexts), the MCU Configuration window will automatically open to prompt you to choose your target MCU and confirm or modify its option and memory configuration (see [Section 3.13](#) on page 33).

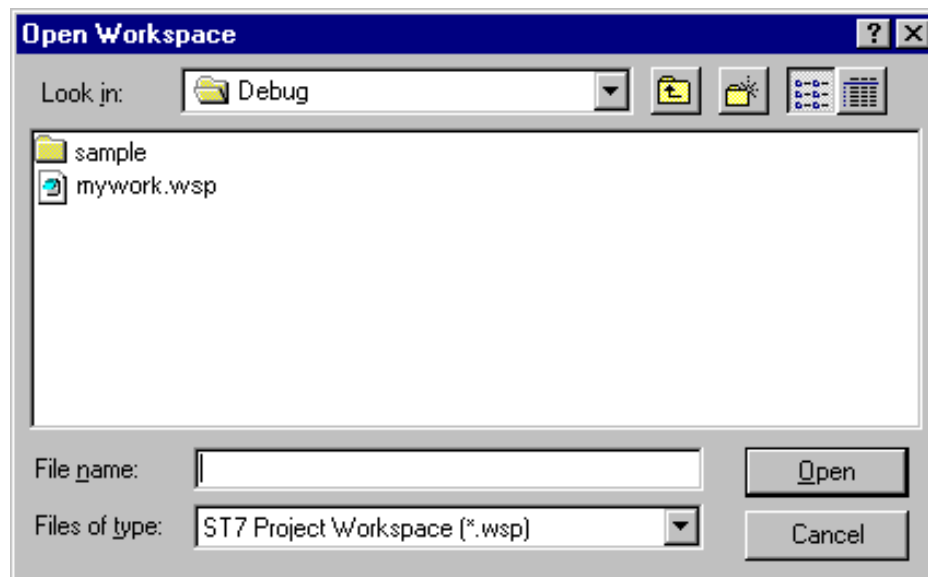
3.7 Opening an existing workspace

If you have already created a workspace, you simply need to open it in order to load all of your project settings into the STVD7.

Note: There are a number of sample workspaces provided with STVD7 that you can open to get familiar with STVD7. These samples are listed in [Table 1](#) on page 20.

- 1 From the main menu, select **File>Open Workspace**.

This command opens a window where you can browse to any folder you wish, and select an existing workspace.

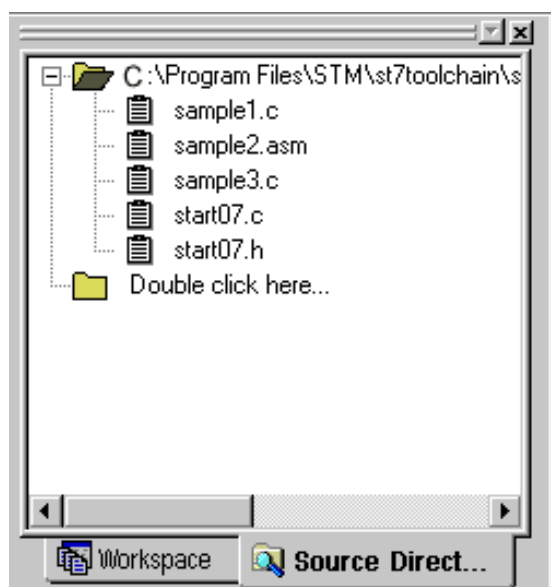
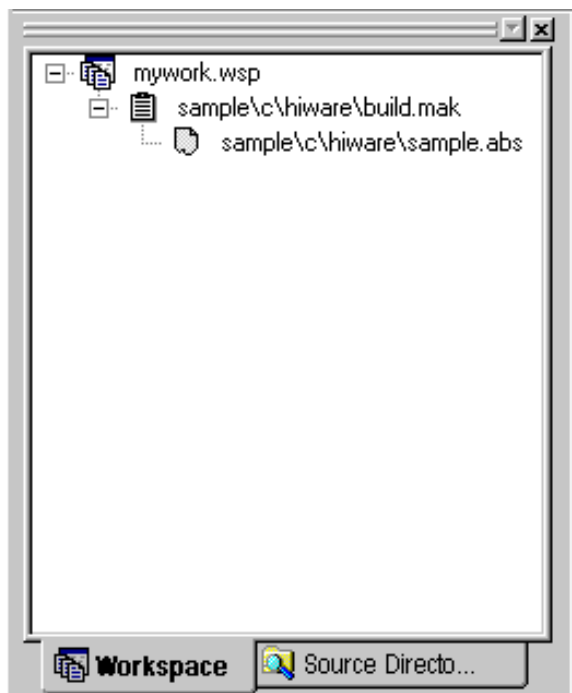


2 The Workspace window opens.

When a workspace is opened, all of the predefined project settings are loaded into the STVD7. The **Workspace** window will show a structured representation of the project. For example, `mywork.wsp` shows that it uses `build.mak` as the make file and `sample.abs` as the executable file.

Note: Although the name of the executable file is shown in the Workspace window, it has not yet been loaded into the emulation memory—see [page 26](#).

If you click on the **Source Directory** tab, the window will show every source and intermediate file type (*.c, *.s, *.asm, *.h or *.o) in the selected directory.



3 If there are no source files shown in the Source Directory tab of the Workspace window, or you wish to list additional files stored in another folder, you may browse to them by clicking the **Double Click here...** folder. The **Add Source Directory** window pops up allowing you to enter or browse for a new directory,