LITE0 Exercises

ST7Flite0 InDart Demo Board

USER MANUAL AND TUTORIAL EXERCISES

Release 1.1

Ref: DOC-LITE0-Exercises

LITE0 Exercises

ST7Flite0 InDart Demo Board

USER MANUAL AND TUTORIAL EXERCISES

May 2002

DOC- LITE0- Exercises

Table of Contents

1 INTRODUCTION

This document describes how to configure the ST7Flite0 InDart Demo Board and install and run the Tutorial Exercises available on the “MCU ON CD” CD-ROM.

In order to perform the Tutorial Exercises you must have the following equipment:

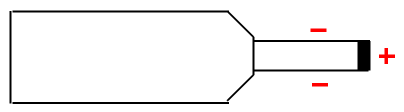
- Softec Microsystems InDart-ST7FLITE0
- Software development tools available from the “MCU ON CD” CD-ROM.
- Power adapter (not delivered)

1.1 POWER ADAPTER

To be able to install and run the ST7Flite0 InDart Demo Board, you need a power adapter.

Its polarity must be as follows:

Figure 1. Jack plugpolarity



The adapter output must have the following characteristics:

Output :

– Voltage: 9VDC max.

– Current: 500mA min.

1.2 GETTING ASSISTANCE

For more information, application notes, FAQs and software updates on all the ST7 microcontroller family, check out our website: **http://mcu.st.com**

For assistance on all ST microcontroller subjects or if you need help with using your ST7Flite0 InDart Demo Board, feel free to contact your local distributor or your local sales office.

2 ST7 DEMO BOARD DESCRIPTION

The demo board (which features DIP-switches, jumpers, LEDs, push-buttons, a potentiometer, prototyping area and a standard ICP connector) can be used for evaluation/tests in case you do not have your own target application board.

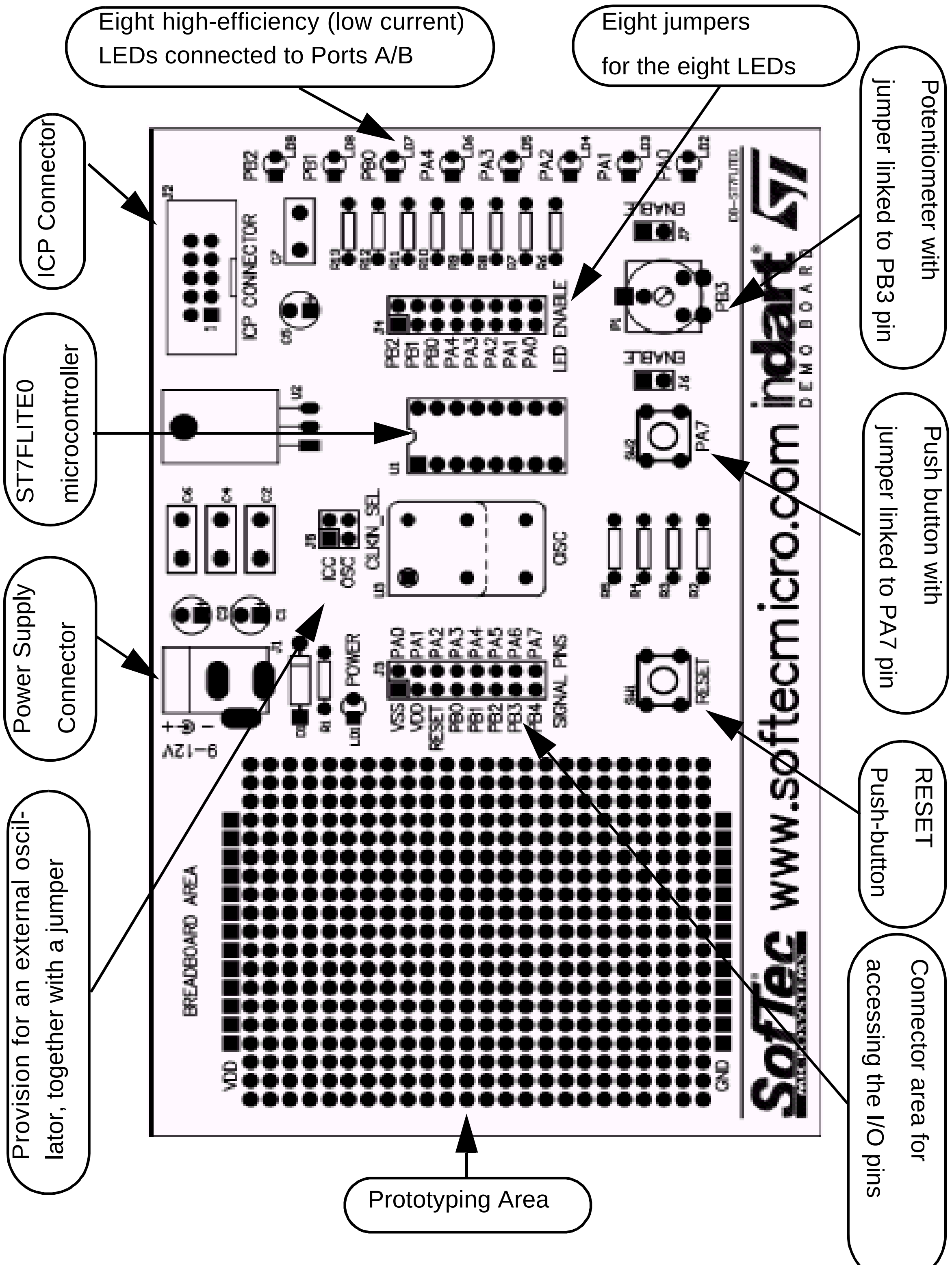
Note:

Please refer to the Softec InDart user's manual to get more information.

2.1 IDB-ST7FLITE0 (INDART DEMO BOARD) FEATURES

The IDB-ST7FLITE0 Demo Board has the following hardware features (refer to the schematic drawing on the following page.)

Figure 2:The IDB-ST7FLITE0 Demo Board

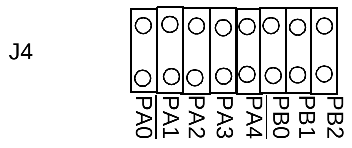
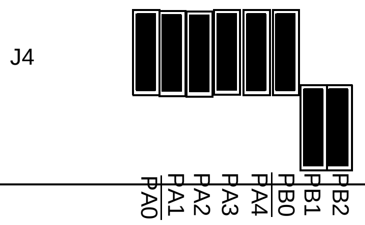

Eight high-efficiency (low current) LEDs connected to Ports A/B
Eight jumpers for the eight LEDs
Potentiometer with jumper linked to PB3 pin
ICP Connector
ST7FLITE0 microcontroller
Push button with jumper linked to PA7 pin
Power Supply Connector
RESET Push-button
Provision for an external oscillator, together with a jumper
Connector area for accessing the I/O pins
Prototyping Area
ICP CONNECTOR
LED ENABLE
ENABLE
PA7
RESET
SIGNAL PINS
OSC
CLKIN_SEL
ICC OSC
POWER
9-12V
BREADBOARD AREA
VDD
GND
www.softecmicro.com
indart DEMO BOARD
SofTec

2.2 IDB-ST7FLITE0 DEMO BOARD JUMPER OPTIONS AND CONNECTORS

2.2.1 LED jumpers (J4)

You can connect / disconnect each of the eight LEDs from their respective Ports on Port A and Port B via these eight jumpers (as shown in Table 1.).

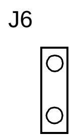
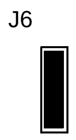
Table 1. J4 jumpers

Setting	Function
	Disconnect the eight LEDs from the ports
	Connect the eight LEDs to the ports

2.2.2 PA7 port jumper (J6)

You can connect / disconnect a push-button switch with a jumper on the PA7 pin. (i.e Table 2)

Table 2. J6 jumper

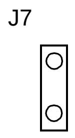
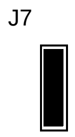
Setting	Function
	Disconnect the push button from the PA7 pin
	Connect the push button to the PA7 pin

2.2.3 PB3 port jumper (J7)

You can connect / disconnect a trimmer with a jumper on the PB3 pin.

(i.e Table 3)

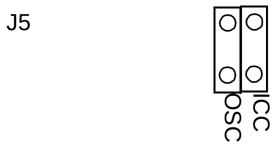
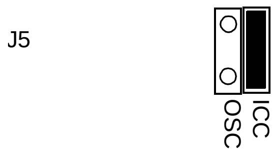
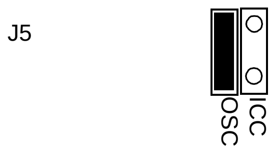
Table 3. J7 jumper

Setting	Function
	Disconnect the trimmer from the PB3 pin
	Connect the trimmer to the PB3 pin

2.2.4 Clock selection jumper (J5)

This jumper selects the microcontroller clock source to be external oscillator or the OSC_CLK pin on the ICC connector. (i.e Table 4)

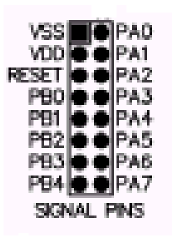
Table 4. J5 jumper

Setting	Function
	Deselect External oscillator and OSC_CLK ICC pin as clock source
	Select the OSC_CLK ICC pin as clock source
	Select the External oscillator as clock source

2.2.5 Connector for accessing the I/O pins (J3)

On the demo board , you can access the I/O pins of the ST7FLITE0 via a connector for expansion prototyping (i.e Table 5).

Table 5. I/O pin connector (J3)

Setting	Function
J3  VSS VDD RESET PB0 PB1 PB2 PB3 PB4 PA0 PA1 PA2 PA3 PA4 PA5 PA6 PA7 SIGNAL PINS	To access the I/O pins for expansion prototyping.

2.2.6 IDB-ST7FLITE0 Resource allocation

The following table lists the ST7Flite0 InDart Demo Board resources with the corresponding pin of the wire wrap connector and indicates the jumper that disconnects each resource:

Table 6. IDB-ST7FLITE0 Resource allocation

On-board resource	Jumper	Wire-Wrap Connector (J3)	
		Pin Name	PIN No.
		Vss	1
		Vdd	2
RESET push-button SW1 (ICP Connector)		$\overline{\text{RESET}}$	3
LD7 (Led)	J4 pin 6	PB0	4
LD8 (Led)	J4 pin 7	PB1	5
LD9 (Led)	J4 pin 8	PB2	6
Potentiometer P1	J7	PB3	7
ICP Connector J2	J5	PB4	8

Wire-Wrap Connector (J3)		Jumper	On-board resource
Pin No.	Pin Name		
16	PA0	J4 pin 1	LD 2 (Led)
15	PA1	J4 pin 2	LD 3 (Led)
14	PA2	J4 pin 3	LD 4 (Led)
13	PA3	J4 pin 4	LD 5 (Led)
12	PA4	J4 pin 5	LD 6 (Led)
11	PA5		ICP Connector J2
10	PA6		ICP Connector J2
9	PA7	J6	PA7 push-button SW2

3 SOFTWARE INSTALLATION

To perform the exercises which are detailed in this manual, the following software packages must be installed:

The inDART-ST7 user interface setup program is located in the SofTec Microsystems "System Software" CD-ROM provided with the board. The setup program will copy the required files to your hard disk.
To install the inDART-ST7 user interface:

1. Insert the "System Software" CD-ROM into the CD-ROM drive of your computer.
2. A startup window should automatically appear (if the startup window doesn't appear automatically, manually run the Setup.exe file located on the CD-ROM root directory). Choose "Install Instrument Software" from the main menu.
3. A list of available software should appear. Click on the "Install inDART-ST7 Toolchain" option.
4. Follow the on-screen instructions.

When the installation is complete, click on the Finish button and then reboot your computer to have the autoexec.bat modifications taken into account.

3.1 REALIZER

The free REALIZER BRONZE version is available from the Actum website: **www.actum.com**

3.2 TUTORIAL EXERCISES

The Tutorial Exercises can be installed from the "MCU ON CD" CD-ROM. To install them, follow these instructions:

1. Place the "MCU-ON-CD" CD-ROM in your CD-ROM drive. The CD-ROM's autorun feature opens up a welcome screen on your PC. If the autorun feature does not work, use the Windows File Manager or Windows Explorer to browse to the CD-ROM's root folder, and double-click on **welcome.exe**.
2. Select **Install Your Development Tools** from the list of options. A new screen appears listing the different families of STMicroelectronics MCUs.
3. Using the mouse, place the cursor over the **ST7 TOOLS** option. Choose **InDart ST7Flite0** and **ST7-EVAL** from the lists that appear.
4. The install wizard is launched. Follow the instructions that appear on the screen.

The Tutorial Exercises are stored in 3 sub-directories:

– **Manual**: contains the User Manual in PDF format.

– **Init**: This directory contains all the files for each exercise.

– **Results**: This directory contains all the exercise results.

Note: the install wizard also copies the **options.s19** file to the **Result** directory for each program (exercises and examples). This file must be used in order to burn the MCU Option Bytes.

If you choose not to install the Tutorial Exercises in the default installation directory, you will have to create a new project environment for each program in the **Result** directory in order to restore all the parameters and links.

4 EXERCISES

The following exercises, located in the **.\Exercise\Init** directory, will allow you to learn how to work with ST7FLITE0 MCUs and tools. By doing them you will learn how to program using the assembly and C (Cosmic and Metrowerks) languages. The first exercise will help you to develop applications with the REALIZER (for this exercise we will use the ST72251 device).

Note:

For this exercise we have chosen to use the ST72251 device because the REALIZER BRONZE version does not support the ST7FLITE0.

We will use the InDart interface for all the exercises.

4.1 EXERCISE 1: HOW TO USE THE A/D CONVERTER WITH THE REALIZER

The goal of this section is to help you to get started quickly with the REALIZER by using a example. The correct file is located in the directory **.\Exercise\Result\Ex1.** The following exercise has been developed with the free REALIZER BRONZE version available from the Actum web site (www.actum.com).

This exercise consists of writing a program that displays the voltage of a trimmer. The voltage is measured by the A/D converter and displayed on 3 levels as follows:

- Level 1: LD0 (0V to 1.5V)
- Level 2: LD1 (1.5V to 3.5V)
- Level 3: LD2 (3.5V to 5V)

The following code uses the A/D interrupt.

In order to make the code easy to understand, the program is divided into the following parts:

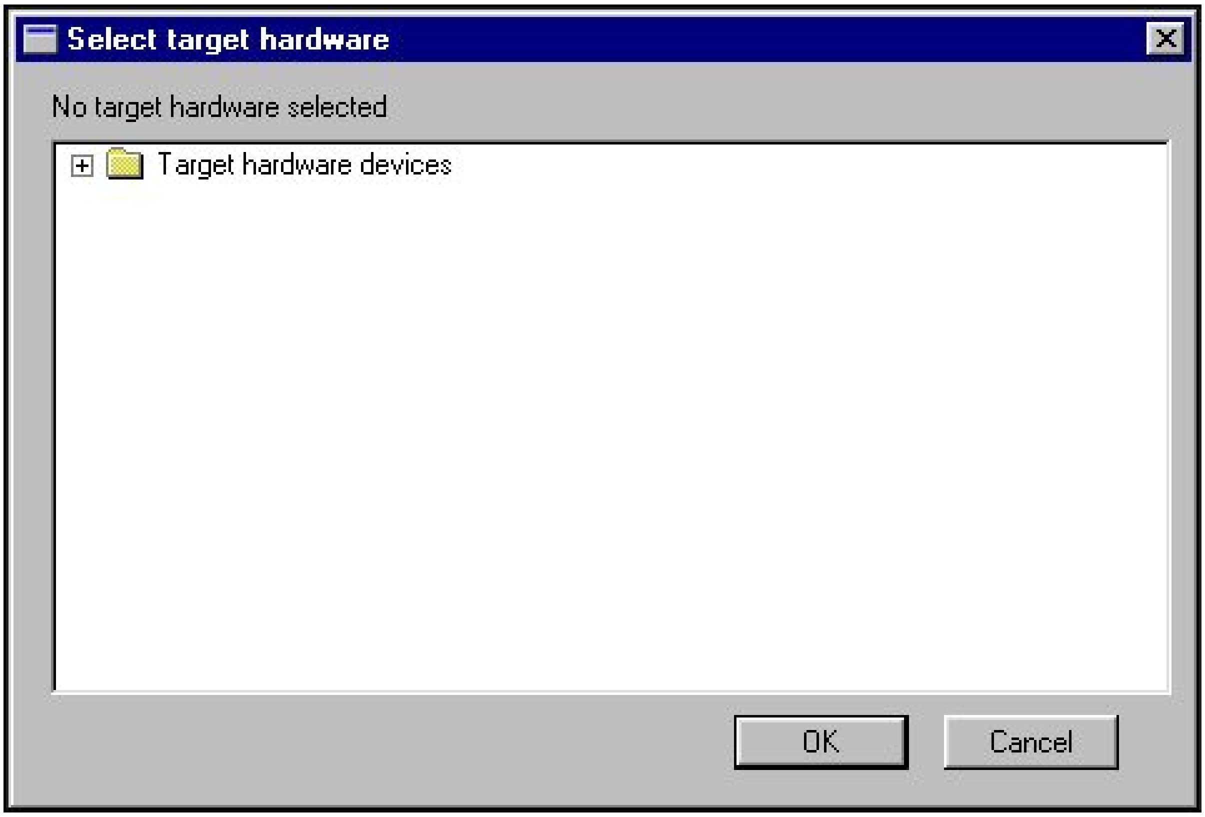
- Initialize the ST7 properly (Reset routine).
  - Initialize PA0, PA1 and PA2 as output push-pull.
  - Initialize PC0 as analog input.

4.1.1 Setting up the REALIZER software on your PC

1. Double-click on the REALIZER icon to start the tool after the software installation.
2. Now you need to create a new project environment. To create a new project, go to the

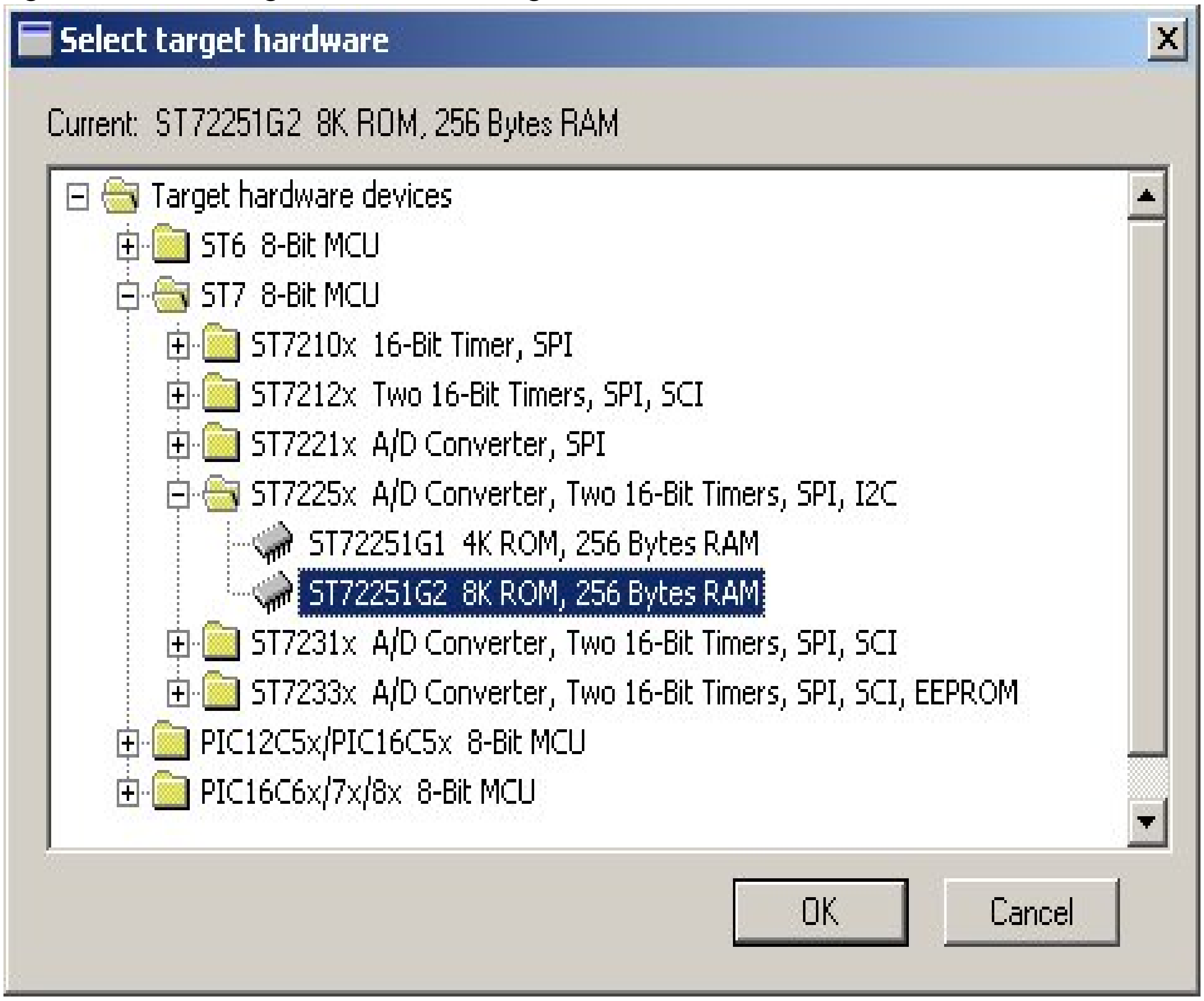
Project menu and click on **New.** When you have found the path where you want to create your project, name your file with the extension **.rpf.** A dialog box as shown in Figure 3 will appear.

Figure 3. Select target hardware dialog box



3. Choose the ST72251 device in the tree structure as shown on the following page and click on the OK button.

Figure 4. Select target hardware dialog box



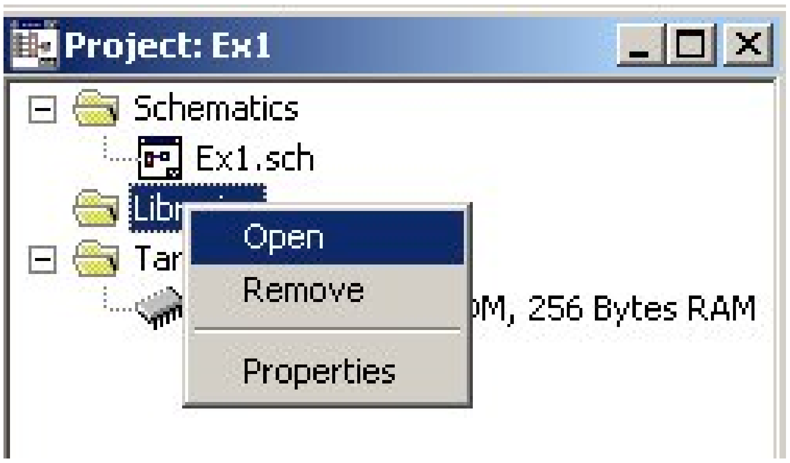
Select target hardware

Current: ST72251G2 8K ROM, 256 Bytes RAM

```
Target hardware devices
    ST6  8-Bit MCU
    ST7  8-Bit MCU
        ST7210x  16-Bit Timer, SPI
        ST7212x  Two 16-Bit Timers, SPI, SCI
        ST7221x  A/D Converter, SPI
        ST7225x  A/D Converter, Two 16-Bit Timers, SPI, I2C
            ST72251G1  4K ROM, 256 Bytes RAM
            ST72251G2  8K ROM, 256 Bytes RAM
        ST7231x  A/D Converter, Two 16-Bit Timers, SPI, SCI
        ST7233x  A/D Converter, Two 16-Bit Timers, SPI, SCI, EEPROM
    PIC12C5x/PIC16C5x  8-Bit MCU
    PIC16C6x/7x/8x  8-Bit MCU
```

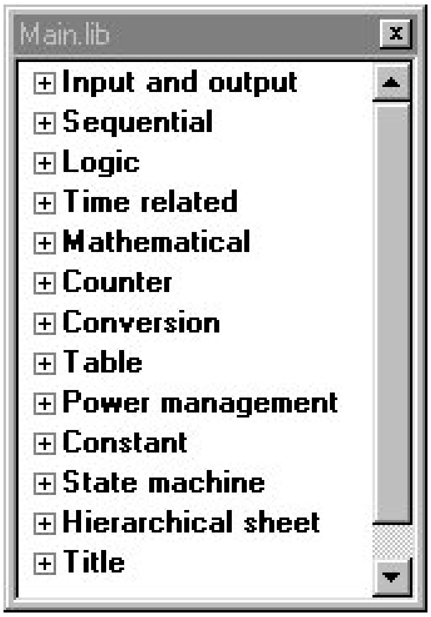
4. Now you have a new **Project** window with the name of your project.
5. Click on the **Libraries** icon with the right mouse button. A menu will be displayed as follows.

Figure 5. Libraries menu



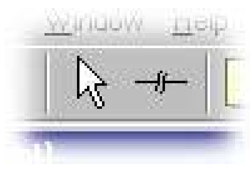
6.Click on **Open** to add the library you want to work on. Choose the **main.lib** file. When you select this file, it will display a new window, as shown in Figure 6, and you can choose any of the symbols in the various trees to draw your scheme in the schematic window.

Figure 6. Main library window menu



7. Draw your scheme. In this exercise, you may need to use the **adc, digout, nor2** and **comp** symbols which are in the **Input and output, Logic** and **Conversion** trees of the main library. To place in them the schematic window or to obtain more information on each of the symbols you would like to use, click with the right mouse button and choose **Place** or **Information**.

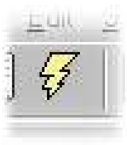
You then have to connect them with wires using the smart icons:



4.1.2 Analysing the created schematic

1. In the project window, click with the right mouse button on **Target hardware** and choose **Hardware settings**. The following dialog box will appear.

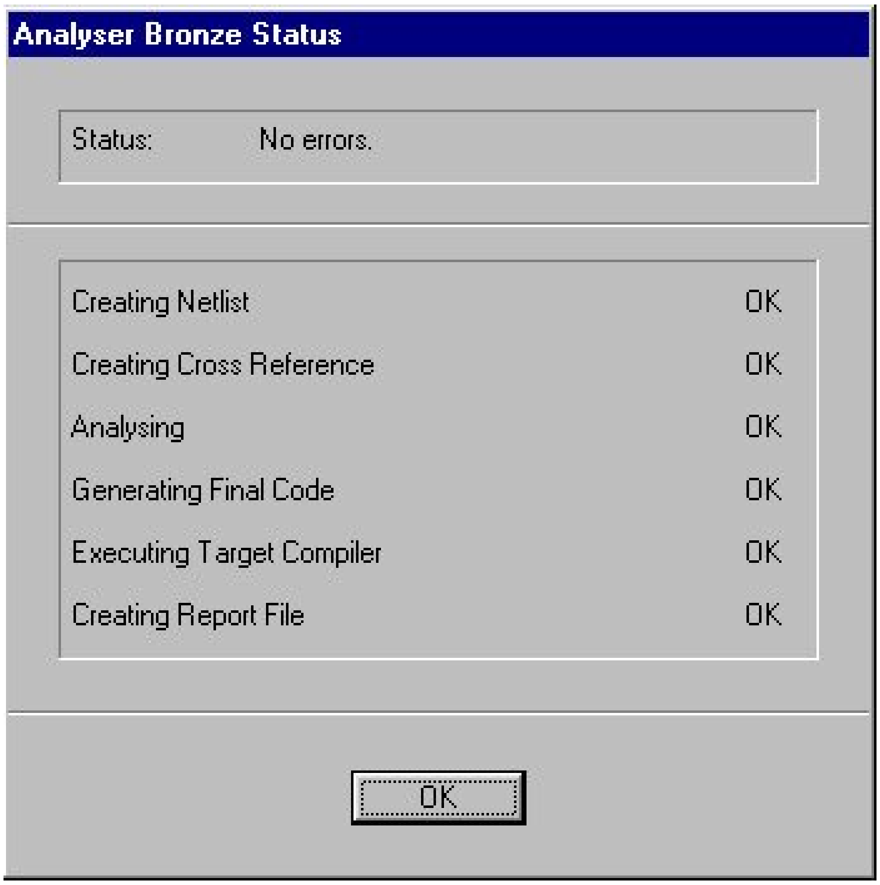
I2.Now, start the analyser **Project > Analyse** or use the smart icon:



This action will create the **asm**, **inc** and **hex** files. The **hex** file can be directly used to burn a chip with a Starter kit for example, using the windows Epromer.

If there are no errors in your scheme, you should obtain a dialog box as shown in Figure 7.

Figure 7. Analyser status dialog box



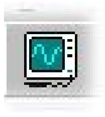
Analyser Bronze Status

Status: No errors.

Creating Netlist	OK
Creating Cross Reference	OK
Analysing	OK
Generating Final Code	OK
Executing Target Compiler	OK
Creating Report File	OK

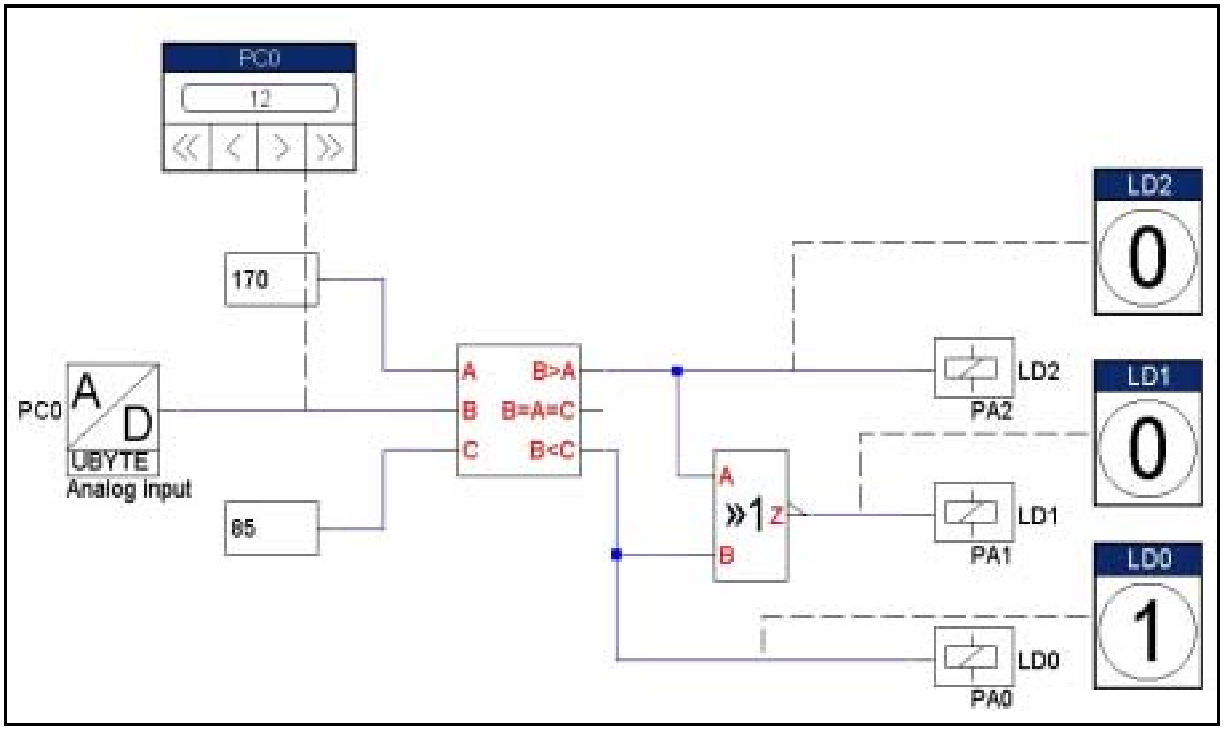
4.1.3 Simulating the created schematic

1. Start the simulator by clicking on **Tools > Simulator**. A new window will appear. You can also use the smart icon shown below:



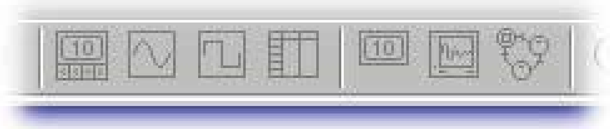
2. Now you need to create a new simulation environment. The simulation environment file stores all the simulation parameters like settings and options. To create a new simulation environment, go to the **File** menu and click on **New > Simulation environment.** Create your simulation in the same path as your project and name your file with the **.sef** extension. Your scheme will appear.

Figure 8: Simulation Mode



3. Click on the wires you would like to monitor. Input adjusters and Output probes will be displayed in bold as follows:

Figure 9. Simulation toolbar



Here is a list of the actions invoked by the toolbar buttons from left to right:

– Create and place a **numeric adjuster**

– Create and place a **sine wave adjuster**

– Create and place a **square wave adjuster**

– Create and place a **time table wave adjuster**

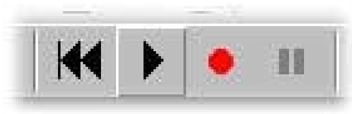
– Create and place a **numeric probe**

– Create and place an **oscilloscope probe**

– Create and place a **state machine probe**

4. Once all the probes are implemented, use the following toolbar to **Initialize, Start, Record,**

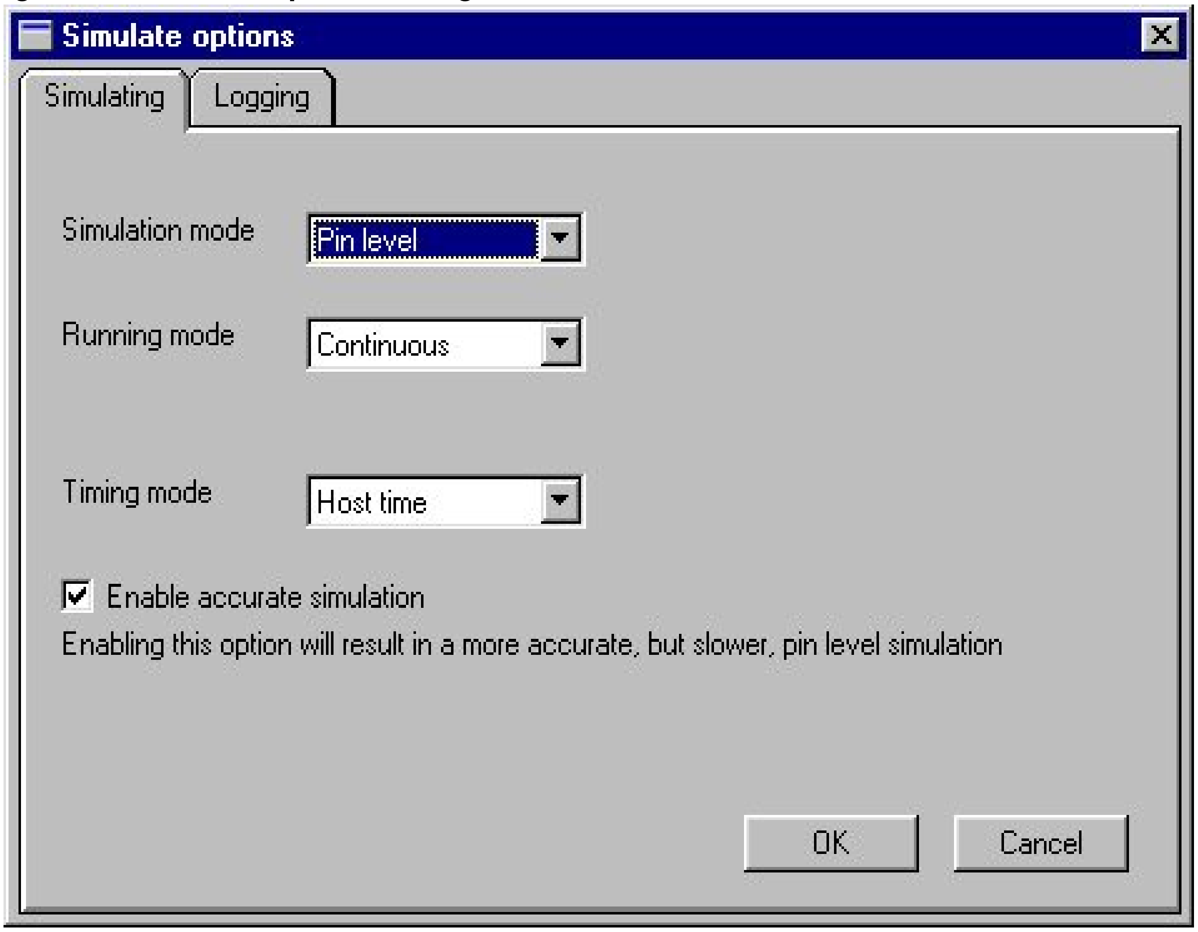
or **Pause/Stop** the simulation.



If you want a pin level simulation mode you must :

1.) Select **Options > Simulate.** A dialog box, like the one shown in Figure 10, will appear.

Figure 10. Simulate options dialog box



Simulation mode: Pin level

Running mode: Continuous

Timing mode: Host time

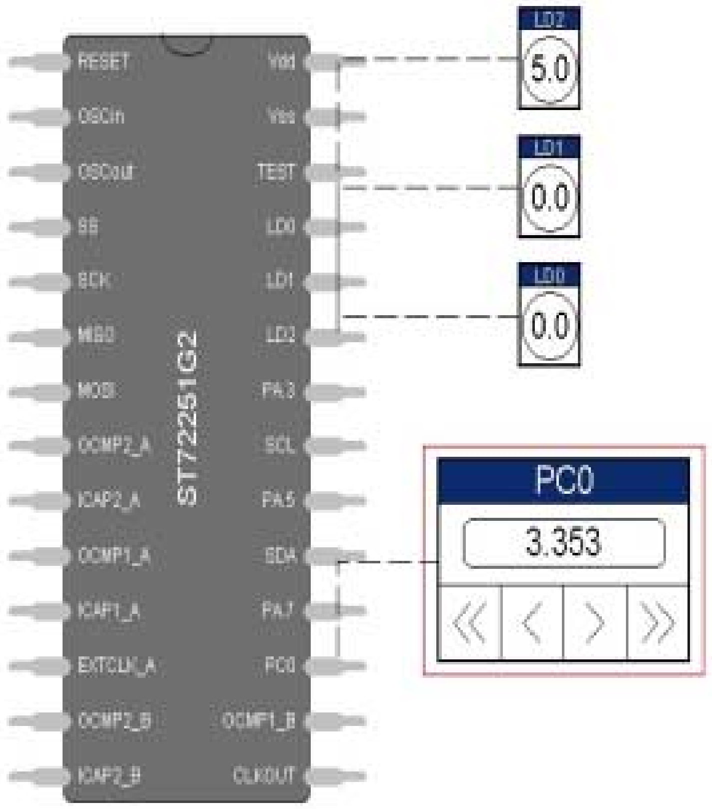
☑ Enable accurate simulation

Enabling this option will result in a more accurate, but slower, pin level simulation

2.) Click on the pins you would like to monitor. Input adjusters and Output probes will be displayed as for the schematic diagram .

Figure 11 gives an example of simulation mode :

Figure 11. Pin level simulation mode overview



4.2 EXERCISE 2: ASSEMBLER SYNTAX

The purpose of the first exercise is to improve your knowledge of the ST7 assembler syntax and the STVD7 environment.

1. Start the STVD7 Simulator:

– Enter the Toolchain directory paths in the dialog box or use the browse button.

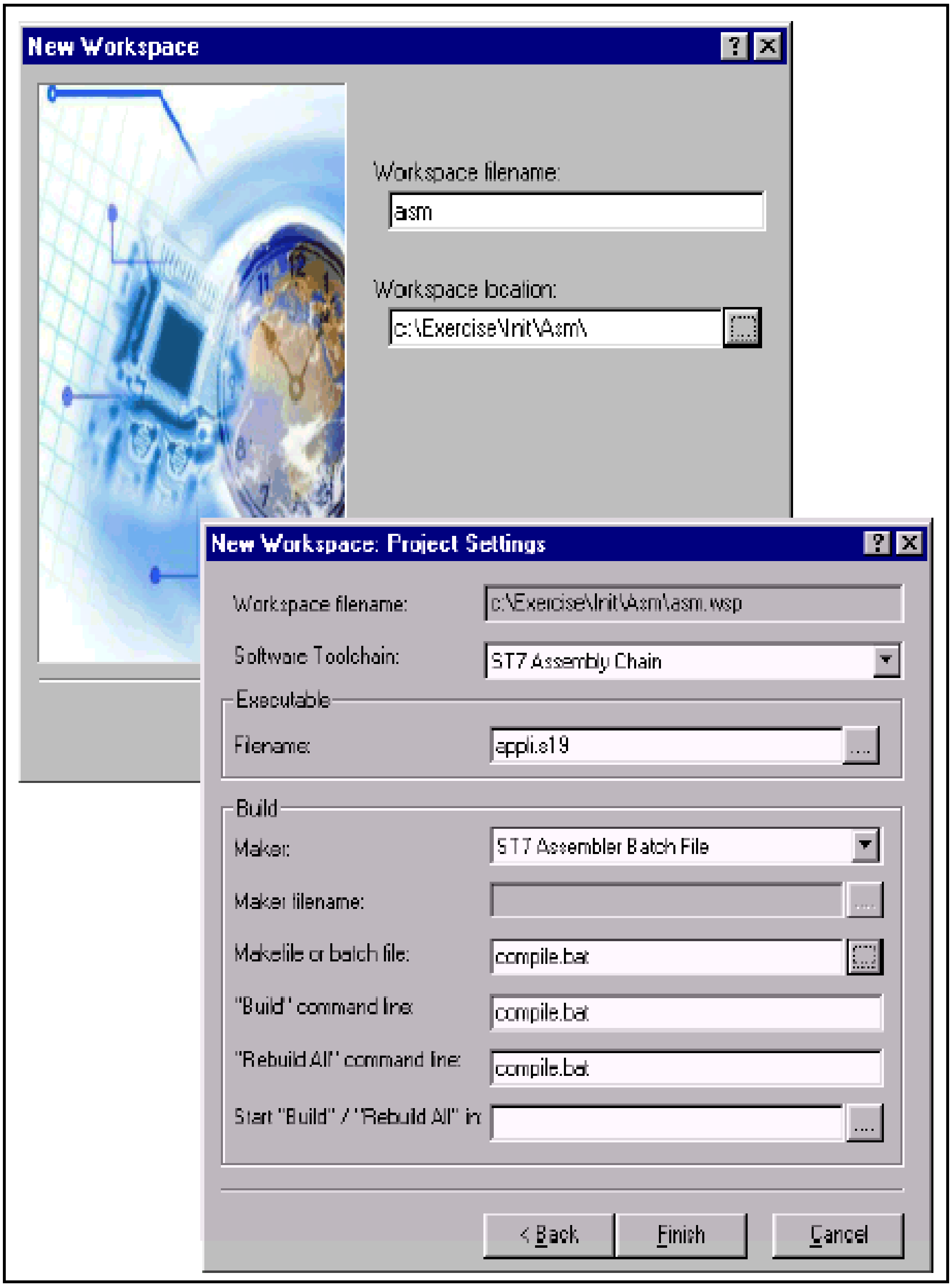
Note: If you have chosen the default directory during installation, you will not have to change the assembly toolchain path. For Metrowerks the correct path is C:\Metrowerks\CodeWarrior_STM_V1.1\prog and for Cosmic it is C:\COSMIC\evalST7.

– Create a new project:

– Click on “File, New Workspace” and enter the requested information (see Figure 12)

– Click on the “Source Directory” tab in the Workspace window and double click on the directory : your working directory is displayed by default.

Figure 12 : Create a new project



Workspace filename: asm

Workspace location: c:\Exercise\Init\Asm\

Workspace filename: c:\Exercise\Init\Asm\asm.wsp

Software Toolchain: ST7 Assembly Chain

Executable

Filename: appli.s19

Build

Maker: ST7 Assembler Batch File

Maker filename:

Makefile or batch file: compile.bat

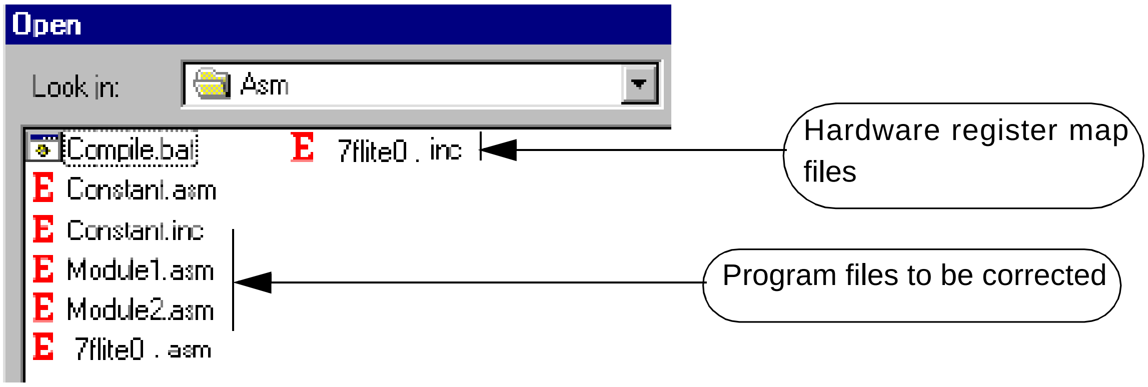
"Build" command line: compile.bat

"Rebuild All" command line: compile.bat

Start "Build" / "Rebuild All" in:

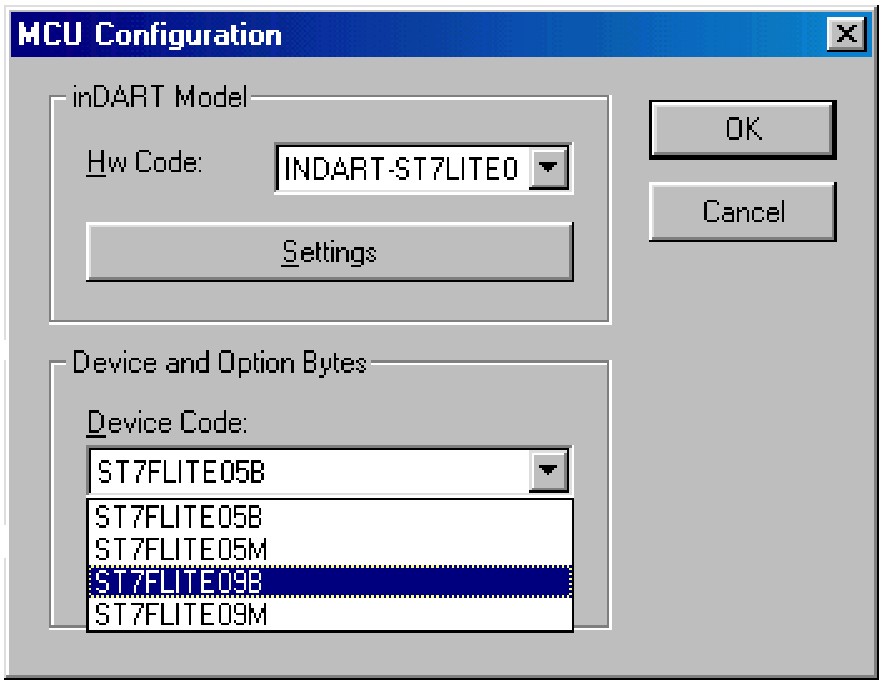
2. Modify the files located in the following environment (C:\exercise\Init\Ex2, see Figure 13) to correct the assembler syntax. Some syntax errors have been introduced by replacing the correct directives or code by '?????'.Find them and correct them.

Figure 13: Environment files



3. When everything is corrected, write the compile.bat batch file to assemble, link and create a .S19 file (give it any name you want). The batch file also generates the S19, SYM, MAP and OBJ files. Use the technical manual for help.
4. Then, click on "Project, Build" or directly on the Build shorcut to build your application.
5. If some errors occur, double click on them in the output window to be place on them, correct them and rebuild.
6. If you want to debug, just click on the blue D in the tool bar. Then select "Tools, MCU configuration" to configure the MCU name (see Figure 14), and the option byte (software watchdog ,...).

Figure 14: MCU Configuration


MCU Configuration

inDART Model

Hw Code: INDART-ST7LITE0

Device and Option Bytes

Device Code: ST7FLITE05B

ST7FLITE05B
ST7FLITE05M
ST7FLITE09B
ST7FLITE09M

4.3 EXERCISE 3: HOW TO USE THE DATA EEPROM

This exercise consists of creating a program that writes and reads data in EEPROM.

This program is divided into two parts:

- Programming a table of values in the data EEPROM (we will take the character string: “STMicroelectronics” as a table).
- Reading the data EEPROM and displaying what is read on the LEDs.

You read the values by pressing the SW2 push-button.

In order to make the code easy to understand, the program is divided into the following parts:

- Initialize the ST7FLITE0 properly (Reset routine).
  - Initialize PA4 to PA0 as output push-pull
  - Initialize PA7 as input pull-up
  - Initialize PB2 to PB0 as output push-pull
- Write the two subroutines (Writing subroutine and Reading and Display subroutine).

NOTES:

- Both routines will be called in the Main.c.
- LEDS connected to PA0 to PA3 display the 4 LSB, and LEDs connected to PA4, PB0 to PB2 display the 4 MSB.
- Please refer to the ST7FLITE0 datasheet to get information on the peripherals.

4.3.1 Exercise 3 in Assembly and C (Cosmic and Metrowerks) languages

You must modify the files located in the C:\Exercise\Init\Ex3 directory to write a correct program.

Follow the comments and replace all '?????' with the corresponding code. When everything is corrected, write the Ex3.bat batch file to assemble, link and create a .S19 file (give it any name you want). The batch file also generates the S19, SYM, MAP and OBJ files. Use the technical manual for help.

Note:
Use the C:\Exercise\Init\Ex3\Asm directory for Exercise 3 in Assembler language.
Use the C:\Exercise\Init\Ex3\C_ cosmic directory for Exercise 3 in Cosmic language.
Use the C:\Exercise\Init\Ex3\C_ metro directory for Exercise 3 in Metrowerks language.

4.4 EXERCISE 4: BASIC USE OF THE ADC

This exercise is divided into two parts :

- Ex4a : the purpose of this program is to convert an analog value from the PB3 port (connected to a trimmer) into a 8-bit digital value and to display the converted value on LEDs.
- Ex4b : the purpose of this program is to read the analog input of the PB3 port pin and to display the voltage value on LED indicators as follows:
  - 0V-1.5V LED connected to PB0
  - 1.5V-3.5V LED connected to PB1
  - 3.5V-5V LED connected to PB2

In the ADC directory, you will find the ST7FLITE0 environment in assembly language as well as the structure of the main module and the batch file for compiling efficiently.

The programs in the ex4a.asm module and in the ex4b.asm module have to be modified as follows:

- The 'ST7_init' a sub-routine for initializing the ports and the ADC .
- The main program which waits for the end of the current conversion and outputs the suitable result on in the LEDS.

4.4.1 Exercise 4 in Assembly and C (Cosmic and Metrowerks) languages

You must modify the files located in the directory: C:\Exercise\Init\Ex4 to write a correct program.

Follow the comments and replace all '?????' with the corresponding code. When everything is corrected, write the a.bat batch files (ie. Ex4a.bat and Ex4b.bat) to assemble, link and create a .S19 file (give it any name you want). The batch file also generates the S19, SYM, MAP and OBJ files. Use the technical manual for help.

Note:

Use the C:\Exercise\Init\Ex4(a and b)\Asm directory for Exercise 4 in Assembler language.

Use the C:\Exercise\Init\Ex4(a and b)\C_ cosmic directory for Exercise 4 in Cosmic language.

Use the C:\Exercise\Init\Ex4(a and b)\C_ metro directory for Exercise 4 in Metrowerks language.

4.5 EXERCISE 5: HOW TO USE THE SERIAL PERIPHERAL INTERFACE (SPI)

This exercise consists of writing two programs to allow two ST7FLITE0 devices to communicate with each other via SPI.

Then this exercise is divided into two parts (programs):

- Configure the Master device. Then write the transmit routine to transfer a byte.
- Configure the Slave device. Then write the reception routine to recover the byte.

The read value will have to be displayed on the LEDs.

In order to make the code easy to understand, the program is divided into the following parts:

- Initialize the ST7FLITE0 properly (Reset routine).
  - Initialize PA4 to PA0 as output push-pull
- Write the two routines (Master Mode and Slave Mode routines).

NOTES:

- Both routines will be called in the Main.c
- Please refer to the ST7FLITE0 datasheet to get information on the peripherals.

4.5.1 Exercise 5 in Assembly and C (Cosmic and Metrowerks) languages

You must modify the files located in the directory C:\Exercise\Init\Ex5 to write a correct program.

Follow the comments and replace all '?????' with the corresponding code. When everything is corrected, write the Ex5.bat batch files to assemble, link and create a .S19 file (give it any name you want). The batch file also generates the S19, SYM, MAP and OBJ files. Use the technical manual for help.

Note:

Use the C:\Exercise\Init\Ex5\Asm directory for Exercise 5 in Assembler language.

Use the C:\Exercise\Init\Ex5\C_ cosmic directory for Exercise 5 in Cosmic language.

Use the C:\Exercise\Init\Ex5\C_ metro directory for Exercise 5 in Metrowerks language.

To test the Exercice 5 we must use two InDart Demo Boards and connect them via the Vss, MOSI and SCK pins .

4.6 EXERCISE 6: SWITCH ON A LED EVERY 0.5 SECONDS

This exercise can be done in 3 steps.

- Switch a LED on and off every 0.5 seconds
- Shift the LED from Px to Px+n with the previous time base
- Double the frequency by pressing the board push-button.

The result of each step of your application can be checked on the demo board.

4.6.1 First part

The purpose of the first part is to use the output compare capability of the timer to create a real time clock. The internal time basis is 0.5 sec. Every 0.5 seconds a timer interrupt switches on a LED connected to PA0 and switches it off after the same period.

- Write a routine called “ST7_init” which contains the output port configuration (push-pull output).
- Write a routine called “init_timer” which contains the code for configuring the timer (output compare interrupt enable, fCounterClock = fLTIMER, fills the output compare 1 register with the time base value). This value is declared as a word and is initialized in the constant files.
- Write the timer interrupt routine “artim_oc_rt” which switches the LED connected to PA0 on or off, adds to the content of DCR0 the value corresponding to 0.5 sec and clears CMPF0 (compare flag)

The main program calls the two initialization routines and enters an infinite loop after resetting the interrupt mask.

4.6.2 Second part

Using the same time base and the previous code:

- Modify the timer interrupt routine to shift the lit LED from Px to Px+1 . The content of the ports to be displayed is stored in the 8-bit word “shiftdata”. This data has to be declared in the constant files and initialized in the main program.
- In the “ST7_init” routine, configure the necessary ports as push-pull outputs.

There will be no wait time between the moment the lit LED switches from PB2 to PA0.

4.6.3 Third part

Using an external interrupt on PA7 (pressing the push-button to hold the PA7 pin low) double the timer frequency or divide it by 2.

- Write the external interrupt routine ‘ext1_rt’ which processes the frequency change every time you detect the button has been pressed. You can change the timer frequency by modifying the time base value.

- Modify the “init” routine to configure the PA7 pin as input with interrupt and to select the falling edge interrupt sensitivity.

Then you have to create a new project under STVD7.

4.6.4 Exercise 6 in Assembly and C (Cosmic and Metrowerks) languages

You must modify the files located in the directory C:\Exercise\Init\Ex6 to write a correct program.

Follow the comments and replace all ‘?????’ with the corresponding code. When everything is corrected, write the a.bat batch files (Ex6a.bat, Ex6b.bat and Ex6c.bat) to assemble, link and create a .S19 file (give it anyname you want). The batch file also generates the S19, SYM, MAP and OBJ files. Use the technical manual for help.

Note:

Use the C:\Exercise\Init\Ex6(a, b and c)\Asm directory for Exercise 6 in Assembler language.

Use the C:\Exercise\Init\Ex6(a, b and c)\C_ cosmic directory for Exercise 6 in Cosmic language.

Use the C:\Exercise\Init\Ex6(a, b and c)\C_ metro directory for Exercise 6 in Metrowerks language.

The correct files are located in the following directory: C:\Exercise\Result\Ex6 (a, b and c).