# DYNA-51EB

## User's Manual

**February 2011 Rev 1.0**

**Kailash-Vaibhav,**
**G-wing, 3rd Floor, Parksite,**
**Vikhroli (W), Mumbai – 400 079**
**INDIA**

**Tel: +91 22 4233 0000; Ext.: 109**
**Fax: +91 22 4233 0010 / 20**
**Visit:: www.dynalogindia.com**

# 8051 BASED MICROCONTROLLER EVALUATION BOARD MANUAL

## Overview

The 8051 based DYNA-51EB board is a useful tool for embedded control projects for both students and hobbyists.

Its versatile design and programmable microcontroller lets you access numerous peripheral devices and program the board for multiple uses. The board has many I/O connectors and supports a number of programming options including 8051 assembly and C

The 8051 trainer board is based on NXP P89V51RD2 Micro controller. The port lines of P89V51RD2 are terminated on 10-Pin FRC connectors. Each port has separate 10-pin connector. It also has RS232 interface with DB9 male connector based on MAX232. The RS232 level converted RX and TX signals are terminated on this DB9 connector. The microcontroller can be programmed through this connector. To interface Dynalog make PIO cards, one 26-pin FRC connector is provided. The pin configuration of this connector is compatible to PIO cards. The PIO cards include stepper motor, analog-to-digital and digital- to-analog converters, LEDs, switches, relays, buzzers and more.

It can be programmed using Flash Magic programming utility of NXP's own programming utility via serial port.

### Features:

- Microcontroller: NXP P89V51RD2 with 11.0592MHz crystal.
- The Pull-up resistors for the Port P0 is provided on a 9-Pin socket
- Double sided high quality PTH PCB board to provide extra strength to the connector joints for increased reliability.
- Power: +5VDC @100mA without any PIO board attached.
- Four 10-pin FRC connectors are provided to access all P89V51RD2 ports of P0, P1, P2, and P3 for connection to external devices such as Dynalog PIO boards.
- One 26-pin FRC connector is provided to access the ports P0, P1 and P2. This connector is compatible with Dynalog make PIO cards.
- ALE, PSEN and Reset signals are terminated on a separate 3 Pin Relimate connector.

- Support for the On-chip serial programmer through DB9 connector.
- Interrupt signals are terminated on 4-in Relimate connector.
- Two Push buttons are provided to simulate high to low interrupt signals on INTO & INT1 inputs.
- Compatible with NXP's Flash Magic programming utility

## Functional Description

The 8051 is designed for embedded control and robotic applications as well as microprocessor experimentation.

The 8051 has an on-chip loader/programmer: The loader / programmer is accessed via Serial COM Port DB9 Connector.

The 8051 Trainer features a flexible power supply routing system with VCC and GND pin available on DC jack connector for powering the ICs on the board as well as Dynalog PIO cards connected to the Trainer board.

Dynalog PIO cards can be connected to the 8051 Trainer board via 10pin FRC connector or 26Pin FRC connector depending upon the PIO card selected. Dynalog has Varity of PIO cards. Contact your nearest Dynalog office for more details.

The signals ALE, PSEN and RESET are terminated on a separate 3 pin Relimate. By using these signals along with Ports 0 , 1 & 2 user can design memory mapped I/O Hardware. Contact nearest Dynalog office for more details.

The signal pins INT0, INT1, RX, and TX which multifunction pins of port 3 are also terminated on

10-pin FRC connector assigned for Port 3

## Power supply

The 8051 Trainer board may be powered via the dedicated power supply connector. The Trainer board is designed for operation at 5V. Using a voltage other than 5V can damage the 8051 trainer or the connected devices.

## Crystal Oscillator

The 8051 Trainer has an 11.0592 MHz oscillator crystal. 11.0592 MHz oscillator crystal makes you enable to connect the 8051 trainer board to PC by COM port and the serial transfer error will be as low as zero.

## User I/O Pins

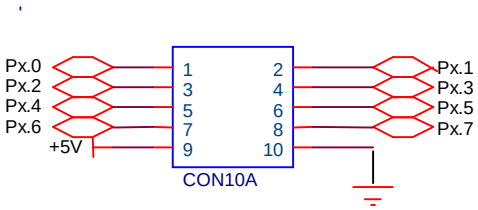
The 8051 Trainer board has 4 10-pin FRC connectors, one 26-pin FRC connector and two Relimate connectors for user to access all the ports of the 8051 microcontroller

## 8051 Trainer Headers Connection

Note: All 8051 ports can be used as general purpose I/Os or as address, data and control signals for designing external memory mapped I/O. Refer P89V51RD2 datasheets for more details

### 10-Pin FRC configuration

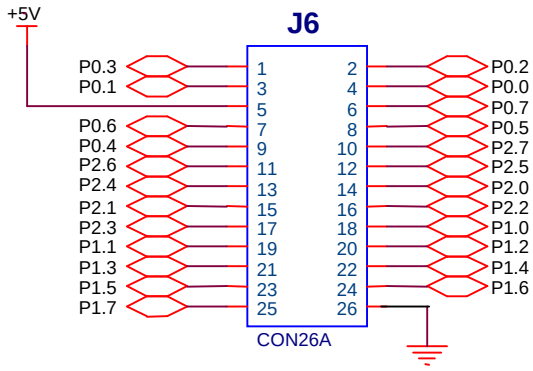
10-pin FRC port pin configuration. Where Px denotes the Port number




The table below shows the 10-pin RC connectors assigned for different ports

| Connector Name | Port Number |
|---|---|
| J2 | Port 0 |
| J3 | Port 1 |
| J4 | Port 3 |
| J5 | Port 2 |

### 26-Pin FRC configuration




**J7- DB9 Connector(serial Port)**

| Pin No. | Signal |
|---|---|
| 2 | Trainer board Tx data |
| 3 | Trainer board Rx data |
| 5 | Ground |

**J1- Relimate Connector (External Interrupt Input)**

| Pin No. | Signal |
|---|---|
| 1 | 5V |
| 2 | INT0 |
| 3 | INT1 |
| 4 | Ground |

**J9-Relimate Connector (control signals)**

| Pin No. | Signal |
|---|---|
| 1 | RESET |
| 2 | PSEN |
| 3 | ALE |

**JP1 – JUMPER SETTING ( EA SIGNAL)**

JP1 : 1-2 – ON : Select External Program Memory

JP2 : 2-3 – ON : Select Internal Program Memory of P89V51RD2 (Default Factory setting)

# Loading Firmware on Development Board

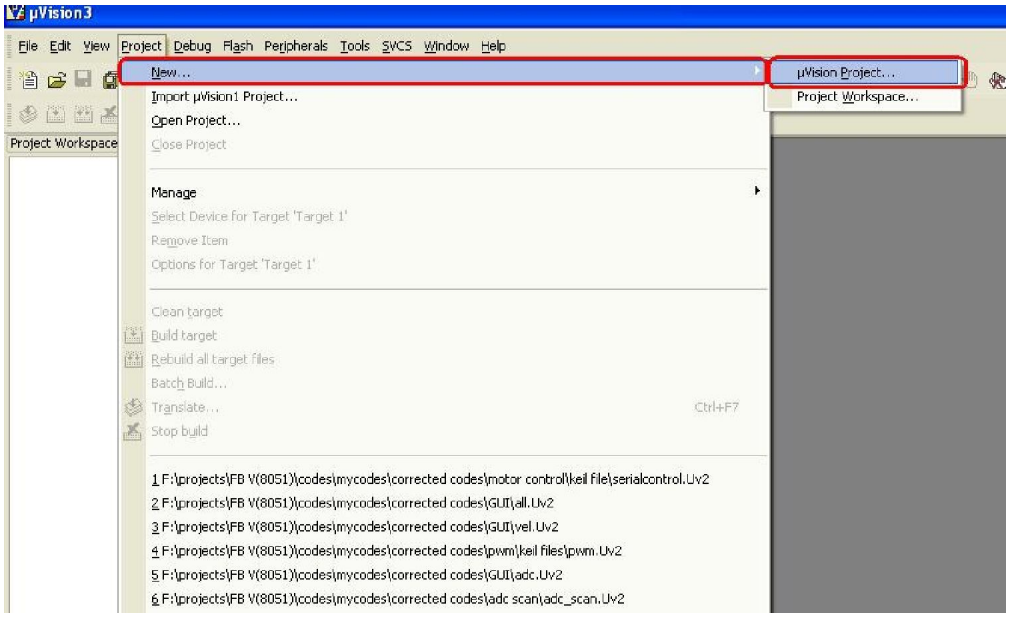
P89V51RD2 Development Board programming involves two steps. First step is to write and compile the code and generate the "*.hex" file. Second step is to load this "*.hex" file on the microcontroller using Flash Magic software provided by NXP (formerly Phillips)

We are going to use Keil-U-Vision (Version 4) software for writing the code for the microcontroller. We can also use any other open source of proprietary software supporting P89V51RD2 microcontroller. P89V51RD2 Development Board CD contains free version of the uVision 4 software. You can also download latest version from http://www.keil.com/dd/chip/3711.htm and click C51 Evaluation Software

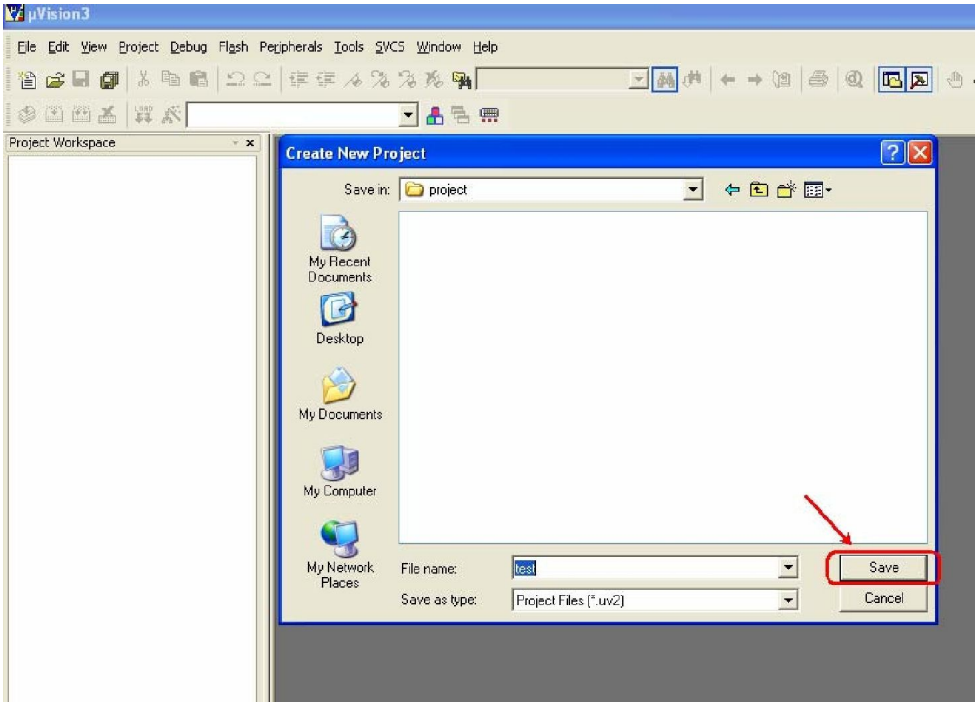
**Writing program in KEIL uVision3**

Start with Keil-U-Vision

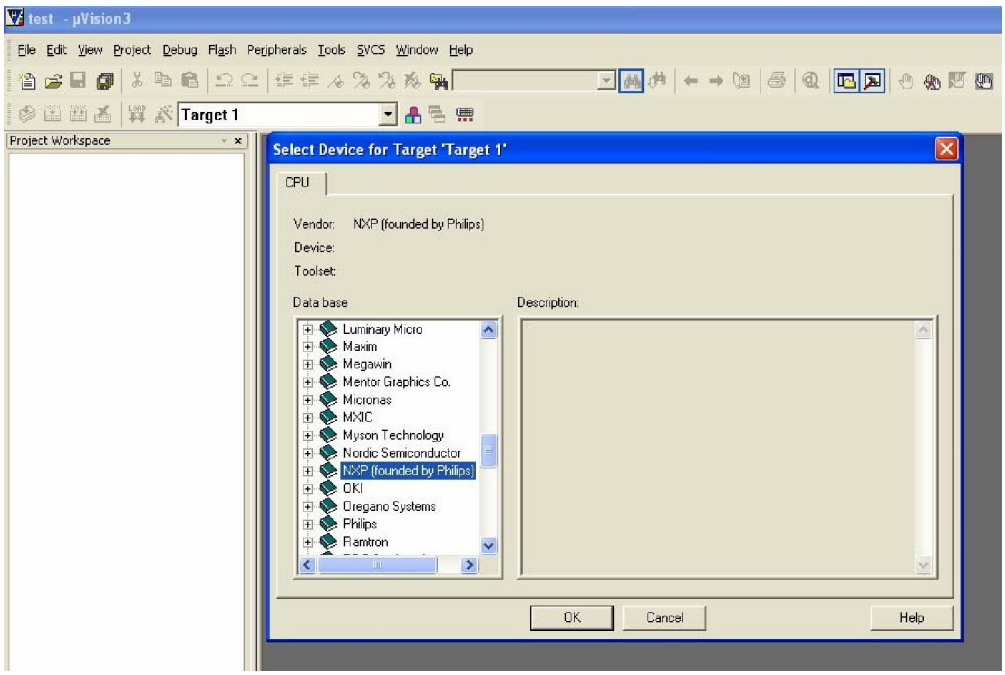

Go to project and start new project.

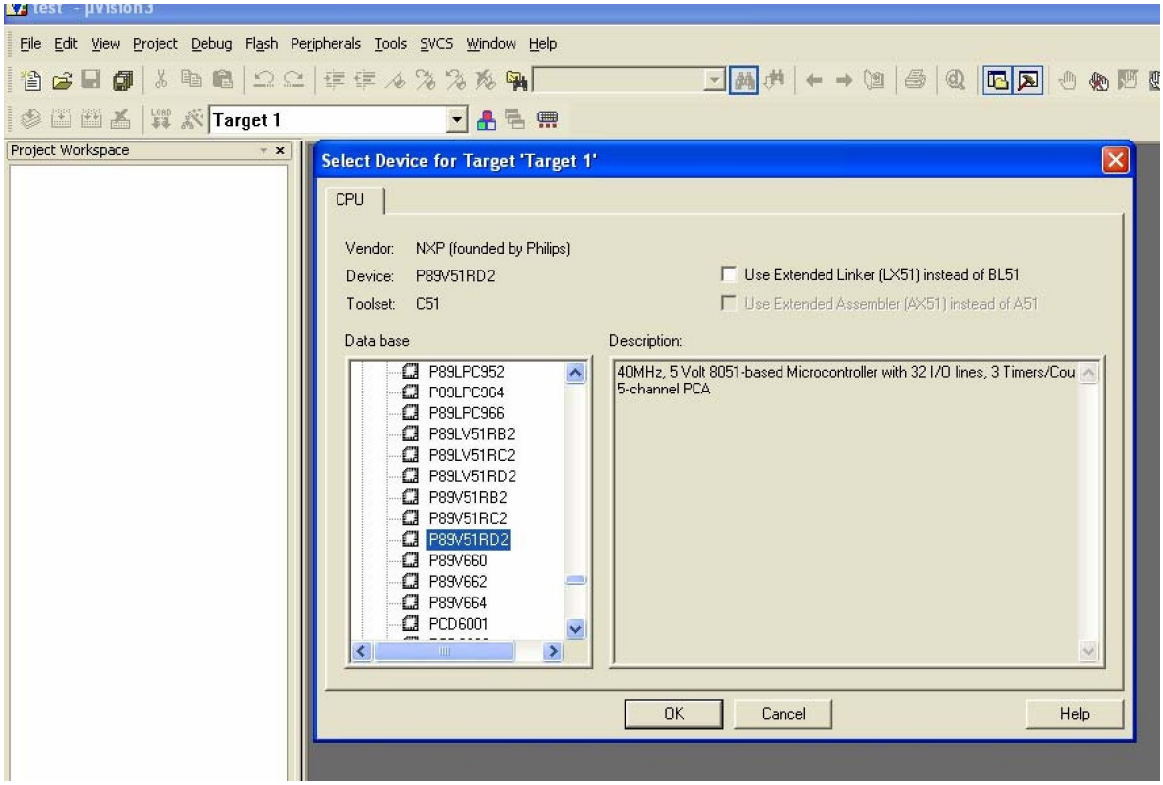


Name your project and save it in your project folder. (Always create new folder for new Project)

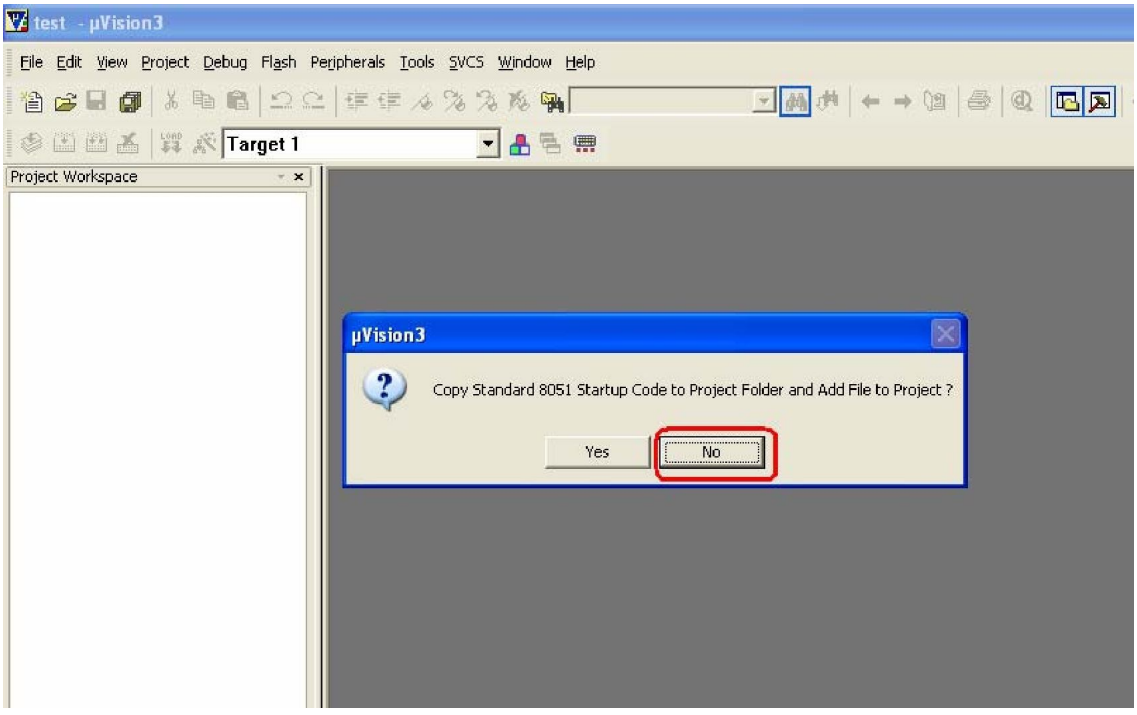



A dialogue box will open asking you to select your device used. Select the appropriate device for e.g. P89V51RD2.Then click OK. P89V51RD2 can be found in the NXP (founded by Philips) directory.

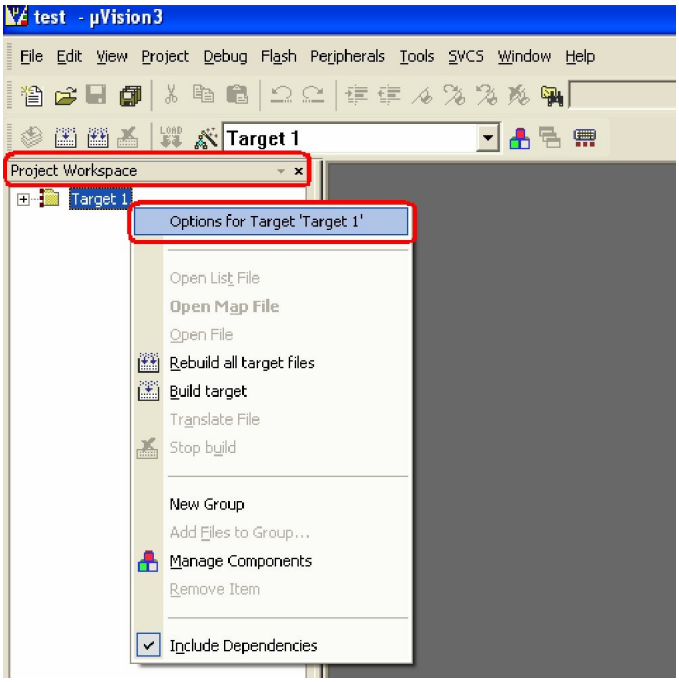




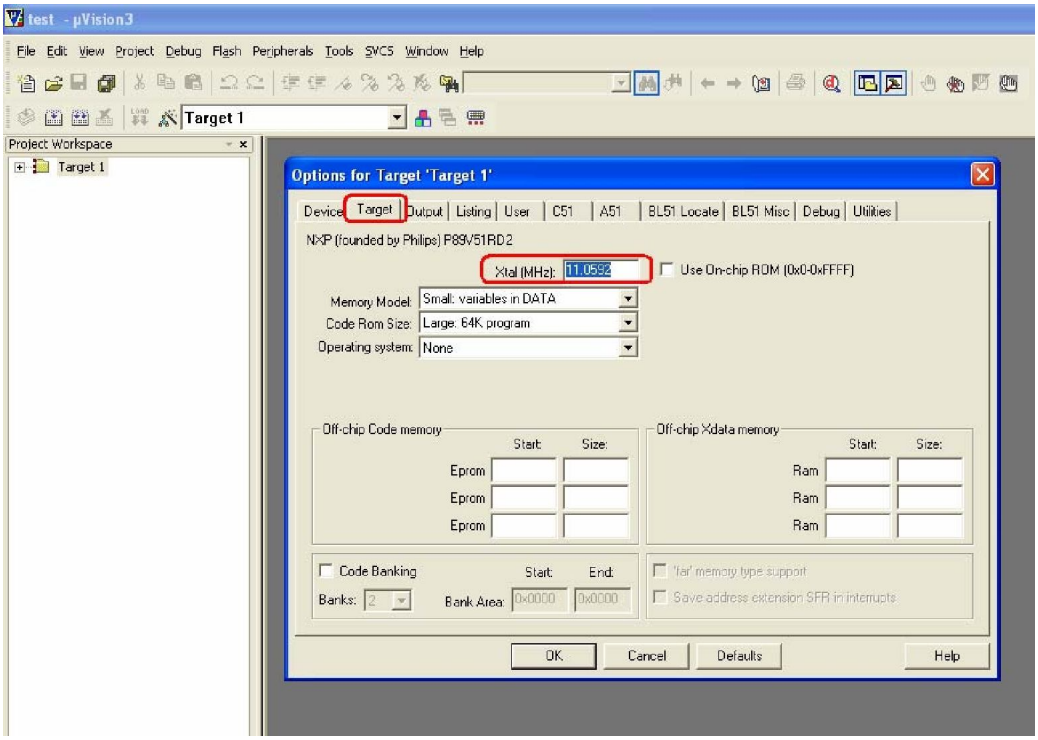

Next it will ask you if you want to add the A51STARTUP code.
Say “NO”



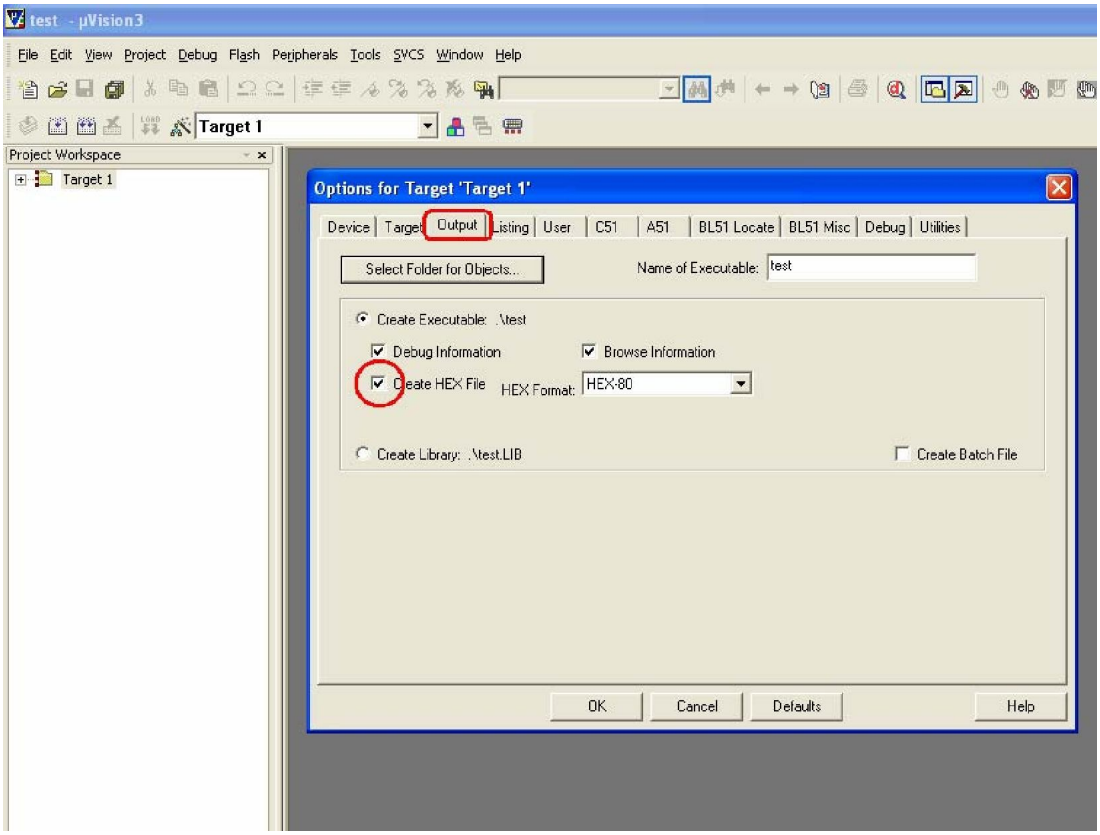
In the Project Workspace right click on ‘Target 1’ and select options for target as shown. A dialogue box to choose different options will open.



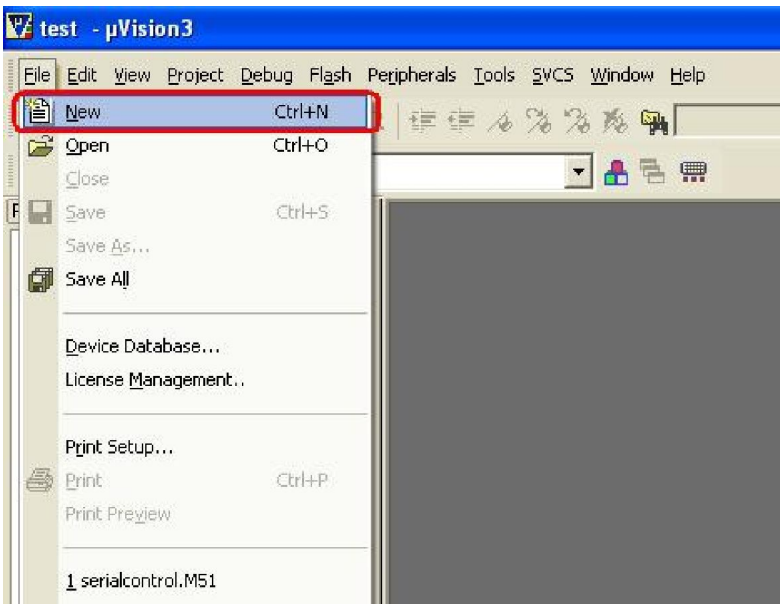

Click on the ‘Target’ tab. Enter the frequency of the crystal. For Development Board its 11.0592 MHz.



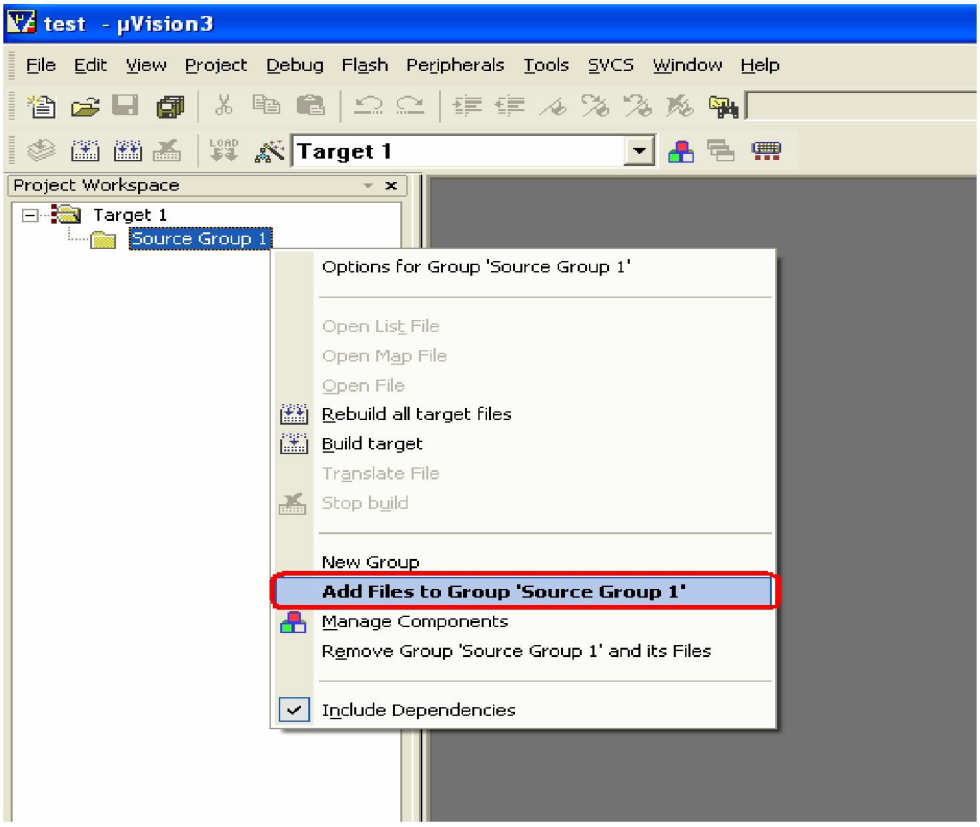
Go to the output tab and tick on the file to create HEX-file. Then click OK to save your options. Other tabs can be left with the default options.

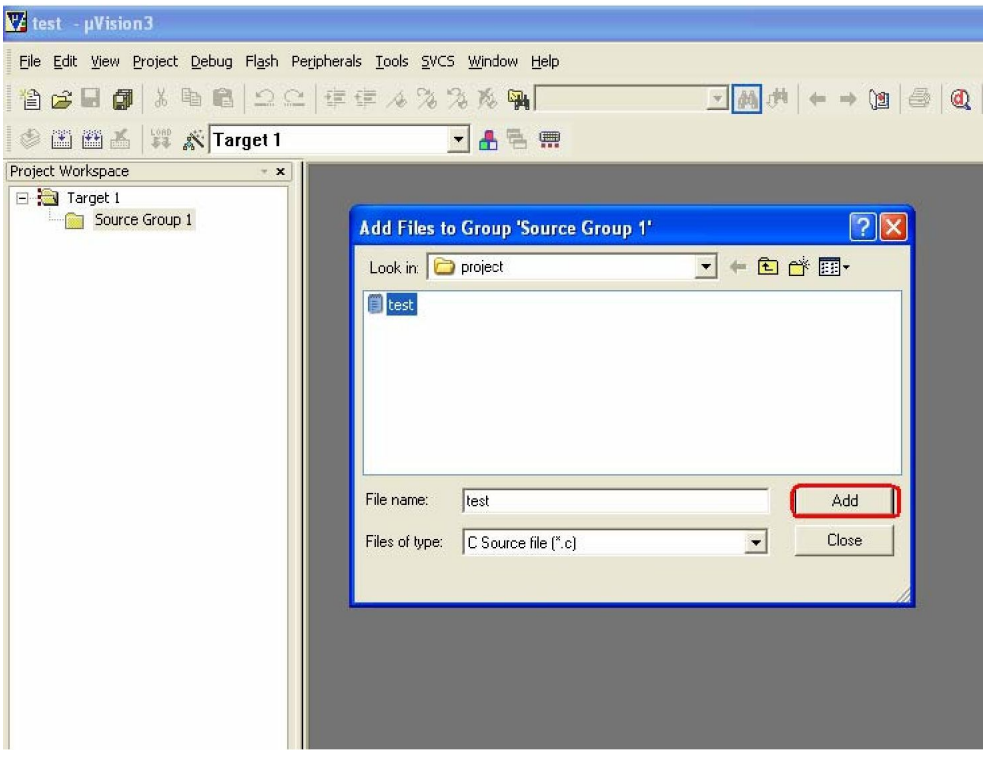



After this is done, open a new file and save it with the project file as a ASM file i.e. with the extension ".asm".

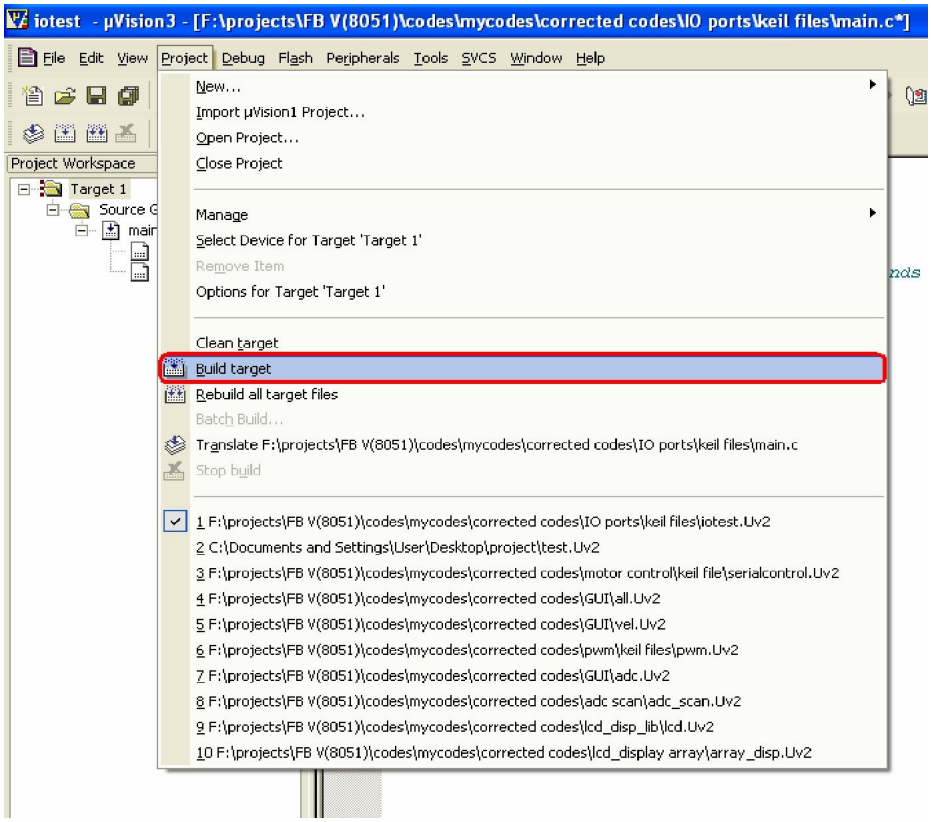


Add this file to the project by Right-clicking on “Source Group” and choosing to add files to group. Select the appropriate ‘.ASM’ file to be added.



Now click on the “Project” tab and choose to “Build Target” as shown.




Check for any errors in the ‘Output Window’.
If there are no errors then ‘Hex’ file will be created and stored in the Project folder. You can then download this file onto your microcontroller using the In System Programming (ISP) software i.e. using flash tools like Flash Magic or FLIP or parallel programming as supported by your microcontroller.

**Note:** Use ‘rebuild all target files’ option if you are including multiple header file

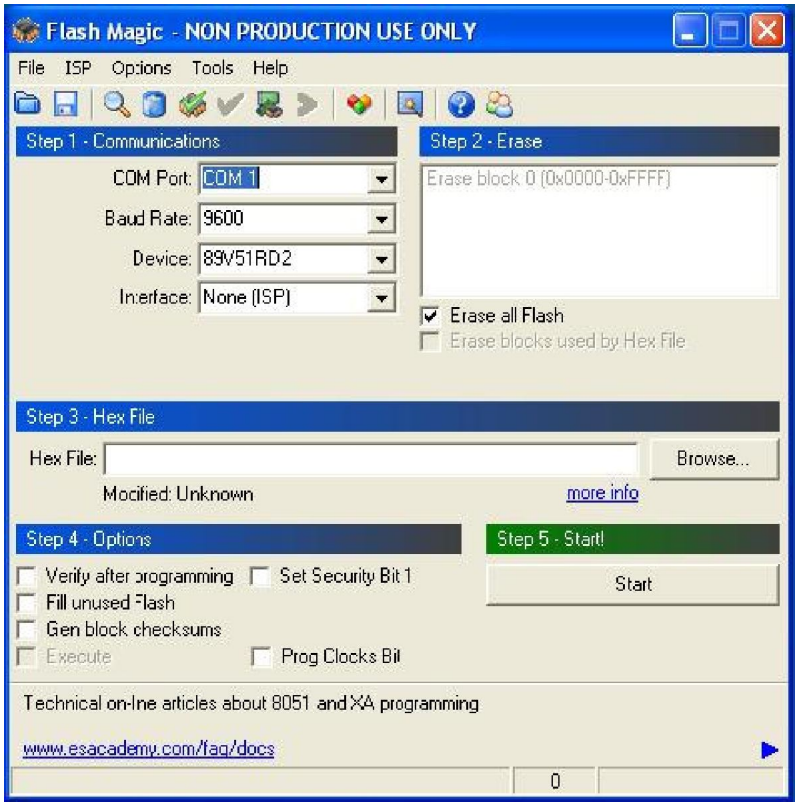
Loading the generated Hex file on the microcontroller using Serial Port

Flash Magic is Windows software which allows easy access to all the In System Programming (ISP) features provided by the devices.

These features include:
• Erasing the Flash memory (individual blocks or the whole device)
• Programming the Flash memory
• Reading 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 Boot loader mode

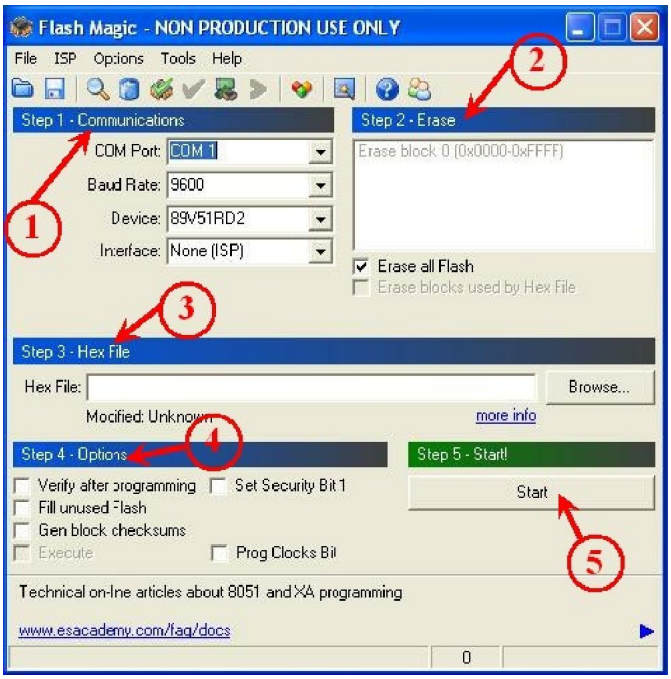
Flash Magic provides a clear and simple user interface to these features

Go to '*Flash Magic*' Icon, it will open the main window as show below.

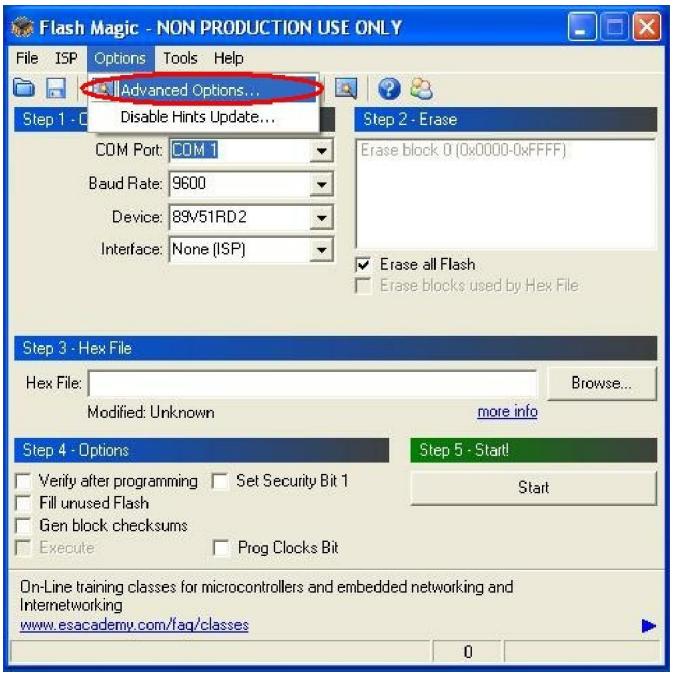


In main window you can see five types of sections.

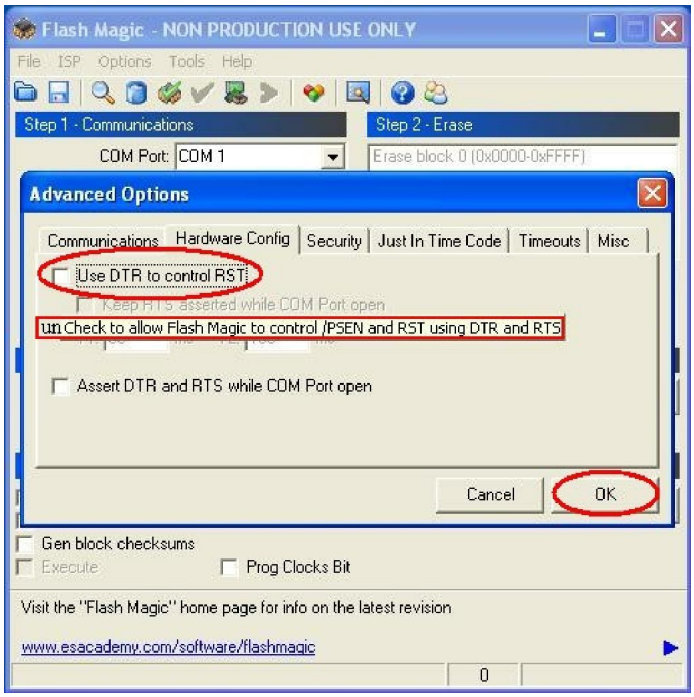
1. Communications
2. Erase
3. Hex File
4. Options
5. Start



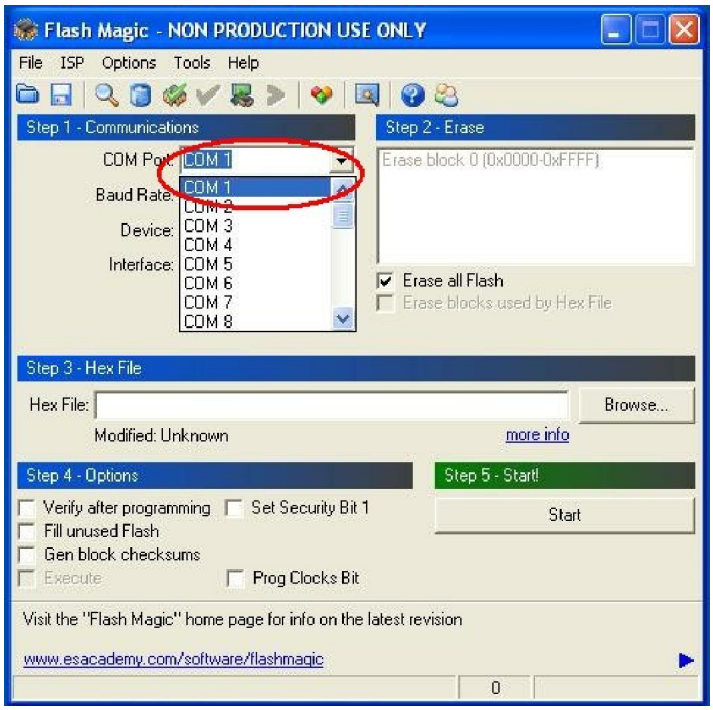

Go to "*Option*" in the toolbar, and select first menu '*Advance options.*'



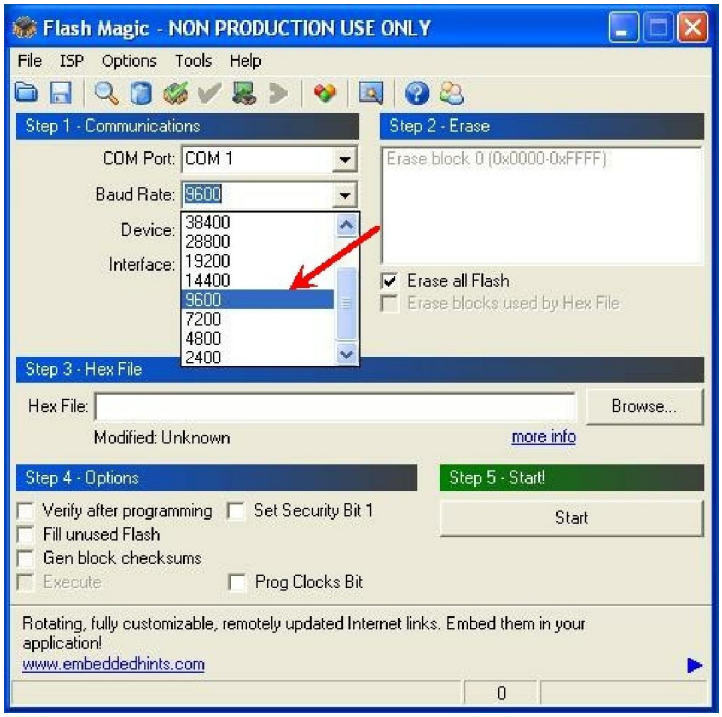
It will show you extracted window with many options.



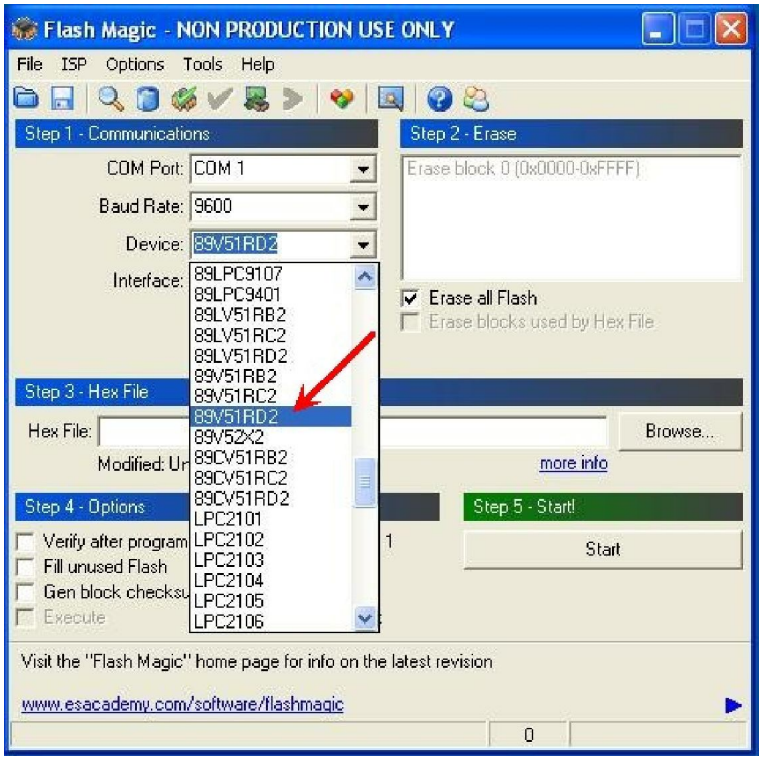

Uncheck the box which is highlighted to don't allow Flash Magic to control PSEN & RST using DTR & RTS. Now go for Communication selection, select '*COM 1'* from '*COM PORT*' Option if you are using serial port. If you are using USB to serial converter from NEX Robotics then find out COM port number with the help of documentation provided with the USB to serial converter.



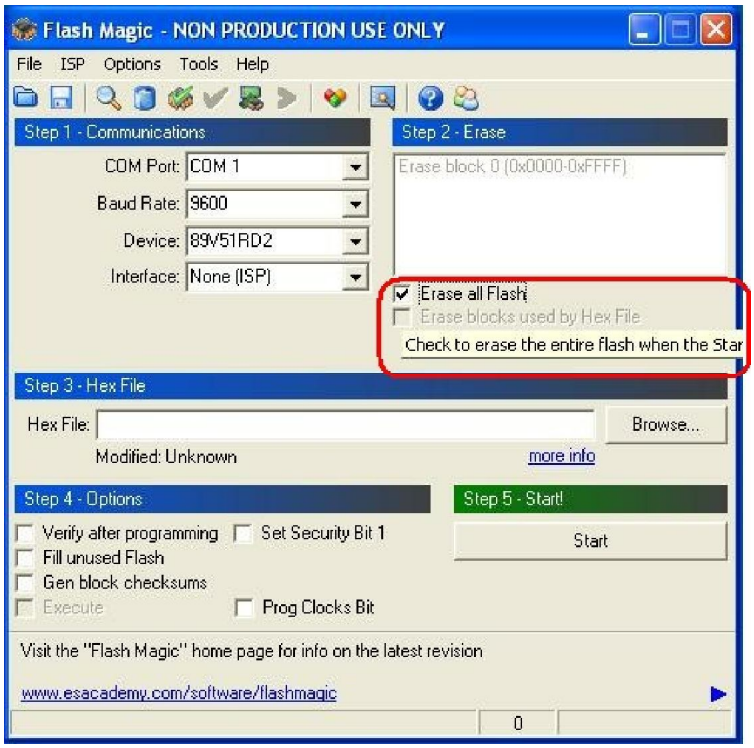
Select '*Baud Rate*' 9600 from Baud Rate option.




Select proper Device '*89V51RD2*' from Device option.

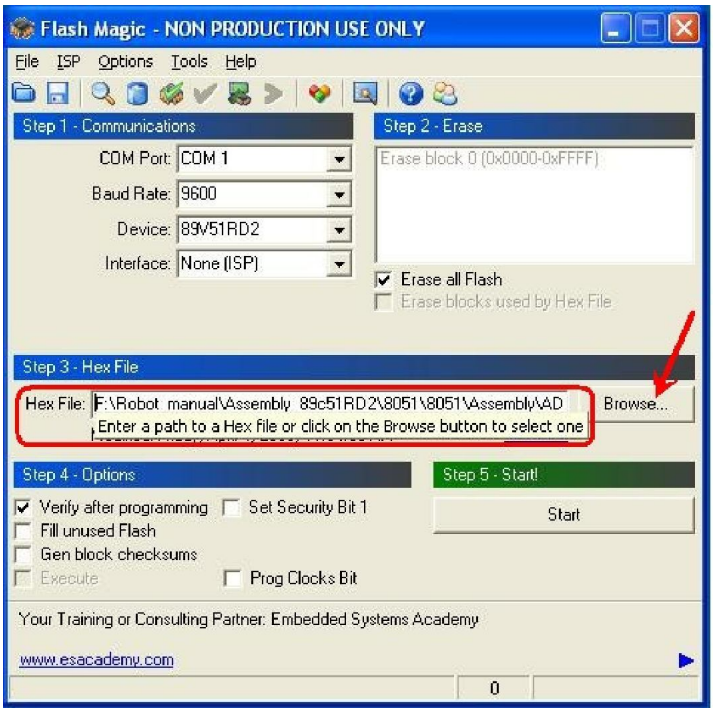


Go to '*Erase*' section for erasing the all '*Flash*' or Blocks used by '*HEX File*'

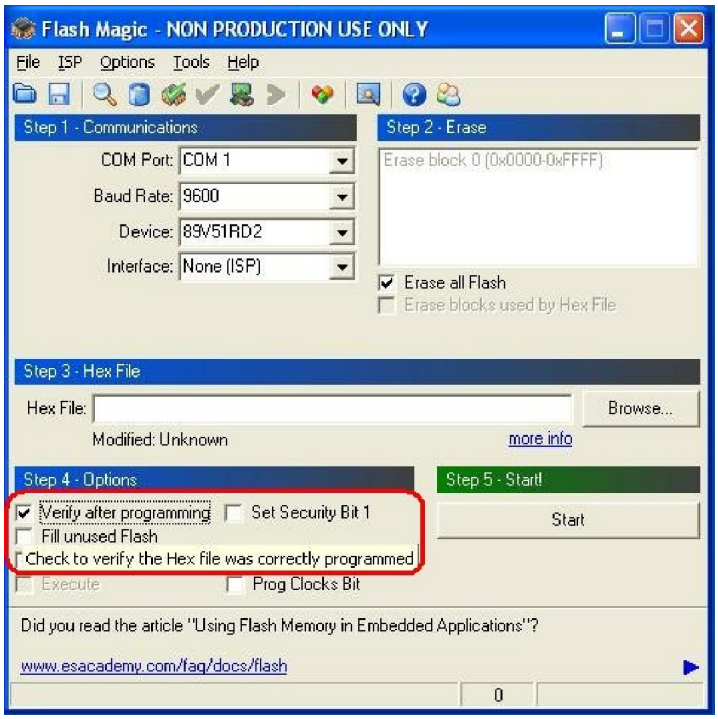



Check on the check box to Erase all Flash.

Go to '*HEX File*' section click '*Browse*' to select proper Hex file.



Go to '*Option*' section to select *'Verify after programming'* option. This will verify hex file after loading.




After doing all the required settings, connect serial cable between Development Board and PC. Turn on the Board and click on the *'Start'* , it will ask to *'RESET TO DEVICE IN TO ISP MODE'*, now press RESET Switch on the Development Board, Flash magic will load hex file on the robot and verify it for correctness. If you are using USB to serial converter from then find out COM port number with the help of documentation provided with the USB to serial converter.

# Sample Programs for 8051

**Program 1:**

```
*********************************************************************************
: Objective        : Arithmetic programs with AT89S8253
: Program          : Write a program to ADD, SUBSTRACT, DIVIDE, MULTIPLY
                         two numbers. (Debug using step mode).
: Compiler         : KEIL uvision3
*********************************************************************************

Org     0000h
        mov a,#10h
        mov b,#5h
        add a,b
        mov r1,a
        clr a
        mov b,#00h

        mov a,#20h
        mov b,#10h
        subb a,b
        mov r2,a
        clr a
        mov b,#00h

        mov a,#2h
        mov b,#2h
        mul ab
        mov r3,a
        clr a
        mov b, #00h

        mov a,#21h
        mov b,#2h
        div ab
        mov r4,a
        mov r5,b

end
```

**Program 2:**

```
*********************************************************************************
: Objective       : Logical Operation programs with AT89S8253
: Program         : Write a program to AND, OR, EXOR. (Debug using step mode).
: Compiler        : KEIL uvision3
*********************************************************************************

Org     0000h
        mov a,#1h
        anl a,#1h
        mov r1,a
        clr a
        mov a,#0h
        orl a,#1h
        mov r2,a
        clr a
        mov a,#1h
        xrl a,#1h
        mov r3,a
        clr a
 end
```

**Program 3:**

```
*********************************************************************************
: Objective             : Generate square wave with AT89S8253
: Program               : Write a program to generate square wave on port P1.0
: Compiler              : KEIL uvision3
: Development Board     : DYNA51-EB
*********************************************************************************

Org     0000h
        up:setb p1.0
        lcall delay
        clr p1.0
        lcall delay
        sjmp up

delay : mov r4,#0ffh
upx :   mov r5,#0ffh
up1 :   djnz r5, up1
        djnz r4,upx
        ret
end
```

**Program 4:**

```
*********************************************************************************
: Objective              : Interface LED with AT89S8253
: Program                : Write a program for rolling display of LED
: Compiler               : KEIL uvision3
: Development Board      : DYNA51-EB
*********************************************************************************


Org     0000h
start:
        mov r7,#08h             // Initilizing the counter   //
        mov a,#7fh              // data 0111 1111 port 0.0 to port 0.7 for rotating the led
                                // one on others off //
Rotate : mov p1,a
        mov p0,a
        mov p2,a
        mov p3,a
        call delay
        call delay
        rl a                    //for rotating the data in accumulator use this
                                // instruction  rl a or rr a //
        djnz r7,rotate          // decrement the contents of r7 & jump if not
                                // zero to label rotate //
        sjmp start
 delay:
        mov r2,#0ffh
        load:
        mov r1,#0ffh
        here:  djnz r1,here
        djnz r2,load
        ret
  end
```

**Program 5:**

```
*********************************************************************************
: Objective                : Serial Communication using RS232 with AT89S8253
: Program                  : Transmit a character "YES" continuously at 9600 baud
: Crystal frequency        : 11.0592 MHz
: Compiler                 : KEIL uvision3
: Development Board        :  DYNA51-EB
*********************************************************************************
org       0000h
          mov tmod, #20h
          mov th1, #-3
          mov scon, #50h
          setb tr1
again :   mov a, #"Y"
          acall trans
          mov a, #"E"
          acall trans
          mov a, #"S"
          acall trans
          mov a, #" "
          acall trans
          sjmp again
trans :   mov sbuf, a
here  :   jnb ti, here
          clr ti
          ret
 end
*********************************************************************************
: Objective                : Serial Communication using RS232 with AT89S8253
: Program                  : Transmit a character "A" continuously at 4800 baud
: Crystal frequency        : 11.0592 MHz
: Compiler                 : KEIL uvision3
: Development Board        :  DYNA51-EB
*********************************************************************************
org       0000h
          mov tmod, #20h
          mov th1, #-6
          mov scon, #50h
          setb tr1
          again: mov sbuf, #"A"
          here: jnb ti, here
          clr ti
          sjmp again
  end
```

**Program 6:**

```
*********************************************************************************
: Objective              : Serial Communication using RS232
: Program                : Receive a character on serial port modify it & then
                           Retransmit it at 9600 bits / sec using Interrupt technique.
: Crystal Frequency      : Crystal Frequency IS 11.0592 MHz
: Compiler               : KEIL uvision3
: Development Board      : DYNA51-EB
*********************************************************************************


Org             0000h
                lcall start

Org             0023h
                lcall serialISR
Start:          mov IE, #90h            // enable serial Interrupt
                acall serialInit
gut:            sjmp gut
serialISR:      jb RI,rec
                clr TI
                reti
rec:            clr RI
                mov A,SBUF
                inc A
                mov SBUF,A
                reti
serialInit:     mov SCON,#50h
                mov TH1,#0fdh
                mov TMOD,#20h
                setb TR1
                ret
end
```

**Program 7:**

```
*********************************************************************************
: Objective              : Interfacing of Seven Segment Display with AT89S8253
: Program                : Write a program for rolling the display 1234 on the Seven
                           Segment Display through right entry mode.
: Compiler               : KEIL uvision3
: Development Board      : DYNA51-EB, Seven segment display
*********************************************************************************
Org     0000h
start:  setb p1.0
        clr p1.1
        clr p1.2
        clr p1.3
        mov p0,#0f9h
        call delay
        setb p1.1
        clr p1.0
        clr p1.2
        clr p1.3
        mov p0,#0a4h
        call delay
        setb p1.2
        clr p1.0
        clr p1.1
        clr p1.3
        mov p0,#0b0h
        call delay
        setb p1.3
        clr p1.0
        clr p1.1
        clr p1.2
        mov p0,#99h
        call delay
        sjmp start

delay:  mov r2,#0ffh            // Random Delay
load:   mov r1,#0ffh
here:   djnz r1,here
        djnz r2,load
        ret
end
```

**Program 8:**

```
********************************************************************************
: Objective              : Interfacing of Seven Segment Display with AT89S8253
: Program                : Write a program for rolling the display 1234 on the Seven
                           Segment Display through right entry mode using push button
                           P1.0
: Compiler               : KEIL uvision3
: Development Board      : DYNA51-EB, Seven segment board, 4x4 matrix keypad.
********************************************************************************
org     0000h
here1:  jb p1.0,here1
start:
        setb p1.0
        clr p1.1
        clr p1.2
        clr p1.3
        mov p0,#0f9h
        call delay
        setb p1.1
        clr p1.0
        clr p1.2
        clr p1.3
        mov p0,#0a4h
        call delay
        setb p1.2
        clr p1.0
        clr p1.1
        clr p1.3
        mov p0,#0b0h
        call delay
        setb p1.3
        clr p1.0
        clr p1.1
        clr p1.2
        mov p0,#99h
        call delay
        sjmp start
        delay:
        mov r2,#0ffh
        load:
        mov r1,#0ffh
        here:
        djnz r1,here
        djnz r2,load
        ret     end
```

**Program 9:**

```
*********************************************************************************
: Objective              : Interfacing Stepper Motor with AT89S8253
: Program                : Run the stepper motor in continuous mode.
: Compiler               : KEIL uvision3
: Development Board      : DYNA51-EB, DYNA STP-Pio card, Stepper motor.
*********************************************************************************
```

```
 Org     0000h
Main :  mov a, #05h
        mov p0,a
        call delay
        mov a,#09h
        mov p0,a
        call delay
        mov a,#0ah
        mov p0,a
        call delay
        mov a,#06h
        mov p0,a
        call delay
        sjmp main
delay :  mov r0,#10h
 l2 :    mov r1,#0ffh
 l1 :    djnz r1,l1
        djnz r0,l2
        ret
end
```

CONNECTION DETAIL:

Connect pin of

1) Port 1.1 pin to the pin no1 of 26 pin FRC Cable Connected to Stepper Motor Card.
2) Port 1.2 pin to the pin no2.
3) Port 1.3 pin to the pin no3.
4) Port 1.4 pin to the pin no4.
5) Ground should be common

```
*******************************************************************************

: Program              : Run the stepper motor in step mode.
: Compiler             : KEIL uvision3
*******************************************************************************


org     0000h
main:   mov a,#05h
        mov p0,
        call delay
        mov a,#09h
        mov p0,a
        call delay
        mov a,#0ah
        mov p0,a
        call delay
        mov a,#06h
        mov p0,a
        call delay
        sjmp main
        delay: mov r0,#10h
 l2 :   mov r1,#0ffh
 l1 :   mov r3,#0ffh
 l3 :   djnz r3,l3
        djnz r1,l1
        djnz r0,l2
         ret
 end
```

CONNECTION DETAIL:

Connect pin of

6) Port 1.1 pin to the pin no1 of 26 pin FRC Cable Connected to Stepper Motor Card.
7) Port 1.2 pin to the pin no2.
8) Port 1.3 pin to the pin no3.
9) Port 1.4 pin to the pin no4.
10) Ground should be common.

**Program 10:**

```
*********************************************************************************
: Objective              : Interfacing 16X2LCD with AT89S8253 in 4-bit mode.
: Program                : To display text on the LCD screen.
: Compiler               : KEIL uvision3
: Development Board      : DYNA51-EB, LCD Module.
*********************************************************************************
ORG 0000H
                LJMP MAIN
                ORG 0030H
MAIN:           NOP
                EN    EQU   P1.0
                RS    EQU   P1.2
//              RW    EQU   P1.1                    .1
                DAT     EQU   P1
                LCALL LCD_INT
                LCALL CLEAR
                LCALL LINE1
                MOV DPTR,#MYDATA
                LCALL LOOP
                LCALL LINE2
                MOV DPTR,#MYDAT2
                LCALL LOOP
                LCALL LINE3
                MOV DPTR,#MYDAT3
                LCALL LOOP

again:          SJMP again

;=====================================================================
W_NIB:          PUSH  ACC               ;Save A for low nibble
                ORL   DAT,#0F0h         ;Bits 4..7 <- 1
                ORL   A,#0Fh            ;Don't affect bits 0-3
                ANL   DAT,A             ;High nibble to display
                SETB  EN
                CLR   EN
                POP   ACC               ;Prepare to send
                SWAP  A                 ;...second nibble
                ORL   DAT,#0F0h         ; Bits 4...7 <- 1
                ORL   A,#0Fh            ; Don't affect bits 0...3
                ANL   DAT,A             ;Low nibble to display
                SETB  EN
                CLR   EN
                RET
;=====================================================================
```

```
LCD_INT:    CLR  RS
//          CLR  RW
            CLR  EN
            SETB  EN
            MOV   DAT,#028h
            CLR  EN
            LCALL SDELAY
            MOV   A,#28h
            LCALL COM
            MOV   A,#0Ch
            LCALL COM
            MOV   A,#06h
            LCALL COM
            LCALL CLEAR
            MOV A,#080H
            LCALL COM
            RET
;=========================================================================
CLEAR:      CLR  RS
            MOV   A,#01h
            LCALL COM
            RET

;=========================================================================
DATAW:      SETB  RS
//          CLR RW
            LCALL W_NIB
            LCALL LDELAY
            RET
;=========================================================================
SDELAY:     MOV R6,#1
HERE2:      MOV R7,#255
HERE:       DJNZ R7,HERE
            DJNZ R6,HERE2
        RET
;=========================================================================
LDELAY:     MOV R6,#100
HER2:       MOV R7,#255
HER:        DJNZ R7,HER
            DJNZ R6,HER2
        RET
;=========================================================================
COM:        CLR RS
//          CLR RW
            LCALL W_NIB
```

```
                LCALL SDELAY
                RET
;=========================================================================
LINE1:          MOV A,#80H
                LCALL COM
                RET
LINE2:          MOV A,#0C0H
                LCALL COM
                RET
LINE3:          MOV A,#0c9H
                LCALL COM
                RET
//LINE4:        MOV A,#0D0H
//              LCALL COM
//              RET
;=========================================================================
LOOP:           CLR A
                MOVC A,@A+DPTR
                JZ GO_B2
                LCALL DATAW
                LCALL SDELAY
                INC DPTR
                SJMP LOOP
GO_B2:          RET
;=========================================================================
MYDATA:             DB " DYNA-I2C-ADDA  ",0
MYDAT2:             DB " ADC i/p ",0
MYDAT3:             DB "=",0

                END
```

**Program 11:**

```
*********************************************************************************
: Objective              : Interfacing ADC 0808 with AT89S8253.
: Program                : To display text on the LCD screen.
: Compiler               : KEIL uvision3
: Development Board      : DYNA51-EB, LCD Module, PIO ADC-01
*********************************************************************************

ORG 0000H
                LJMP MAIN
                ORG 0030H
MAIN:           NOP
                EN     EQU   P2.0
                RS     EQU   P2.2
//              RW     EQU   P1.1
                DAT      EQU   P2
                LCALL LCD_INT
                LCALL CLEAR
                LCALL LINE1
                MOV DPTR,#MYDATA
                LCALL LOOP
                LCALL LINE2
                MOV DPTR,#MYDAT2
                LCALL LOOP
                 LCALL LINE5
                lcall dispH
again:          lcall adconvert
                LCALL LINE3
                lcall adisph
                LCALL LINE4
                lcall adispl

                SJMP again

;=========================================================================
W_NIB:          PUSH  ACC               ;Save A for low nibble
                ORL   DAT,#0F0h         ;Bits 4..7 <- 1
                ORL   A,#0Fh            ;Don't affect bits 0-3
                ANL   DAT,A             ;High nibble to display
                SETB  EN
                CLR   EN
                POP   ACC               ;Prepare to send
                SWAP  A                 ;...second nibble
                ORL   DAT,#0F0h         ; Bits 4...7 <- 1
                ORL   A,#0Fh            ; Don't affect bits 0...3
```

```
                ANL   DAT,A             ;Low nibble to display
                SETB  EN
                CLR   EN
                RET
;=====================================================================
LCD_INT:        CLR   RS
//              CLR   RW
                CLR   EN
                SETB  EN
                MOV   DAT,#028h
                CLR   EN
                LCALL SDELAY
                MOV   A,#28h
                LCALL COM
                MOV   A,#0Ch
                LCALL COM
                MOV   A,#06h
                LCALL COM
                LCALL CLEAR
                MOV A,#080H
                LCALL COM
                RET
;=====================================================================
CLEAR:          CLR   RS
                MOV   A,#01h
                LCALL COM
                RET

;=====================================================================
DATAW:                  SETB  RS
//              CLR RW
                LCALL W_NIB
                LCALL LDELAY
                RET
;=====================================================================
SDELAY:         MOV R6,#1
HERE2:          MOV R7,#255
HERE:           DJNZ R7,HERE
                DJNZ R6,HERE2
                RET
;=====================================================================
LDELAY:         MOV R6,#1
HER2:           MOV R7,#255
HER:            DJNZ R7,HER
                DJNZ R6,HER2
```

```
                RET
;=========================================================================
COM:            CLR RS
//              CLR RW
                LCALL W_NIB
                LCALL SDELAY
                RET
;=========================================================================
LINE1:          MOV A,#80H
                LCALL COM
                RET
LINE2:          MOV A,#0C0H
                LCALL COM
                RET
LINE3:          MOV A,#0c9H
                LCALL COM
                RET
LINE4:          MOV A,#0cah
                LCALL com
                RET
LINE5:          MOV A,#0cbh
                LCALL com
                RET
;=========================================================================
LOOP:           CLR A
                MOVC A,@A+DPTR
                JZ GO_B2
                LCALL DATAW
                LCALL SDELAY
                INC DPTR
                SJMP LOOP
GO_B2:          RET

ADisph:
                MOV a,r3
                LCALL DATAW
                LCALL SDELAY
                ret
ADispl:
                MOV a,r4
                LCALL DATAW
                LCALL SDELAY
                ret

;=========================================================================
```

```
MYDATA:     DB " PIO-ADC-01  ",0
MYDAT2:     DB " ADC i/p= ",0
dispH:
            MOV a,#48H  ;ascii for H
            LCALL DATAW
            LCALL SDELAY
            ret

;=====================================================================
adconvert:
;-------------------------------------------------------
START  EQU  P1.0          ; Pin 6 Start
EOC    EQU  P1.3          ; Pin 7 EOC
OE     EQU  P1.1          ; Pin 9 Output Enable
ALE    EQU  P1.2          ; Pin 22 ALE
adata  EQU  P0            ; Data Lines

;-------------------------------------------------------
                           ; Read one byte of data from adc.
                           ; Performs a analog conversion cycle.
                           ; address of channel in register "ADDRESS",
                           ; Returns data in BUFFER
                           ; Destroys A.
       MOV adata,#0FFH     ; Data lines for input
       SETB OE             ; Disable output
        SETB ALE           ; Latch the address
       NOP
       nop
       nop
       NOP
       SETB START          ; Start the conversion
       NOP
       NOP
       NOP
       CLR START
       NOP
       NOP
EOCLOOP:
       JNB EOC, EOCLOOP    ; Do until EOC high
       CLR OE              ; Output Enable
       MOV a,adata         ; Get data in buffer
       SETB O
       CLR ALE

;-------------------------------------------------------
```

```
; binary to ascii conversion
;
*********************************************************************************
        mov     R1,a
        anl   a,#0f0h           ; a=xxxx0000    ; A && #f0h (get the high nibble)
        swap  a                 ; a=0000xxxx    ; swap nibbles
        orl   a,#30h            ; a=0011xxxx    ; add #30h, if nibble is
        mov r2,a
        mov b, #3ah             ; stores #3ah in B
        div ab                  ; divide A/#3ah
        jz recupera1            ; if zero, nibble < #0Ah
nibble1_ok:
        mov a,r2
        add a, #07h             ; adds #07h to get ASCII of A-F
        mov  r3,a
        jmp nibble2

 recupera1:
        mov a,r2
        mov r3,a
        nibble2:
        mov     a,r1
        anl   a,#0fh            ; a=0000xxxx    ; A && #0fh (get low nibble)
         orl   a,#30h           ; a=0011xxxx    ; add #30h, if nibble is
        mov   r2,a              ; store acc. at Reg.R2
        mov b, #3ah             ; stores #3ah in B
        div ab                  ; divide A/#3ah
        jz  recupera2           ; if zero, nibble < #0Ah
        mov a,r2
        add a, #07h             ; adds #07h to get ASCII of A-F
        mov  r4,a
        ret                     ; return to main routine
        recupera2:
        mov     a,r2
        mov r4,a
        ret                      ; return to main routine
end
```

**Program 12:**

```
*********************************************************************************
: Objective            : Interfacing DAC 0809 with AT89S8253.
: Program              : To display text on the LCD screen.
: Compiler             : KEIL uvision3
: Development Board    : DYNA51-EB, LCD Module, PIO DAC-01
*********************************************************************************
```

**For ramp wave form**

```
org     0000h
        mov p0,#00h
        mov p1,#00h
        mov p1,#01h
up:     mov a,#00h
next:   mov p0,a
        inc a
        cjne a,#0ffh,next
        ljmp up
end
```

**For Square waveform**

```
org     0000h
        mov p0,#00h
        mov p1,#00h
        mov p1,#01h
        mov a,#00h
next:   mov p0,a
        inc a
        cjne a,#0ffh,next
tt:     dec a
        mov p0,a
        cjne a,#00h,tt
        ljmp next
end
```