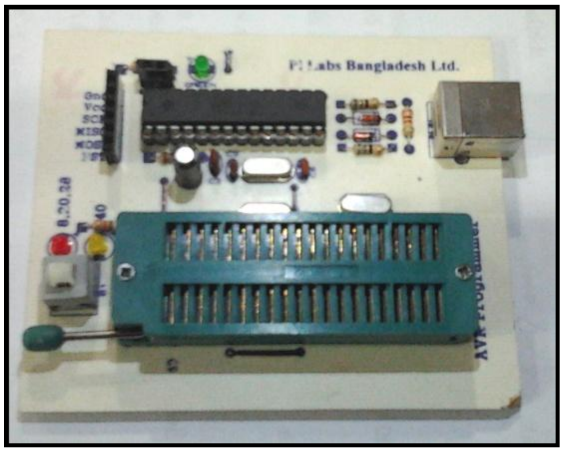
User Manual Of AVR Programmer

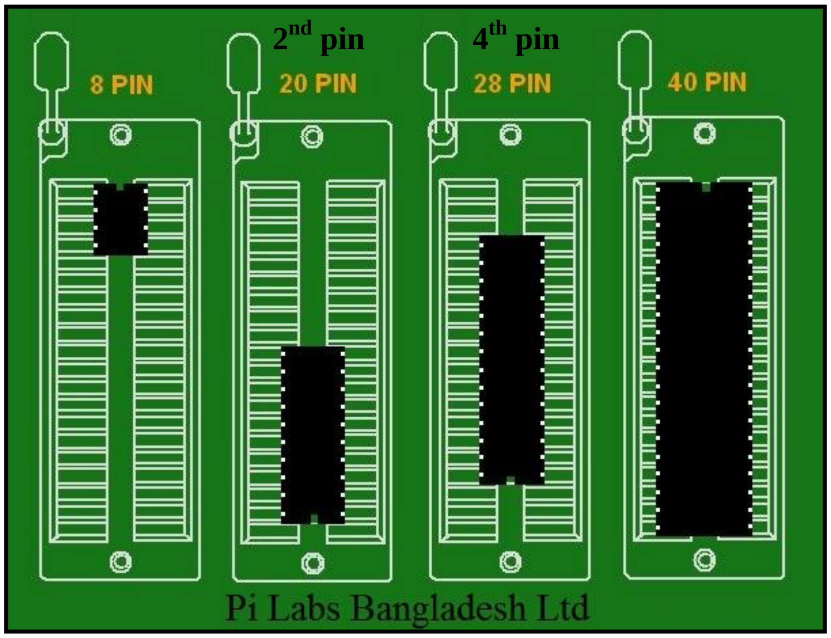


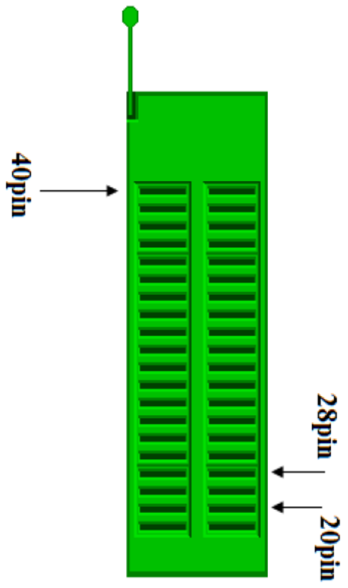
Pi Labs Bangladesh Ltd.

Contents

1) Placement

For 40pin AVR microcontroller press the button (B1) and yellow LED will lit. For other AVR microcontroller release the button and red LED will lit. This Programmer supports all the 5 volt AVR Microcontroller.





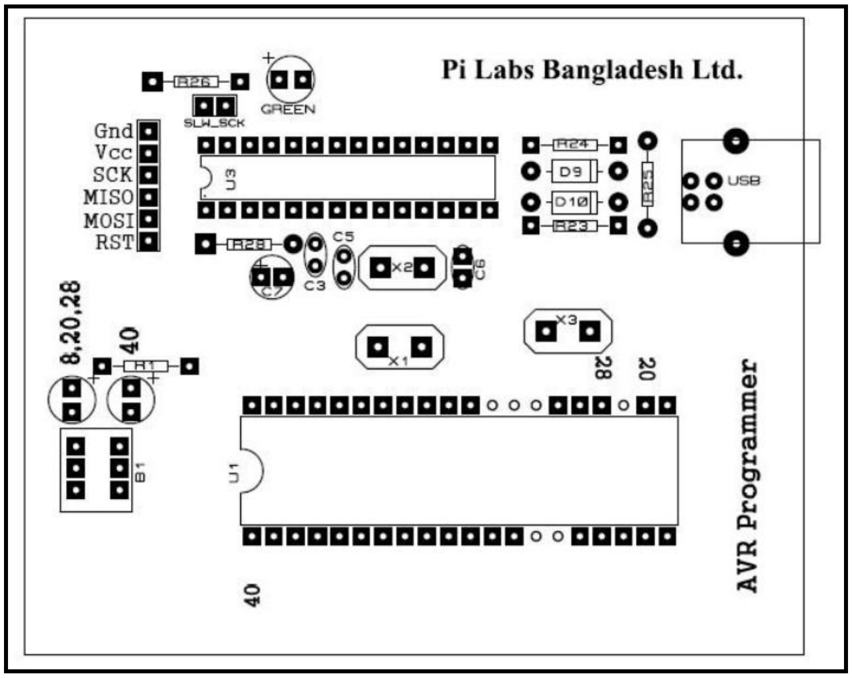


Figure: AVR Programmer Layout

2) Burner Software

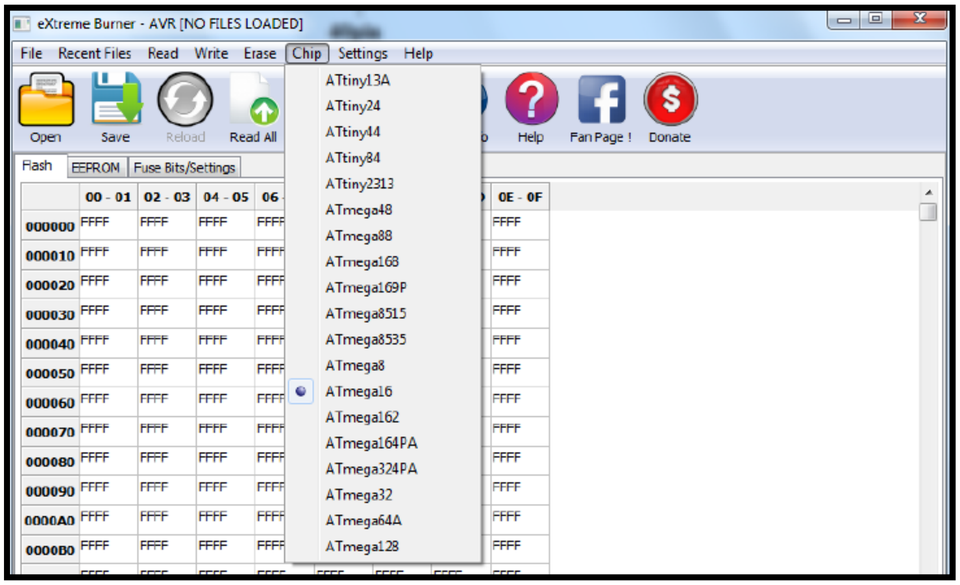
We provide two open source software:

1) Extreme Burner (site link):

At the time I am writing this manual, V1.2 is running. No need to install Driver for USBasp, its included in the Extreme Burner Installation.

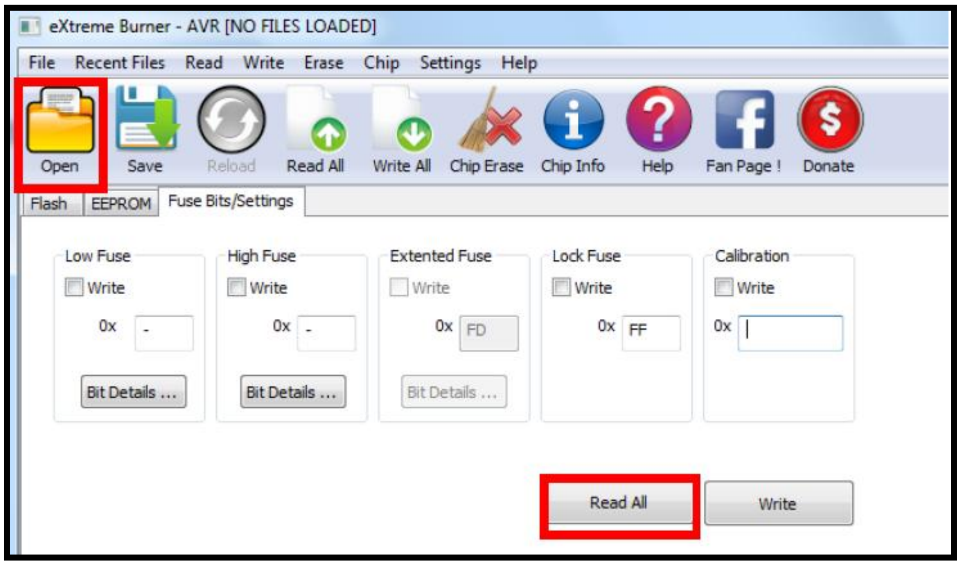
Programming Steps:

- Plug the AVR Programmer into your Windows PC.
- Install the "Extreme Burner V1.2". Driver will be installed with the installation process.
- Open Extreme Burner.
- Under drop-down menu choose your desired chip.



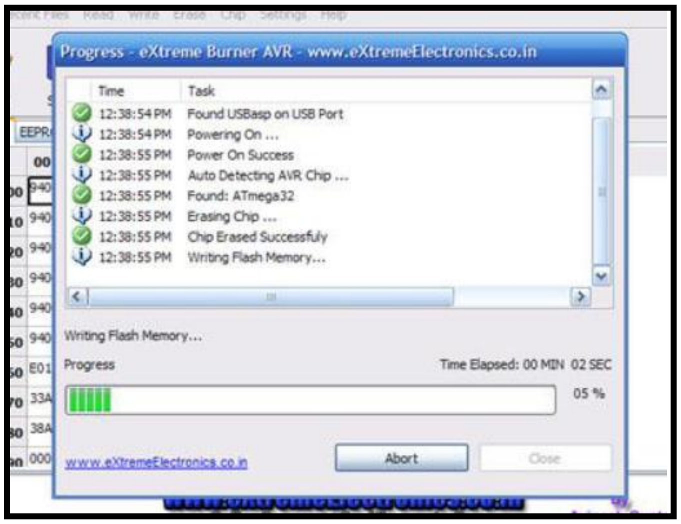
```
        00 - 01  02 - 03  04 - 05  06     0E - 0F
000000  FFFF     FFFF     FFFF     FFFF   FFFF
000010  FFFF     FFFF     FFFF     FFFF   FFFF
000020  FFFF     FFFF     FFFF     FFFF   FFFF
000030  FFFF     FFFF     FFFF     FFFF   FFFF
000040  FFFF     FFFF     FFFF     FFFF   FFFF
000050  FFFF     FFFF     FFFF     FFFF   FFFF
000060  FFFF     FFFF     FFFF     FFFF   FFFF
000070  FFFF     FFFF     FFFF     FFFF   FFFF
000080  FFFF     FFFF     FFFF     FFFF   FFFF
000090  FFFF     FFFF     FFFF     FFFF   FFFF
0000A0  FFFF     FFFF     FFFF     FFFF   FFFF
0000B0  FFFF     FFFF     FFFF     FFFF   FFFF
```

- Click on the Fuse Bits/Settings tab and choose "Read All" for read your chip's Fuse Bytes.



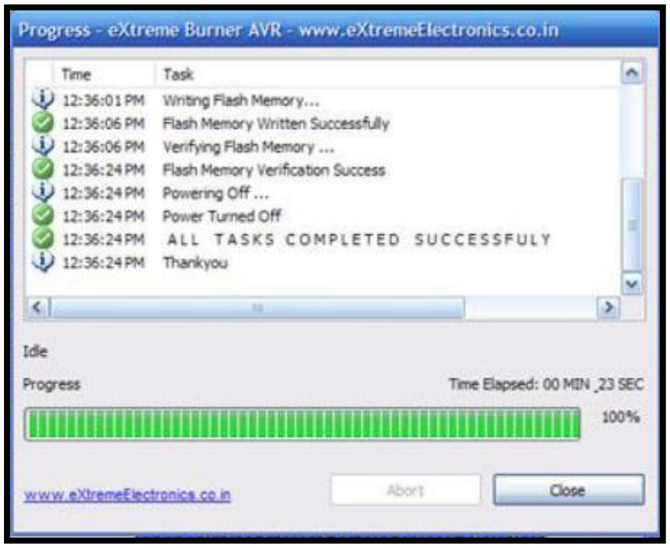
Low Fuse	High Fuse	Extented Fuse	Lock Fuse	Calibration
☐ Write	☐ Write	☐ Write	☐ Write	☐ Write
0x -	0x -	0x FD	0x FF	0x

- If you want to write Fuse Bytes into your chip, write down it on the blank boxes, check the "write", and then press write. For more on Fuse Bits settings see chapter 3. Also you can visit: http://www.engbedded.com/fusecalc/
- For loading Hex file press "Open" and browse the hex file.
- From the drop-down menu choose Write > Flash


Progress - eXtreme Burner AVR - www.eXtremeElectronics.co.in
Abort
Close
www.eXtremeElectronics.co.in

```
Time          Task
12:38:54 PM   Found USBasp on USB Port
12:38:54 PM   Powering On ...
12:38:55 PM   Power On Success
12:38:55 PM   Auto Detecting AVR Chip ...
12:38:55 PM   Found: ATmega32
12:38:55 PM   Erasing Chip ...
12:38:55 PM   Chip Erased Successfuly
12:38:55 PM   Writing Flash Memory...

Writing Flash Memory...
Progress                         Time Elapsed: 00 MIN 02 SEC
                                                        05 %
```

- If programming is successful you will see just like below:


Progress - eXtreme Burner AVR - www.eXtremeElectronics.co.in
Abort
Close
www.eXtremeElectronics.co.in

```
Time          Task
12:36:01 PM   Writing Flash Memory...
12:36:06 PM   Flash Memory Written Successfully
12:36:06 PM   Verifying Flash Memory ...
12:36:24 PM   Flash Memory Verification Success
12:36:24 PM   Powering Off ...
12:36:24 PM   Power Turned Off
12:36:24 PM   ALL TASKS COMPLETED SUCCESSFULY
12:36:24 PM   Thankyou

Idle
Progress                         Time Elapsed: 00 MIN 23 SEC
                                                        100%
```

To know more about this software, you can visit either the official site (eXtreme Electronics) or go through our tutorial site (link 3).

Link:

1) http://extremeelectronics.co.in/avr-tutorials/gui-software-for-usbasp-based-usb-avr-programmers/
2) http://extremeelectronics.co.in/software/extreme-burner-goes-social/
3) http://techshopbd.com/index.php/tutorial-categories/avr-tutorial/microcontroller-input

2) AVRpal.net: This software is developed by Hasanur Rashid, one of the closest people of Pi Labs Bangladesh Ltd. At the time I am writing this manual, I have V2.3.0 in my hand.

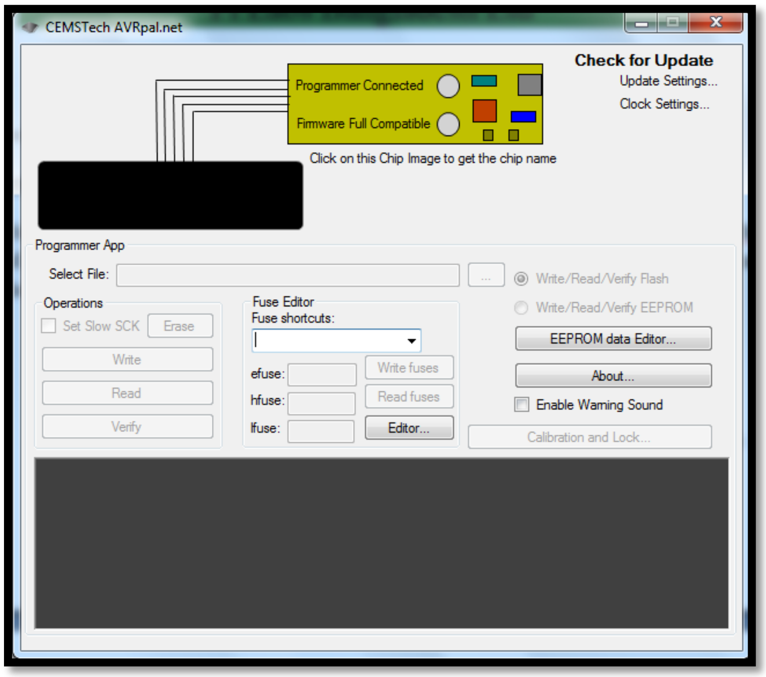
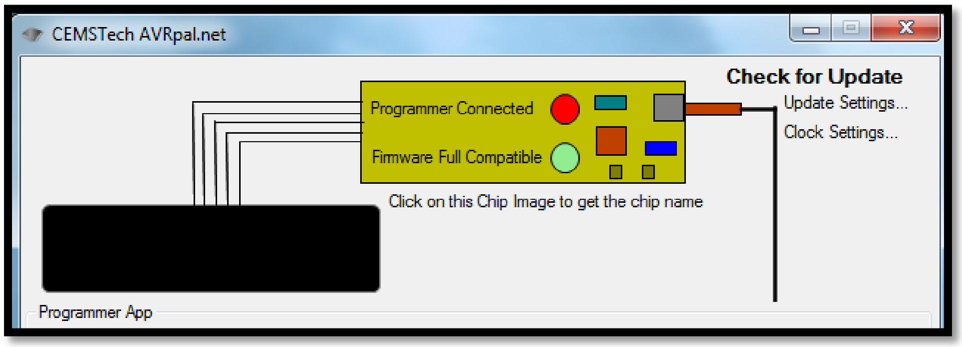
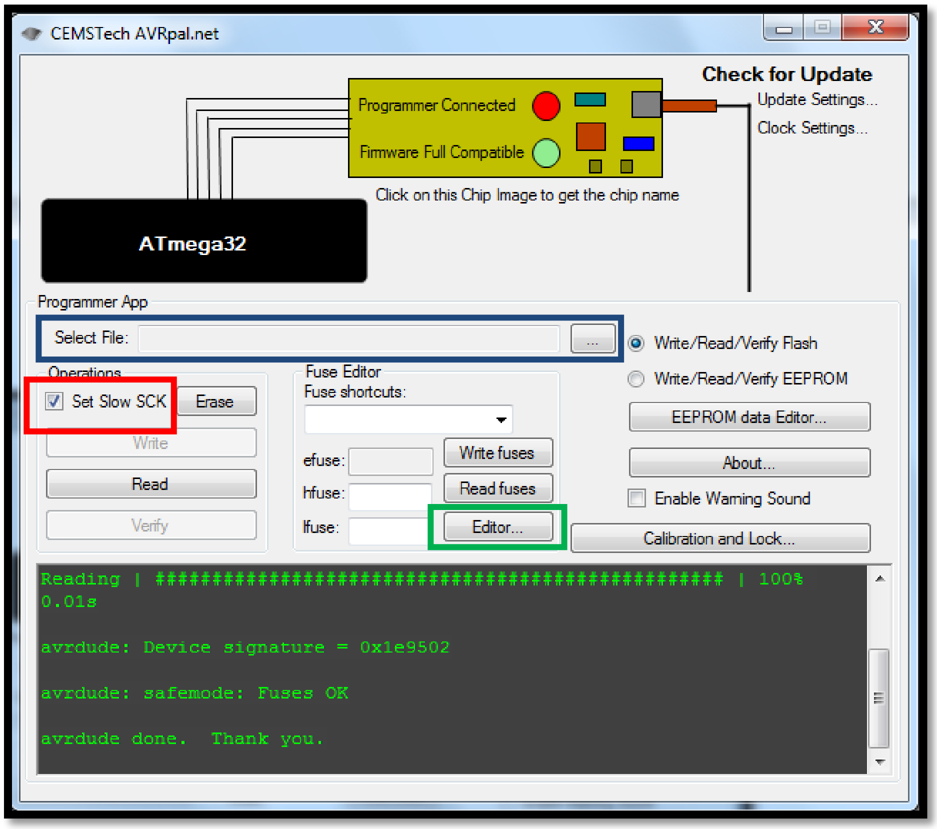


Fig: User Interface of AVRpal.net

If AVRpal.net detect you programmer, then “Programmer Connected” and “Firmware Full Compatible” LED will lit.



Now click on the “Black Box” (which actually represents IC), and software will detect which mcu is present the zif socket. Make sure you checked “Set Slow SCK” (red box). A detail on “Slow SCK” is discussed in “Fuse Byte Tutorial” section (Chapter 3).

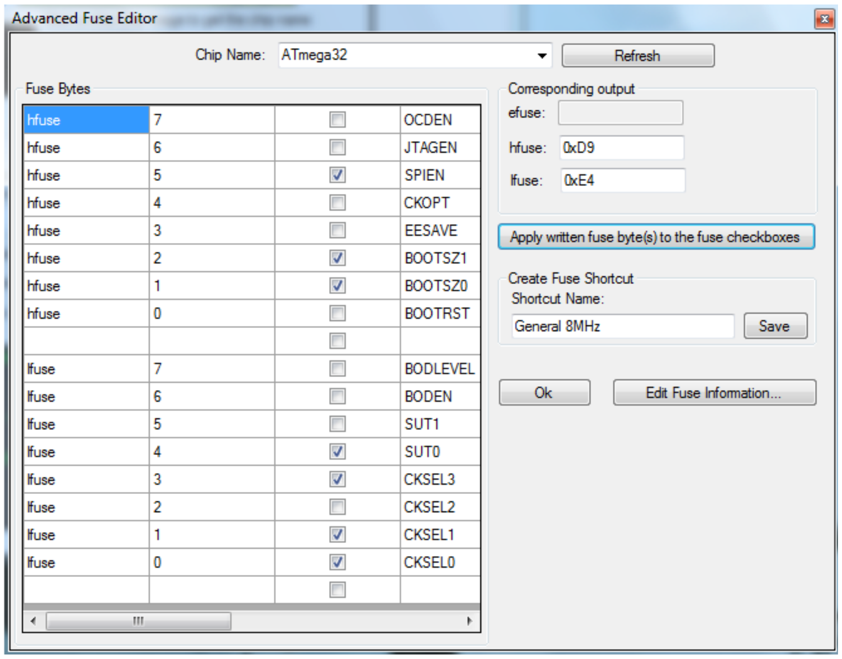


Set Slow SCK [✓]
Write/Read/Verify Flash [✓]

```
Reading | ################################################## | 100%
0.01s

avrdude: Device signature = 0x1e9502

avrdude: safemode: Fuses OK

avrdude done.  Thank you.
```

So, it detects my mcu (ATmega32). To load “Hex” file browse it from “Select File” (blue box). You can also read/write fuses. Take great care when you change the fuse bits. If do not know anything about fuse bits, please read “Fuse byte Tutorial” section.

By using Fuse Editor (green box) you can save fuse byte for specific mcu.



Chip Name: ATmega32

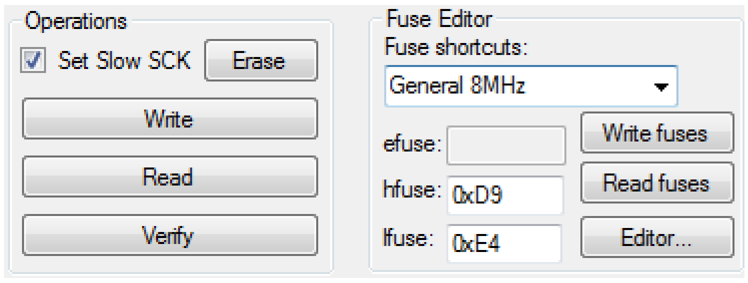
Fuse Bytes

hfuse	7	☐	OCDEN
hfuse	6	☐	JTAGEN
hfuse	5	✓	SPIEN
hfuse	4	☐	CKOPT
hfuse	3	☐	EESAVE
hfuse	2	✓	BOOTSZ1
hfuse	1	✓	BOOTSZ0
hfuse	0	☐	BOOTRST
		☐	
lfuse	7	☐	BODLEVEL
lfuse	6	☐	BODEN
lfuse	5	☐	SUT1
lfuse	4	✓	SUT0
lfuse	3	✓	CKSEL3
lfuse	2	☐	CKSEL2
lfuse	1	✓	CKSEL1
lfuse	0	✓	CKSEL0
		☐	

Corresponding output
efuse:
hfuse: 0xD9
lfuse: 0xE4

Shortcut Name: General 8MHz

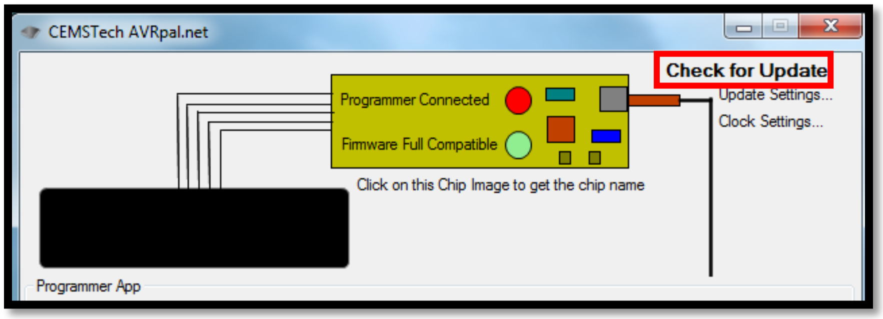
For instance, I frequently use L=0xE4 / H=0xD9 fuse bytes. So I named it "General 8MHz" and then press "Save".


Operations
Set Slow SCK ✓

Fuse Editor
Fuse shortcuts: General 8MHz
efuse:
hfuse: 0xD9
lfuse: 0xE4

Now if you click on the "Fuse Shortcuts" drop down menu, you'll see "General 8MHz". If you select it, corresponding fuse bytes will load. If there is another mcu in zif socket other then ATmega32, it won't be available, which prevents write wrong fuse bytes.

As this software is still in beta testing, so always look for updates (red box). Also if you find any bug, please report it on the site mentioned above.


CEMSTech AVRpal.net
Programmer Connected
Firmware Full Compatible
Click on this Chip Image to get the chip name
Programmer App

3) Fuse Byte Tutorial

I guess many of you were confused when programming AVR fuse bytes. I get many newbie questions like I programmed AVR but it doesn’t work. 90% of them always set wrong fuse bytes and make them DEAD or unusable. In this tutorial first I will discuss about fuse bits and then how to active so called “DEAD” chips.

Introduction

Fuses are an extremely important part programming a chip, but are rarely explained thoroughly. You only need to set them once, but if you don't do it right, it's a disaster!

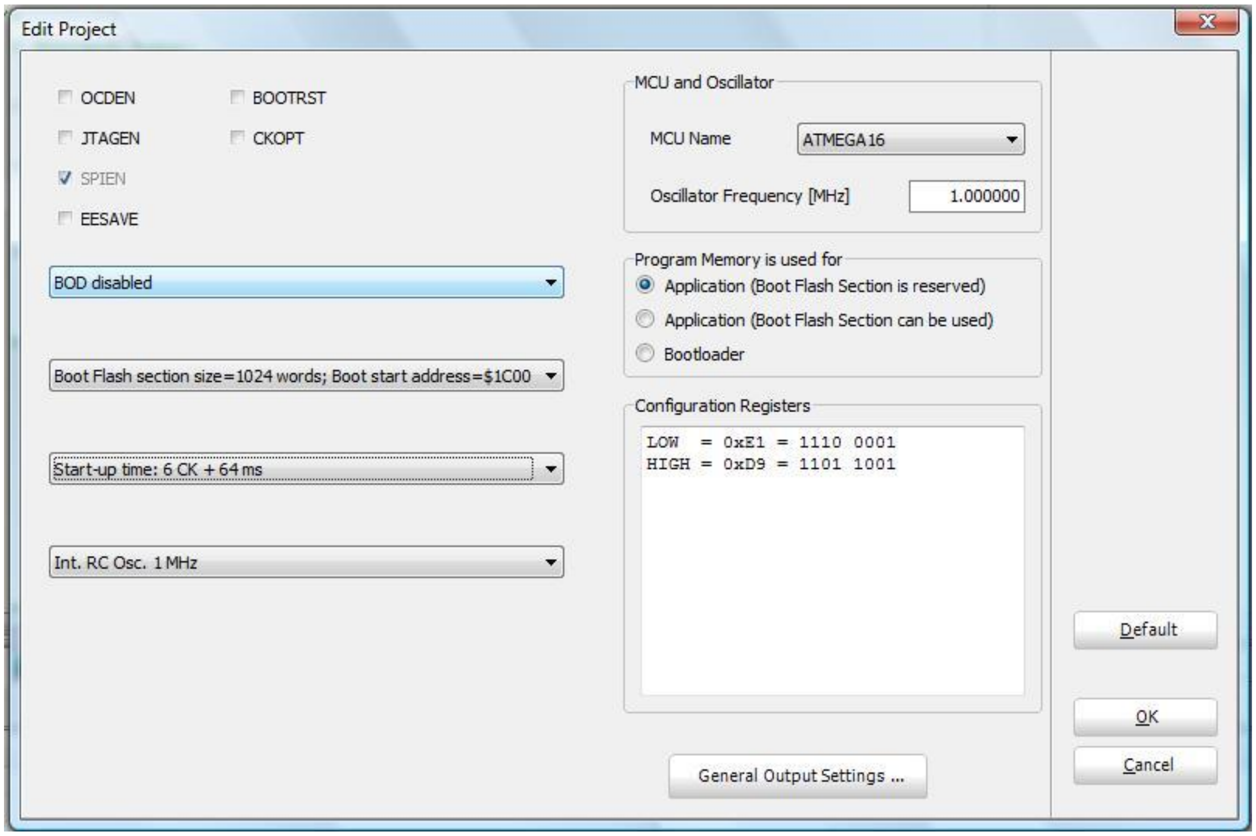
You know about flash, EEPROM and RAM as parts of the chip. What I did not mention is that there are also 3 bytes of permanent (by permanent I mean that they stick around after power goes out, but that you can change them as many times as you'd like) storage called the fuses. The fuses determine how the chip will act, whether it has a bootloader, what speed and voltage it likes to run at, etc. Note that despite being called 'fuses' they are re-settable and don’t have anything to do with protection from overpowering (like the fuses in a home).

The fuses are documented in the datasheets, but the best way to examine the fuses is to look at a fuse calculator such as in “MikroC” compiler. Please collect the latest version. When I am writing this tutorial I have version 4.6 in my hand.

If you want to enable something in AVR what you do? Most probably you set corresponding bit as ‘1’, right? For Fuse bit its opposite. Here ‘1’ means un-programmed or disable and ‘0’ means programmed or enable. Always remember these.

Fuse bit = 0 => fuse bit is PROGRAMMED
Fuse bit = 1 => fuse bit is UN-PROGRAMMED

Open a project. I am assuming that you are using ATmega16. Go to “Project Menu” and click on “Edit Project” (Shifht+Ctrl+E). A window will open just like below:


Edit Project

- [ ] OCDEN
- [ ] BOOTRST
- [ ] JTAGEN
- [ ] CKOPT
- [x] SPIEN
- [ ] EESAVE

BOD disabled

Boot Flash section size=1024 words; Boot start address=$1C00

Start-up time: 6 CK + 64 ms

Int. RC Osc. 1 MHz

MCU and Oscillator

MCU Name: ATMEGA16

Oscillator Frequency [MHz]: 1.000000

Program Memory is used for

- ● Application (Boot Flash Section is reserved)
- ○ Application (Boot Flash Section can be used)
- ○ Bootloader

Configuration Registers

```
LOW  = 0xE1 = 1110 0001
HIGH = 0xD9 = 1101 1001
```

SPIEN bit is for programming. It should be enable for load program into chip. If you disable it you can't program no more. So 'MikroC' make this fixed and you can't change it. If you check other bits (Like: JTAGEN) you will see the corresponding bits in "Configuration Registers" box (bottom right) changed to Zero. You can know about other fuse bits in ATmega16 datasheet page 255. I like to mention one thing before I go to next stage. By default in ATmega16, JTAG enabled by default. That's why you can't use PORTC2-5 as digital I/O. So you have to disable it.

Brown-out Detect (BOD)

The first drop down menu is "Brown Out Detect". These fuses set what voltage to turn the **Brownout** protection circuitry on. A brownout for a chip means that the power voltage is too low for it to run reliably at the speed of the clock.

For example, the attiny2313 can run as fast at 20MHz but only if the power voltage is between 4.5V and 5.5V. If the voltage is lower than that, it may behave erratically, erasing or overwriting the RAM and EEPROM. It may also start running random piece of the flash program. To keep it from doing that, set the brownout voltage to 4.3V, then if the voltage dips, the chip will turn off until the voltage returns. It will then reset and start over.

If the chip is meant to run at 5V, set the brown-out to 4.3V. If the chip can run as low as 3.3V you can set the brown-out to 1.8V. If the chip is a 'low voltage compatible' chip such as the attiny2313V (which can run as low as 1.8V if its clocked at 4MHz or less) then you can set the brownout to 1.8V.

For simplicity, disable it.

Clock selection

The 2nd option is how the chip is clocked. Every CPU uses a clock, in general one assembly code instruction is run every clock cycle. The one in your PC has a clock that runs at 1GHz or higher. This little chip runs much slower. If you look at the menu you'll see a huge list of options, but looking carefully you'll see there are two groupings, the **Clock Source** and the **Startup Time**.

The **Clock Source** can be either of the following:

```
External Clock, Internal 8MHz clock, Internal 4MHz clock, Internal 128KHz clock,
External Crystal (0.4-0.9 MHz), External Crystal (0.9MHz - 3.0MHz), External
Crystal (3.0MHz - 8.0MHz) or External Crystal (8.0MHz +)
```

External Clock means that a square wave is being input into the **CLOCK-IN** pin. This is pretty rare unless you have a clock generating chip. Don't use this unless you're sure you mean to.

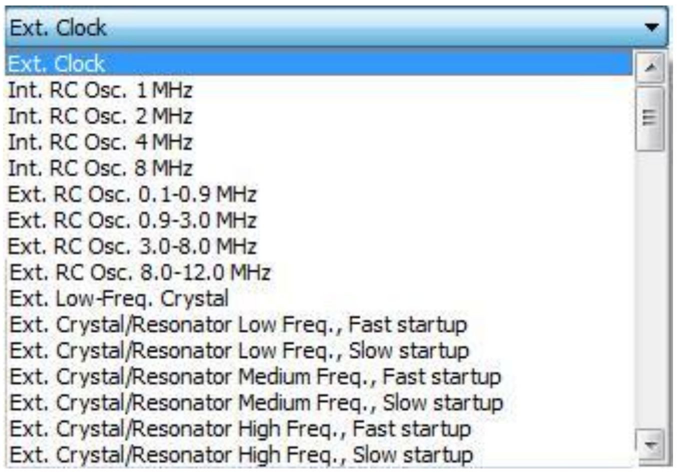
Internal Clock means that there’s a little oscillator inside the chip, it’s not very precise but good for most projects that don’t have fine timing issues. The clock varies with temperature and the power supply voltage. You can choose from a 1MHz, 2MHz, 4MHz or 8MHz clock. Having an internal oscillator means we don't need to wire up a crystal and we can use the clock pins for our own nefarious purposes.

External Crystal
If you need a special clock rate, like 3.58MHz or 12MHz or a high precision clock that won't drift with the temperature, you'll want an external crystal or oscillator.

Crystal Oscillator

Ceramic Oscillator

Clock settings has following options: →

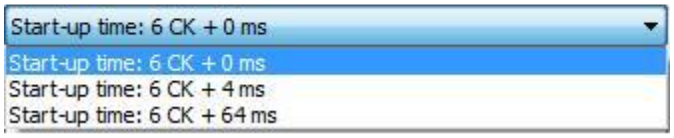

Ext. Clock
Ext. Clock
Int. RC Osc. 1 MHz
Int. RC Osc. 2 MHz
Int. RC Osc. 4 MHz
Int. RC Osc. 8 MHz
Ext. RC Osc. 0.1-0.9 MHz
Ext. RC Osc. 0.9-3.0 MHz
Ext. RC Osc. 3.0-8.0 MHz
Ext. RC Osc. 8.0-12.0 MHz
Ext. Low-Freq. Crystal
Ext. Crystal/Resonator Low Freq., Fast startup
Ext. Crystal/Resonator Low Freq., Slow startup
Ext. Crystal/Resonator Medium Freq., Fast startup
Ext. Crystal/Resonator Medium Freq., Slow startup
Ext. Crystal/Resonator High Freq., Fast startup
Ext. Crystal/Resonator High Freq., Slow startup

Beginners usually use internal 1MHz to 8 MHz. So choose internal oscillator. For this tutorial I choose Int. RC Osc. 1 MHz.

Startup time

The **Startup time** can be either of the following: **6CK + 0 ms, 6CK + 4 ms, 6CK + 64 ms**.

The **Startup Time** is just how long the clock source needs to calm down from when power is first applied. Always go with the longest setting **6CK + 64ms** unless you know for a fact your clock source needs less time and 64ms is too long to wait.
The "Start-up Time" menu has following options. Choose the 64ms:


Start-up time: 6 CK + 0 ms
Start-up time: 6 CK + 0 ms
Start-up time: 6 CK + 4 ms
Start-up time: 6 CK + 64 ms

By default, chips that come from the factory have the **Internal 1 MHz clock with 6CK + 0ms Startup.**

So, in configuration register box I see the fuse bits are: LOW: 0xE1   HIGH: 0xD9

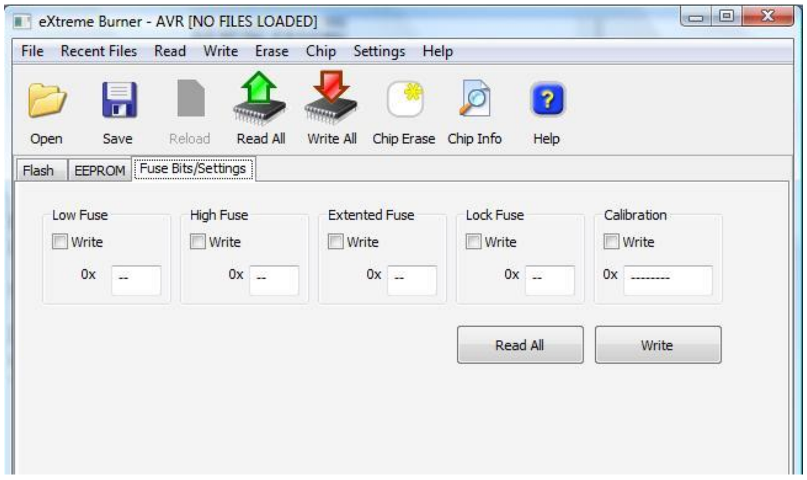
It's time to load this bit into chip. If you use USBasp hardware for loading, then you must be using "Extreme Burner" or "AVRpal.net" software I guess.

By AVRpal.net:

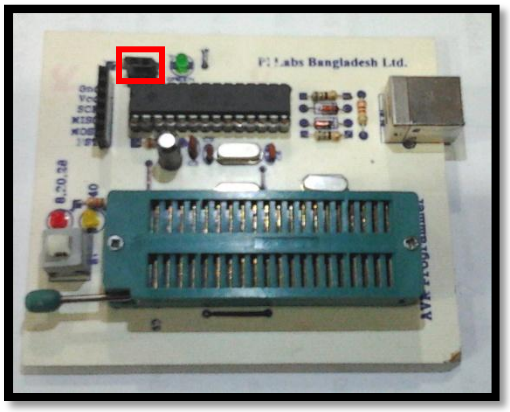
If your chip is brand new, then by default its clock frequency set into internal 1 MHz. So, in AVRpal.net you have to check "Set Slow SCK" box. If you change the fuse bits for higher frequency then you can either "check" or "uncheck" Slow SCK, your choice. The difference is, when "Slow SCK" is checked the program will load slowly and vice versa. REMEMBER, if clock set to 1 MHZ, you have to check "Slow SCK" box. Otherwise it won't work.

By eXtreme Burner:

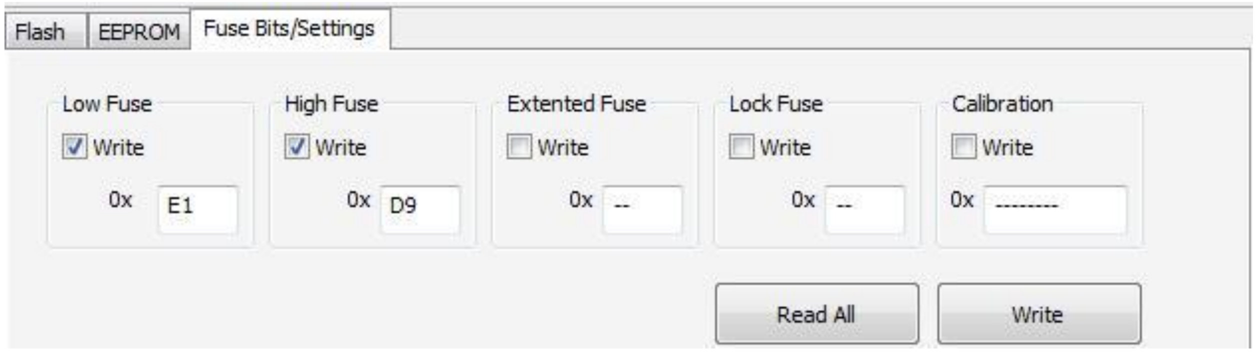
Open the program. Change the tab from "Flash" to "Fuse bits/Settings", just like below:


eXtreme Burner - AVR [NO FILES LOADED]
File Recent Files Read Write Erase Chip Settings Help
Open Save Reload Read All Write All Chip Erase Chip Info Help
Flash EEPROM Fuse Bits/Settings
Low Fuse Write 0x --
High Fuse Write 0x --
Extented Fuse Write 0x --
Lock Fuse Write 0x --
Calibration Write 0x --------
Read All
Write

In this software there is no "Slow SCK" option. You have to do so through hardware (red box). Short it to activate "Slow SCK" and vice versa.



Change the fuse bits and check the "Write" box as below and click "Write" button:



Low Fuse	High Fuse	Extented Fuse	Lock Fuse	Calibration
[✓ Write]	[✓ Write]	Write	Write	Write
0x E1	0x D9	0x --	0x --	0x --------

Also before write the fuse bit, you can also read the default value by clicking "Read All".

4) Troubleshooting

I think the above tutorial will help you set fuse bits properly. But though what if you set wrong fuse bit? (Most of the time it happens with pony prog user). Here comes the solution:

If extreme burner does not detect you chip:

1) First check, is “SLW_SCK” pin shorted (for eXtreme Burner) or checked (for AVRpal.net)? If not, then short/check it and try.
2) If till problem remains then put a crystal (8 or 12MHz, I prefer 12MHz) between XTAL1 and XTAL2 pin, with “SLW_SCK” pin shorted/checked. Then click “Read All”. Most of the time problem will solve in this stage.
3) If till problem exist then maybe you disable SPIEN bit, which is unlikely, a rare situation, but definitely it isn’t for wrong clock fuse bit. If that is the situation then you need high voltage programmer (like TopWin). Also you can use “Fuse bit Doctor” (Google it!).
4) Check the ATmega16’s Vcc and GND pin, see if they get between 4.5 to 5.5V.
5) Test short circuit of MOSI, MISO, SCK, RST pin with your multimeter between your programmer and your chip.

I hope above five steps will solve your problem.

Thank you for using Pi-Labs’ Product.

Fahad Mirza
Embedded System Engineer
Pi Labs Bangladesh Ltd.
fahadmirza80@yahoo.com

LIMITATIONS
Hardware:
This package includes a circuit diagram. This circuit can only be used for programming 5V target systems. For other systems a level converter is needed.