

UISP, Mote Programming, and Mote Fuse HELP GUIDE

UISP HELP GUIDE	1
OVERVIEW	2
UISP PARALLEL PORT PROBLEMS	2
GENERAL	2
UISP AND LINUX	3
ATMEL JTAG POD.....	3
ATMEL AVR ISP IN-SYSTEM-PROGRAMMER.....	4
<i>Required Items</i>	4
<i>Installation</i>	5
ATMEGA128 FUSE CONTROL	6
CONFIGURATION	6
PROGRAMMING FUSES VIA UISP	7

Overview

This document discusses several problems and workarounds encountered by Mote users when programming motes with UISP

UISP Parallel Port Problems

General

Some users experience problems with the UISP download program. These problems manifest themselves in several modes:

1. UISP cannot detect the presence of the Atmega128 processor on the Mote Processor Board (MPR Board)
2. Once programmed the part cannot be reprogrammed.
3. Flash errors occur after download, during the verify cycle.

The following usually cause these problems: (Assuming that the programming board is correctly connected to the PC and the mote power supply/battery voltage is correct)

1. The PC's printer port driver is not supported correctly by the UISP program
2. The PC is outputting serial programming data to the Mote too fast.

If this happens the Atmega's fuses may be set incorrectly. Depending on which fuses have been incorrectly set, this can be fatal and require replacement of the Mote CPU chip. Normally, the fuses can be reset to the correct state using the uisp program (see below) or a JTAG ice.

The following items should be checked if problems occur:

1. Use the simplest printer port driver in the BIOS. These drivers go by various names (SPP, AT..). The printer driver EPP and ECP can cause these problems. To change the driver, reboot the computer, enter the bios (before Windows boots) then change the printer driver.
2. In the makerules file (in the tinyos /apps directory) slow down the uisp clock.

Replace the following:

```
PROGRAMMER_FLAGS=-dprog=dapa $(PROGRAMMER_EXTRA_FLAGS)
```

With:

```
PROGRAMMER_FLAGS=-dprog=dapa -dt_sck=40 $(PROGRAMMER_EXTRA_FLAGS)
```

This will slow down the uisp serial clock to 80usec (40 hi, 40lo)
Faster times may also work.

3. If using batteries, check the battery voltage. If the battery voltage is less than 3.0V the flash may not be reprogrammed correctly. This can also cause the Atmega128 fuses to be set incorrectly which will defeat any further reprogramming. Users are advised to use a Crossbow MIB500 that detects low battery voltages. This unit also accepts an external wall mounted power supply. (5-15VDC)

UISP and LINUX

On RH Linux 8.0 and a Thinkpad T30, UISP version 20010909 as well as the UCB version of UISP 20020626 seems to work: It does not require `-dlpt=3` or `-dt_sck=40`. However, version 20020626 from elsewhere (savannah.nongnu.org) does not seem to work.

<http://webs.cs.berkeley.edu/tos/hardware/hardware.html> to download UCB, UISP 200220626

For other issues on RedHat Linux or non PC Platform, send email to:
tinyos-help@Millennium.Berkeley.EDU

Atmel JTAG Pod

If UISP fails to work users can purchase the Atmel JTAG pod (\$300 from an Atmel distributor) and use AVR Studio to download code into the mote.

http://www.atmel.com/dyn/products/tools.asp?family_id=607

The MIB500 programming board has a JTAG connector (J3, 10 pin header) to attach the JTAG pod. This also requires adding the following lines into the Makerules file in the `tinyos/apps` directory.

below this line:

```
MAIN_SREC = $(BUILDDIR)/main.srec
```

add this line:

```
MAIN_HEX = $(BUILDDIR)/main.hex
```

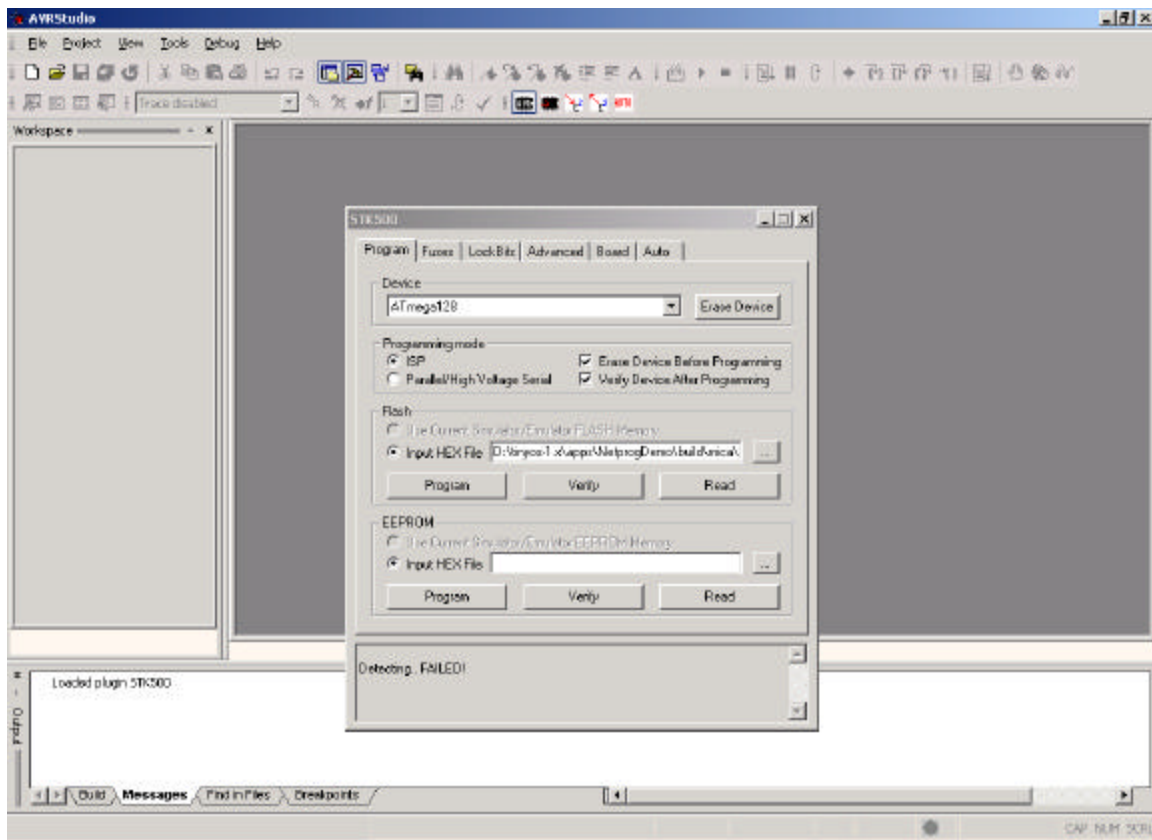
below this line:

```
$(OBJCOPY) --output-target=srec $(MAIN_EXE) $(MAIN_SREC)
```

add this line:

```
$(OBJCOPY) --output-target=ihex $(MAIN_EXE) $(MAIN_HEX)
```

This will generate a hex file along with the standard srec file after executing a “make mica..” command. To load the file through the JTAG pod run AVR studio and use the menu tools/sdk500 menu:



Navigate to the directory containing the hex file and select Program.

Also note that the above dialog box can be used to point and click configure the ATmega fuses. Remember that if your Mote is not programming, the fuses are likely configured incorrectly. (See Fuse section below).

Atmel AVR ISP In-System-Programmer

Required Items

1. Atmel AVR-ISP In System Programmer Kit. See www.atmel.com for details. Available from www.digikey.com pn ATAVRISP-ND \$29.00.
2. Crossbow MIB500 Mote Interface Board
3. AVR-ISP to MIB500 Adapter Cable (see attached schematic).
4. AVRStudio v 4.x installed on PC w/ serial port
5. Target MOTE (MICA2, MICA2DOT)

Installation

1. Modify MIB500 to supply Vcc power to AVRISP pod

Add jumper wire from MIB500's DB25 pin 15 to U6 pin5 or C3 (side closest to the mounting screw). Use AWG30 wire.

2. Install AVRISP-MIB Adapter cable and PC-AVRISP Cable

Open AVRISP Pod and plug 6position connector of Adapter Cable into socket. Note the connector is polarized. DO NOT CLOSE POD

Connect MIB to the DB25 connector end of Adapter cable

Connect PC Serial Port to AVRISP using supplied DB9 cable.

3. Power-Up MIB/AVRISP

Observe AVRISP Green Led is ON indicating power at the AVRISP.

4. Install AVRStudio v 4.x from www.atmel.com website or CD-ROM supplied with AVRISP kit.

5. Update AVRISP Firmware if required

Run AVRStudio and select TOOLS/STK500-AVRISP menu.

If AVRISP Control Panel opens up your AVRISP is up-to-date. Close up AVRISP Pod and proceed to Step 6.

If AVRStudio reports AVRISP Firmware update is required:

- Press CANCEL (the automatic update feature does not work)
- Select HELP/AVR Tools Guide
- Select AVRISP User's Guide and Getting Started
- Select AVRISP Manual Firmware Upgrade (last item in list)
- Follow the Manual Firmware Upgrade procedure
- Close

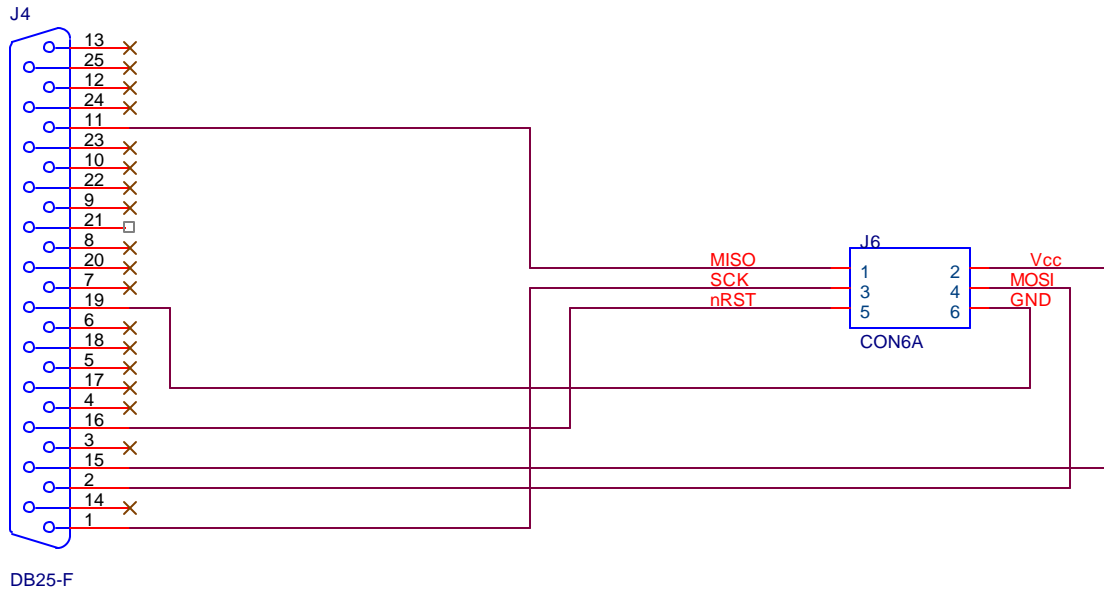
6. Cycle Power on MIB/AVRISP

7. Verify AVRISP Operation

Run AVRStudio and open AVRISP Control Panel via TOOLS/STK500-AVRISP.

Verify reading FUSE settings.

At this point the Programming System is configured and ready for program downloading. Refer to AVRISP User Manual accessible from AVRStudio HELP menu.



DB25-F
AVRISP to MIB Adapter Cable Drawing

ATMega128 FUSE CONTROL

This is a procedure for controlling ATMega processor fuses via a PC parallel port.

The setup is identical as used for general UISP programming of a Mica mode (eg make install.15 mica)

Configuration

You must use a version of UISP.exe that is v2002.06.26 or later.

To determine the uisp.exe version type (in the cygwin window):

```
$ uisp -h
```

which will display something like:

```
uisp version 20020626
```

```
(c) 1997-1999 Uros Platise, 2000-2002 Marek Michalkiewicz
```

If the version is not correct:

1. Install UISP v 2002.06.26 or later

Usually this is installed in cygwin folder under `..\usr\local\bin`

2. Install and run giveio.sys driver if running under NT/2000/XP as follows (you need giveio.sys and loaddrv.exe. A source of both is at National Semiconductor.

Programming Fuses via UISP

1. Run CYGWIN
2. Install MICA PROGRAMMING module (w/ Mica mote installed) on LPT1: port of PC. Power up board – power is NOT available from LPT1 port.

At Cygwin prompt (assumed to be '\$'):

```
$ uisp -dprog=dapa --rd_fuses //note the two (2) dashes - - in front of 'rd_fuses'
Atmel AVR ATmega128 is found.
pulse

Fuse Low Byte      = 0xfe
Fuse High Byte     = 0x19
Fuse Extended Byte = 0xfd
Calibration Byte   = 0xa8 -- Read Only
Lock Bits          = 0xff
  BLB12 -> 1
  BLB11 -> 1
  BLB02 -> 1
  BLB01 -> 1
  LB2 -> 1
  LB1 -> 1
```

This indicates everything is working ok.

To *disable* JTAG and ONCHIP DEBUGGING

```
$uisp -dprog=dapa --wr_fuse_h=0xD9
```

To *enable* JTAG (doing so is required for JTAG download and debugging):

```
$uisp -dprog=dapa --wr_fuse_h=0x19
```

It is recommended the fuse state be verified by reading the settings following a write:

```
$uisp -dprog=dapa --d_fuses
```