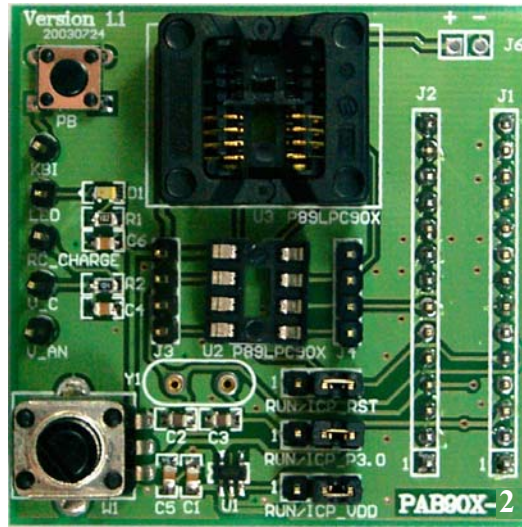


PAB90x-2

Programming and Evaluation Board for LPC906, LPC907, LPC908



Getting Started Guide

2003 August 15

Revision History

Version	Date	Notes
1.0	08/15/03	First release

Table of Contents

1. Overview
2. PAB90x-2 Programming Board Setup
3. Programming the P89LPC90x using PAB90x-2
4. Running User code on the LPC90x using PAB90x-2
5. Evaluating the P89LPC90x using PAB90x-2
6. Code Generation for LPC900 using Code Architect

1. Overview

PAB90x-2 (V1.1 and up) is a Programming and Evaluation Adapter Board for the Philips 8-bit microcontrollers LPC906, LPC907, LPC908 in an 8-pin package. (Whenever “LPC90x” is mentioned in this document, it refers to LPC906/907/908. To program other LPC90x devices, e.g. LPC901/902/903, you need a different programming adapter board called PAB90x-1.)

The PAB90x-2 board is designed as an add-on board to the MCB900 board from Keil Software. (More information: <http://www.keil.com/mcb900>)

PAB90x-2 Programming functionality:

To program the 1K of on-chip Flash memory of LPC906/907/908 devices with PAB90x-2, an MCB900 board from Keil Software is necessary. Before any devices can be programmed, the P89LPC932 on the Keil MCB900 board needs to be programmed with the ISP-to-ICP bridge code using the ISP software FlashMagic (included on the CD). FlashMagic then also supports the programming of LPC90x devices, however only in combination with the ISP-to-ICP bridge code on the MCB900 as noted above. See sections 2 and 3 for details.

PAB90x-2 Evaluation Board functionality:

After the user code has been programmed using PAB90x-2 + MCB900 as described above, PAB90x-2 can be used as a small stand-alone board to evaluate the user software. The following elements are available for evaluation:

- **LED:**
PAB90x-2 provides one low-active green LED (D1) that can be hooked up to any port pin of the LPC90x using the LED header pin.
- **Potentiometer:**
PAB90x-2 provides a potentiometer, which delivers 0V to 3.3V to the V_AN (“analog voltage”) header pin.

- **Push Button:**
A push button (PB) is provided as an external trigger in an application or to wake up the LPC90x from Power-Down mode.
- **Crystal Oscillator socket**
Different Crystal oscillator values can be tested in socket Y1.
A 12MHz crystal is provided.
- **Access to LPC90x port pins:**
The header rows J3 and J4 provide access to the 8 port pins of the LPC90x.

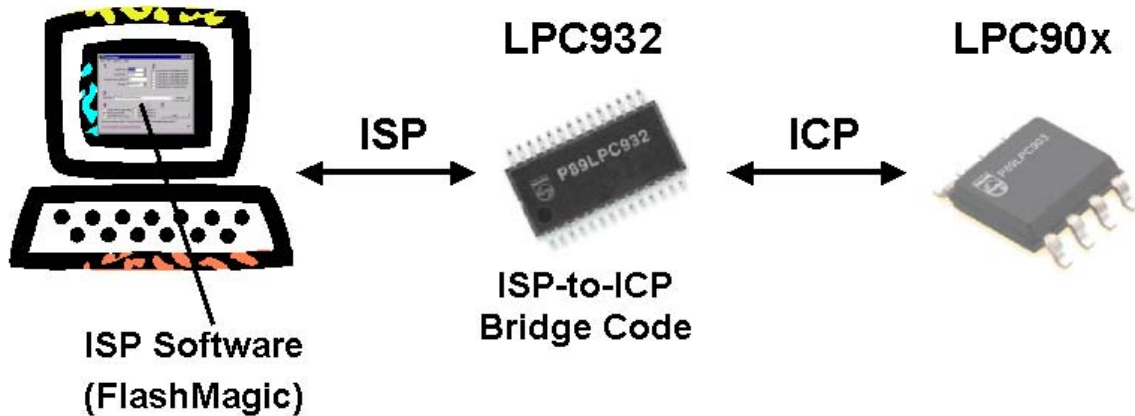
For details see section 5, *"Evaluating the LPC90x using PAB90x-2"*

2. PAB90x-2 Programming Board Setup

Introduction

The LPC90x devices are programmed using a programming method called “**ICP**” (**In-Circuit Programming**). In contrast to some of the larger LPC900 family members, the LPC90x devices do not offer other programming methods like Parallel Programming, In-System Programming (ISP) or complete In-Application Programming (IAP). Details on the ICP specification can be found on the included CD (folder “ICP Programming Specification”)

FlashMagic is the preferred software used to perform ISP on the Philips Flash Microcontrollers that support this programming method. To provide the same programming Interface but still do the ICP programming method that the LPC90x devices require, Philips Semiconductors developed **ISP-ICP bridge software** that can translate between ISP and ICP. In other words, if this ISP-to-ICP bridge code is downloaded to a microcontroller (e.g. LPC932), FlashMagic can be used to program the LPC90x devices (see figure below).



The ISP-ICP bridge application

As already mentioned in the overview, the PAB90x-2 programming board is designed as an add-on board to the MCB900 board from Keil Software. Before any P89LPC90x devices can be programmed, the LPC932 microcontroller on the Keil MCB900 board needs to be programmed with the ISP-ICP bridge code using the ISP software FlashMagic (included on the CD). FlashMagic then also supports the programming of P89LPC90x devices, however only in combination with the ISP-to-ICP bridge code on the MCB900 as noted above.

Step 1) Install FlashMagic from the included CD

Go to folder “FlashMagic” on the included CD ROM and execute “FlashMagic.exe”. Follow the instructions to install FlashMagic.

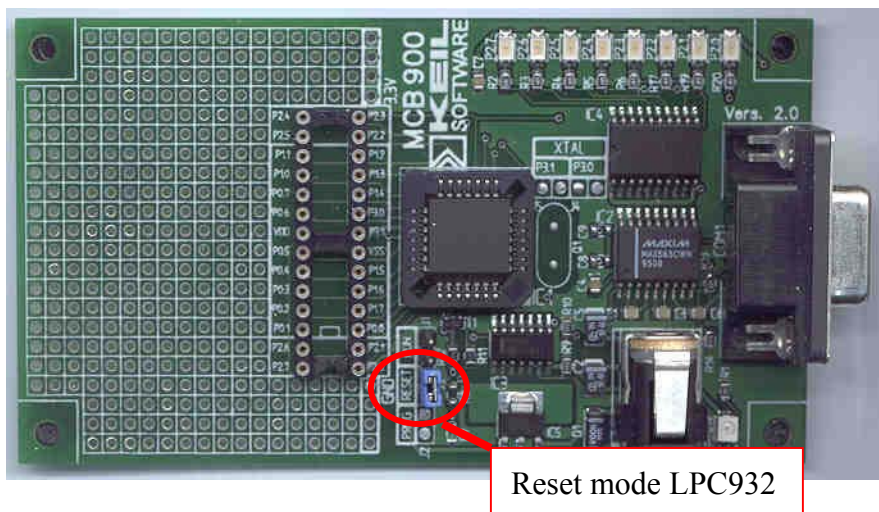
(The latest version of FlashMagic can always be found at

www.esacademy.com/software/flashmagic)

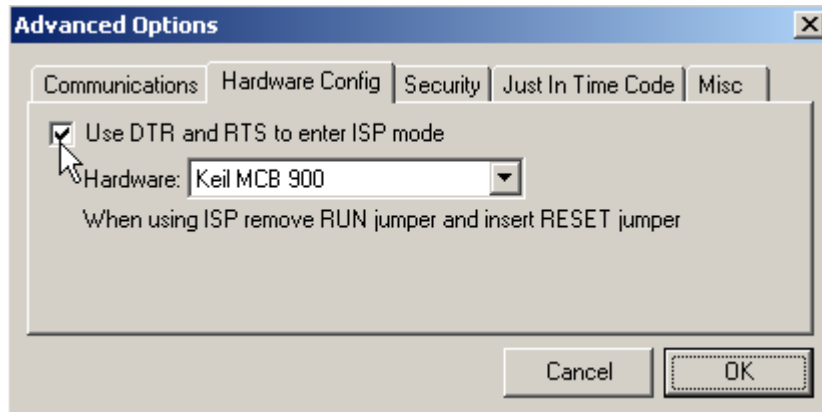
Step 2) Program the LPC932 device on the MCB900 board with the ISP-ICP bridge code using the FlashMagic software you installed in step 1

The ISP-ICP bridge code is included on the CD (folder “ISP ICP bridge source code”).

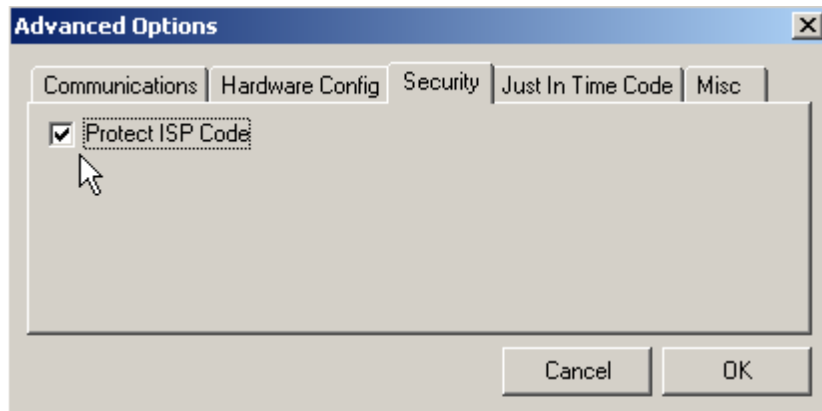
- Set the jumper on the MCB900 board to the RESET position:



- Connect the MCB900 to your PC COM port using a serial cable.
- Power up the MCB900 board.
- Start FlashMagic (Start | Programs | FlashMagic | FlashMagic)
- Go to “Options | Advanced Options | Hardware Config” and make sure the box “Use DTR and RTS to enter ISP mode” is clicked:

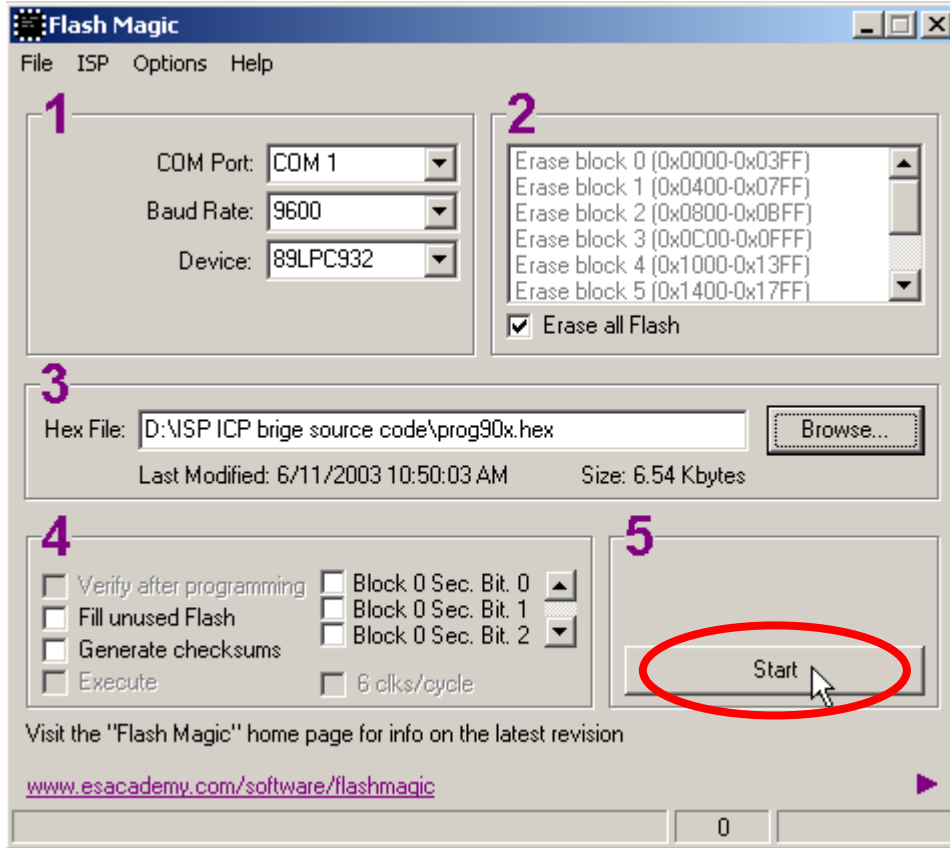


- Go to “Options | Advanced Options | Security” and make sure the “Protect ISP code” option is selected:



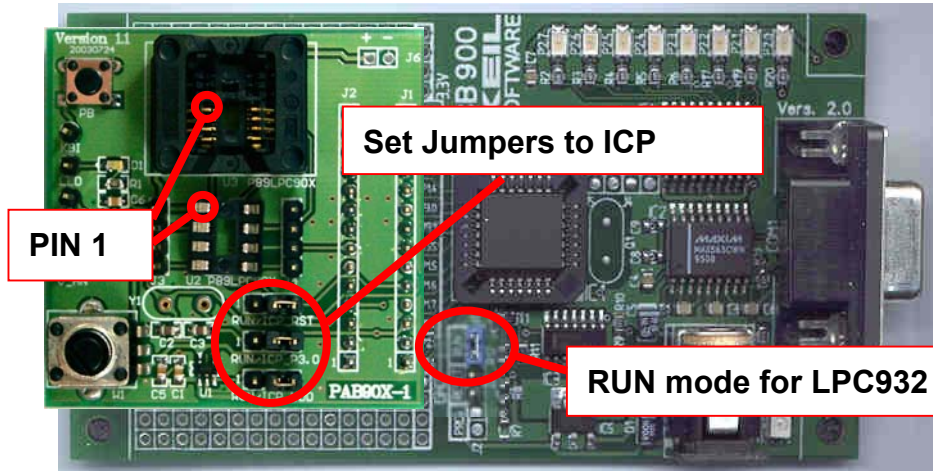
- Select the COM port you will be using to program the LPC932.
- Select the P89LPC932 as the device to be programmed
- Select baud rate 9600
- Browse to the prog90x.hex file that is on the CD in the “ISP ICP bridge source code” folder
- Check the “erase all” Flash box.

- Click start program the ISP-ICP bridge code into the P89LPC932:



Step 3) Attach the PAB90x-2 board to the MCB900 board

- Disconnect the power cable from the MCB900 board
- Set the jumper on the MCB900 board from “Reset” to “RUN”
- Plug the PAB90x-2 programming board onto the MCB900. Make sure all 28 pins align correctly. Be aware of the locations for pin position 1 for when you insert a device later on (see image below).



NOTE: If you have an MCB900 board without a 28-pin DIP socket, please use the included 28-pin DIP socket and solder it into the MCB900 board.

NOTE: There are 3 jumpers on the PAB90x-2 board, these have to be all set to the position “ICP_xxx” closest to the edge of the board in order for the LPC90x to enter ICP mode.

Your MCB900 / PAB90x-2 programming hardware is now set up.
To program a device, continue with the following section.

3. Programming the LPC90x using PAB90x-2

Once all setups from the previous chapter have been made, using FlashMagic the LPC90x devices can be programmed in 5 steps corresponding to the numbered steps on the FlashMagic User Interface.

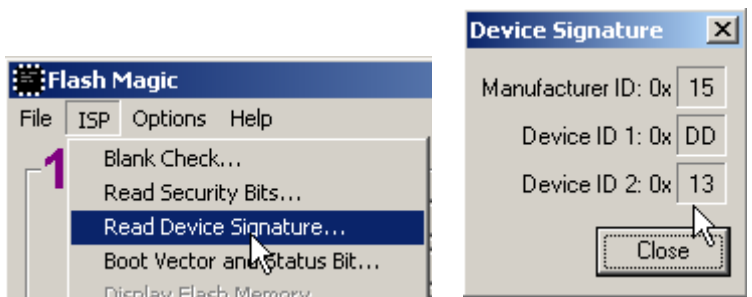
Step 0.

- Disconnect the power cable from the MCB900 board
- Insert an LPC90x device in **EITHER** the SO8 socket **OR** the DIP8 socket of the PAB90x-2 board. **Make sure PIN 1 is aligned correctly.**
- **Make sure all jumpers are set to “ICP_....”.**
- **Disconnect any cables from J3 and J4 as connected hardware might interfere with the programming signals.**
- Plug the power cable back in to enter ICP mode on the LPC90x (the ICP mode is only entered on power-up of the MCB900). LED P2.3 on the MCB900 board should light up.
- Start FlashMagic (Start | Programs | FlashMagic | FlashMagic).

Step 1.

- Select the COM port (make sure it is not already taken by another application, e.g. some PDA software)
- Select a baud rate of 19200 baud (the ISP-ICP bridge application has a fixed baud rate of 19200 baud)
- Select the LPC90x device you would like to program (the PAB90x-2 board can program LPC906, LPC907, and LPC908. (For programming of LPC901 / LPC903 / LPC903, you need the PAB90x-1 board due to the differences in pinout.)

- At this point you can do a quick read of the device signature bytes to make sure you're connected to the LPC90x:



If FlashMagic cannot read the Device Signature or you get all FF's, please check your setup. The correct signature bytes are:

LPC906: 15h DDh 11h; **LPC907:** 15h DDh 12h; **LPC908:** 15h DDh 13h

Step 2.

Select which Flash blocks should be erased before the programming operation.

Step 3.

Select the hex file to be programmed.

Step 4.

Can be used to set security bits and fill unused Flash (optional)

Step 5.

Start programming!

“Hello World” Example

Provided on the CD provided is some example code for the LPC907/908, which uses the UART to send out the string “Hello World :)”. (The LPC906 does not have a UART.)

The example code can be programmed using the 5 steps described above.

To run the user code after you programmed it, please refer to section 4.

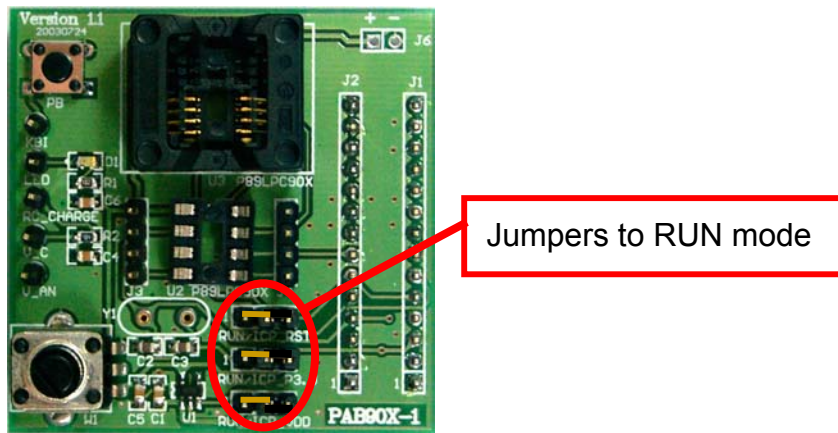
4. Running User code on the LPC90x using PAB90x-2

When a device has been successfully programmed, use the following steps to run the programmed code:

Step 1: Disconnect the power cable from the MCB900 board

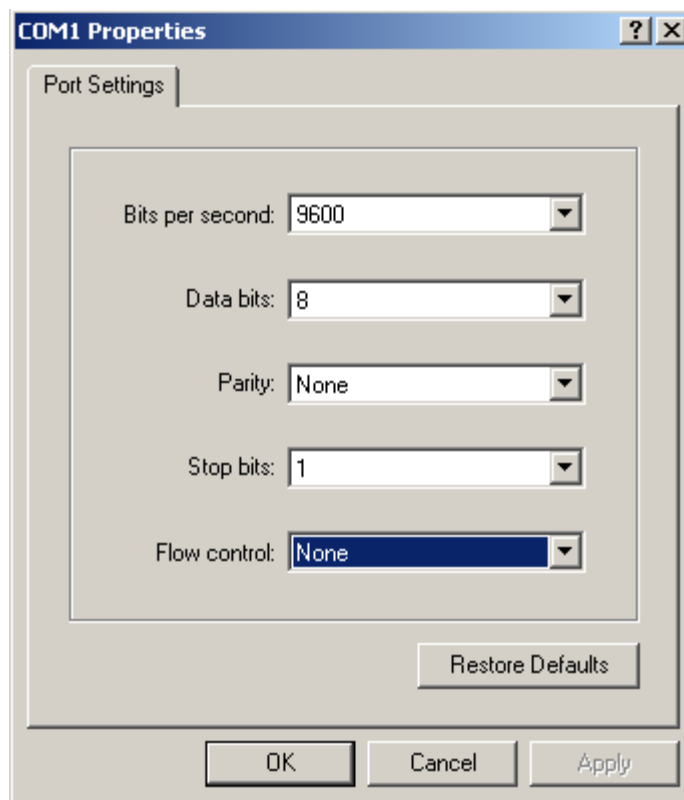
Step 2: Put the 3 jumpers on the PAB90x-2 board to RUN mode

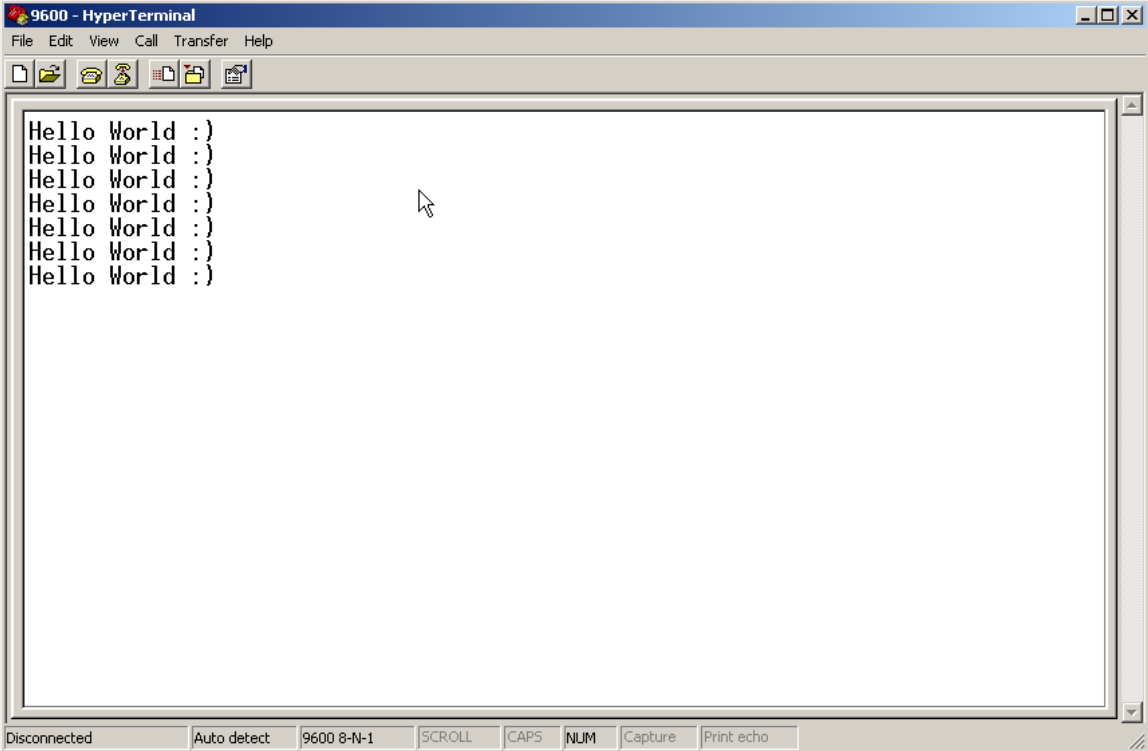
Step 3: Reconnect power – your user code is now running.



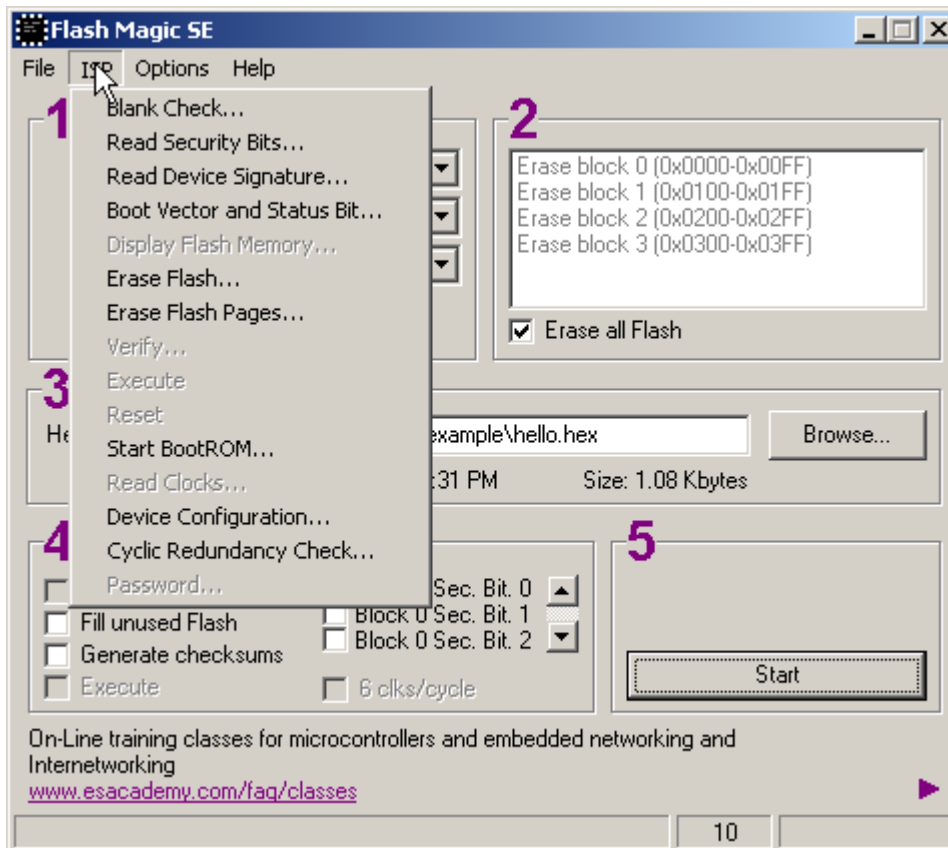
To see the UART sending the string from the “Hello World” example you can open a terminal program, e.g. HyperTerminal:

- To have access to the COM port of the PC, start the program 'HyperTerminal'. (START | PROGRAMS | ACCESSORIES | COMMUNICATIONS | HYPERTERMINAL)
- Click on 'Properties' and check the following settings: 9600 baud, 8 data bits, 1 stop bit and no flow control.
- The P89LPC907/908 will send "**Hello World :)**" through the UART with 1 sec delays, which is displayed on the screen.





Other functions can be found in the ISP menu:



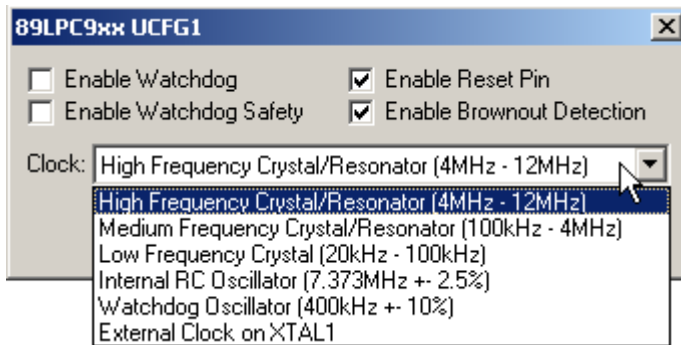
5. Evaluating the P89LPC90x using PAB90x-2

As described in section 4, the user code that has been programmed using PAB90x-2 + MCB900 can be run and PAB90x-2 can now be used as a small stand-alone board to evaluate the user software. The following elements are available for evaluation:

- **LED:**
PAB90x-2 provides one low-active green LED (D1) that can be hooked up to any port pin of the LPC90x. A LOW level at the LED header pin turns the LED on, a HIGH level turns it off.
- **Potentiometer:**
PAB90x-2 provides a potentiometer, which delivers 0V to 3.3V to the V_AN (“analog voltage”) header pin. This analog voltage can directly be fed to the on-chip comparator on the LCP90x or the header pins V_C and RC_CHARGE can be used to do a slope conversion Analog-to-Digital Converter (together with resistor R2 and capacitor C4). This slope conversion ADC is described in detail in Application Note AN10187, “Low-cost A/D-Conversion with Philips LPC Microcontrollers”. This Application Note (among others) is included on the CD.
- **Push Button:**
A push button (PB) is provided as an external trigger in an application or to wake up the LPC90x from Power-Down mode using the keyboard interrupt feature. If header pin KBI is hooked up to a port pin in quasi-bidirectional mode (providing an internal pull-up resistor), the level at the port pin will be HIGH if the button is not pressed. It will go LOW when the button is pressed. Please note that the button might “bounce”, so you might get several transitions at the port pin.

- **Crystal Oscillator socket**

Different Crystal oscillator values can be tested for the LPC906 in socket Y1. To use an external crystal, FlashMagic offers you to reprogram UCFG1, the User-Configuration Register of the LPC90x devices (ISP | Device Configuration):

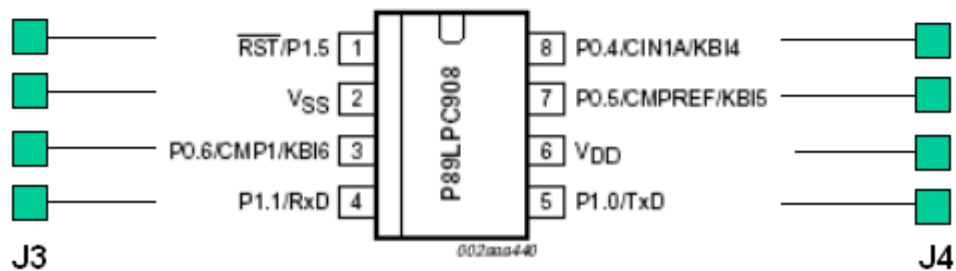
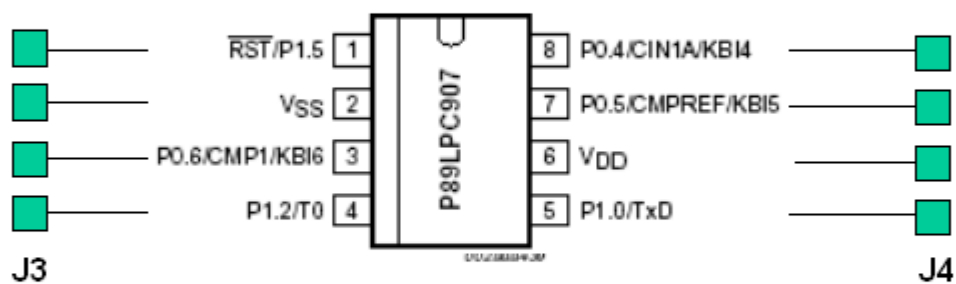
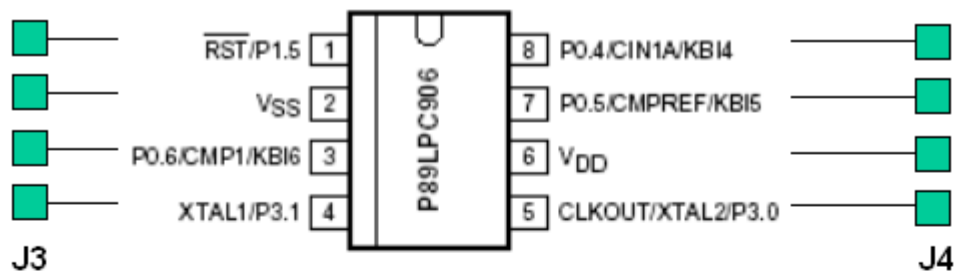


Please note: Only LPC906 features external crystal pin inputs. LPC907 and LPC908 can only be run from the internal RC oscillator or the internal Watchdog oscillator.

Please note: The capacitors C2 and C3 are the capacitors belonging to the crystal oscillator circuit. These capacitors are just recommendations. Since many factors play a role in oscillator circuits (board design, board layout, tolerances of components etc.), it is **HIGHLY RECOMMENDED** to do extensive testing of the oscillator circuit in your own hardware to make sure your oscillator circuitry works reliably under different conditions like temperature and voltage.

- **Access to LPC90x port pins:**

The header rows J3 and J4 provide access to the 8 port pins of the LPC90x (see next page).



Using the LED, the Potentiometer and the Push-Button in a pre-programmed example

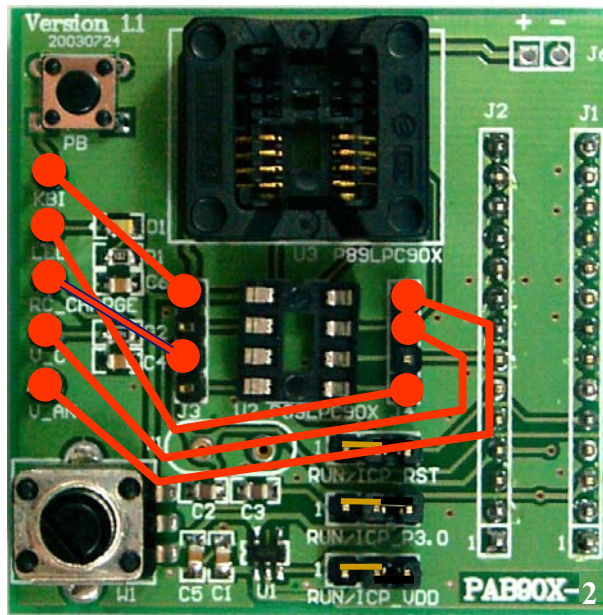
There is an example program preprogrammed into the device sample that came with your PAB90x-2 board. The source code is included on the CD, in folder “LPC906_7_8 LED blinker and SD ADC”.

The software blinks the LED with a frequency that depends on the position of the potentiometer. While the push button is pressed, the device is held in Reset.

To run the software, do the following:

(please also see “1_minute_demo_setup.pdf” on the included CD):

- If you already erased the sample that came with the PAB90x-2 board: Program the “D:\LPC906_7_8 LED blinker and SD ADC\blinky2.hex” file into the LPC90x as described in section 3.
- Connect the KBI, LED, RC_CHARGE, V_C and V_AN header pins to the LPC90x as described in the image below:



- Make sure all three jumpers are set to “RUN” (left position)
- Attach the provided battery pack to J6 (red cable = “+”) and turn it ON
- The LPC90x demo runs.
 - The LED will blink according to analog voltage selected with potentiometer (turn potentiometer to adjust frequency)
 - Push Button will reset the application

The Demo code makes use of the on-chip analog comparator on the LPC90x. Together with resistor R2 and capacitor C4, the software uses the on-chip analog comparator as a **Sigma-Delta Analog to Digital Converter**. This method is described in detail in **AN10187, “Low-cost A/D-Conversion with Philips LPC Microcontrollers”**. This Application Note (among others) is included on the CD. The digital representation of the analog voltage provided by the potentiometer controls the frequency with which the LED is turned on and off.

While the push button is pressed, it holds the application in Reset via a LOW signal on the Reset pin. Please note that the LPC90x is equipped with an **on-chip Power-On Reset (POR) circuit** that automatically provides a clean reset signal to the device on power-up. The external Reset pin is **ONLY** needed if the chip needs to be reset manually or by another device in the application. Most of the time, the internal power-on reset is sufficient and this pin (P1.5) becomes a valuable general-purpose input pin.

6. Code Generation for LPC900 using

For faster time to market, the Embedded Systems Academy (www.esacademy.com) developed a Code Generation tool for Philips LPC900 called **Code Architect**.

Code Architect is a Web and PC based software tool that allows source code to be quickly and easily generated for Philips Microcontrollers. By selecting user-friendly input options, customized C source code will be generated for Keil and Raisonance 8051 Compilers with only a few mouse clicks. The source code may then be cut and pasted or saved and added to a Keil or Raisonance project.

Code Architect allows multiple complex peripherals to be configured and used with a minimum of effort and time.

Code Architect Versions

Code Architect comes in two versions - Internet and Windows.

The Internet version is available at www.codearchitect.org and provides all the features of Code Architect except automated saving of generated files. However, generated code may be cut and pasted from the web page. Note that generated code is not retained. Nor is it linked to any specific users or IP addresses.

The Windows version is available for download by following the relevant link at the bottom of the page. It is a 32-bit Windows application and provides all the features of Code Architect plus the ability to automate saving the generated files. The system requirements are:

- Internet Explorer 4.0 or later
- Javascript enabled in Internet Explorer
- 3Mb disk space
- Pointing device

The Internet and Windows versions of Code Architect share a common project file format, therefore projects saved in one version may be loaded into the other version. This allows access to your Code Architect projects regardless of whether you are using a public PC, friend's PC, on a business trip, at the office or whether you have an Internet connection or not. The user interface is nearly identical in both Internet and Windows versions providing familiarity and ease of use.

Using Code Architect

Code Architect is very straightforward to use.

Across the top of the page there is the project title and links to project options, such as saving a project, loading a project and creating a new project. When Code Architect is first started a new project with the name "Untitled" is automatically created. When code is generated, the settings will automatically be saved to the project. Once you have finished generating code you may choose the Save Project option to save your project file.

Note: If you do not save your project file you will lose all settings you have made since starting Code Architect. The Windows version will ask you if you wish to save the project when you attempt to close the application. The Internet version will not. Also if you stop using the Internet version for 30 minutes or more, you will lose all settings you have made since accessing the web site.

Generating code is divided into three steps:

Step 1 is to select the peripheral or module you wish to generate code for.

1. Select the desired microcontroller and peripheral:

[Start](#) > [89LPC932](#) > [I2C](#)

Selection of the correct device first will be required before you are presented with a list of peripherals available for that device. Simply click on the name of the device or device family followed by the peripheral.

Please note that by selecting the LPC932, code can be generated for many other devices (LPC90x, LPC91x, LPC92x) since these devices share the same peripherals – in other words, the LPC932 is a superset of the other devices.

Step 2 is to enter the options for the peripheral.

2. Configure the I2C and click on Generate:

Input

CPU Clock Frequency (MHz):

Options

Clock Source:

Data Rate (kHz): (0 - 400kHz)

Interrupt

Interrupt Register Bank:

Interrupt Priority:

Interrupt function name:

Module Interface Functions

Initialization function name:

Transmit function name:

Receive function name:

Callback Functions

Master Transmitter Get Byte function name:

Master Receiver Received Byte function name:

Master Transmitter Last Byte function name:

Master Receiver Last Byte function name:

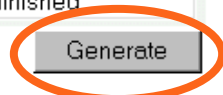
Slave Transmitter Get Byte function name:

Slave Receiver Received Byte function name:

Slave Transmitter Last Byte function name:

Slave Receiver Last Byte function name:

Transfer Finished function name:



The options have been made as user-friendly as possible, however sometimes reference to the datasheet will be required to understand some of the options.

Once selections have been made, clicking on the Generate button will generate customized C source code based on your selections.

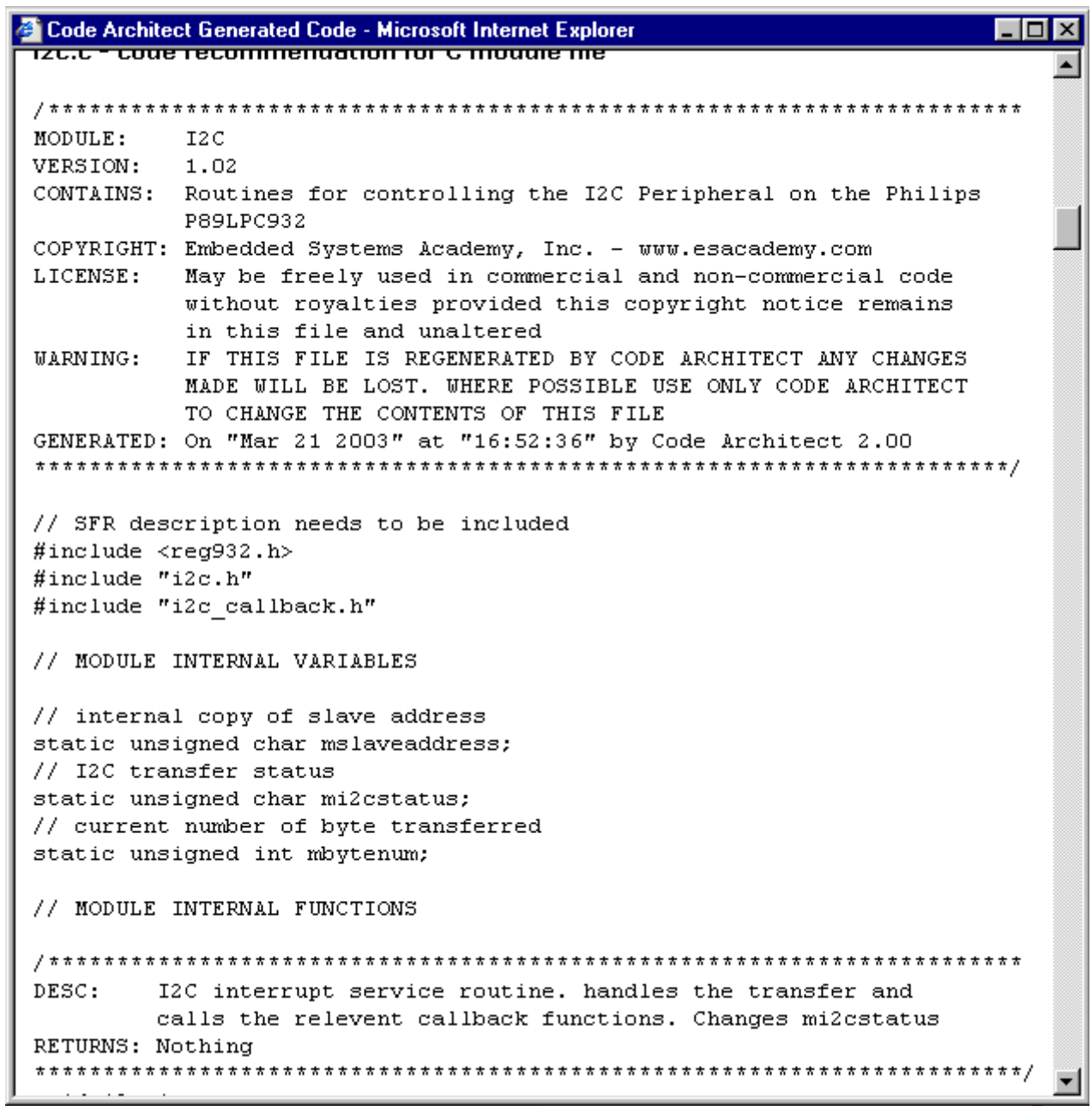
In some situations, notes may be displayed along with the generated code. Notes are shown immediately before the generated code. Typical reasons for notes appearing include:

- Attempting to use a baud rate that is not possible. In this case the actual baud rate that will be generated will be given along with the error.
- Flash locations in the device must also be configured along with using the generated code. In this case the relevant settings for the Flash locations will be given.

Resource conflicts may also be shown along with the generated code. Resource conflicts occur when more than one peripheral that has been configured uses the same device pin or other resource such as a timer. When a resource conflict occurs, the resource in question along with which peripherals use it are shown.

Step 3 is to either save or cut and paste the generated code.

3. Copy generated code:



```
Code Architect Generated Code - Microsoft Internet Explorer
I2C.C - Code Recommendation for C module file

/*****
MODULE:      I2C
VERSION:    1.02
CONTAINS:   Routines for controlling the I2C Peripheral on the Philips
            P89LPC932
COPYRIGHT:  Embedded Systems Academy, Inc. - www.esacademy.com
LICENSE:    May be freely used in commercial and non-commercial code
            without royalties provided this copyright notice remains
            in this file and unaltered
WARNING:    IF THIS FILE IS REGENERATED BY CODE ARCHITECT ANY CHANGES
            MADE WILL BE LOST. WHERE POSSIBLE USE ONLY CODE ARCHITECT
            TO CHANGE THE CONTENTS OF THIS FILE
GENERATED:  On "Mar 21 2003" at "16:52:36" by Code Architect 2.00
*****/

// SFR description needs to be included
#include <reg932.h>
#include "i2c.h"
#include "i2c_callback.h"

// MODULE INTERNAL VARIABLES

// internal copy of slave address
static unsigned char mslaveaddress;
// I2C transfer status
static unsigned char mi2cstatus;
// current number of byte transferred
static unsigned int mbytenum;

// MODULE INTERNAL FUNCTIONS

/*****
DESC:      I2C interrupt service routine. handles the transfer and
            calls the relevent callback functions. Changes mi2cstatus
RETURNS:   Nothing
*****/
```

Usually, the generated code is provided in the form of a header file and C source file; therefore two files must be saved or created. In both the Internet and Windows version cutting and pasting is provided. In the Windows version the source files may be saved by choosing Save Source Files... from the File menu. A link will be provided in both versions to open the generated code in a separate window as an aid to cutting and pasting.

Support

Support for Code Architect may be obtained from the following location. Note also that a link is provided at the bottom of each Code Architect web page.

Embedded Systems Academy, Inc.

Email: support@codearchitect.org

Feedback

If you have any feedback, bug reports or suggestions regarding Code Architect please send them to support@codearchitect.org