

# *HandsOn<sup>®</sup> Technology*

## **8051 $\mu$ Controller Starter Kits**

***FLASH  $\mu$ CONTROLLER PROGRAMMER/DEVELOPMENT SYSTEM***

***MODEL: HT-MC-02***

***8051 is one of the most popular 8-bit  $\mu$ Controller architectures in use today,  
learn it the practical and HandsOn<sup>®</sup> way  
Suitable for Industrial Embedded Systems Control & Applications***

***From Novice to Expert***

<b>HT-MC-02 Getting Started Manual</b>
--

The **HT-MC-02** is a compact design platform development specifically for use with the Philips P89C51Rx2xxx series of 8051 core flash  $\mu$ controller. **HT-MC-02** is ideal platform for small to medium scale embedded systems development, and quick 8051 embedded design prototyping. **HT-MC-02** can be used as stand-alone Flash programmer for the Philips P89C51Rx2 series  $\mu$ C or as an 8051 development, prototyping and educational platform. Assembled Hex files are programmed via the serial port and stored directly in on-chip Flash Program memory. Programs stored in on-chip flash memory are retained in the absence of power and are automatically run when the Flash Memory **HT-MC-02** board is power on and reset.

**FEATURES:**

- 8051 Central Processing Unit.
- FLASH EPROM internal program memory with block erase.
- On-chip Flash Program Memory with In-System Programming (ISP) and In-Application Programming (IAP) capability.
- Flash memory reliably stores program code even after 10,000 erase and program cycles.
- Internal 1Kbyte fixed boot ROM, contains low level Flash programming routines for code downloading via RS232 cable.
- Supports in-circuit programming for Philips P89C51RX2 series flash 8051  $\mu$ C.
- 6 clocks per machine cycle or 12 clocks per machine cycles operation.
- Three 16-bit timer/event counters.
- LED power status indicator.
- Switch between Program & Run modes with the flip of a switch.
- Onboard processor reset button.
- 32 I/O pins on header connectors located on pcb edge for easy port access.
- On board rectifier and regulator accepting 9~12V AC/DC input supply.
- Additional regulated 5V DC power supply for use in interface board that required power supply.

**1. Ports Description:**

**1.1 Port 0<sup>NOTE1</sup> :**

Port 0 is an open-drain, bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application, it uses strong internal pull-ups when emitting 1s. By inserting jumper into location K2 will enable the external pull-up through the 10K resistors array RP1 to Port 0.

P0.1	P0.3	P0.5	P0.7	GND
P0.0	P0.2	P0.4	P0.6	GND

Pin1

**J3 (Port 0) Header Pin Assignment**

**1.2 Port 1<sup>NOTE1</sup>:**

Port 1 is an 8-bit bi-directional I/O port with internal pull-ups on all pins except P1.6 and P1.7, which are open drain. Port 1 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pull-ups.

P1.1	P1.3	P1.5	P1.7	GND
P1.0	P1.2	P1.4	P1.6	GND

Pin1

**J4 (Port 1) Header Pin Assignment**

**1.3 Port 2<sup>NOTE1</sup>:**

Port 2 is an 8-bit bi-directional I/O port with internal pull-ups. Port 2 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 2 pins that are externally being pulled low will source current because of the internal pull-ups. Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX

@DPTR). In this application, it uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOV @Ri), port 2 emits the contents of the P2 special function register. P2.7 must be an "1" to program and erase the device.

P2.1	P2.3	P2.5	P2.7	GND
P2.0	P2.2	P2.4	P2.6	GND

Pin1

**J5 (Port 2) Header Pin Assignment**

**1.4 Port 3 <sup>NOTE1:</sup>**

Port 3 is an 8-bit bi-directional I/O port with internal pull-ups. Port 3 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pull-ups.

P3.1	P3.3	P3.5	P3.7	GND
P3.0	P3.2	P3.4	P3.6	GND

Pin1

**J6 (Port 3) Header Pin Assignment -- As normal I/O pins**

To configure Port3 as 8bits data bus + 3 control lines LCD interface, the pin assignment as below:

P1.7	P1.5	P3.1	P3.3	P3.5	P3.7	GND
P1.6	P1.4	P3.0	P3.2	P3.4	P3.6	GND

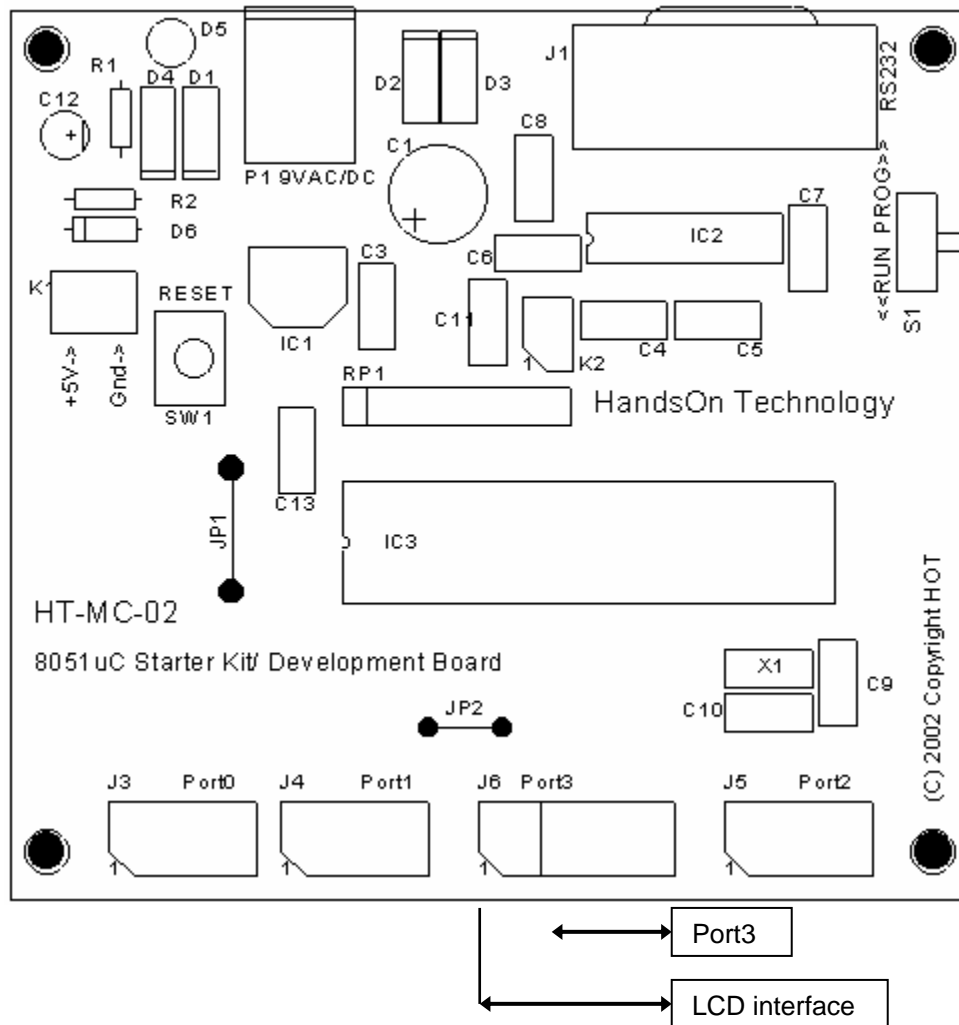
Pin1

**J6 (Port 3) Header Pin Assignment -- As LCD Interface port**

P1.4 ~P1.7 can be assign for LCD RW, EN, RW and Backlit control line, P3.0 ~P3.7 as 8bits LCD data lines. Refer to LCD modules specification for pins assignment. Industry commonly used LCD datasheet is accompanied in the CDROM free with HT-MC-02 Programmer Board.

**NOTE1: REFER TO P89C51RB2/RC2/RD2 DATA SHEET FOR DETAIL DESCRIPTION OF PINS FUNCTIONS.**

**2. Parts Placement Diagram:**



### **3. Using the C Compiler:**

Now it's time to get down to business! What we want to do is to write the first program, translate it and send it to the uC. For this we need some software. We will use the well-known evaluation software from Keil – uVision2 Integrated Development Environment, which is located on the accompanied CDROM.

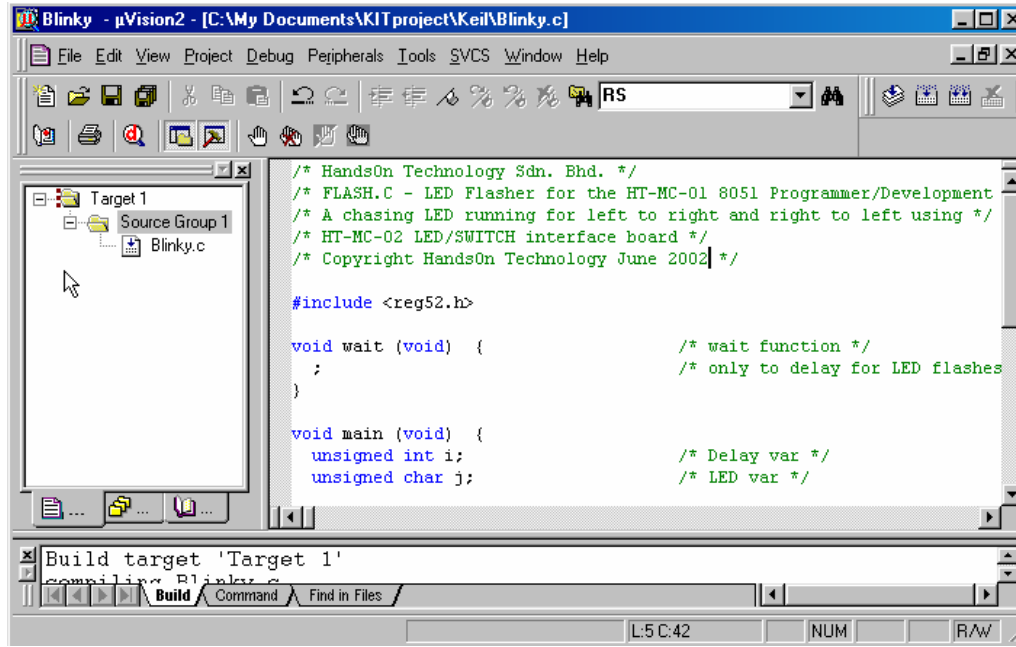
The Keil Software 8051 development tools are programs you use to compile your C code, assemble your assembly source files, link and locate object modules and libraries, create HEX files, and debug your target program.

The  $\mu$ Vision2 IDE is a Windows-based software development platform that combines a robust editor, project manager, and make facility.  $\mu$ Vision2 supports all of the Keil tools for the 8051 including the C compiler (a C Compiler translates a source text into pure machine code), macro assembler, linker/locator, and object-HEX converter.  $\mu$ Vision2 helps expedite the development process of your embedded applications by providing the following:

- Full-featured source code editor,
- Device database for configuring the development tool setting,
- Project manager for creating and maintaining your projects,
- Integrated make facility for assembling, compiling, and linking your embedded applications,
- Dialogs for all development tool settings,
- True integrated source-level Debugger with high-speed CPU and peripheral simulator,
- Links to development tools manuals, device datasheets & user's guides.

The  $\mu$ Vision2 screen provides you with a menu bar for command entry, a tool bar where you can rapidly select command buttons, and windows for source files, dialog boxes, and information displays.  $\mu$ Vision2 lets you simultaneously open and view multiple source files.

Below show the screenshot of the  $\mu$ Vision2 IDE:



This section describes the **Build Mode** of  $\mu$ Vision2 and shows you how to use the user interface to create a sample program. A full user manual for  $\mu$ Vision2 can be found in the accompanied CDROM.

## **4. Creating Projects**

$\mu$ Vision2 includes a project manager that makes it easy to design applications for the 8051 family. You need to perform the following steps to create a new project:

- Start  $\mu$ Vision2, create a project file and select a CPU from the device database.
- Create a new source file and add this source file to the project.
- Add and configure the startup code for the 8051 device
- Set tool options for target hardware.
- Build project and create a HEX file for PROM programming.

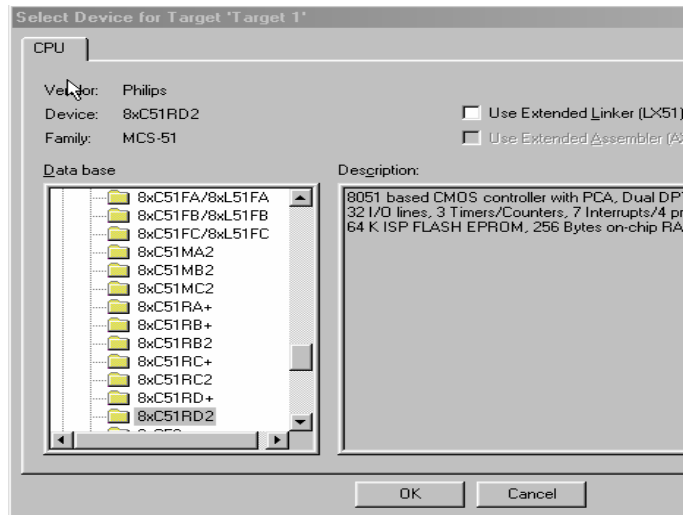
The description is a step-by-step tutorial that shows you how to create a simple  $\mu$ Vision2 project.

### **4.1 Starting $\mu$ Vision2 and Creating a Project (Blinking LED )**

$\mu$ Vision2 is a standard Windows application and started by clicking on the program icon. To create a new project file select from the  $\mu$ Vision2 menu **Project – New Project....** This opens a standard Windows dialog that asks you for the new project file name.

We suggest that you use a separate folder for each project. You can simply use the icon **Create New Folder** in this dialog to get a new empty folder. Then select this folder and enter the file name for the new project, i.e. in this example **Blinky** and click "SAVE" icon.

The "Select Device for Target" dialog box will pop up shows the  $\mu$ Vision device database.



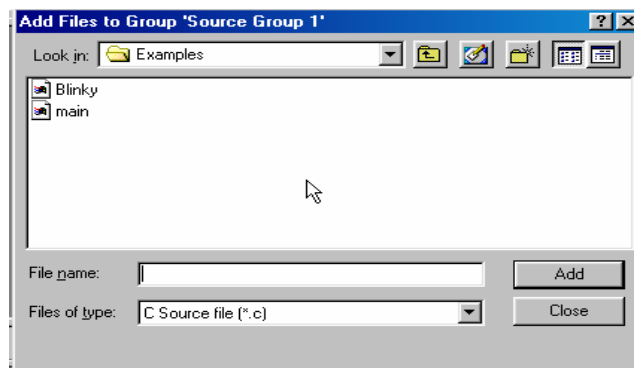
We are using for our examples the Philips P89C51RB2/RC2/RD2 CPU. This selection sets necessary tool options for the P89C51RD2 device and simplifies in this way the tool configuration.

$\mu$ Vision2 creates a new project file with the name **Blinky.UV2** which contains a default target and file group name. You can see these names in the **Project Window**:

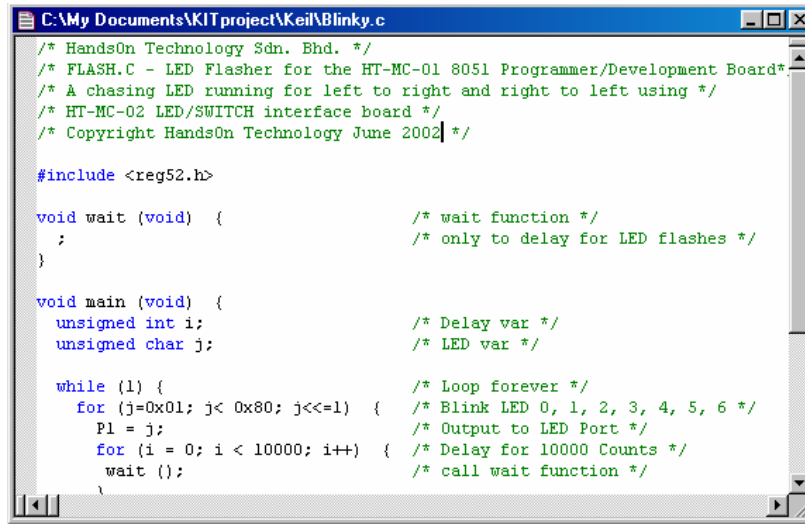


## 4.2 Creating New Source Files

You may create a new source file with the menu option **File – New**. This opens an empty editor window where you can enter your source code.  $\mu$ Vision2 enables the C color syntax highlighting when you save your file with the dialog **File – Save As...** under a filename with the extension \*.C. We are saving our example file under the name **Blinky.C**. Or you can import the blinky.c source file by the following: Move mouse pointer to “Source Group1” icon in the Project Window and right click mouse buttons and select “Add File to Group ‘Source Group 1’”. “Add File to Group ‘Source Group 1’” Dialog appeared.



Select Blinky by double clicking on or select “Add” button follow by “Close” button. You can view the C source file by double clicking the file name in the Project Window.



```
C:\My Documents\KITproject\Keil\Blinky.c
/* HandsOn Technology Sdn. Bhd. */
/* FLASH.C - LED Flasher for the HT-MC-01 8051 Programmer/Development Board*/
/* A chasing LED running for left to right and right to left using */
/* HT-MC-02 LED/SWITCH interface board */
/* Copyright HandsOn Technology June 2002 */

#include <reg52.h>

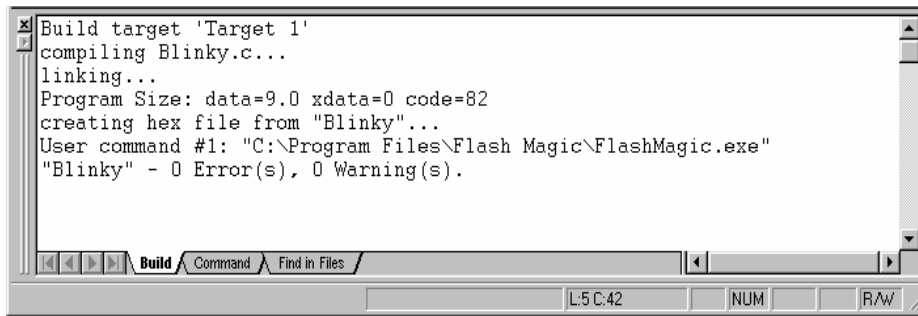
void wait (void) { /* wait function */
; /* only to delay for LED flashes */
}

void main (void) {
unsigned int i; /* Delay var */
unsigned char j; /* LED var */

while (1) { /* Loop forever */
for (j=0x01; j< 0x80; j<<=1) { /* Blink LED 0, 1, 2, 3, 4, 5, 6 */
P1 = j; /* Output to LED Port */
for (i = 0; i < 10000; i++) { /* Delay for 10000 Counts */
wait (); /* call wait function */
}
}
}
}
```

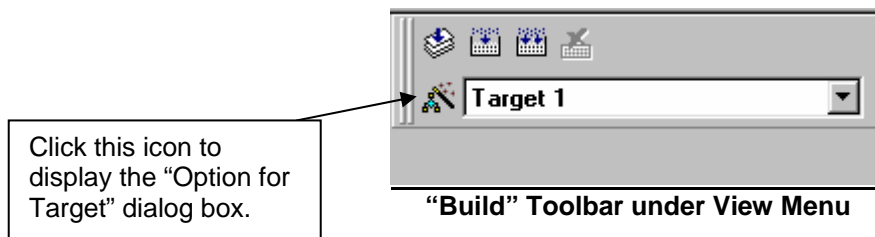
### **4.3 Building Projects and Creating a HEX Files**

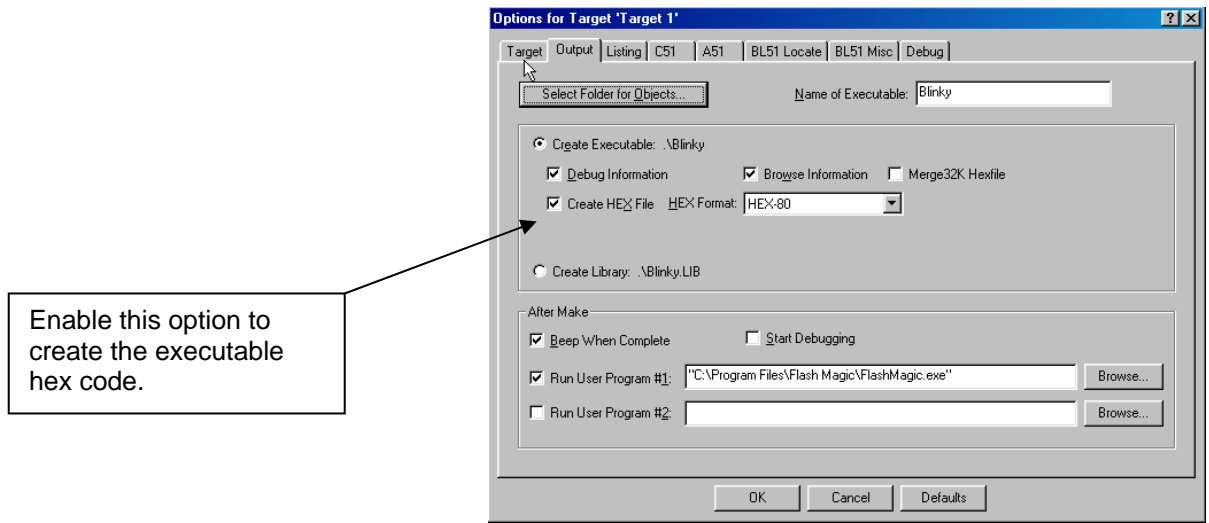
Typical, the tool settings under **Options – Target** are all you need to start a new application. You may translate all source files and line the application with a click on the **Build Target** toolbar icon. When you build an application with syntax errors,  $\mu$ Vision2 will display errors and warning messages in the **Output Window – Build** page. A double click on a message line opens the source file on the correct location in a  $\mu$ Vision2 editor window.



“Output” window under View Menu

After you have tested your application, it is required to create an Intel HEX file to download the software into a Flash programmer.  $\mu$ Vision2 creates HEX files with each build process when **Create HEX file** under **Options for Target – Output** is enabled. You may start your Flash programming utility after the make process when you specify the program under the option **Run User Program #1**.





“Option for Target” dialog box

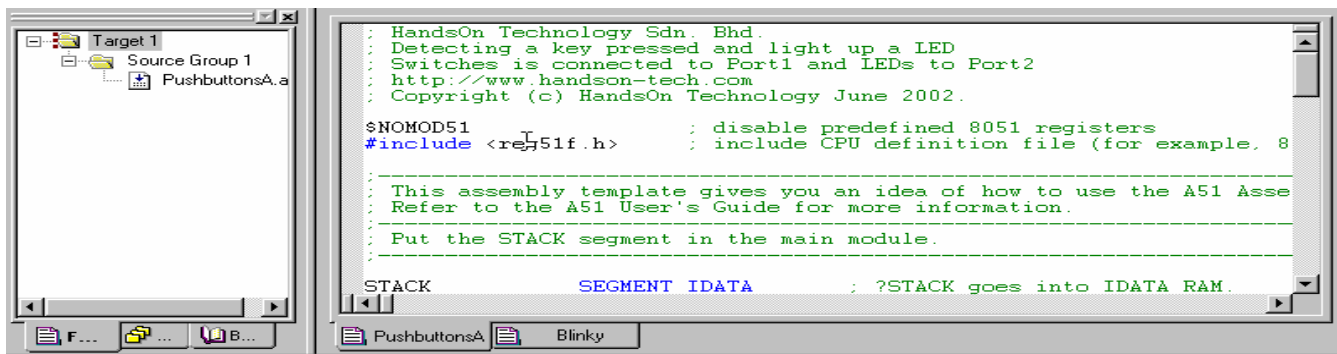
## 5. Using Assembler

### What is an Assembler?

An assembler is a software tool designed to simplify the task of writing computer programs. It translates symbolic code into executable object code. This object code may then be programmed into a  $\mu$ controller and executed. Assembly language programs translate directly into CPU instructions which instruct the processor what operations to perform. Therefore, to effectively write assembly programs, you should be familiar with both the  $\mu$ computer architecture and the assembly language.

This section shows you how to create **8051** program, developed in assembly language. The program will light up one of the eight LEDs when one of the eight push buttons is pressed using 8LEDs+Switches interface board.

Follow the procedure as in Section 4 above to open up a project “Pushbutton.uv2” project file. Assemble the program to create the HEX file to download it to the **HT-MC-02 8051-PROGRAMER/DEVELOPMENT STARTER KIT**.



## 6. Programming the 8051:

### 6.1 Introduction

To load a program into the  $\mu$ C on the **HT-MC-02** Starter Kit, you will need the Windows program **Flash Magic** programming software from Embedded System Academy which can be found on the accompanied CDROM or free



download it from the **Embedded System Academy** Website: <http://www.esacademy.com> and install this to your C drive.

Flash Magic is Windows software from the Embedded Systems Academy that allows easy access to all the ISP features provided by the P89C51Rx2 devices. These features include:

- Erasing the Flash memory (individual blocks or the whole device)
- Programming the Flash memory
- Modifying the Boot Vector and Status Byte
- Reading Flash memory
- Performing a blank check on a section of Flash memory
- Reading the signature bytes
- Reading and writing the security bits
- Direct load of a new baud rate (high speed communications)
- Sending commands to place device in BootROM mode

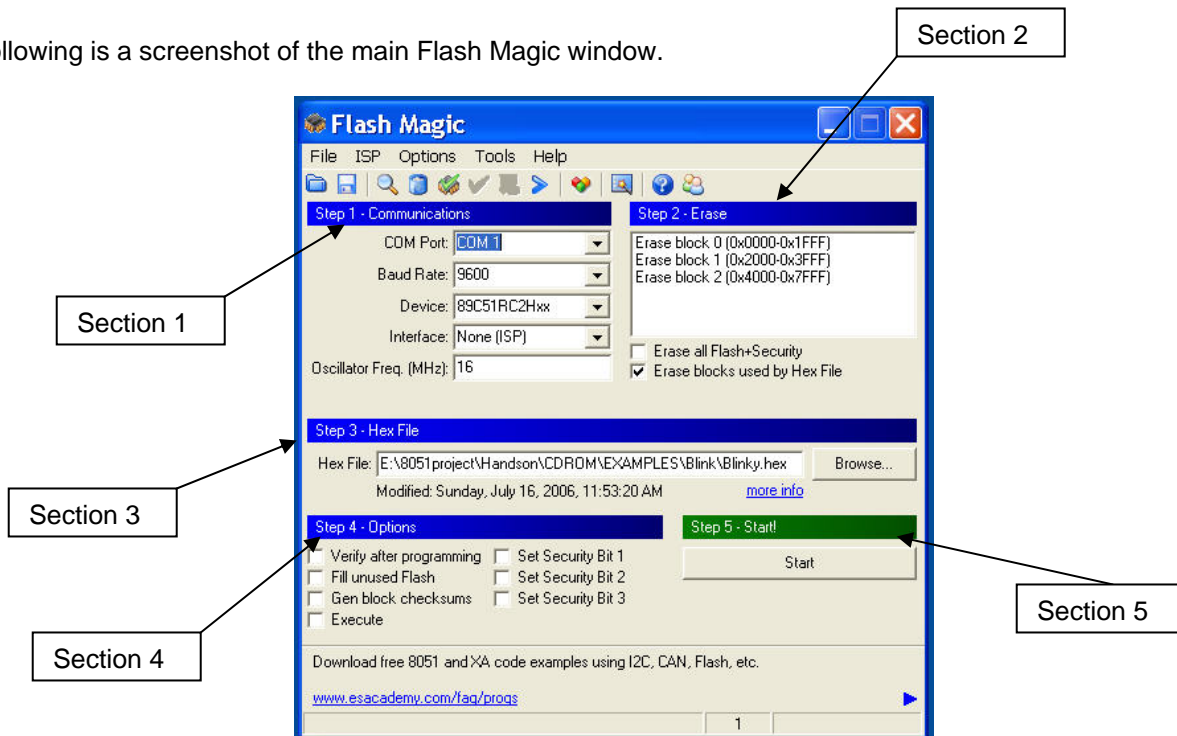
Flash Magic provides a clear and simple user interface to these features and more as described in the following sections.

Under Windows, only one application may have access the COM Port at any one time, preventing other applications from using the COM Port. Flash Magic only obtains access to the selected COM Port when ISP operations are being performed. This means that other applications that need to use the COM Port, such as debugging tools, may be used while Flash Magic is loaded.

## **7. Flash Magic User Interface Tour**

### **7.1 Main Window**

The following is a screenshot of the main Flash Magic window.



The window is divided up into five sections. Work your way from section 1 to section 5 to program a device using the most common functions. Each section is described in detail in the following sections.

At the very bottom left of the window is an area where progress messages will be displayed and at the very bottom right is where the progress bar is displayed.

Just above the progress information EmbeddedHints are displayed. These are rotating Internet links that you can click on to go to a web page using your default browser. If you wish to quickly flick through all the hints then you can click on the fast forward button:

## 7.2 Menus

There are four menus, File, ISP, Options and Help. The File menu provides access to loading and saving Hex Files, loading and saving settings files and exiting the application. The ISP menu provides access to the less commonly used ISP features. The Options menu allows access to the advanced options and includes an item to reset all options. The Help menu contains items that link directly to useful web pages and also open the Help About window showing the version number.

The Loading and Saving of Hex Files and the other ISP features are described in the following sections.

## 7.3 Tooltips

Throughout the Flash Magic user interface extensive use has been made of tool tips. These are small text boxes that appear when you place the pointer over something and keep it still for a second or two.

Note that tool tips do not appear for items that are disabled (grayed out).

## 7.4 Saving Options

The options in the main window and the Advanced Options window are automatically saved to the registry whenever Flash Magic is closed. This removes the need for an explicit save operation. When Flash Magic is restarted the main window and the Advanced Options window will appear as you left it, so you do not have to repeatedly make the same selections every time you start the application.

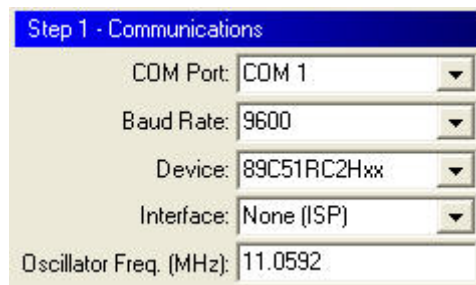
If you wish to reset the options to the original defaults then choose Reset from the Options menu.

## 7.5 Five Steps Programming

For each step there is a corresponding section in the main window as described in the User Interface Tour.

### 7.5.1 Step 1 – Connection Settings

Before the device can be used the settings required to make a connection must be specified.



Select the desired COM port from the drop down list or type the desired COM port directly into the box. If you enter the COM port yourself then you must enter it in one of the following formats:

- COM n
- n

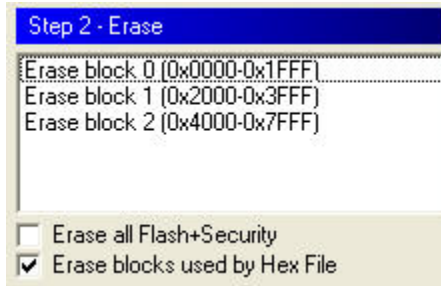
Any other format will generate an error. So if you want to use COM 5 (which is not present on the drop down list) you can directly type in either "COM 5" or "5". Select the baud rate to connect at. Try a low speed first. The maximum speed that can be used depends on the crystal frequency on your hardware. You can try connecting at higher and higher speeds until connections fail. Then you have found the highest baud rate to connect at.

Alternatively, some devices (Rx2 and 66x families) support high speed communications. Please refer to the High Speed Communications section for information. Enter the oscillator frequency used on the hardware. Do not round the frequency, instead enter it as precisely as possible. Select the device being used from the drop down list. Ensure you select the correct one as different devices have different feature sets and different methods of setting up the serial communications. Once the options are set ensure the device is running the on-chip BootROM. Note that the connection settings affect all ISP features provided by Flash Magic.

**Note: A good & reliable Baud Rate to start off is 7200 bits/s.**

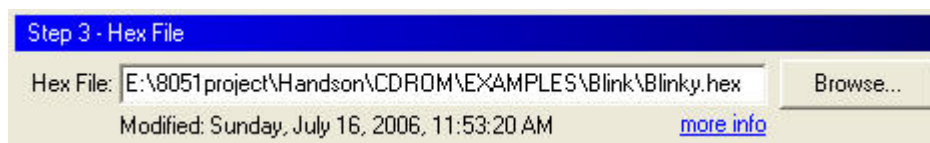
### 7.5.2 Step 2 – Erasing

This step is optional, however if you attempt to program the device without first erasing at least one Flash block, then Flash Magic will warn you and ask you if you are sure you want to program the device.



Select each Flash block that you wish to erase by clicking on its name. If you wish to erase all the Flash then check that option. If you check to erase a Flash block and all the Flash then the Flash block will not be individually erased. Erasing all the Flash also results in the Boot Vector and Status Byte being set to default values, which ensure that the BootROM will be executed on reset, regardless of the state of the PSEN pin. Only when programming a Hex File has been completed will the Status Byte be set to 00H to allow the code to execute. This is a safeguard against accidentally attempting to execute when the Flash is erased. On some devices (not the Rx+ family) erasing all the Flash will also erase the security bits. This will be indicated by the text next to the Erase all Flash option. On some devices erasing all the Flash will also erase the speed setting of the device (the number of clocks per cycle) setting it back to the default. This will be indicated by the text next to the Erase all Flash option.

### 7.5.3 Step 3 – Selecting the Hex File

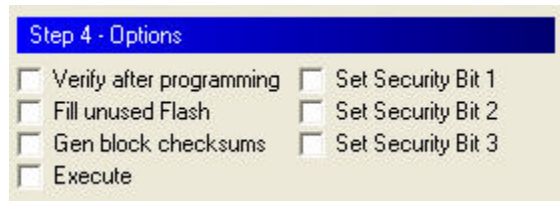


This step is optional. If you do not wish to program a Hex File then do not select one. You can either enter a path name in the text box or click on the Browse button to select a Hex File by browsing to it. Also you can choose Open... from the File menu.

Note that the Hex file is not loaded or cached in any way. This means that if the Hex File is modified, you do not have to reselect it in Flash Magic. Every time the Hex File is programmed it is first re-read from the location specified in the main window.

### 7.5.4 Step 4 – Options

Flash Magic provides various options that may be used after the Hex File has been programmed.



This section is optional, however Verify After Programming, Fill Unused Flash and Generate Checksums may only be used if a Hex File is selected (and therefore being programmed), as they all need to know either the Hex File contents or memory locations used by the Hex File. Also note that if one or more of the security bits are set on the device or the clocks bit (6 clks/cycle) is set on the device, then those set bits will be disabled in this section, indicating that they cannot be reprogrammed. If the device erases the security bits (and clocks bit) when a full Flash erase is performed then you can select the Erase all Flash option in section 2 and all the security bits (and clocks bit) will be enabled, indicating that you can select which ones you wish to program after the Hex File has been programmed. If the device does not erase the security bits then even if Erase all Flash is checked, the set security bits will remain disabled. On these devices only a Parallel Programmer can erase the security bits.

Checking the Verify After Programming option will result in the data contained in the Hex File being read back from Flash and compared with the Hex File after programming. This helps to ensure that the Hex File was correctly programmed.

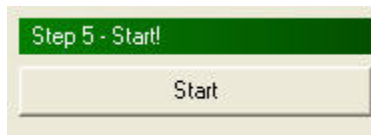
Checking the Fill Unused Flash option will result in every memory location not used by the Hex File being programmed with the value 00H. Once a location has been programmed with 00H it cannot be reprogrammed with any other value, preventing someone from programming the device with a small program to read out the contents of Flash or altering the application's operation.

Checking the Generate Checksums option will instruct Flash Magic to program the highest location in every Flash block used by the Hex File with a special "checksum adjuster value". This value ensures that when a checksum is calculated for the whole Flash Block it will equal 55H, providing the contents of the Flash block have not be altered or corrupted. Please refer to the Checksums section for more information.

Checking the Execute option will cause the downloaded firmware to be executed automatically after the programming is complete. Note that this will not work if using the Hardware Reset option or a device that does not support this feature. The 66x and old Rx2 families only support this feature from revision G onwards.

### 7.5.5 Step 5 – Performing the Operations

Step 5 contains a Start button.



Switch the S1 on HT-MC-02 Programmer board to "Prog" position and press SW1 Reset buttons once.

Clicking the Start button will result in all the selected operations in the main window taking place. They will be in order:

- Erasing Flash
- Programming the Hex File
- Verifying the Hex File
- Filling Unused Flash
- Generating Checksums
- Programming the clocks bit
- Programming the Security Bits
- Executing the firmware

Once started, progress information and a progress bar will be displayed at the bottom of the main window. In addition the Start button will change to a cancel button. Click on the cancel button to cancel the operation.

Note that if you cancel during erasing all the Flash, it may take a few seconds before the operation is cancelled.

Once the operations have finished the progress information will briefly show the message "Finished...". To execute the code download to the  $\mu$ Controller, switch the S1 to position Run and press SW1 Reset button once. You should observe the program performing the task or control you designed.

**Important: If after pressing the start button, the below message pop-up, please do the following:**



- I. Unplug the power supply and plug in again for the main power supply reset.
- II. Unplug any interface connected to port2 and port3, press Reset button and try again.

## 8. System Requirements

- PC with Pentium, Pentium-II or compatible processor,
- Windows 95, Windows-98, Windows NT 4.0, or higher,
- 16 MB RAM minimum,
- 20 MB free disk space.
- Mouse
- COM Port. See Note:1 for the cable type to be used.

## 9. Resources:

The following are development resources for the 8051 with samples code and other helpful utilities.

1. NXP Semiconductors (Formerly known as Philips Semiconductors): 8051 sample code, P89C51Rx2 datasheets & Application Note for 8051. <http://www.standardics.nxp.com/microcontrollers/>
2. Embedded System Academy: <http://www.esacademy.com/faq/progs> -- for **Flash Magic**.
3. Vault Information Services has one of the best online tutorials for the 8051 that we have seen to date. <http://www.8052.com>.
4. MCU Tools supplier, emulators, compilers & assembler: <http://www.mcu123.com>
5. Professional Industrial compiler & emulator . <http://www.keil.com> , evaluation available for free download.

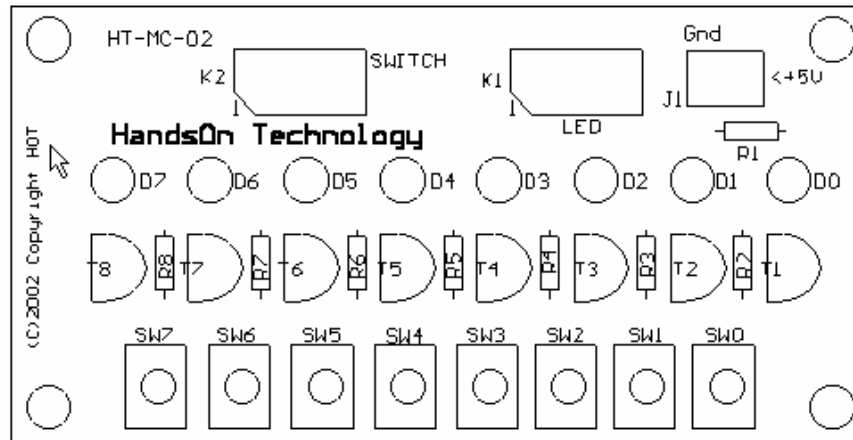
## 10. CDROM Contents:

Useful files in the accompanied CDROM:

1. **ek51v701** application file. This is the Keil C-Compiler and Assembler windows program. Locates this file in the CDROM and run this program to install the Keil Compiler to the PC.
2. **FlashMagic323** application file. This is Flash Memory  $\mu$ C programming software. Locates this file in the CDROM and install it into the hard disc.
3. **AcrobatReader705** application file. Acrobat Reader program to read and print all the documentations contain in this CDROM. Run this application program to install the Acrobat Reader 7 into the hard disc.
4. **8051 $\mu$ C** tutorial, application note and IC data sheets related to this Starter kit.
5. **Examples Files**. A hands-on examples code for you to get familiar with the compiler, assembler and programming the **HT-MC-02 Starter Kit**.
6. **Project Folder**: Collection of constructional projects based on 8051 uC.
7. Schematic diagram and technical detail for **HT-MC-02 Starter Kit**.

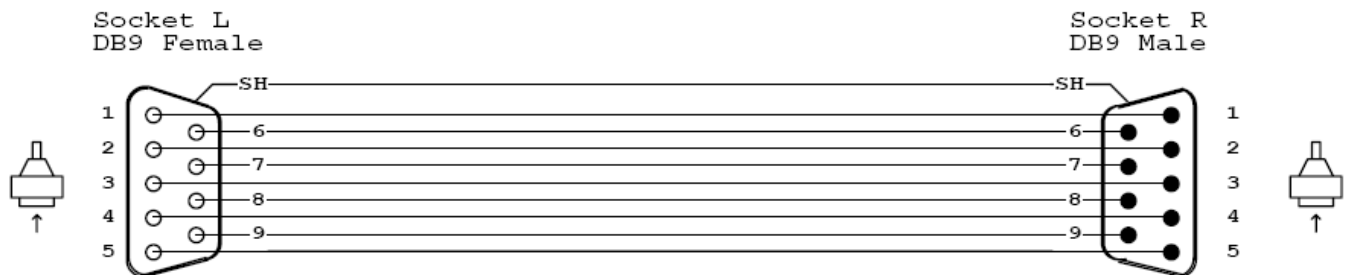
## 11. LEDs & SWITCH BOARD

8 switches + 8 LEDs test and experiment board free with HT-MC-02 Starter Kits.



***This experimental board is free with the HT-MC-02 Starter Kit. (While stock last)***

**Note1: RS232 Direct 1-to-1 Cable Connection to be used for this board.**



**Email address:**

For Sales : [sales@handsontec.com](mailto:sales@handsontec.com)  
For technical support: [techsupport@handsontec.com](mailto:techsupport@handsontec.com)  
For general inquiry: [inquiry@handsontec.com](mailto:inquiry@handsontec.com)