SGS-THOMSON MICROELECTRONICS

ST624x-KIT

STARTER KIT FOR ST624x MCU FAMILY

HARDWARE FEATURES

- Immediate evaluation of ST6240 with demonstration examples
- Program debugging within the user's real application environment
- On board programming of ST62E46 and ST62T46
- In-circuit programming of ST62E4x and ST62T4x devices on the user's application board

SOFTWARE FEATURES

- Software simulator including LCD display and I/O read/write
- Assembler, linker, debugger
- EPROM/OTP programming utilities
- Application examples

Table of Contents

Table of Contents

1 INTRODUCTION

The ST624x Starter Kit provides you with all you need to start designing, developing and evaluating programs for ST624x microcontrollers immediately.

The ST624x Starter Kit includes:

- The ST6 assembler and linker, AST6 and LST6.
- The ST6 Windows debugger, WGDB6.
- The Windows ST6 microcontroller programmer, Epromer.
- The ST6 Starter Kit board, which serves as a demonstration board and low-cost debugging tool.
- Some demonstration programs that show how ST6 microcontrollers use the Starter Kit board resources.
- Some example programs.
- Two ST62E46BF1 microcontrollers.
- A complete set of paper documentation and online help.

The demonstration programs, that come pre-loaded on the ST62T40B microcontroller, show how the powerful features of ST6 microcontrollers operate in a real environment. The demonstration programs use the hardware resources provided on the Starter Kit board, which include an LCD, hexadecimal keyboard, a resistance trimmer and an 8 MHz and a 32 KHz oscillator.

Using the ST6 assembler and linker, AST6 and LST6, you can assemble and link ST6 programs. The "ST6 Family Software development tools AST6, LST6, WGDB6" User Manual will guide you through the steps of developing, assembling and linking programs for the ST6. The Starter Kit software includes a set of example programs of typical ST6 applications. These are installed in the directory C:\st6tools\sk624Xi1\examples.

For a fast-track solution for developing bug-free programs for the ST6, without the hassle of writing assembler code, try out the ST6-Realizer program.

Once you have developed your ST6 program, you can use the Windows-based ST6 program debugger, WGDB6/SIMULATOR, together with the Starter Kit board, as a low-cost but powerful debugging tool. WGDB6 includes an ST6 simulator, that simulates the execution of your program, and uses the ST6 that is plugged into the Starter Kit board to emulate all transactions that are performed with the data space. Thus, using the Starter Kit board with WGDB6, you can view how the microcontroller peripherals behave when your program is executed. WGDB6 includes powerful debugging features, such as source-level debugging, instruction and conditional memory access breakpoints and selective trace recording. The "ST6 Family

Software development tools AST6, LST6, WGDB6" User Manual and online help will lead you through the debugging process using WGDB6.

When your program is ready, Epromer provides you with an easy-to-use Windows interface, which lets you prepare executable code, then write it to the ST62E46B microcontroller that can be plugged into the SDIP56 ZIF socket on the Starter Kit board, or your own in-circuit application board that is connected to the Starter Kit board.

The ST62T40B can't be erased or programmed, it is already programmed and contains the demonstrations program.

To observe and evaluate the consequences of your program on the resources it controls, you can run it on the ST62T40B microcontroller that is soldered on the Starter Kit board in Hardware Simulation mode. If it controls a resource that is not included on the Starter Kit board, you can connect your own resource to the board.

Instructions for use - Warning

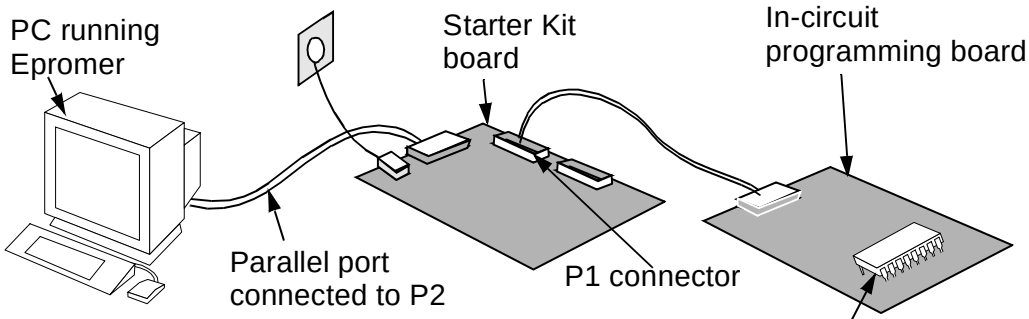
This product conforms with the 89/336/EEC directive; it also complies with the EN55022 emissions standard for ITE, as well as with generic 50082-1 immunity standards.

The product is a **Class A apparatus**. In a residential environment this device may cause radioelectrical disturbances which may require that the user adopt appropriate precautions.

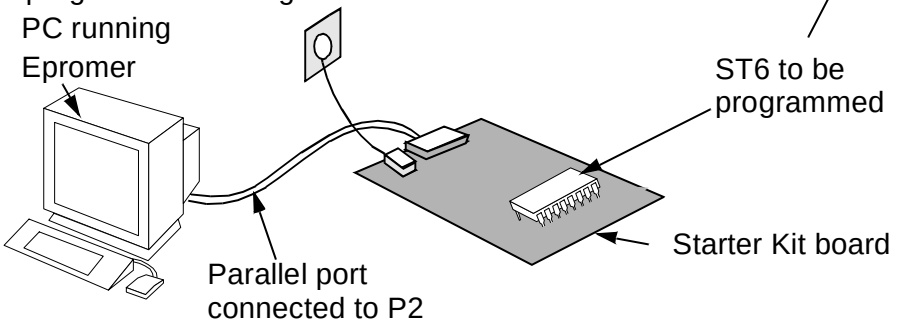
The product is not contained in an outer casing, and cannot therefore be immune against electrostatic discharge (ESD): **it should therefore only be handled at static safe work stations.**

The following diagram summarises the possible uses of the Starter Kit board and the hardware setup required for each one.

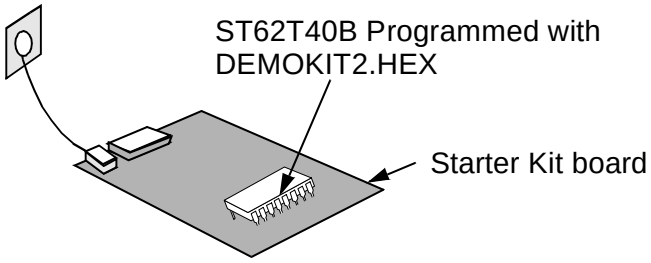
To program ST6s on your own in-circuit programming board:



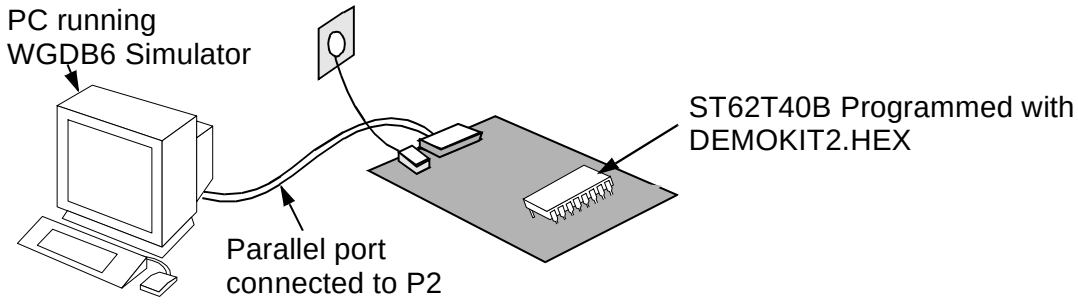
To program ST6s using the Starter Kit board:



To run the demonstrations:



To use the Starter Kit board as a hardware simulator:



1.1 Where to go from here...

The following table directs you to where you should look for further information about using the ST6 Starter Kit

To:	Refer to:
Find out about the Starter Kit board and ST6 microcontrollers provided with the kit.	“The Starter Kit Hardware” on page 8 of this book.
Install the Starter Kit software, and connect the power supply to the board.	“Installing the Starter Kit” on page 15 of this book.
Find out what the demonstration applications do, and run them.	“Running the Demos” on page 17 of this book.
Learn how to develop source code for AST6 and LST6.	"ST6 Family Software development tools AST6, LST6, WGDB6" User Manual.
Prepare the Starter Kit board for use as an ST6 hardware simulator with WGDB6.	“Using The Starter Kit Board as a Hardware Simulator” on page 22 of this book.
Learn how to use WGDB6 for debugging your programs.	"ST6 Family Software development tools AST6, LST6, WGDB6" User Manual.
Prepare the Starter Kit board for programming ST6 microcontrollers using Epromer.	“Programming ST6 Microcontrollers” on page 32 of this book.
Prepare the Starter Kit board for connecting your own in-circuit programming board.	“In-Circuit Programming” on page 33 of this book.
Learn how to use Eprommer for programming ST6 microcontrollers.	The Epromer online help.
Connect your own hardware resource or LCD to the Starter Kit board.	“Connecting External Resources to the Starter Kit Board” on page 19 of this book.
Perform some introductory excercises using WGDB6.	“Exercises” on page 26.
Learn how the LCD interface works.	“LCD Interface” on page 36.

2 THE STARTER KIT HARDWARE

This section describes the ST6 microcontrollers and the Starter Kit board that come with the ST6 Starter Kit. A full schematic of the Starter Kit board is provided in “Hardware Information” on page 47.

2.1 The ST6 Microcontroller

The Starter Kit includes two ST62E46BF1 microcontrollers.

The ST62T40B microcontroller is pre-loaded with the code **DEMOKIT2.HEX**, which includes the demonstration programs (see “Running the Demos” on page 17), as well as the communications protocol program, that enables you to use the Starter Kit board as a simulator (see “Using The Starter Kit Board as a Hardware Simulator” on page 22).

2.2 The Starter Kit Board

The Starter Kit board includes the following resources:

- A Reset button.
- An 8-alphanumeric digit LCD.
- A hexadecimal keyboard.
- A LED indicator.
- A resistance trimmer.
- One 8 MHz and one 32 KHz oscillator.
- A SDIP56 ZIF socket to program the ST62E46B or ST62T46B.

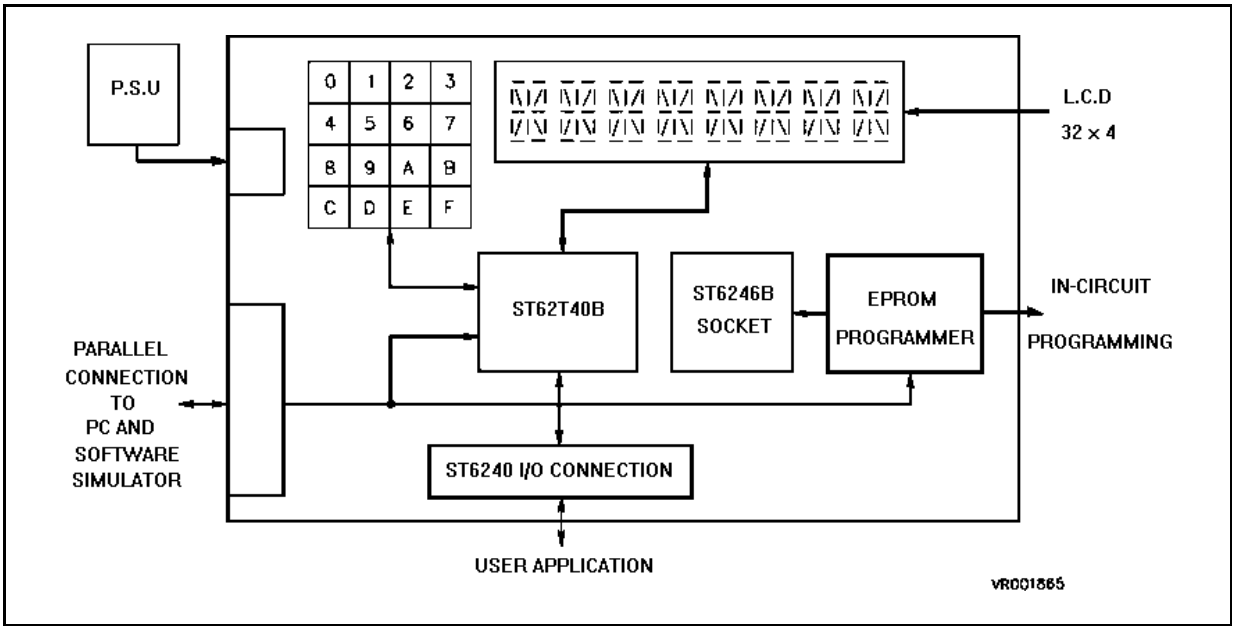
It comes with its own power supply unit that can be plugged into an AC mains source, or a DC source with the following characteristics:

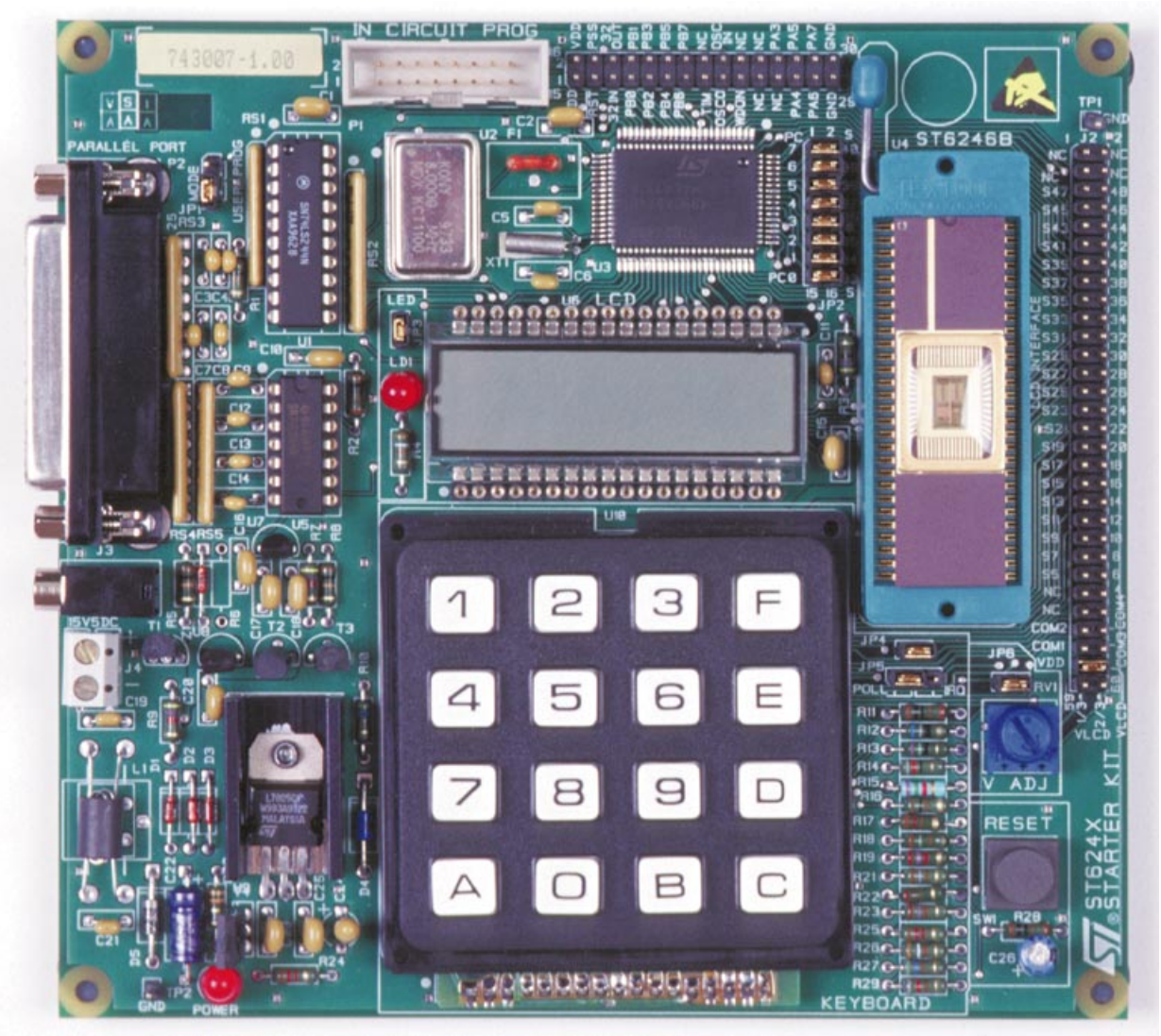
- Voltage: 16V min./20V max.
- Current: 100 mA min.

It includes the following connectors:

- A parallel port connector (P2) for connection to the host PC when it is used as a hardware simulator or for programming.
- A remote resource I/O interface (J1).
- An in-circuit ST6 programming board connector (P1).
- A remote LCD connector (J2) to which you can connect your own LCD.

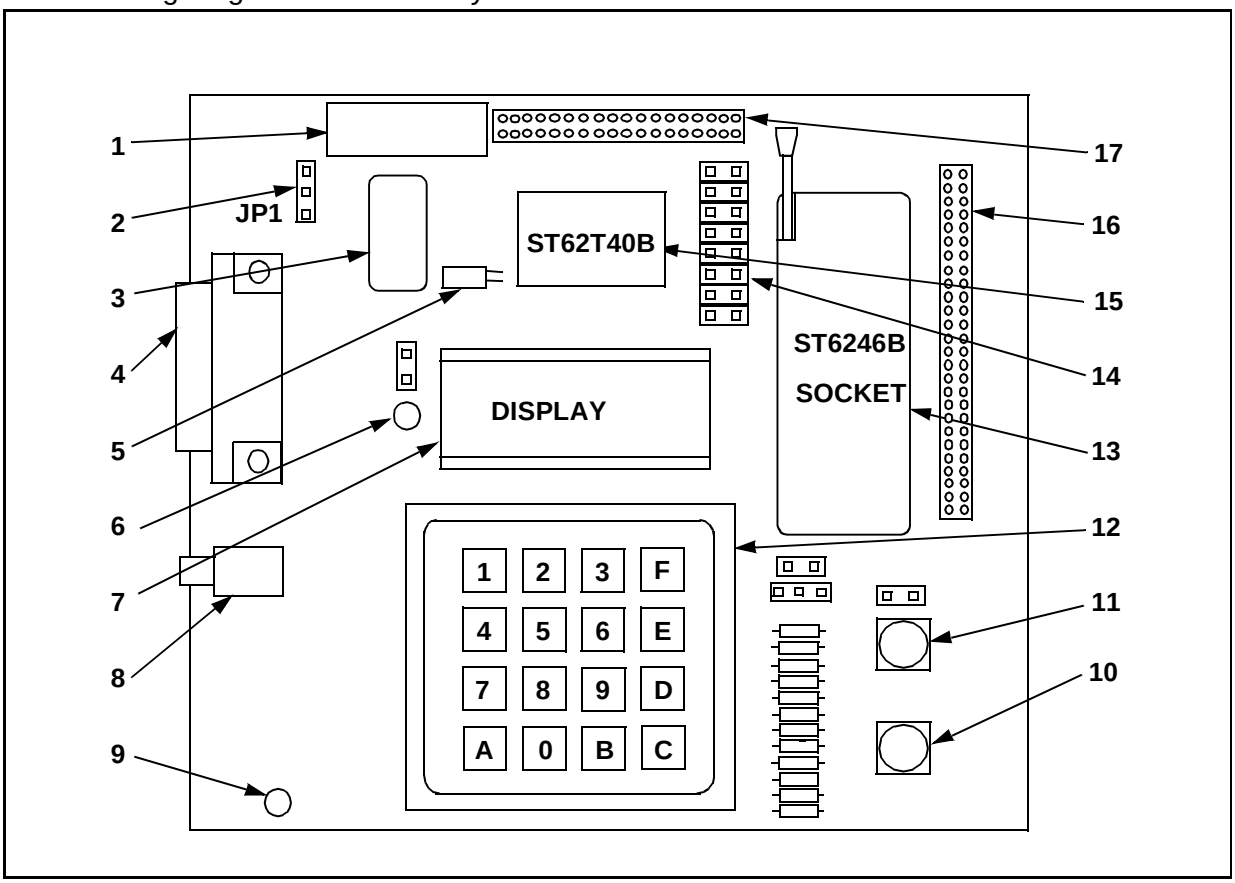
Below is a block diagram of the Starter Kit board:





ST624x Starter Kit Board

The following diagram shows the layout of the Starter Kit board.



1	In-circuit programming connector P1.	17	Remote resource I/O interface J1.
2	“Programming” or “User” operating mode selection jumper JP1.	16	Remote LCD interface connector J2.
3	8 Mhz oscillator.	15	ST62T40B MCU
4	PC connector P2.	14	LCD protection with jumper JP2 if the combi-port PC0-7 is used.
5	32.768 KHz oscillator.	13	SDIP56 ZIF MCU socket.
6	LED indicator LD1.	12	Keyboard
7	LCD display.	11	Voltage trimmer
8	Power supply JACK connector J3.	10	RESET button.
9	Power supply LED indicator LD2.		

2.3 8 MHz and 32 KHz Oscillators

An oscillator feeds the ST62T40B OSCIN input with an 8 MHz clock signal.

A 32 KHz oscillator is delivered with the board. The required components: crystal XT1 and capacitors C5, C6 are connected to the ST62T40B as described in the ST6240 Data Book.

2.4 8-alphanumeric Digit LCD

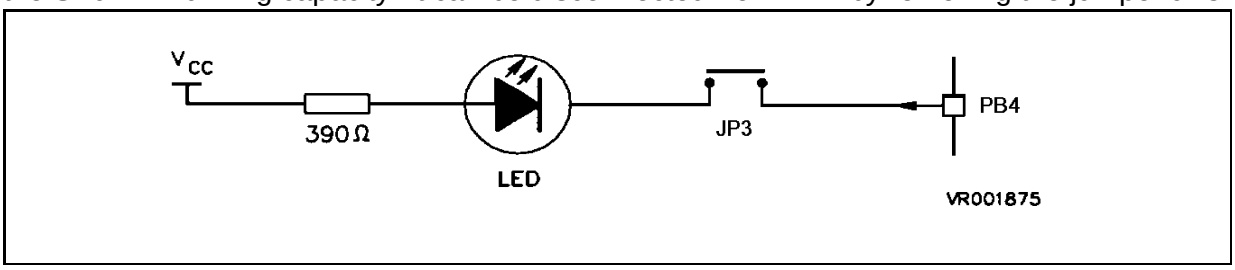
An 8-alphanumeric digit LCD is connected to the ST62T40B LCD driver outputs. It has 32 segments that are driven by 4 COM outputs. You can disconnect a part of the LCD by removing jumper JP2 if you want to use the combi-port PC0-7 on the connector interface J2. For full details about the LCD see “LCD Interface” on page 36.

2.5 Reset Button

This activates the ST62T40B RESET input when pressed. A power-on reset circuit is also provided.

2.6 LED Indicator

A LED is connected to the ST62T40B PB4 I/O pin (which is defined as output) to demonstrate the ST6 LED-driving capacity. It can be disconnected from PB4 by removing the jumper JP3.



2.7 Hexadecimal Keyboard

A hexadecimal keyboard is connected to the ST62T40B PB0 I/O pin (defined as A/D Converter input), via an analog interface resistor array.

The voltage value on the A/D converter input is equal to 5V/16 x the key number, thus giving an image of the pressed key.

The following table lists the Resistor array values and their corresponding voltage/key values:

Resistor Array Values	Theoretical Voltage Values
RT: 1KΩ	NO KEY: 5V
R0: 68Ω	K0: 0V
R1: 75Ω	K1: 0.312V
R2: 82Ω	K2: 0.625V
R3: 100Ω	K3: 0.937V
R4: 120Ω	K4: 1.250V
R5: 150Ω	K5: 1.562V
R6: 180Ω	K6: 1.875V
R7: 220Ω	K7: 2.187V
R8: 270Ω	K8: 2.500V
R9: 390Ω	K9: 2.812V
R10: 560Ω	KA: 3.125V
R11: 820Ω	KB: 3.437V
R12: 1.2KΩ	KC: 3.750V
R13: 2.7KΩ	KD: 4.062V
R14: 7.5KΩ	KE: 4.375V
	KF: 4.687V

Jumper JP5 sets the keyboard operation mode: polling or IRQ, according to the following diagram.

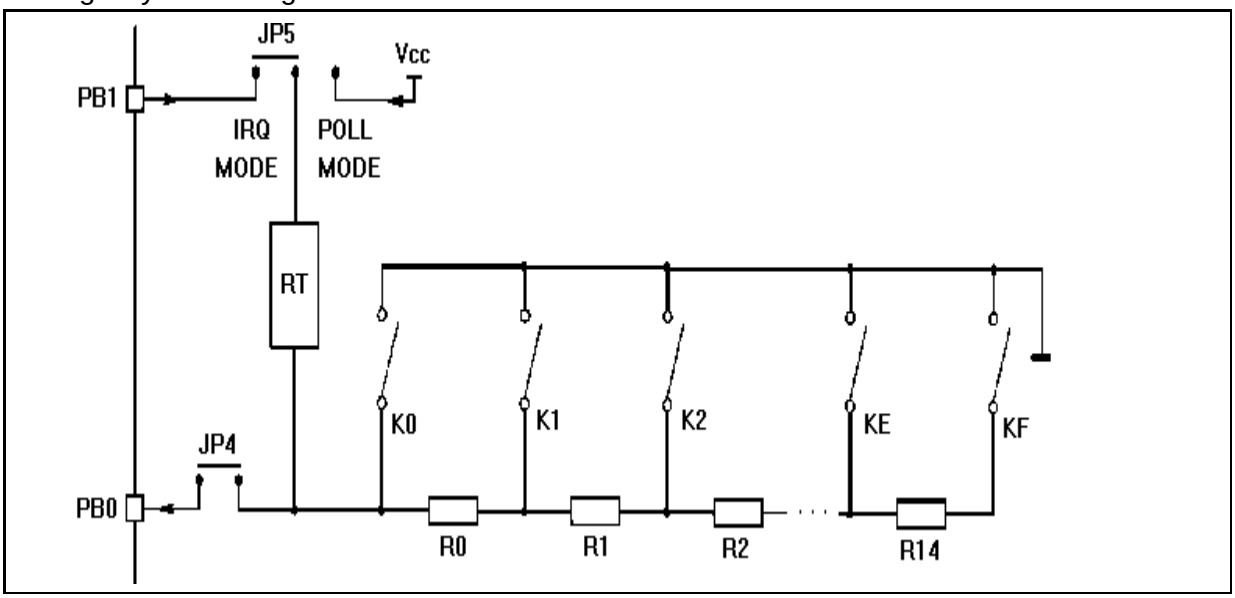
This position selects polling.

This position selects IRQ.

Jumper JP4 disconnects the keyboard output from PB0 when it is removed, enabling you to connect your own external source to PB0 via the J1 connector.

For an example of the analog keyboard application, refer to the SGS-Thomson application note AN431: Using ST6 Analog Inputs for Multiple Key decoding.

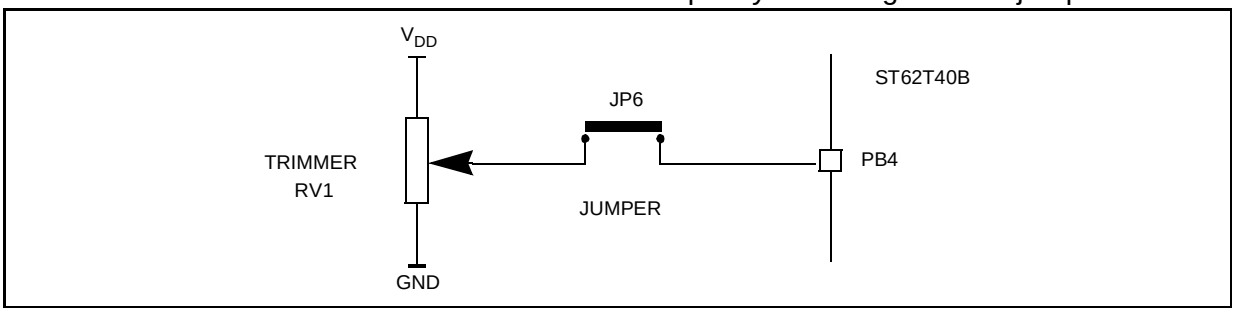
Analog Keyboard diagram:



2.8 Resistance trimmer

A 10 KΩ resistance trimmer feeds the ST62T40B PA4 I/O pin (when programmed as an A/D Converter input) with a variable voltage (0 to 5V DC). It is used for A/D conversion demonstration/evaluation.

The trimmer can be disconnected from the PA4 I/O pin by removing the JP6 jumper.



2.9 Combi-ports PC0-7

The port C of the ST6240B is used for the LCD segments. It can be used as a normal port in hardware simulation mode.

The port PC0-7 can be accessed on J2, prior to use it, remove the 8 jumpers on JP2 (marked 14 on the Starter Kit board diagram on page 11). Thus, the LCD won't be affected by the inputs on the port PC0-7 and won't be damaged.

3 INSTALLING THE STARTER KIT

3.1 Hardware and Software Requirements

To be able to install and run the ST6 Starter Kit, you need a PC with:

- A 3 1/2” Floppy Disk Drive
- A free Centronics compatible parallel port connector
- MS-Windows™ 3.11, NT or 95.
- A CD-ROM Disk Drive

3.2 Installing the Software

If diskettes are provided, you must install the software with them in order to have the latest release:

1 Place the SK624XI1 diskette into your floppy disk drive.

2 In Windows Explorer or File Manager, view the contents of the diskette, then double-click the **Setup** file or icon.

3 Follow the instructions as they appear on screen.

If only the ST62 CDROM is provided, then:

1 Place the ST62 CDROM provided into your CDROM disk drive.

2 In Windows Explorer or File Manager, view the contents of the CDROM, browse to st62oncd\ftools\sk624Xi1 and double-click the **Setup** file or icon.

3 Follow the instructions as they appear on screen.

3.3 Connecting the Power Supply

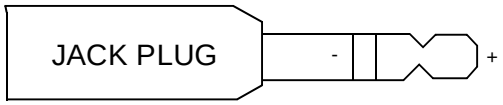
If you have AC mains supply, connect the Jack plug on the power supply cable provided to the J3 input socket, then connect the mains plug to a mains source.

If you have DC mains supply, connect the male plug on the power supply cable provided to the J3 input socket, then connect the mains plug to a mains source with the following characteristics:

- Voltage: 16V min./20V max.
- Current: 100 mA min.

To avoid a short circuit, always connect the power input cable to the starter kit board before connecting it to a mains power supply.

If you use your own 3.5 mm power supply plug, its polarity must be as follows:



4 RUNNING THE DEMOS

This section describes the demonstration programs that are provided with the Starter Kit and explains how to run them.

4.1 What the Demos Do

The following paragraphs describe the demos that come pre-loaded with the ST6 Starter Kit demos. See “Running the Demonstration Programs” on page 18 below for details on how to select and run a demo.

The source files of these demos are provided with the Starter Kit software in the file **C:\st6tools\sk624Xi1\sk624Xli\DEMOKIT2.ASM.**

4.1.1 Demo 1 - Key Display

1 Initialises the pin PB0 (which is connected to the keyboard) as an analog input.

2 Polls the A/D converter to detect whether a key is pressed.

3 When a key is pressed, it shifts the contents of the LCD one place to the left and displays the value of the pressed key on the right side of the LCD.

To stop the demonstration, press the Reset button. To quit the demonstration routine and by-pass the presentation message, press and release the Reset button while pressing any key on the Starter Kit keyboard.

4.1.2 Demo 2 - Voltmeter

1 Initialises the pins as follows:

This pin:	Is initialised as:
PA4	Analog input. Connected to the trimmer RV1.
PB4	20 mA direct LED drive output.

2 Reads digital value of the voltage present on PA4 from the A/D converter data register.

3 Displays the read voltage value on the LCD panel. When the read value reaches 4 volts, the LD1 LED is switched on indicating that the voltage value is reaching its upper limit.

You can adjust the voltage value using the voltage trimmer (marked 11 on the Starter Kit board diagram on page 11).
To stop the demonstration, press the Reset button. To quit the demonstration routine and by-pass the presentation message, press and release the Reset button while pressing any key on the Starter Kit keyboard.
The LED flickers when around 4 volts is reached. This is because of the power supply voltage noise >20 mV or instability of the resistor trimmer.

4.2 Running the Demonstration Programs

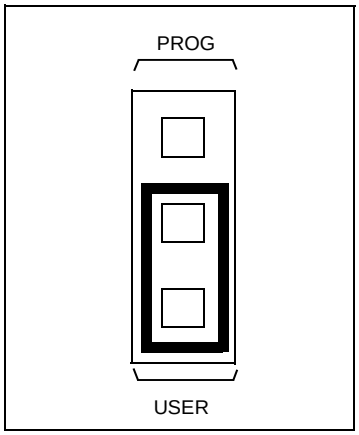
The ST62T40B microcontroller labelled DEMOKIT2 is programmed with the demonstration software.

To run the demonstrations:

1 Power down the Starter Kit board.

2 Make sure that the ST62E46B is not plugged into the SDIP56 ZIF MCU socket.

3 Select the USER mode using the jumpers marked JP1 (marked 2 on the Starter Kit board diagram on page 11), as shown in the diagram below:



4 Disconnect the cable from the parallel port (P2) connection, if it is connected.

5 Power up the Starter Kit board.

6 Press and release the Reset button on the Starter Kit board. When the message “Press key 1\2” is displayed on the LCD, press either 1 to run the Keyboard demonstration or 2 to run the Voltmeter demo.

To stop the current demonstration and view the other demonstration, repeat step 6 above. Or, to avoid the display of the presentation message, press any key on the Starter Kit keyboard then, while keeping the key pressed, press and release the Reset button.

5 CONNECTING EXTERNAL RESOURCES TO THE STARTER KIT BOARD

You can connect your own external resources to the pre-programmed ST62T40B to debug or evaluate your programs, using the connector J1 (marked 17 on the Starter Kit board diagram on page 11).

You can also connect your own LCD to the ST62T40B LCD driver using the connector J2 (marked 16 on the Starter Kit board diagram on page 11).

To be able to use either of these connectors, you must disconnect the resources that are already connected to the ST62T40B, to avoid external resource/Starter Kit board resource conflicts.

The following tables list the Starter Kit board resources and the corresponding J1 and J2 connections, and indicates the jumper that disconnects each resource

Table 1. J1 User's I/O Interface Connector

Signal	Pin Number	Pin Number	Signal
5V supply (1)	1	2	5V supply (1)
RESET	3	4	PSS
OSC32in (2)	5	6	OSC32out (2)
PB0 (3)	7	8	PB1 (4)
PB2	9	10	PB3
PB4 (5)	11	12	PB5
PB6	13	14	PB7
TIMER	15	16	NMI
OSCout	17	18	OSCin
WDON	19	20	Unused
Unused	21	22	Unused
Unused	23	24	PA3
PA4 (6)	25	26	PA5 (7)
PA6 (7)	27	28	PA7 (7)
GND	29	30	GND

NOTE:

1. The 5V supply is available up to 200mA max current.
2. 32KHz oscillator is mounted on PCB at XT1, C5 and C6 locations.

3. PB0 is connected to Analog Keyboard Array output. It can be disconnected by removing jumper JP4.
4. PB1 may be connected to Analog Keyboard reference voltage input (for interrupt mode). It can be disconnected by removing jumper JP5.
5. PB4 is connected to LED LD1. It can be disconnected by removing jumper JP3.
6. PA4 is connected to the trimmer RV1. It can be disconnected by removing jumper JP6.
7. PA5, PA6, PA7 are used to perform data transfer in programming mode (PROG configuration). In case these signals are connected to external user's sources, those sources must be disconnected (or set to High Z state) during programming operations.
   PA0, PA1, PA2 are used to perform data transfer with PC simulator software. They are not available for external usage in this operating mode. Thus, they are not present on J1 connector.

Table 2. J2 User's LCD Interface Connector

Signal	Pin Number	Pin Number	Signal
Unused	1	2	Unused
Unused	3	4	Unused
S47	5	6	S48
S45	7	8	S46
S43	9	10	S44
S41	11	12	S42
S39/PC6 (1)	13	14	S40/PC7 (1)
S37PC4 (1)	15	16	S38/PC5 (1)
S35PC2 (1)	17	18	S36/PC3 (1)
S33/PC0 (1)	19	20	S34/PC1 (1)
S31	21	22	S32
S29	23	24	S30
S27	25	26	S28
S25	27	28	S26
S23	29	30	S24
S21	31	32	S22
S19	33	34	S20
S17	35	36	S18
S15	37	38	S16
S13	39	40	S14
S11	41	42	S12

Signal	Pin Number	Pin Number	Signal
S9	43	44	S10
S7	45	46	S8
S5	47	48	S6
Unused	49	50	S4
Unused	51	52	Unused
COM2	53	54	COM4
COM1	55	56	COM3
5V (2)	57	58	VLCD (2)
VLCD1/3 (3)	59	60	VLCD2/3 (3)

NOTE:

1. The combi-port PC0-7 can be accessed on J2. It is normally connected to the LCD pins but, however, these LCD pins can be disconnected with JP2 jumper in order to use the combi-port. After Reset, these pins are configured automatically as LCD segment.
2. The VLCD input pin is connected to 5V supply voltage through a jumper between pins 57 and 58. To connect another voltage source, first remove this jumper and then feed pin 58 with the new VLCD voltage.
3. For the Starter Kit LCD, VLCD1/3 and VLCD2/3 are generated by the ST62T40B internal resistor divider. If needed, an external divider can be connected to these two pins as described in ST6240 data book, especially if VLCD < 4.5V.

6 USING THE STARTER KIT BOARD AS A HARDWARE SIMULATOR

WGDB6, the ST6 debugger that runs under Windows, lets you test your programs without having to program the EPROM of your target ST6. Depending how much information you want, and how close to real life you want your test environment to be, you can use WGDB6 in one of three ways:

- As a software simulator. If you use WGDB6 as a simulator, you need not attach any additional hardware to your PC. The ST6 simulator program, that comes with WGDB6 and is run when you run WGDB6/Simulator, simulates the execution of your program, letting you step through the code and see what happens as the program runs. WGDB6 simulator includes Wave Form Editor, which simulates the output of the pins on your target ST6 in relation to inputs that you define, enabling you to see how its peripherals react to the inupts they receive.
- With an ST6 hardware emulator. Emulators are hardware systems that act as your target microcontroller, at the same time capturing detailed information, such as which areas of memory are accessed by the program and what happens when they are accessed. In this case, WGDB6/Emulator provides an interface between the emulator and your PC, displaying data captured by the emulator and letting you implement the WGDB6 features in the emulator, such as software or hardware breakpoints.
- With the Starter Kit board as a hardware simulator. This is a cross between the above two. The WGDB6 software simulator simulates the execution of your program, but each time the data space is accessed, it accesses that of the ST6 that is plugged into your Starter Kit board. Thus, using the Starter Kit board with WGDB6, you can view how the real microcontroller peripherals behave when your program is executed.

This section describes the third option, how to use the Starter Kit board as a hardware simulator.

You can use the Starter Kit board to emulate any ST624x microcontroller.

When simulating programs designed for other microcontrollers, make sure that you do not use resources that are not available on the microcontroller your application is designed for.

6.1 The Data Transmission Driver

Data is transferred between the simulated peripheral registers and the ST62T40B registers via the host PC's parallel port. The DEMOKIT2.HEX program, which is in the ST62T40B microcontroller that is soldered on the Starter Kit board includes the transmission driver.

The data transfer driver uses the following pins:

PC parallel port	ST62T40B	SIMULATOR USAGE
D2	PA2	Synchronisation
D3	PA0	Write data to MCU
D4	RESET	Hardware reset of peripherals
D6	NMI	Initiates data transfer
SDOP	PA1	Read data from MCU

Note: Do not connect any external resources to the corresponding J1 connector pins when using the Starter Kit board as a peripheral emulator.

6.2 Technical Limitations

The Starter Kit board has the following limitations when used with WGDB6 as a hardware simulator:

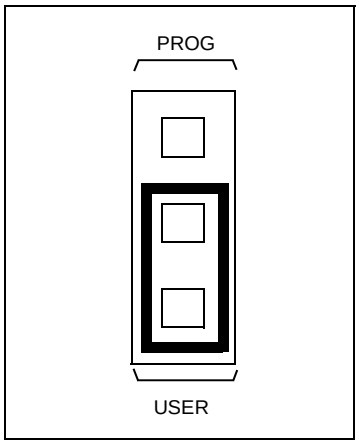
- Real-time program execution is not supported. The program execution speed depends on your PC.
- Resetting the ST62T40B by power up, pressing the Reset button or external reset does not reset the simulated ST6 core. To perform a complete simulated reset, use the WGDB6 reset command instead.
- Interrupts sent by the ST62T40B microcontroller are not supported by the WGDB6 simulator.
- The pins: NMI, PA0, PA1 and PA2 on the ST62T40B microcontroller are used for communications with the host PC, and are thus not available for simulation.

To use the Starter Kit board as a hardware simulator:

1 Power down the Starter Kit board.

2 Make sure that the ST62E46BF1 is not plugged into the SDIP56 ZIF MCU socket.

3 Select the USER mode using the jumper JP1 (marked 2 on the Starter Kit board diagram on page 11), as shown in the diagram below:



4 Connect the Parallel port P2 on the Starter Kit board to a spare parallel port on your PC using the cable provided with the Starter Kit.

5 Power up the Starter Kit board.

To run WGDB6:

- If you are using Windows 95, click the **Start** button, point to **Programs**, then **ST6 Tools**, then click **WGDB6/Simulator**.
- If you are using Windows 3.x, double-click the appropriate **WGDB6/Simulator** icon in the **ST6 Tools** program group.

Refer to "WGDB6 User Guide" in the "ST6 Family Software Development Tools AST6, LST6, WGDB6" User Manual for full instructions on how to use WGDB6.

6.3 Error Messages

The following table lists the error messages you may encounter when using WGDB6 with the Starter Kit board:

Error message	Description
Error 116 Port A protected when using board.	This means that WGDB6 tried to access the PORT A registers. These are used for communications with the board.
Error 117 Communication error with ST624x board.	This means that a problem occurred during communications between the host PC and the board. Perform the checks listed below.

6.4 Troubleshooting

If there is a communication problem between WGDB6 and the Starter kit board, the title “WGDB6 Simulator” appears in the WGDB6 title bar. In this case, you shoud check the following:

- That the Starter Kit board is correctly powered up.
- That the parallel port cable is correctly connected.
- That the device jumper JP1 is in the USER position.
- That no ST62E46B is plugged into the Starter Kit board.

7 EXERCISES

This section describes two exercises, in which you use your ST6 Starter Kit board as a hardware simulator with WGDB6:

- In the first excercise, you're going to learn how to use WGDB6 to reset the LCD on your Starter Kit board then display the letter A on it.
- In the second excercise, you're going to use WGDB6 to step through the instructions that display messages on the keyboard.

7.1 Exercise 1

1 Connect your Starter Kit board as a Hardware Simulator following the instructions given in "Using The Starter Kit Board as a Hardware Simulator" on page 22.

Press the Reset button on the Starter Kit.

2 Run WGDB6 Simulator:

If you are using Windows 95, click the **Start** button, point to **Programs**, then **ST6 Tools**, then click **WGDB6/Simulator**.

If you are using Windows 3.x, double-click the appropriate **WGDB6/Simulator** icon in the **ST6 Tools** program group.

The message 'REMOTE' is displayed on the LCD of the Starter Kit board.

The WGDB6 main window opens:

3 Load the file **Stktrain.hex**:

i In WGDB6, on the **File** menu, click **Open**.

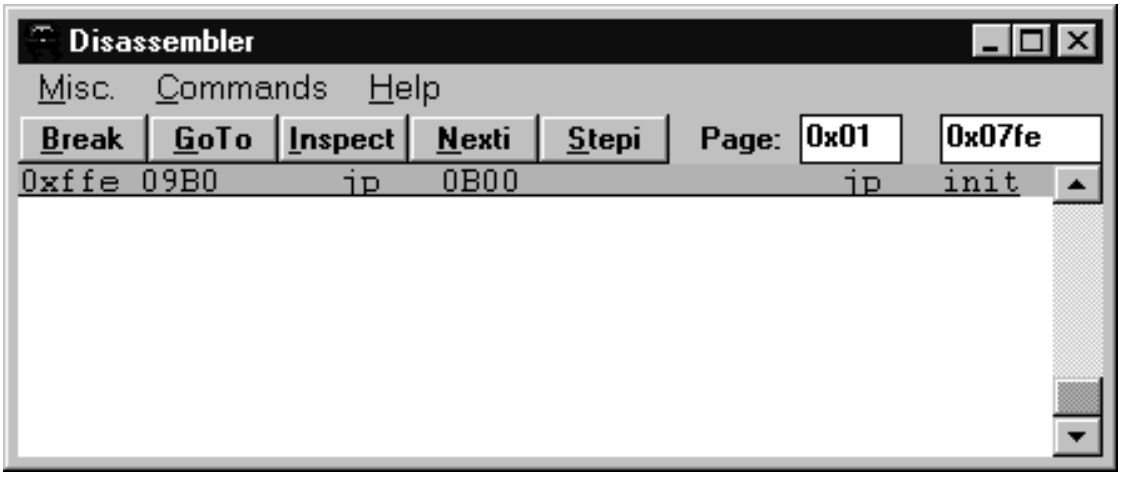
ii Browse to the directory c:\st6tools\sk624Xi1

iii Select the file **stktrain.hex**, then click **OK**.

4 The Disassembler window now opens, displaying the source code of **stktrain.hex**.

5 Reset the simulated program: on the **Commands** menu, click **Reset**.

The disassembler window opens, displaying the line **0xffe**, which is set by the reset vector.

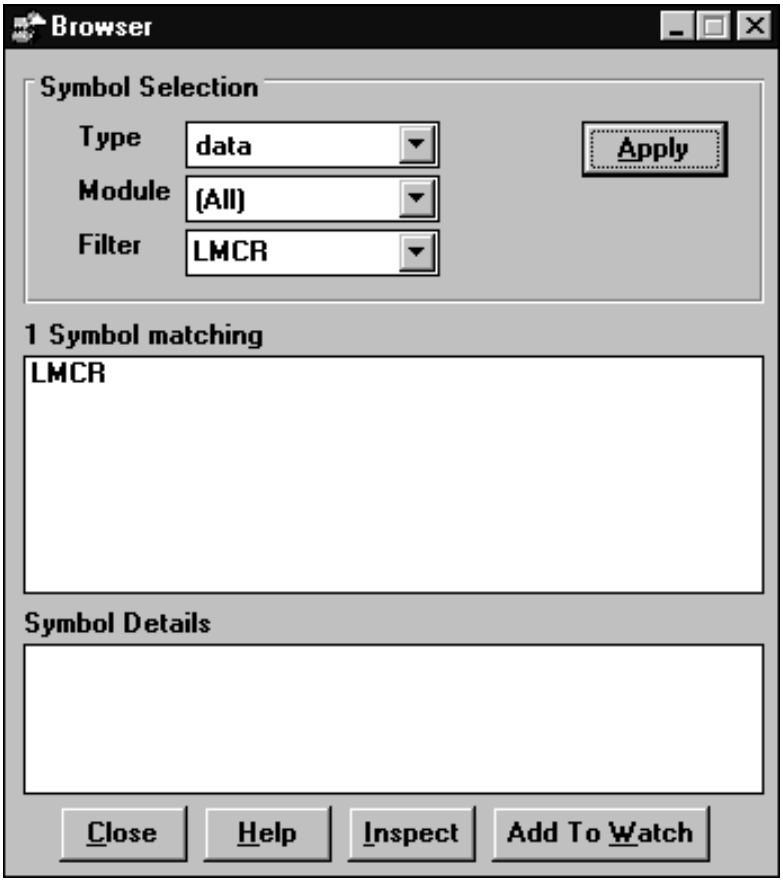


The LCD panel on the Starter Kit board is now clear, indicating that the WGDB6 simulator performed a physical reset of the ST62T40B plugged into the starter kit board. When the reset was performed, all segment outputs were set to OFF by setting the HFE bits 2, 1 and 0 of the LMCR register to 0. The LMCR register sets the LCD control mode. For full details about the LMCR, see the ST6240 Data Book.

You are now going to locate the register LMCR, and change its value:

1 In the WGDB6 main window, on the **Windows** menu, click **Browser**.

The Browser dialog box opens:



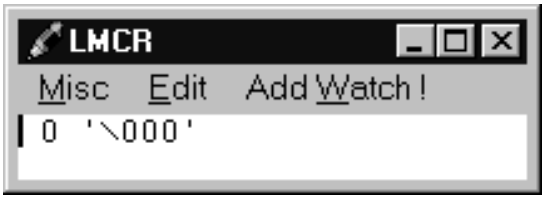
2 From the **Type** drop down list, choose data, as shown above (LMCR is a data address).

3 In the **Filter** field, type LMCR, as shown above.

4 Click **Apply**. LMCR is now displayed in the **Symbols** matching box, as shown above.

5 Select LMCR in the **Symbols matching** box, then click **Inspect**.

The Inspect window opens, displaying the value of LMCR (which is 0):



You are now going to change the value of LMCR to 36:

1 In the Inspect window, select the value '0', then on the **Edit** menu, click **Modify**, and overtype the value 0 with 36, and click **Set**. The contents of the inspect window are now 36 '$', indicating that the new LMCR value is 36. The value 36 corresponds to the following configuration:

bits DS1-0 = hold the values 0 0, defining 1/4 duty.

bits HF2-0 = hold the values 1 1 0, setting %256 main osc.

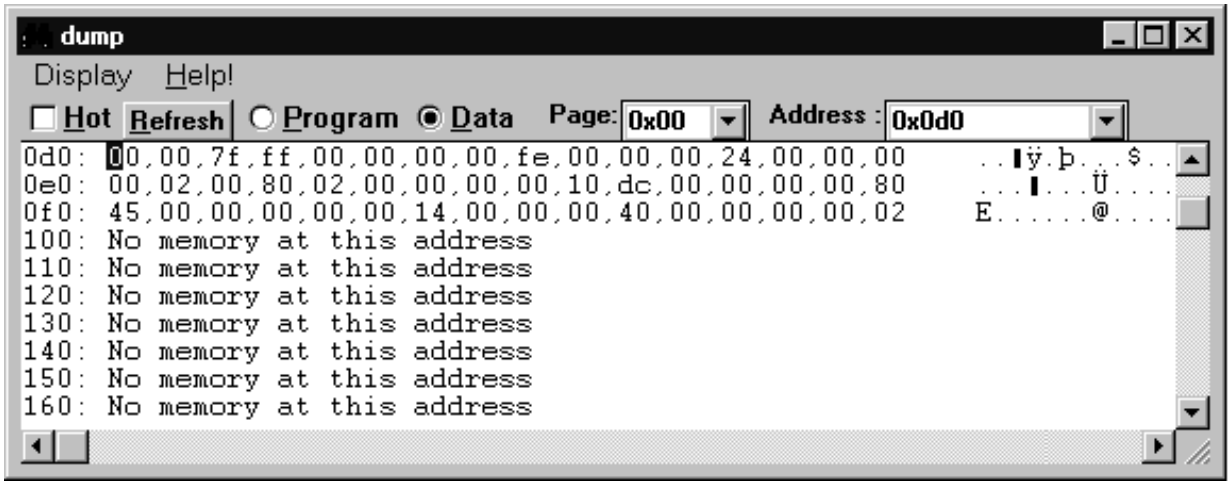
bits LF2-1 = hold the values 1 1 0, setting the LCD frequency to 512 Hz.

2 Look at the LCD on the Starter Kit board, it now displays the message 'REMOTE'.

(If you already used the program, the message may not be 'REMOTE'. It depends on what is in the memory)

You are now going to manually build the character 'A' that will be displayed on the LCD:

1 In the **Inspect** window, on the **Edit** menu, click **Dump**. The dump window opens, click on 'Data' to display the contents of the data space starting at the address 0d0.



2 In the **Address** field in the Dump window, type the address 0e0, then press the Enter key on your keyboard. This moves the cursor to the beginning of the LCD RAM area. Observe the values from 0E0 to 0F7: these correspond to the letters 'REMOTE' that are displayed on the LCD. Overtype all these values with the value '00', by repeatedly pressing the 0 key on your keyboard: you will note that all LCD panel segments are turned off.

3 To build the letter A, in the Dump window overtype the values 00 with the following values:

At this address:	Which corresponds to this value:	Type this value:
0e1	COM1	02
0e7	COM2	0e
0ed	COM3	07
0f3	COM4	00

Look at the LCD: the character ‘A’ is now displayed at digit 0. Now try writing some of your own characters, using the information provided in “LCD Interface” on page 36.

7.2 Exercise 2

1 With **stktrain.hex** still loaded in WGDB6, in the WGDB6 main window, on the **Commands** menu, click **Reset** to reset the program.

The Disassembler window opens, with the line `jp init` highlighted: this is the line pointed to by the reset vector.

2 In the WGDB6 main window, click the **Cont** button. The program counter jumps to the init address, which is the beginning of the program.

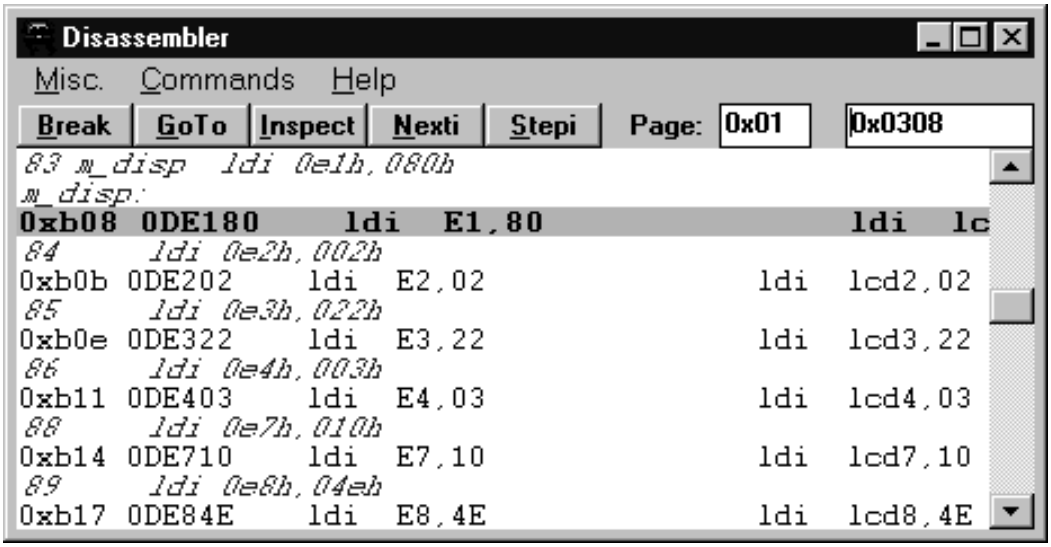
Look at the Starter Kit LCD: it is first cleared, then the word ‘DISPLAY’, followed by ‘KEY’ is displayed on it. Press some keys on the starter kit keyboard, and note the result: the keys you press are displayed on the right side LCD. Bear in mind that the program is being simulated, and thus is not running in real time mode, so you may have to keep keys pressed for several seconds until they are displayed on the LCD.

3 In the WGDB6 main window, click the **Stop** button.

4 In the Disassembler window, type `m_disp` in the top right field, next to the **Page** field, and press the Enter key on your PC keyboard.

The disassembled code around the address `m_disp` is displayed, with the `m_disp` address highlighted. Click the **Break** button in the Disassembler window to set a breakpoint

on the `m_disp` address: program execution will now stop when the PC reaches this address.



5 In the Disassembler window, on the **Commands** menu, click **Reset**.

6 Type `init` in the top right field, next to the **Page** field, and press the Enter key on your PC keyboard. The program counter now jumps to the beginning of the program.

7 On the **Commands** menu, click **Continue**. This continues running the program, until the breakpoint you placed on the line of `m_disp` is reached. Wait for program execution to stop.

8 Click the **Next** button to execute the next line of the program, and look at the LCD : it displays one segment, that has been turned on by the instruction: `ldi E1, 80`. Keep pressing the **Next** button to execute each instruction in turn, watching the effects each one has on the LCD. You will notice that each time you press the **Next** button, a few more segments are turned on, until the word ‘DISPLAY’ appears on the LCD.

Now that you are familiar with WGDB6, try stepping through the program yourself, observing the effects each command has on the LCD, and modifying data to see the effect it has on the LCD.

8 PROGRAMMING ST6 MICROCONTROLLERS

You can use the Starter Kit board, in conjunction with the program Epromer, to program ST62E46B or ST62T46B microcontrollers. You can also perform in-circuit programming of ST62E4X or ST62T4X OTP/EPROM microcontrollers using your own board, connected to the Starter Kit board via the connector P1 (marked 1 on the Starter Kit board diagram on page 11).

NOTE:

The PA5, PA6, PA7 and RESET pins are used to perform programming operations. If these pins are connected to an external resource via J1, you must either disconnect them (see “Connecting External Resources to the Starter Kit Board” on page 19) or set them to high impedance state during programming operations.

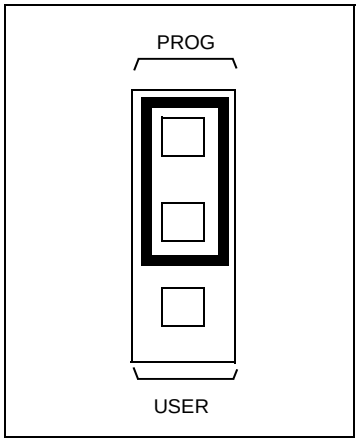
This section describes how to set up the Starter Kit board for programming microcontrollers, and lists the connection requirements for in-circuit application boards.

8.1 Setting Up the Starter Kit Board

1 Power down the Starter Kit board.

2 Plug the ST62E46B/T46B microcontroller you want to program into the MCU socket on the Starter Kit board.

3 Select the PROG mode using the jumper JP1(marked 2 on the Starter Kit board diagram on page 11), as shown in the diagram below:



4 Connect the Parallel port P2 on the Starter Kit board to a spare parallel port on your PC using the cable provided with the starter kit.

5 Power up the Starter Kit board.

You can now use Epromer to program the microcontroller that is plugged into the Starter Kit board.

NOTE:

Epromer does not work under Windows NT.

To run Epromer from Windows 3.x, double-click the Epromer icon in the ST6 Tools group.

To run Epromer from Windows 95, click **Start**, **Programs**, **ST6 Tools**, then **Epromer**.

For instructions on how to operate Epromer, click **Help** in the Epromer main window.

8.2 In-Circuit Programming

You can perform in-circuit programming of ST62E4x and ST62T4x EPROM/OTP microcontrollers using your own board, connected to the Starter Kit board via the connector P1 (marked 1 on the Starter Kit board diagram on page 11).

8.2.1 Application Board Connections

The following paragraphs specify the connection requirements between your application board and the Starter Kit board.

The application board must have a suitable 16-way connector (8x2 header HE10) to be connected via a 16-way cable to connector P1 (marked 1 on the Starter Kit board diagram on page 11) on the Starter Kit board.

The following table shows the required pin connections:

Table 3. Signal interconnection to different ST6 families

8x2 Connector	I/O	ST621X/2X	ST624X	ST626X/9X
Pin 1	O	PB6	PA6	PB3
Pin 3	O	PB5	PA5	PB0
Pin 5	O	OSCin	OSCin	OSCin
Pin 7	I	PB7	PA7	PB2
Pin 9	O	RESET	RESET	RESET
Pin 11	I	OSCout	OSCout	OSCout
Pin 13	O	VPP/TM	VPP/TM	VPP/TM
Pin 14+16	Power	VDD	VDD	VDD
Pin 2+4+8	Power	VSS	VSS	VSS

V_{DD}

Use of the V_{DD} connection is optional, depending on whether the application board supply can or cannot be disconnected. If the application board supply is disconnected, you can supply it through pins 14 and 16 of the connector, as long as the total load current does not exceed 100 mA, and the capacitive load is less than 50 µF.

If the application board has its own power supply, its voltage must be set to 5V, so that logic levels are compatible with those of the Starter Kit board.

OSCin

Synchronises the programming operations using a clock generated by the programming tool. OSCin is located on the application board, and must be directly connected to Pin 5 on the 16-way connector. No isolation is needed as long as a quartz crystal or ceramic resonator is used in the application. If an external clock generator is used in the application, it must be disconnected during in-circuit programming.

RESET

Controls the programming mode entry. To prevent signal level contention, RESET must be directly connected to Pin 9 on the 16-way connector, and must be isolated from other nodes on the application board. Any direct connection to V_{DD}, V_{SS} or an output must be avoided. This pin can be connected to a CMOS input, a 2 KΩ pull-up, a 10 KOhm pull-down or left open (Internal pull-up). The capacitive load of the RESET pin should not exceed 1 µF.

Pins 1 and 7 on the 16-way connector are used to establish communications between the programming tool and the microcontroller.

To prevent signal-level contention, Pins 1 and 7 must be directly connected to PA6 and PA7 on the 16-way connector, and must be isolated from other nodes on the application board. Any direct connection to V_{DD}, V_{SS} or an output must be avoided. These pins may be connected to a CMOS input, a 2 KΩ pull-up, a 10 KOhm pull-down or left open (Internal pull-up).

If **pin 3** on the 16-pin connector is connected to the target device, the same applies. Connection to pin 3 is not necessary if a high voltage level is guaranteed by the board design.

Some **I/O pins** are not connected to the 16-way connector and must be set to a high level during programming. This is normally achieved by the RESET signal sent by the programming tool through the 16-way cable, setting the I/O pins as inputs with an internal 300 KΩ pull-up. To keep these I/O lines high, direct connection of these pins to GND or to any other signal at low level (even temporarily) must be avoided. Only connections to another CMOS input, to an external pull-up or a 10 MΩ pull-down is allowed.

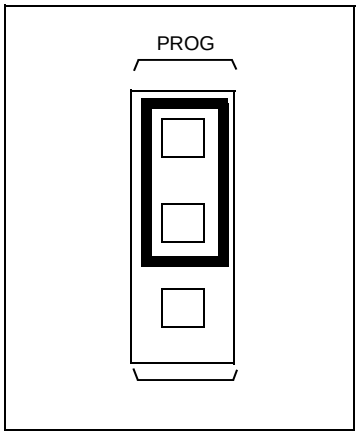
The **Vpp/TM** pin must not be directly connected to GND/V_{SS} on the application board, to avoid any conflict with the programming voltage provided by the programming tool via pin 13 on the

connector. This pin should be pulled down by a resistor with minimum value of 10 KΩ. You must add a 100 nF ceramic capacitor between Vpp/Test and $V_{SS.}$

8.3 Setting Up the Starter Kit Board for In-Circuit Programming

1 Power down the Starter Kit board.

2 Select the PROG mode using the jumpers JP1 (marked 2 on the Starter Kit board diagram on page 11), as shown in the diagram below:



3 Connect the Parallel port P2 on the Starter Kit board to a spare parallel port on your PC using the cable provided with the starter kit.

4 Connect your application board to the connector P1 (marked 1 on the Starter Kit board diagram on page 11) on the Starter Kit board.

5 Power up your Starter Kit board.

You can now use Epromer to program the microcontroller that is on your own board.

NOTE:

Epromer does not work under Windows NT.
To run Epromer from Windows 3.x, double-click the Epromer icon in the ST6 Tools group.
To run Epromer from Windows 95, click **Start**, **Programs**, **ST6 Tools**, then **Epromer**.
For instructions on how to operate Epromer, click **Help** in the Epromer main window.
If your application board is not powered by the Starter Kit, you must connect it to a 5V DC power supply before you start programming.

9 LCD INTERFACE

9.1 ST6240 LCD DRIVER OVERVIEW

This is a quick summary of the features of the ST6240 LCD driver. Refer to the ST6240 data book for more detailed information.

The ST6240 LCD driver comprises LCD control logic, a programmable prescaler, a 24-byte wide dedicated LCD RAM, 45 segments and 4 common outputs. This drives up to 180 LCD segments.

The LCD control logic operates automatically and without interrupting the processor.

The LCD driver configuration is defined by the following ST6240 registers:

- LMCR (LCD Mode Control Register), which defines the LCD backplanes (duty cycle) and the frame frequencies used by the LCD.
- LCD RAM (24 bytes), which sets each segment to ON or OFF by setting or resetting the corresponding bit. According to the contents of the LCD RAM, the drivers generate the segments and common signals which can directly drive an LCD panel.

Hardware configuration requirements are reduced to minimum:

- The VLCD input pin must be fed with VLCD voltage (independently of VMVDD), according to your LCD panel specifications.
- The VLCD1/3, VLCD2/3 pins are connected to intermediate VLCD voltages, to enable external capacitive buffering or resistive shunting. An internal resistor network is implemented so that in most cases, thus no additional components are necessary.

To interface with the LCD panel, all you have to do is:

- Determine the operating LMCR register value according to the LCD panel specifications.
- Generate the correct VLCD voltage and, if needed, connect an external resistor network to the VLCD1/3 and VLCD2/3 pins.
- Determine the most suitable mapping between the LCD panel segments and the LCD RAM bits.

Once you've completed the above steps, controlling the LCD panel becomes as easy as modifying software bits.

9.2 STARTER KIT LCD PANEL INTERFACE

The following paragraphs describe how the starter kit LCD panel interfaces with the ST62T40B microcontroller.

The LCD panel used on ST62T40B Starter Kit board has 8 alphanumeric digits.

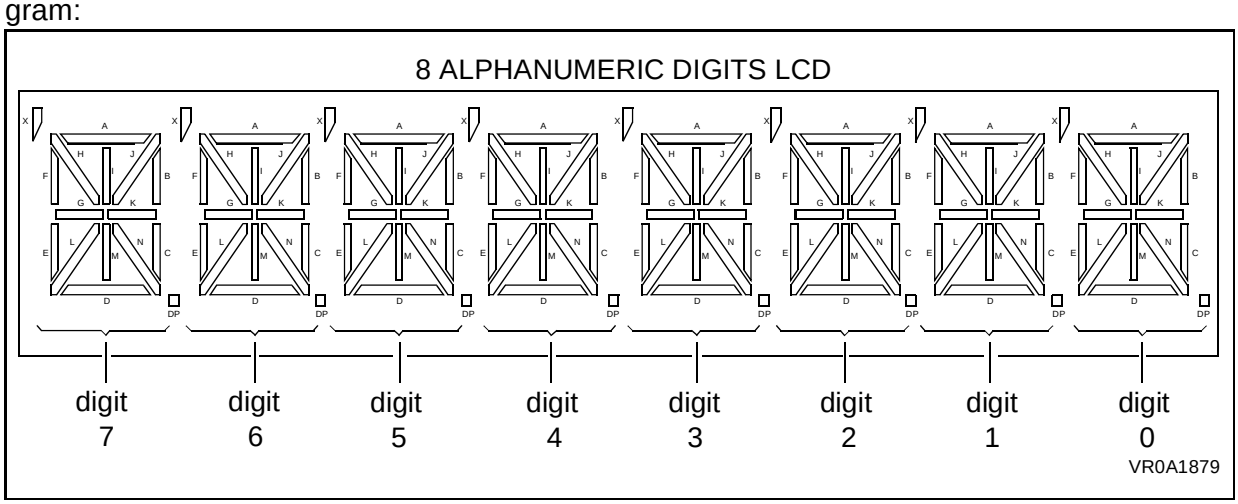
Each digit is based on a 16-segment matrix operating in multiplexed mode (1/4 duty) so that only 4 segment pins are required per digit.

Thus the following pins are required for the 8 digits:

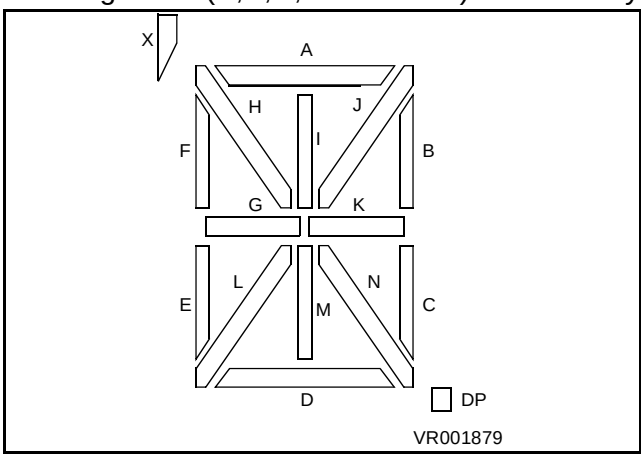
- 32 segment (SEG) pins
- 4 common (COM) pins

VLCD operating mode is 5V.

Digits are annotated from 0 to 7 respectively from right to left as shown by the following diagram:



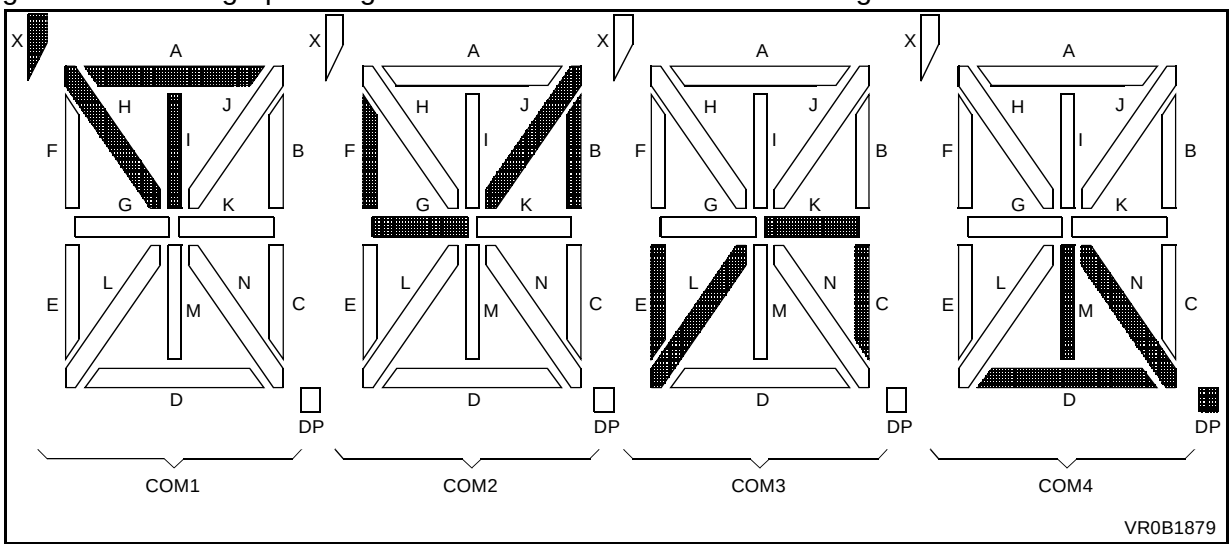
Each digit is made up of 16 segments (A,B,C,D and so on) as shown by the following diagram:



Each segment can be set to ON or OFF, depending on the state of its dedicated Segment and Common lines, according to the boolean equation:

```
segment ON = Seg_line and Com_line active
```

Each DIGIT requires 4 SEG lines X 4 COM lines to be completely defined. The following diagram shows the graphic segments that are switched on according to each active COM:



The following table shows the mapping between graphic elements set and the segment/common pins.

Table 4. Digit Matrix to Segment/COM Pins Mapping

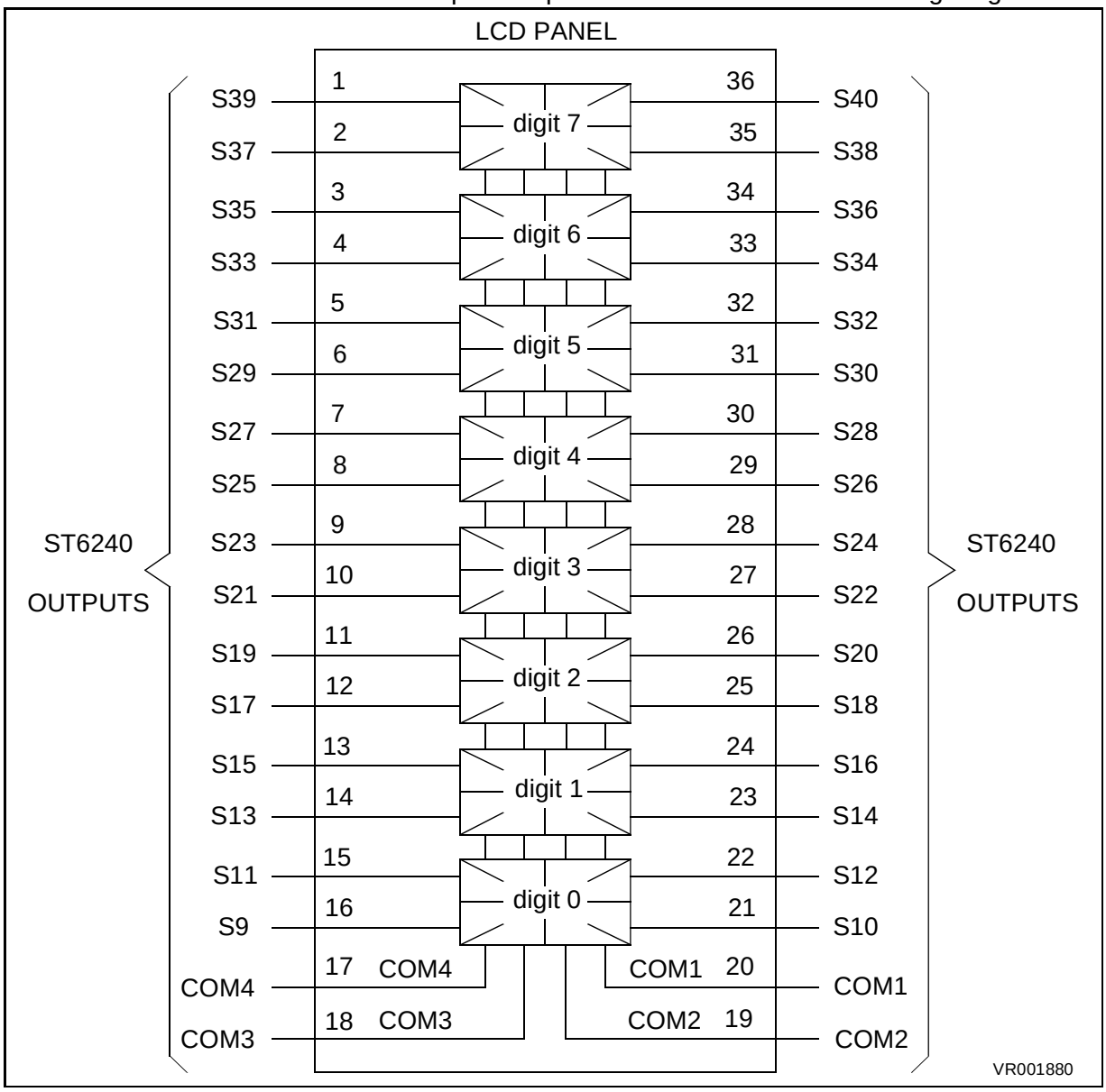
PIN	SEGMENTS				PIN	SEGMENTS				PIN	SEGMENTS			
1	7X	7F	7E	7D	13	1X	1F	1E	1D	25	2A	2B	2C	2DP
2	7I	7J	7K	7N	14	1I	1J	1K	1N	26	2H	2G	2L	2M
3	6X	6F	6E	6D	15	0X	0F	0E	0D	27	3A	3B	3C	3DP
4	6I	6J	6K	6N	16	0I	0J	0K	0N	28	3H	3G	3L	3M
5	5X	5F	5E	5D	17	.	.	.	C04	29	4A	4B	4C	4DP
6	5I	5J	5K	5N	18	.	.	C03	.	30	4H	4G	4L	4M
7	4X	4F	4E	4D	19	.	C02	.	.	31	5A	5B	5C	5DP
8	4I	4J	4K	4N	20	C01	.	.	.	32	5H	5G	5L	5M
9	3X	3F	3E	3D	21	0A	0B	0C	0DP	33	6A	6B	6C	6DP
10	3I	3J	3K	3N	22	0H	0G	0L	0M	34	6H	6G	6L	6M
11	2X	2F	2E	2D	23	1A	1B	1C	1DP	35	7A	7B	7C	7DP
12	2I	2J	2K	2N	24	1H	1G	1L	1M	36	7H	7G	7L	7M
COM	C1	C2	C3	C4	COM	C1	C2	C3	C4	COM	C1	C2	C3	C4

For example, the graphic element 7J (J of digit 7) is driven by SEG line S2 (pin 2) and COM line C2 (pin 19). For digit 2, pin 12 drives segments I,J,K,N when COM lines C1,C2,C3,C4 respectively are active.

9.3 Interfacing The LCD Panel with the ST6240 LCD Driver

When assigning the LCD panel segments to the LCD RAM bits, the mapping used will depend on the LCD you are using. It is recommended that you define the character mapping after defining your software architecture. Make sure, however that this mapping is kept as simple as possible. For example, the description map of each character in LCD RAM should be the same for all the 8 digits on the LCD.

The LCD panel requires 32 segment lines multiplexed through 4 common lines. This is less than the capabilities of the ST6240. Segments S9 to S39 and outputs COM1,2,3,4 of ST6240 LCD driver are connected to the LCD panel inputs as described in the following diagram.



The following table lists the location of each digit definition in the ST6 RAM. Note that the memory is not entirely used.

Table 5. Digit Locations in LCD RAM

	LCD RAM Address	D7 D6 D5 D4	D3 D2 D1 D0
COM1	E1h	Digit 1	Digit 0
	E2h	Digit 3	Digit 2
	E3h	Digit 5	Digit 4
	E4h	Digit 7	Digit 6
COM2	E7h	Digit 1	Digit 0
	E8h	Digit 3	Digit 2
	E9h	Digit 5	Digit 4
	EAh	Digit 7	Digit 6
COM3	EDh	Digit 1	Digit 0
	EEh	Digit 3	Digit 2
	EFh	Digit 5	Digit 4
	F0h	Digit 7	Digit 6
COM4	F3h	Digit 1	Digit 0
	F4h	Digit 3	Digit 2
	F5h	Digit 5	Digit 4
	F6h	Digit 7	Digit 6

Each digit is represented by 4 bits (MSB for odd digits, LSB for even digits) in each COM memory area. Each set of 4 bits x 4 COMs are assigned to the 16 bits defined above as shown in the following table.

Table 6. LCD RAM Bits Mapping

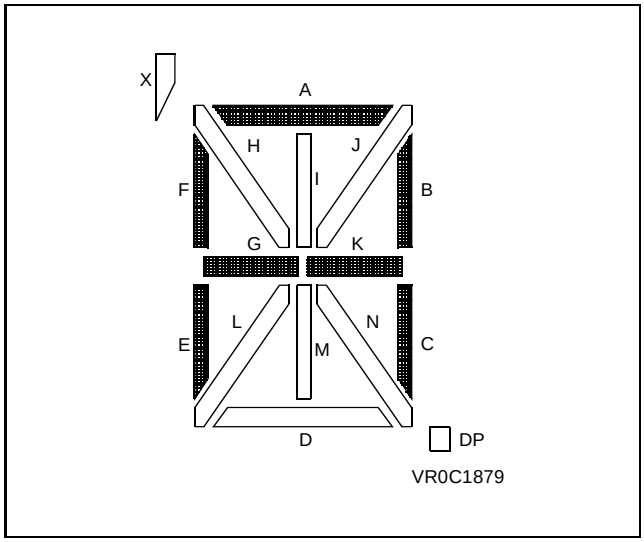
	Bit 7/3	Bit 6/2	Bit 5/1	Bit 4/0
COM1	H	X	A	I
COM2	G	F	B	J
COM3	L	E	C	K
COM4	M	D	DP	N

9.4 Character Definition Examples

The following examples show the definition of characters using the previously described method.

9.4.1 Character A Definition

The following diagram represents the ‘A’ character to be displayed on the LCD panel.



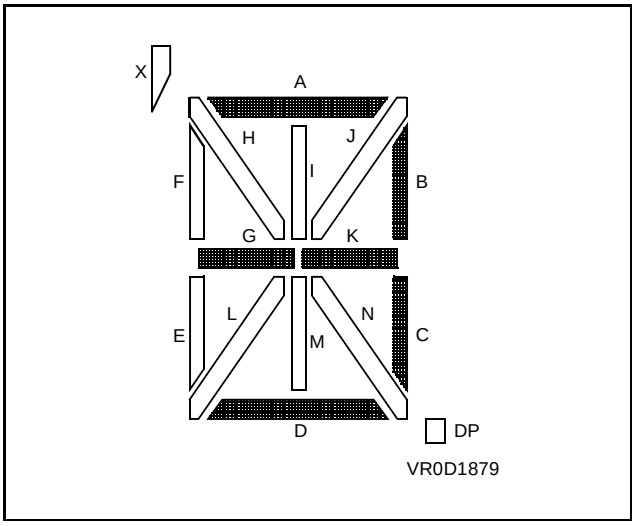
The following table gives the corresponding bit pattern to be set in the LCD RAM to display the character ‘A’ on the LCD panel. The bits set to 0 are left blank, and the bits set to 1 are shaded. The right column provides the corresponding hexadecimal value.

	D3	D2	D1	D0	Hex value
COM1	H	X	A	I	2
COM2	G	F	B	J	E
COM3	L	E	C	K	7
COM4	M	D	DP	N	0

This description is valid for all characters. The position of the displayed character depends only on the addresses and the position (LS/MS nibble).

9.4.2 Character 3 Definition

The following diagram shows the character 3 to be displayed on the LCD.



The following table gives the corresponding bit pattern to be set in the LCD RAM to display the character ‘3’ on the LCD panel.

	D3	D2	D1	D0	Hex value
COM1	H	X	A	I	2
COM2	G	F	B	J	A
COM3	L	E	C	K	3
COM4	M	D	DP	N	4

The set of 4 x 4 bits (representing the character set) can be stored in the ST624x DATA ROM and easily used by display software through indexed accesses.

9.5 Starter Kit LCD Panel Character Set Software Model

As described in the previous paragraphs, LCD connection and character mapping drive a software model of the LCD-displayable objects. These objects are available for DISPLAY routines to drive LCD panel display during program execution.

The following paragraphs describe three methods of displaying characters on the LCD.

9.5.1 Direct Code LCD RAM Patching

This is the most simple way to display digits on an LCD. The mapped character values are directly patched into the corresponding LCD RAM locations. The bit map values are contained in the immediate values of write instructions.

This method should only be used for very simple applications (few characters, displayed few times) or for rapid evaluation.

The following block of code shows how to use direct code LCD RAM patching to display the character ‘A’ at digit 0, where x is the value of the MSB (digit 1):

```
disp_A_0  ldi E1, x2   ;value 2 to digit 0 COM1 addr (E1, LSB)
          ldi E7, xE   ;value E to digit 0 COM2 addr (E7, LSB)
          ldi ED, x7   ;value 7 to digit 0 COM3 addr (ED, LSB)
          ldi F3, x0   ;value 0 to digit 0 COM4 addr (F3, LSB)
```

9.5.2 Indexed Data ROM

The mapped values of character sets are defined into ST6240 Data ROM area (refer to ST6240 databook). The AST6/LST6 `.byte` directive and `.w` and `.d` labels define and access the Data ROM area.

A generic display call subroutine accesses the character definition map through the parameter passed by the calling program. The `disp_0` subroutine (see below) displays a character at digit 0 and clears digit 1 (MSB of the same LCD RAM locations).

An efficient generic display subroutine should manage the digit number (as input parameter) and perform the read/modify/write only on the concerned nibble in the LCD RAM byte (MSB or LSB). This is done by the display subroutine delivered with the ST6240 Starter Kit library (please read the README file provided in the SK6240LIB directory).

A character is defined by 4 x 4 bit words = 16 bits. To optimize the character set size in Data ROM, it is useful to compact these 16 bits into 2 bytes, however this method increases the display routine complexity.

For the ST6240 Starter Kit demo routines, the character set is defined as follows (for example character A):

```
CHAR_A.    .byte 22
           .byte EE
           .byte 77
           .byte 00
```

The character is displayed using the following routine:

```
disp_A_0   ldi RWSR, CHAR_A.w;set Data ROM Window register
           ldi X, CHAR_A.d   ;X set to point to CHAR_A
           call disp_0       ;subroutine that displays digit 0
disp_0     ld E1, (X)        ; output value 2 to digit 0 COM1
                             ;(E1, LSB)
           inc X             ;X set to second byte of CHAR_A
           ld E7, (X)        ;output value E to digit 0 COM2
                             ;(E7, LSB)
           inc X             ;X set to third byte of CHAR_A
           ld ED, (X)        ;output value 7 to digit 0 COM3
                             ;(ED, LSB)
           inc X             ;X set to fourth byte of CHAR_A
           ld F3, (X)        ;output value 0 to digit 0 COM4
                             ;(F3, LSB)
```

Four bytes are used to define a character, with MSB and LSB equal to the 4 bit definition values. In this way, MSB and LSB are easily distinguished using the AND instruction for masking the non-relevant nibble.

9.5.3 Complete Message Display

You can simplify the definition of complete messages using the AST6 .ASCIZ directive. This directive returns the ASCII code pattern of the characters indicated ended by NULL). To enable you to use the .ASCIZ directive, the ST6240 character set is pre-defined in the Data ROM so that it becomes simple to access a mapped character definition through the character's ASCII code.

One Data ROM window contains 64 bytes. As a character definition map requires 4 bytes, 16 characters can be defined in a whole window. This corresponds to a page of ASCII codes (represented by the MSB of the ASCII code).

For example characters @, A, B through O, whose ASCII codes are 40,41,42 through 4F (Hex) respectively, represent the ASCII code page 4, and fill a Data ROM window as shown below:

```
          .ORG x00;beginning of a window
0    CHAR_@.byte 22;address 0 in the window
          .byte AA
          .byte BB
          .byte 44
1    CHAR_A.byte 22;address 3 in the window
          .byte EE
          .byte 77
          .byte 00
through
F    CHAR_O.byte 22;address 3C in the window
          .byte 66
          .byte 66
          .byte 44;last address (3F) in the window
```

For example, the ASCII code of character A is 41 (hex):

- 4 specifies the Data ROM window number
- 1 indexes the number of the character in the Data ROM window

The available LCD character set that is delivered with the ST6240 Starter Kit library is defined using this method. This allows the `asci_dis` subroutine to display a character defined by its ASCII code and `Mdis_mes` subroutine to display a complete message, built using the ASCIZ AST6 directive, on the LCD panel.

Refer to the files SK6240LI.INI and SK6240LI.ASM in the SK624XLI directory for a description of these subroutines.

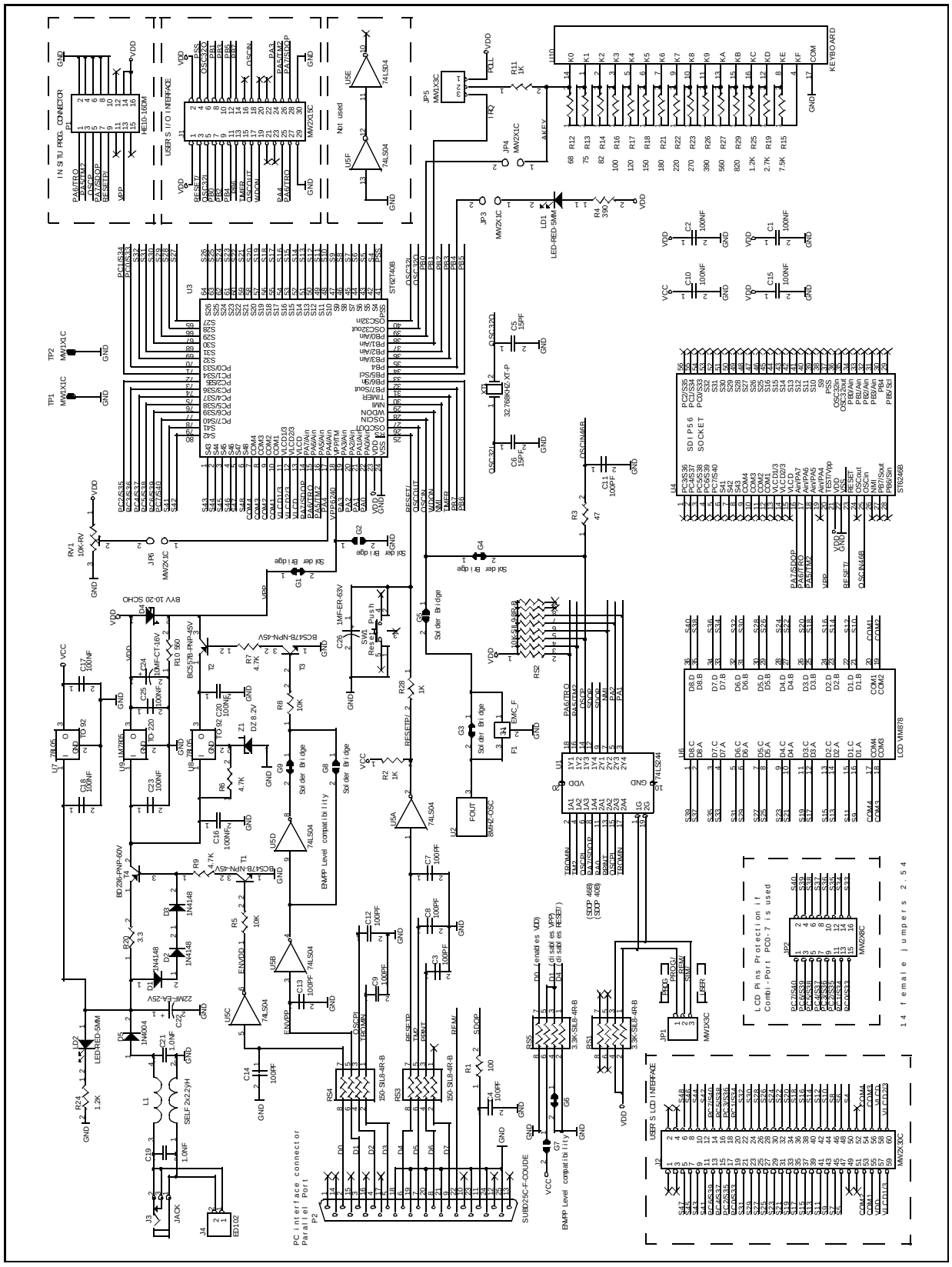
10 HARDWARE INFORMATION

10.1 Part List

Part	Device	Part	Device
U1	74LS244	R1, R16	100Ω
U2	8MHz oscillator	R2, R11, R28	1KΩ
U3	ST62T40B MCU	R3	47Ω
U4	ST62E46B MCU socket	R4, R26	390Ω
U5	74LS04	R5, R8	10KΩ
U6	LCD panel	R6	Not connected
U7, U8	78L05 Voltage Regulator	R7, R9	4.7KΩ
U9	7805 Voltage Regulator	R10, R27	560Ω
U10	Keyboard	R12	68Ω
XT1	32.768 KHz crystal	R13	75Ω
RV1	10KΩ resistor trimmer	R14	82Ω
D1, D2, D3	1N4148 diode	R15	7.5KΩ
D4	BYV 10-20 Schottky	R17	120Ω
D5	1N4004 diode	R18	150Ω
Z1	8.2V Zener diode	R19	2.7KΩ
LD1, LD2	Red LED	R20	3.3Ω
T1, T3	BC547B transistor	R21	180Ω
T2	BC557B transistor	R22	220Ω
T4	BD236 transistor	R23	270Ω
L1	Self 2x2µH	R24, R25	1.2KΩ
C1, C2, C10, C15	100nF (Cd)	R29	820Ω
C16,C17,C18,C20,C23,C25	100nF	RS1, RS5	3.3KΩ SIL8 Array
C3, C4, C7, C8, C9, C11,	100pF	RS2	10KΩ SIL9 Array
C12, C13, C14	100pF	RS3, RS4	150Ω SIL8 Array
C5, C6	15pF	RV1	10KΩ resistor trimmer
C19, C21	1.0nF	P1	Header 2x8
C22	22µF	P2	SUBD25-M Connector
C24	10µF	J1	2x15 pins connector
C26	1µF	J2	2x30 pins connector
SW1	Push-button	J3	Female Jack plug
JP1 to JP6	jumpers	J4	2nd power supply conn.

10.2 Starter Kit Board Schematic

See next page



Notes:

Printed in France by Imprimerie AGL