

Table of Contents

Overview.....	4
Introduction.....	4
MT-X1 Features.....	4
ATxmega128a1 Features.....	5
MT-X1 Hardware.....	7
Board Features.....	7
Solder Jumpers.....	7
Headers / Pin Descriptions.....	8
Buttons / Jumper.....	9
Power / Status LEDs.....	10
Power Supply.....	10
Clock Sources.....	10
Programming Headers.....	10
USB Shield.....	10
Serial Bridge.....	11
MicroSD Card.....	11
RTC Crystal.....	11
Windows Installation.....	12
WinAVR / AVRDUDE.....	12
AVR Studio / AVRISPMkII driver.....	13
FLIP / DFU Bootloader Driver.....	14
MT-X1 Driver / Serial Configuration.....	15
Terminal Emulator.....	17
Linux Installation.....	19
Drivers.....	19
GCC Toolchain.....	19
AVRDUDE.....	19
dfu-programmer.....	19
Terminal Emulator.....	19
Configuration.....	20
AVRISPMkII PDI Programmer.....	22
Using AVR Studio.....	22
Using AVRDUDE.....	26
Serial Bridge.....	27
DFU Bootloader.....	28
FLIP.....	28
dfu-programmer.....	30
XMEGA Demo Program.....	31
Schematic.....	32

About.....33
 Contact Information.....33
 Support Information.....33
 Precautions.....33
Legal Notices.....34

Overview

Introduction

The MT-X1 is a flexible USB development board for the Atmel ATxmega128a1 microcontroller. The XMEGA can be fully programmed over USB using the onboard AVRISPmkII-compatible PDI programmer (no need to purchase an external programmer). The XMEGA can communicate with a computer using the onboard USB to serial bridge. Speeds up to 2Mbps are supported in synchronous mode (1Mbps in asynchronous mode). The Atmel AT90USB162 USB AVR, which provides these features, will automatically sleep when USB is disconnected. The board can be powered via USB or an external header. Voltage is regulated by a 3.3V, 1A LDO regulator. There are several clock options available onboard, including a 32.768KHz crystal, an external 8MHz clock, an external HC49 crystal landing, and several internal clock options. Several peripheral devices are installed and connected to the XMEGA via solder jumpers, which allows use of the pins if the device is not used. Most pins are routed to headers. These devices include a MicroSD card slot, 32KB SPI SRAM, audio amplifier, relay driver, temperature sensor, 4 buttons, 4 LEDs, and an onboard 1.25V precision reference for the ADC. This reference is setup to work with the XMEGA errata. A demo program is preinstalled on the XMEGA demonstrating use of each peripheral device, as well as demonstrating sleep mode. All software used on both the XMEGA and USB AVR is open-source (MIT license).

MT-X1 Features

- Atmel XMEGA 128A1 (chip rev. H), 128KB flash, 8KB RAM
- Onboard USB PDI programmer (no external programmer needed)
 - AVRISPmkII compatible
 - Program flash, EEPROM, fuses, lock bits, and more
 - Works with AVR Studio 4 and 5, AVRDUDE, Codevision, and BASCOM 2.0.6
- USB - Serial Bridge
 - Up to 2MHz baud rate (1MHz async)
 - Synchronous or asynchronous operation
 - Optional USB ready signal
- 3.3V, 1A LDO regulator
- Powered via USB or external header
- 32.768KHz crystal connected to TOSC (RTC) pins
- 8MHz external clock available from USB AVR
- HC49 crystal landing connected to XTAL pins
- Most pins routed to headers (Port A through Port K)
- Solder jumpers can be used to disconnect devices when not used (frees up header pin)
- MicroSD card slot with push-push spring action
- 32KB SPI SRAM chip
- 8 channel relay driver with kickback protection
 - Up to 70mA per channel
 - 5V or 3.3V devices (relays, LCD backlights, etc.)

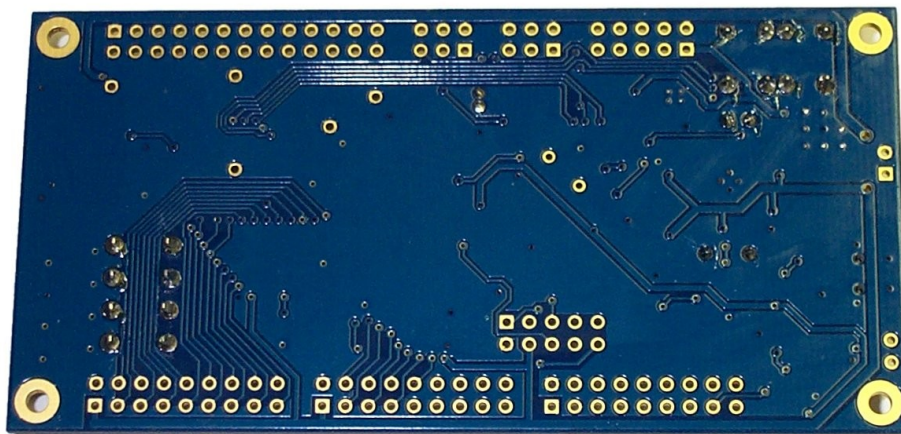
- Can be used as general-purpose low-side driver
- Audio amplifier connected to XMEGA DAC
- Temperature sensor with low-power operation
- 1.25V precision voltage reference
 - Works with XMEGA ADC errata
 - Use for signed differential conversions from 0V to ~2.5V at the pin
 - Routed to both reference inputs via solder jumpers
- 4 buttons
- 4 LEDs
- JTAG (XMEGA), PDI (XMEGA)*, and ISP (USB AVR) headers
- 4 boot modes selectable via jumper and button
 - Serial bridge (default)
 - AVRISPMkII compatible PDI programmer
 - Configuration (uses terminal emulator)
 - DFU bootloader (to update firmware on USB AVR via USB)
- Boot button can be used to toggle between the PDI programmer and the serial bridge
- Preloaded demo program demonstrates onboard peripheral devices as well as sleep mode
- PCB measures 10cm x 5cm
- Compatible with Windows XP/Vista/7 and Linux
- All firmware is open-source (MIT license)
- Uses LUFA USB library and AVRISPMkII clone by Dean Camera (<http://www.fourwalledcubicle.com/>)

ATxmega128a1 Features

- High-performance, Low-power 8/16-bit AVR® XMEGA Microcontroller
- Non-Volatile Program and Data Memories
 - 128K Bytes of In-System Self-Programmable Flash
 - 8K Bytes Boot Section with Independent Lock Bits
 - 2 KB EEPROM
 - 8 KB Internal SRAM
 - External Bus Interface for up to 16M bytes SRAM
 - External Bus Interface for up to 128M bit SDRAM
- Peripheral Features
 - Four-channel DMA Controller with support for external requests
 - Eight-channel Event System
 - Eight 16-bit Timer/Counters
 - 4 Timer/Counters with 4 Output Compare or Input Capture
 - 4 Timer/Counters with 2 Output Compare or Input Capture
 - High-Resolution Extension on all Timer/Counters
 - Advanced Waveform Extension on two Timer/Counters
 - Eight USARTs
 - IrDA modulation/demodulation for one USART

- Four 2-Wire Interfaces w/ dual address match (I2C and SMBus)
- Four SPI (Serial Peripheral Interface) peripherals
- AES and DES Crypto Engine
- 16-bit Real Time Counter with separate Oscillator
- Two Eight-channel, 12-bit, 2 Msps Analog to Digital Converters
- Two Two-channel, 12-bit, 1 Msps Digital to Analog Converters
- Four Analog Comparators with Window compare function
- External Interrupts on all General Purpose I/O pins
- Watchdog Timer with Separate On-chip Ultra Low Power Oscillator
- **Special Microcontroller Features**
 - Power-on Reset and Programmable Brown-out Detection
 - Internal and External Clock Options with PLL and Prescaler
 - Programmable Multi-level Interrupt Controller
 - Sleep Modes: Idle, Power-down, Standby, Power-save, Ext. Stby.
 - Advanced Programming, Test and Debugging Interfaces
 - JTAG (IEEE 1149.1 Compliant)
 - PDI (Program and Debug Interface)
- **I/O and Packages**
 - 78 Programmable I/O Lines
 - 100 - lead TQFP
- **Operating Voltage** (MT-X1 operates at 3.3V)
 - 1.6 - 3.6V
- **Speed performance** (MT-X1 can operate at 0-32MHz)
 - 0 - 12 MHz @ 1.6 - 3.6V
 - 0 - 32 MHz @ 2.7 - 3.6V

Bottom View



MT-X1 Hardware

Board Features

Solder Jumpers

<i>Jumper</i>	<i>Description</i>
J1	USB Shield to gnd (not connected by default)
J2	~5V to relay driver (5V header pin and kickback diodes common cathode)
J3	DACA0 (pin A2) to amplifier audio input
J4	AREF B (pin B0) to 1.25V reference
J5	AREF A (pin A0) to 1.25V reference
J6	ADCA1 (pin A1) to temperature sensor output
J7	External clock input (pin R1) to CLKO from AT90USB162 (8 MHz)
J8	SPI D SS (pin D4) to SRAM chip select (external pullup)
J9	SPI D MOSI (pin D5) to SRAM SI (external pullup)
J10	SPI D SCK (pin D7) to SRAM clock input (external pullup)
J11	SPI D MISO (pin D6) to SRAM SO (external pulldown)
J12	Pin D0 to LED_1
J13	Pin D1 to LED_2
J14	Pin D2 to LED_3
J15	Pin D3 to LED_4
J16	SPI F SCK (pin F7) to relay driver clock input
J17	SPI F MOSI (pin F5) to relay driver SI input
J18	USART F0 RXD (pin F2) to AT90USB162 USART TX
J19	USART F0 TXD (pin F3) to AT90USB162 USART RX (shared with PDI_DATA)
J20	USART F0 XCK (pin F1) to AT90USB162 USART XCK (also USB ready signal)
J21	SPI E MISO (pin E6) to SD card SO (must enable XMEGA pullup)
J22	SPI E SCK (pin E7) to SD card clock input (external pullup)
J23	SPI E MOSI (pin E5) to SD card SI (external pullup)
J24	SPI E SS (pin E4) to SD card chip select (external pullup)

Headers / Pin Descriptions

Pin	Description
External Power Header	Under the default configuration, 5V should be supplied to this pin. Lower voltages may be used down to around 4V (or lower if using less current). Voltages greater than 5.5V require J2 to be disconnected. Disconnecting J2 will disable the 5V output pin and inductive kickback protection of the relay driver. But it will allow voltages up to ~7.5V. This header is reverse polarity protected using a schottky diode.
3.3V output headers (x4)	There are four 2-pin power output headers next to each port header group. The header next to the analog ports (ports A and B) comes from the analog 3.3V rail. Note that if these headers are installed, there will not be enough room to plug in IDC connectors next to each other.
Relay Header 3.3V	This can be used for the positive 3.3V side of a relay or other device.
Relay Header 5V	This can be used for the positive 5V side of a relay or other device. This is also the common cathode of the kickback diodes in the relay driver. Both of these are disabled when J2 is disconnected.
Relay Header 1-8	These are the 8 relay driver outputs. They are open-drain active low. When enabled, the output is connected to ground. When disabled, the pin is in a high impedance state. When driving inductive loads, like relays, free-wheeling diodes provide kickback protection (when J2 connected). Non-inductive loads can also be connected (ie: LCD backlight). Each output is capable of sinking 70mA. Outputs can be combined.
Audio Header	This is the single channel output from the audio amplifier. It can drive 8 ohm loads. 4 ohm loads may also be connected, but under some conditions, distortion or automatic thermal shutdown may occur.
JTAG Header	JTAG header for the XMEGA which can be used for programming, debugging, and JTAG boundary scans. Disable JTAG to gain access to the four underlying analog/GPIO pins.
PDI Header	PDI header for the XMEGA which can be used for programming or debugging. Note that an onboard programmer is already provided. When using this header, J19 MUST be disconnected. This is due to the fact that RX and PDI_DATA are shared. This means that the XMEGA serial TX won't be connected to the USB AVR RX. This doesn't affect programming, but may present a problem in certain situations when debugging. If serial TX is required when debugging, the JTAG header can be used. Alternatively, an external USB-serial bridge can be connected.
ISP Header	ISP header for the USB AVR which can be used for programming or debugging. The USB AVR can be programmed over USB using the DFU bootloader.
Port A	All pins are routed to headers. The 1.25V precision reference can be connected to pin A0 (Vref input) through a solder jumper. The temperature sensor output and audio amplifier input are also connected to this port.

Port B	All pins are routed to headers. The 1.25V precision reference can be connected to pin B0 (Vref input) through a solder jumper. Note that JTAG is connected to pins B4 – B7. JTAG must be disabled to use these pins for other purposes like the ADC.
Port C	All pins are routed to headers. No peripheral devices are connected to this port.
Port D	All pins are routed to headers. This port also connects to the LEDs and 32KB SPI SRAM memory through solder jumpers.
Port E	All pins are routed to headers. This port also connects to the buttons and the MicroSD card slot through solder jumpers.
Port F	All pins are routed to headers. This port also connects to the USART of the USB AVR (RX, TX, and optionally, XCK) as well as the SPI inputs of the relay driver (MOSI and SCK) through solder jumpers. To minimize power consumption, TX should be tristated before entering sleep.
Ports H, J, and K	All pins are routed to headers. No peripheral devices are connected to these ports. They can be used for GPIO or for use with external memory.
Pins Q0 and Q1	The 32.768KHz crystal is connected to these pins, which serve as the TOSC input pins of the RTC.
Pins Q2 and Q3	Pin Q2 is routed to the chip select pin of the relay driver. Pin Q3 is routed to the audio amplifier power-down pin. Neither pin is routed to a header.
Pins R0 and R1	Both of these pins are routed to an HC49 crystal footprint. A 22pF capacitor is also connected to each line. Additionally, R1 can be connected to the USB AVR 8MHz clock output (CLKO) through solder jumper J7.

Buttons / Jumper

There are four modes of operation which are selected using the PROG button and JMP jumper. The button and jumper are sampled when powering up or pressing reset. Additionally, the MT-X1 can be switched between the AVRISPmkII programmer and the serial bridge during runtime by pressing the PROG button. This is useful, for example, to program the XMEGA, then switch to the serial bridge for printf() debugging. The following table lists the mode selection during power-up and reset.

Mode Selection During Power-up and Reset

<i>PROG Button</i>	<i>JMP Jumper</i>	<i>Mode</i>
Pressed	Installed	DFU Bootloader
Not Pressed	Installed	Configuration Mode
Pressed	Not Installed	AVRISPmkII PDI Programmer
Not Pressed	Not Installed	USB Serial Bridge

Power / Status LEDs

There are two green LEDs that are used to indicate USB status, the mode of operation, communication activity, programmer status, and more. The following table lists LED functionality in each mode. Both LEDs are turned off in sleep mode.

LED Functionality

<i>Mode</i>	<i>STS LED</i>	<i>PWR LED</i>
AVRISPmkII Programmer	Programmer Activity	PWM flashing
Configuration Mode	On	On
USB Serial Bridge	RX Activity	TX Activity
DFU Bootloader	On	Off

Power Supply

The MT-X1 can be powered via USB or via an external header. Both sources are connected to the input of a 1A, 3.3V LDO linear regulator through Schottky diodes rated at 2A each. The diodes provide reverse-polarity protection as well as ensuring that current will not flow from one source to the other. For example, if the external header has a greater voltage than the USB VBUS voltage, the diode prevents VBUS from rising to the level of the external voltage.

The 3.3V regulator has thermal protection and foldback current limiting. There is a 10uF capacitor on both the input and output. Note that 10uF is the maximum allowed by the USB specification. When using the external header, additional capacitance may be needed with higher impedance voltage sources (ie: batteries, long cable runs). The regulator input can also be routed to the header pin labeled 5V (near the relay driver).

Clock Sources

TODO (see header / pin descriptions)

Programming Headers

TODO (see header / pin descriptions)

USB Shield

Jumper J1 can be soldered to connect the USB shield to ground. The USB specification calls for the USB shield to be connected to ground on the host side only. However, it may be desired to

ground this on the device side. An 0603 SMT component may be soldered on the solder jumper pads as well.

Serial Bridge

To minimize power consumption, TX should be tristated before entering sleep.

MicroSD Card

Be sure to enable the internal pullup on MISO (all other pins have external pullups).

RTC Crystal

XMEGA Demo uses this for the RTC as well as for the DFLL used to calibrate the internal 32MHz RC oscillator used for the main clock.

Windows Installation

The MT-X1 is supported under Windows XP, Vista (32 and 64 bit), and Windows 7 (32 and 64 bit). There is limited support for Windows 2000. It appears as three different devices to the PC depending on which mode is selected. These devices are the AVRISPmkII compatible programmer, the DFU bootloader (for USB AVR firmware updates), and the USB CDC device (Virtual COM port) which is used for configuration mode and the USB-Serial bridge. Three drivers are required. Two of these drivers are included with software available on the Atmel website. The third driver is included with Windows, but requires an .inf file available on the MattairTech website.

Before plugging in the MT-X1 for the first time, the latest software and drivers must be downloaded. It is important to use the latest versions, especially if installing on Windows Vista or Windows 7. The following table lists the minimum versions of the required software. If the software provides a driver, it is listed as well.

Required Downloads

Software	Version	Driver	URL
WinAVR	20100110	N/A	http://sourceforge.net/projects/winavr/files/WinAVR/20100110/
AVR Studio	4.18 SP3	AVRISPmkII	http://www.atmel.com/dyn/products/tools_cards.asp?tool_id=2725
FLIP	3.4.2	DFU Bootloader	http://www.atmel.com/dyn/products/tools_cards.asp?tool_id=3886
DFU Driver (If FLIP driver fails)	latest	DFU Bootloader	http://www.avrfreaks.net/index.php?module=Freaks%20Academy&func=viewItem&item_type=project&item_id=2196
MT-X1 Driver	latest	CDC driver	http://www.mattairtech.com/
MT-X1 Firmware	latest	N/A	http://www.mattairtech.com/

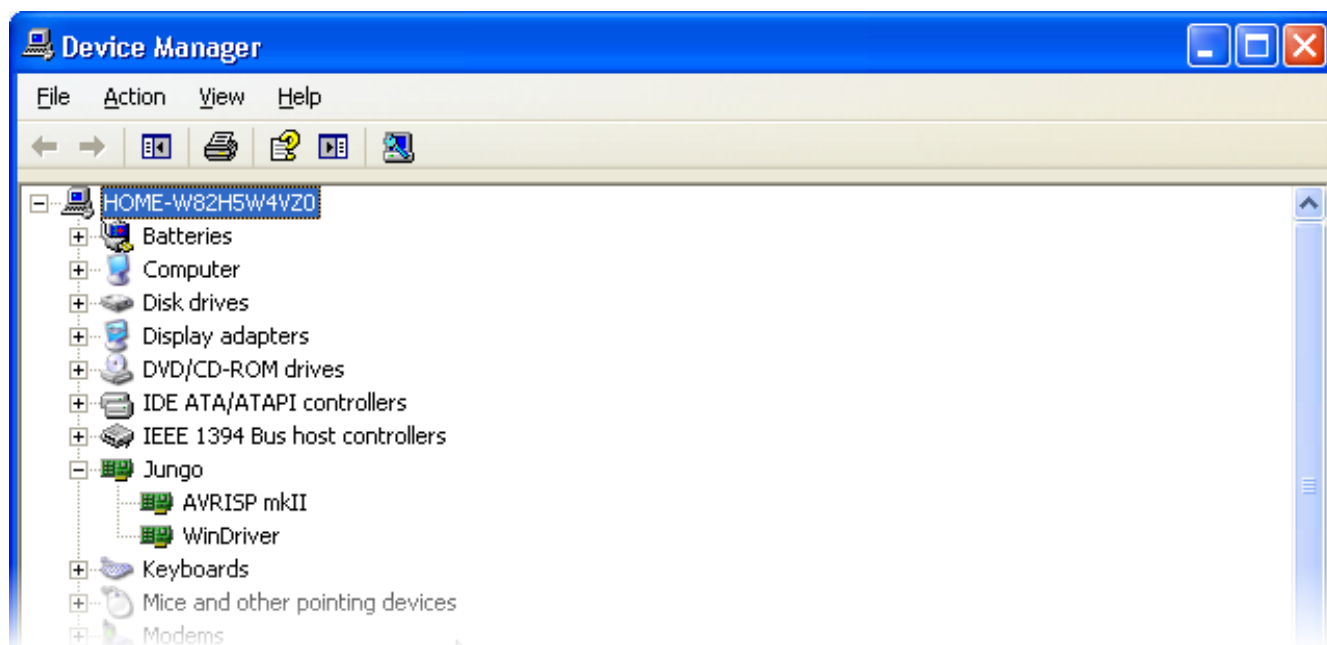
WinAVR / AVRDUDE

WinAVR contains the GNU GCC compiler for C and C++, compiler tools, and libraries (including AVR Libc). It also includes AVRDUDE for Windows, which is a command line tool for transferring firmware to AVR microcontrollers. A graphical tool is included with AVR Studio. Download WinAVR from <http://sourceforge.net/projects/winavr/files/WinAVR/20100110/> and install it first. If you wish to use AVRDUDE, you may need to download and install an update to libusb-win32 available at <http://sourceforge.net/projects/libusb-win32/files/libusb-win32-releases/>. Choose the libusb-win32-devel-filter-x.x.x.x.exe file. Do this only after installing AVR Studio and only if AVRDUDE then does not work.

AVR Studio / AVRISPMkII driver

AVR Studio is an IDE that includes an assembler, debugger, simulator, and an AVR chip programming tool. It works with GCC through a compiler plug-in. Two files must be downloaded, the main program and a service pack. First, download AVR Studio 4.18 from http://www.atmel.com/dyn/products/tools_card.asp?tool_id=2725. Then download SP3 from the same page and begin installation. Be sure to install the Jungo drivers when asked during both setup procedures. They include the AVRISPMkII driver needed by the MT-X1 PDI programmer.

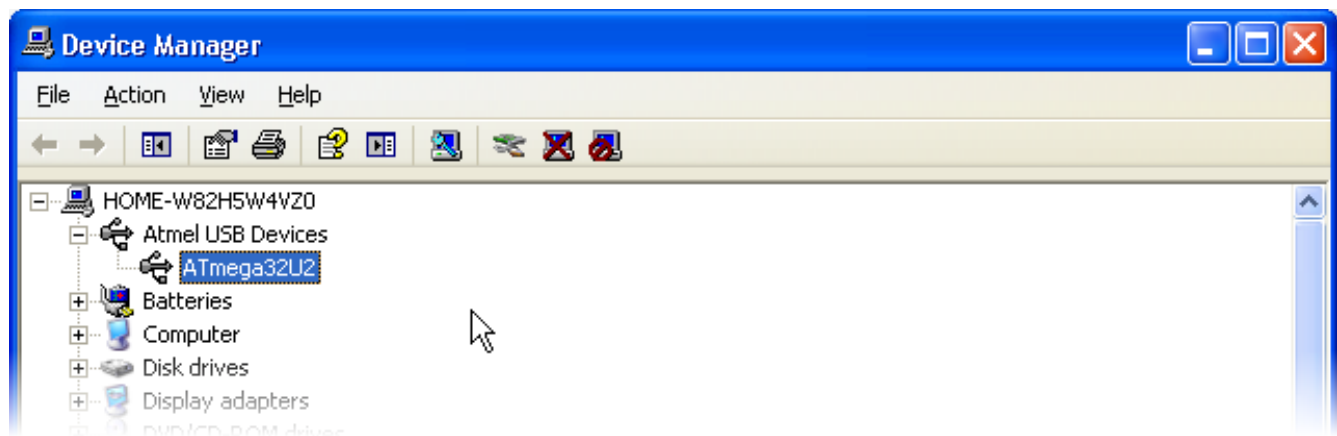
Now the AVRISPMkII driver can be installed. Remove jumper JMP and plug in or reset the MT-X1 while holding down the PROG button. This will run the AVRISPMkII compatible PDI programmer. LED_STS should be lit and LED_PWR should be PWM flashing on and off. Windows will then prompt you for the MT-X1 AVR Programmer driver. By default, this is located in the Program Files/Atmel/AVR Jungo USB directory. Point the installer to the appropriate subdirectory for your PC architecture (usb32 or usb64) and install the driver. Do not use the driver in the AVR Tools/usb directory. Once the driver is loaded, the device will appear as the AVRISPMkII device under Jungo in the device manager.



FLIP / DFU Bootloader Driver

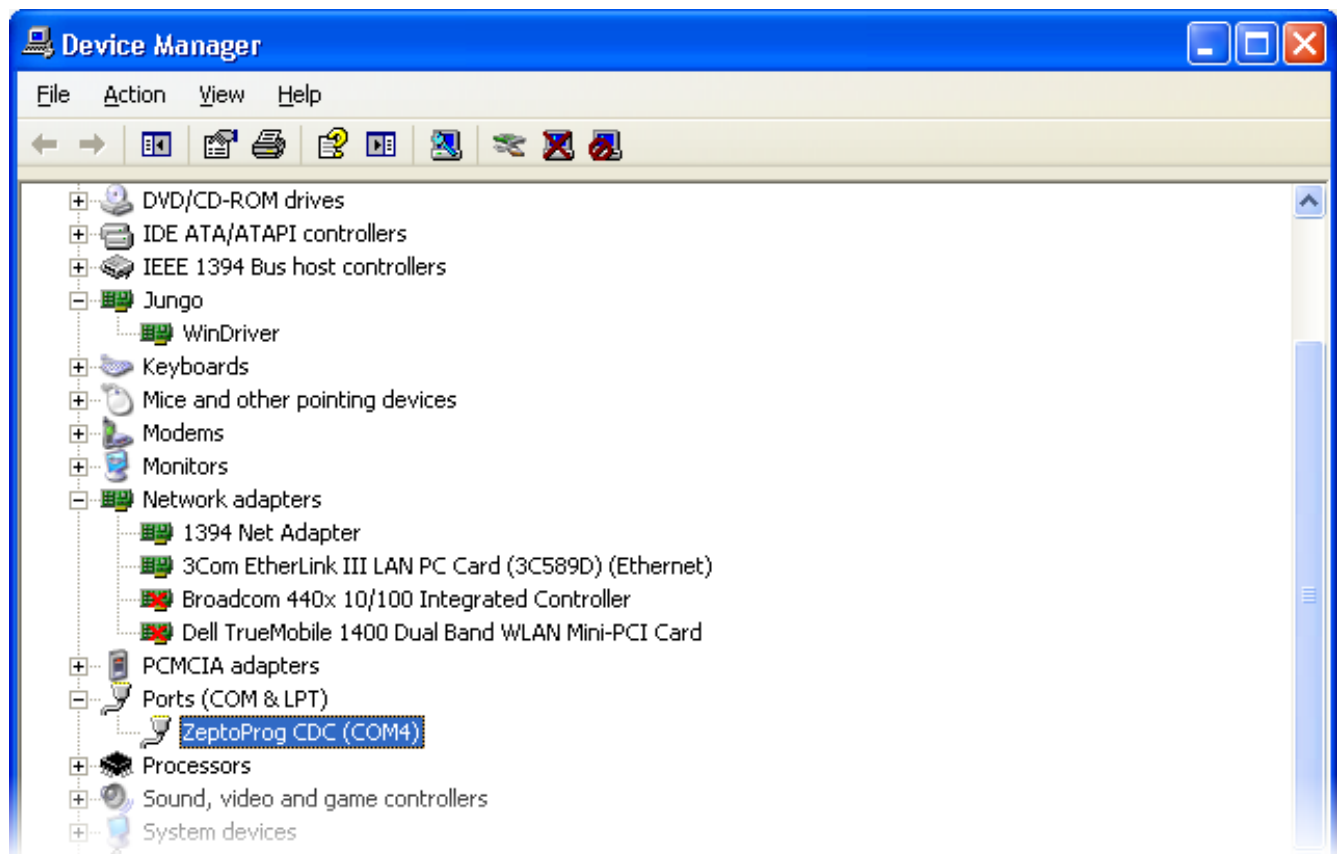
FLIP is a graphical utility used to load firmware updates onto the MT-X1 USB AVR. Updates will be made available at the MattairTech website. FLIP includes the DFU bootloader driver. Download FLIP 3.4.2 or higher from http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3886 and install. If required to install signed drivers and the Atmel drivers do not work, download the signed drivers at http://www.avrfreaks.net/index.php?module=Freaks%20Academy&func=viewItem&item_type=project&item_id=2196 and install.

Once FLIP is installed, the DFU bootloader drivers can be loaded. Install jumper JMP and press button PROG and press reset. This will enter the DFU bootloader. LED_ST5 should be on and LED_PWR should be off. Windows will then prompt you for the AT90USB162 driver. By default, this is located in the Program Files/Atmel/Flip 3.4.2/usb directory. Point the installer to that directory and install. Once the driver is loaded, the device will appear as the AT90USB162 device under Atmel USB Devices in the device manager.

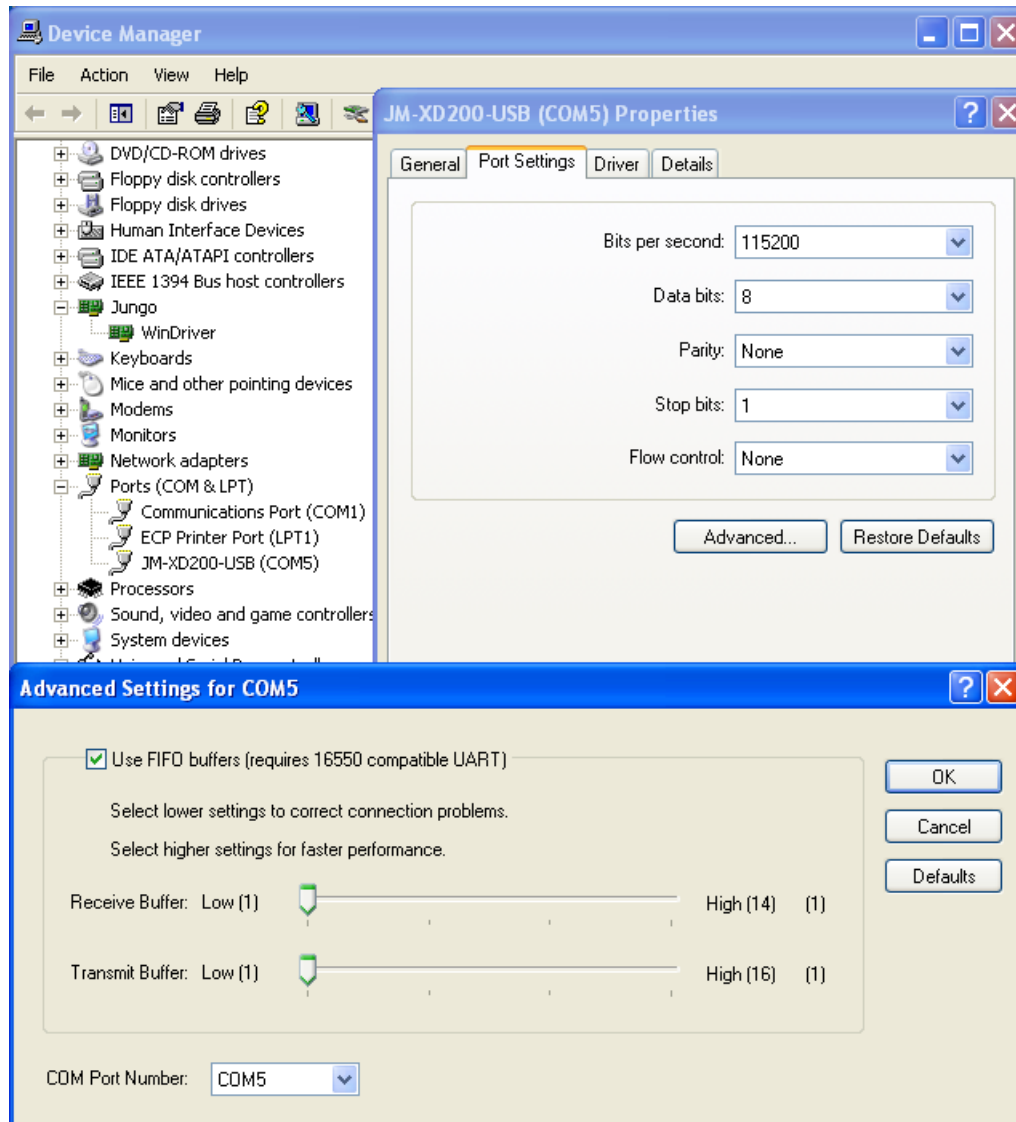


MT-X1 Driver / Serial Configuration

Finally, we will install the MT-X1 CDC driver, which is used by configuration mode and the USB-serial bridge. This driver allows the board to appear as a COM port. The driver itself is included with Windows, but an .inf file is needed to configure it. Download the .inf file from the product page at <http://www.mattairtech.com/>. Now, plug in or reset the MT-X1 with jumper JMP removed. This will run the USB-serial bridge. Both LEDs should be lit. Windows will then prompt you for the MT-X1 CDC driver. Point the installer to the directory where you downloaded the driver and install. Ignore any warnings. Once the driver is loaded, the device will appear as the MT-X1 CDC device using a COM port in the device manager.



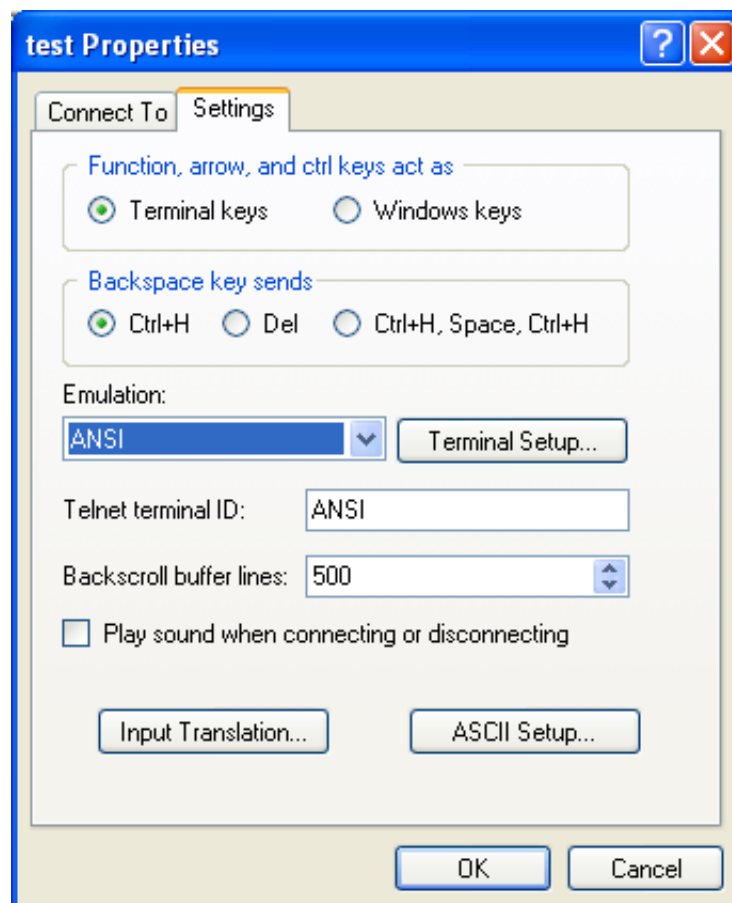
If you wish, double-click on the MT-X1 CDC device entry in the device manager to configure the driver. Click on the Port Settings tab. You can leave the settings at the default values. Note that we are using a virtual COM port and the MT-X1 ignores these settings. The baud rate will always be as fast as possible. Click on Advanced. Here you can adjust the FIFO buffer sizes. If you experience any buffering problems, for example, a delayed response to user input in the configuration, then change both buffer sizes to 1. You may wish to do this now anyway even if you do not currently have any trouble.



Terminal Emulator

Now we are ready to configure the terminal emulator. Windows XP includes HyperTerminal, which has been tested with the MT-X1 and will be documented here. There are several other terminal emulators available freely on the Internet. If you wish to use any of them, it should be no trouble to adapt the instructions presented here. Plug in the MT-X1 with jumper JMP installed. This will run configuration mode which uses the CDC driver. Both LEDs should be on.

Next, start HyperTerminal. Create a new connection. You will refer to this connection again, so give it an appropriate name (after it is configured, you can copy it to your desktop). Select the MT-X1 COM port (ie: COM4) and continue. You can configure the connection as before, but this should not be necessary, as they should be ignored.



After connecting, you may see garbage on the terminal screen. If this is the case, click on the configuration icon and change the emulation to ANSI (or ANSIW). The configuration mode requires an ANSI terminal to allow drawing of the menu system. Normally, when first entering a mode that uses the CDC driver, a message that reads "Press any Key" is printed periodically. If you do not see this message, then too much time has passed between first printing the message and the user connecting the terminal emulator. In this case, the screen may be blank. Just press any key to continue. Note that it may not be possible to switch between modes using the button until a key is pressed.

It is important to always click the disconnect icon before switching to the PDI programmer from configuration mode or the serial bridge. Then click the connect icon a couple seconds after returning. This is required because changing to the AVRISPMkII driver unloads the CDC driver, then loads the AVRISPMkII driver. In order for the terminal to use the same COM port as before, it must be disconnected when returning to the CDC driver so that it does not assign a new COM port. A future firmware release may include the ability to have both drivers loaded simultaneously.

Linux Installation

Linux is supported as well. You must download and build the toolchain from the latest script available at AVR Freaks on the AVR GCC Forum (Script for building AVR GCC sticky at <http://www.avrfreaks.net/index.php?name=PNphpBB2&file=viewtopic&t=42631>). All firmware written for the MT-X1 is developed under Linux using this toolchain.

Drivers

TODO (drivers should already be installed)

GCC Toolchain

TODO (see opening paragraph)

AVRDUDE

TODO (ie: `avrdude -p x128a1 -c avrisp2 -P usb -U flash:w:"myfirmware.hex"`)

dfu-programmer

TODO (must use version 0.5.2 or higher)

Terminal Emulator

TODO (can use minicom, config port (ie: `/dev/tty/ACM0`), save config, run with `minicom -o`)

Configuration

The MT-X1 PDI programmer, serial bridge, and other features can be configured by entering configuration mode. This configuration is stored in non-volatile EEPROM memory. Configuration mode requires an ANSI terminal emulator. Configuration options are highlighted by using the up and down arrow keys, and selected using the enter key. Some dialogs are for entering numbers in hexadecimal. Here, the left arrow key, right arrow key, and backspace can be used. The following lists the structure of the menu system:

- Serial Speed (Serial bridge speed selection)
 - List of selectable speeds: 2400, 9600, 19.2K, 38.4K, 50.0K, 76.8K, 125K, 250K, 500K, 1M, 2M
 - Manual (when selected, configure using Manual Settings below)
- Manual Settings
 - Baud Rate Register (enter value in hex)
 - Clock 2X (async mode only)
- Serial Mode
 - Asynchronous or synchronous
- Sleep Mode (Which sleep mode is used when USB is disconnected or suspended)
 - Power Down or Standby
- Ready Signal (USB ready signal is open-drain active low on XCK pin from USB AVR)
 - Disabled or Enabled
- AVRISPMkII (select which software will be interfacing with the MT-X1 PDI programmer)
 - AVR Studio or AVRDUDE
- Credits (displays list of firmware authors)

The USB AVR automatically enters sleep mode when the USB cable is disconnected or the USB bus is suspended. Sleep mode is by default set to Power Down, which provides for the lowest current consumption. If J7 is connected and the XMEGA is configured to use an external clock, then Standby should be selected. Otherwise, the 8MHz CLKO output from the USB AVR to the XMEGA will be stopped.

The USB ready signal is useful when the XMEGA needs to know when the USB cable is disconnected or the USB bus suspended. The signal is open-drain active-low from the USB AVR XCK line, which may also be used for synchronous serial operation. The XMEGA must enable the pullup on this line before reading it. If it reads low, USB is enumerated and ready. Otherwise, it will read high. If synchronous operation is used, the XCK clock signal will override this. However, when USB is

disconnected or suspended, the clock will stop and the the line driven low.

When changing serial speeds, be sure to also change the speed in the XMEGA.

TODO

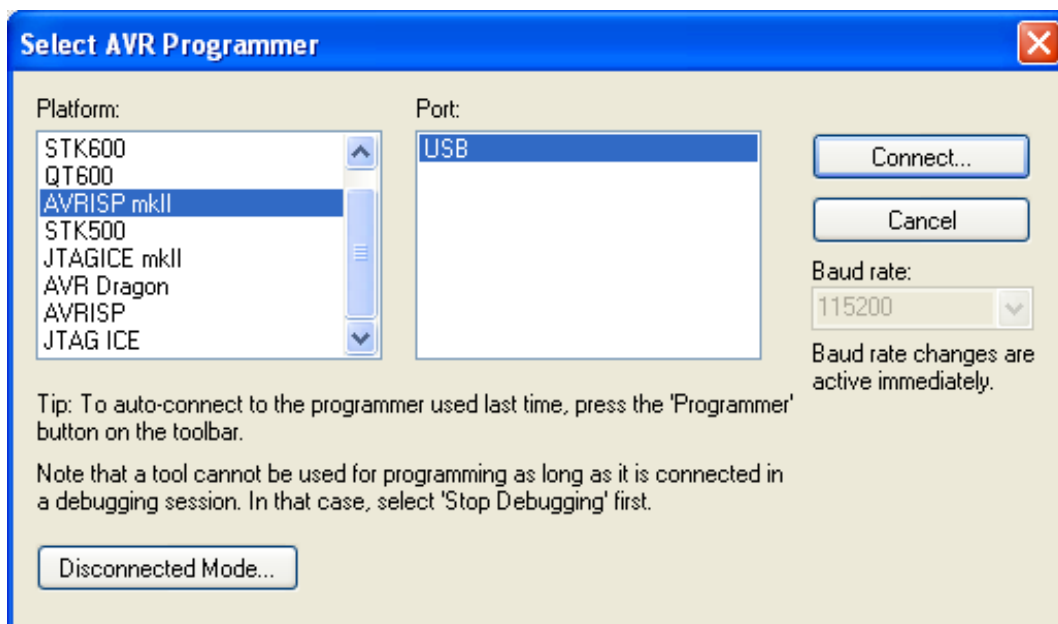
AVRISPmkII PDI Programmer

The MT-X1 PDI Programmer is based on the AVRISPmkII compatible programmer written by Dean Camera (<http://www.fourwalledcubicle.com/>).

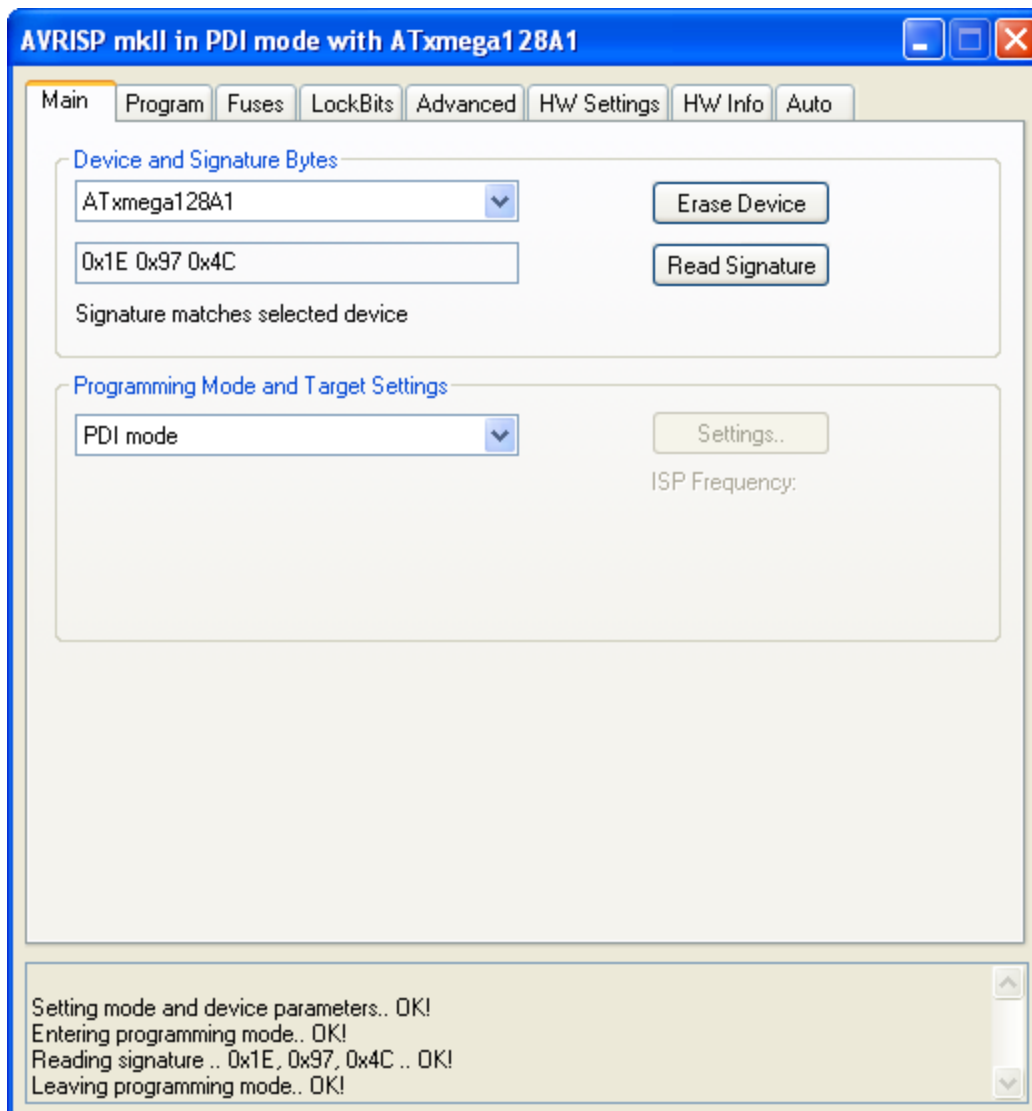
TODO

Using AVR Studio

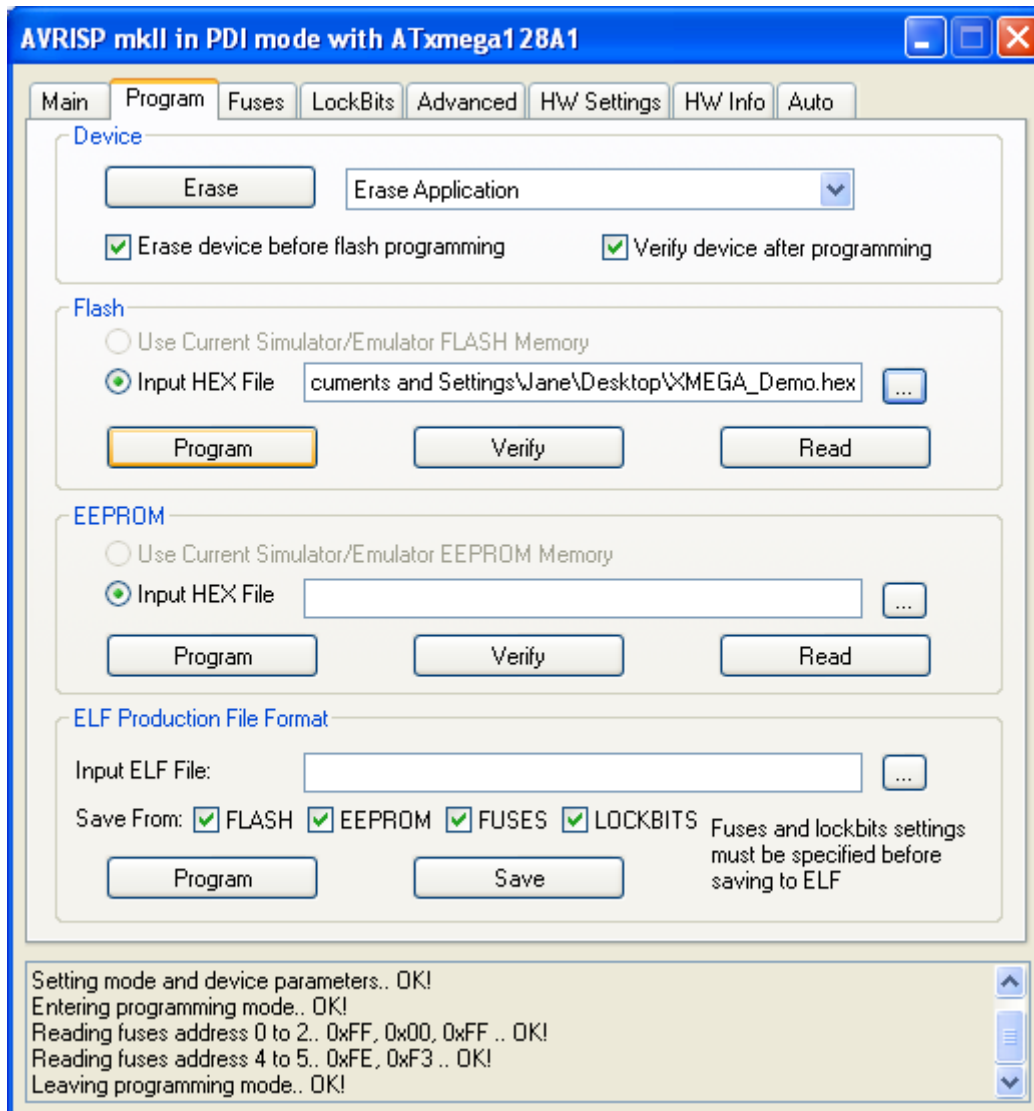
When AVR Studio is run, it will ask you if you want to open or create a new project. For now, you can start a test project. Make it a GCC project. Click Finish. You may also wish to change the project options. Click the gear icon, enter the target clock speed, and select your target device. Now click on the chip icon that reads 'Con'. Select the AVRISPmkII as the programmer and USB as the connection method, then click Connect. If a dialog pops up asking which AVRISPmkII version you are using, select 0000A0012825.



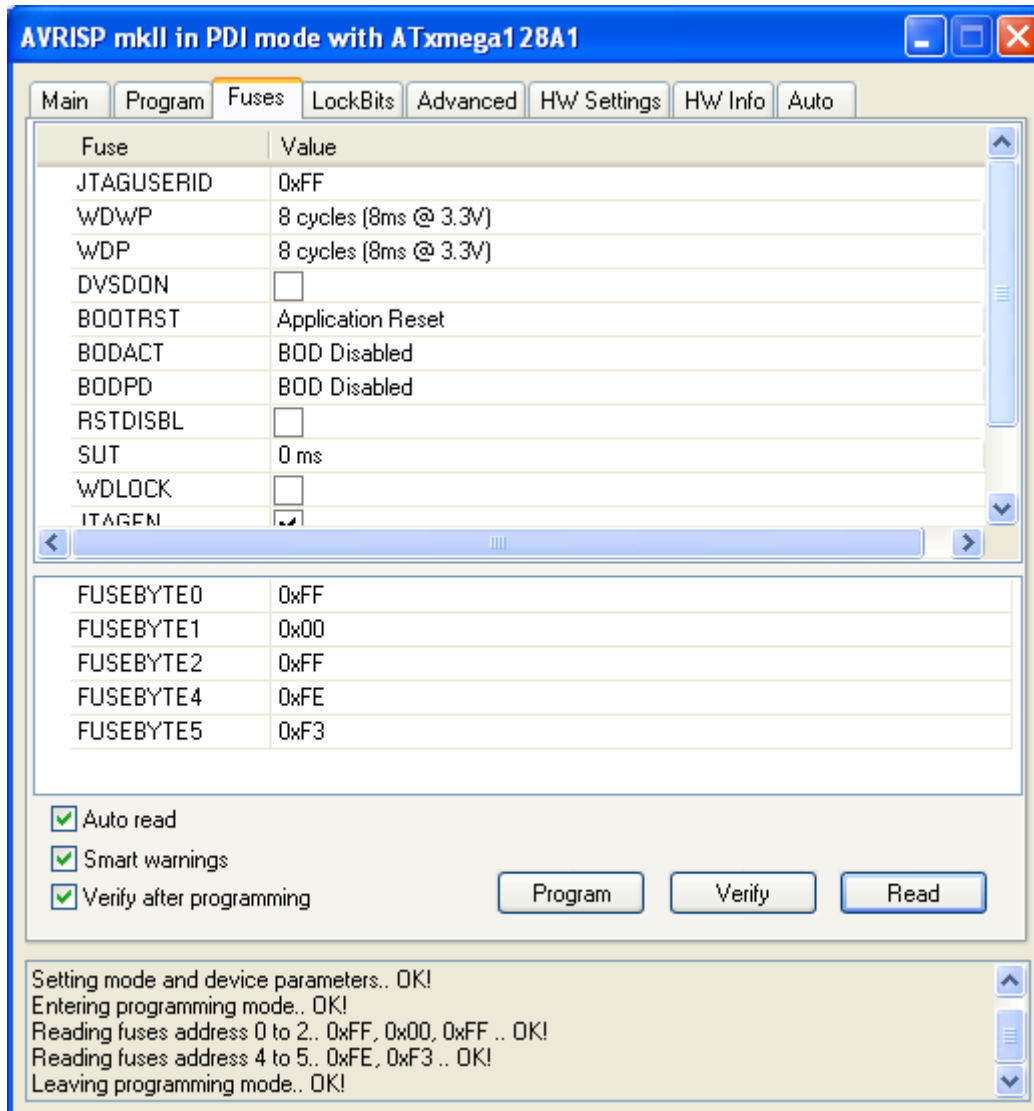
Once connected, a tabbed window will appear. This is the interface to programming the target device. Here, you can load firmware, read/write the EEPROM, modify fuses, change lock bit settings, and more. Click on the first tab. Make sure the correct device is selected, then click Read Signature. It should match the device if all is well. It is recommended to always perform this step first to verify the connection.



Next, click on the Program tab. In the Flash section, a hex file can be programmed into the target's flash memory. Load your hex file, then click Program. You will need to erase the target first if you do not have "Erase device before flash programming" checked. You should also verify the device after programming.



Next, click on the Fuses tab. It is best to leave the fuse settings alone until you understand what they do. In particular, do not set the BOD (Brown-out detection) voltage too close to 3.3V, as this could cause the target to be held perpetually in reset. Due to errata, the BOD should not be enabled in sampled mode when active or idle (BODACT). Sampled mode is OK for other sleep modes (BODPD).



Now you may wish to look at the other tabs. Note that on the HW Settings tab, the reported voltage will always be 3.3V. The Firmware upgrade feature on the same tab should not be used. This is also true for the AVRISPMkII upgrade feature in the AVR Studio tools menu, or any upgrade pop up windows. The MT-X1 PDI Programmer is not an actual AVRISPMkII, it just emulates one, so you should not attempt to update the MT-X1 firmware using AVR Studio. Any firmware updates will be posted to the website and loaded using FLIP or dfu-programmer.

Using AVRDUDE

TODO (ie: avrdude -p x128a1 -c avrisp2 -P usb -U flash:w:"myfirmware.hex")

Serial Bridge

TODO

The Serial Bridge is a USB to serial bridge that allows the XMEGA to communicate with a PC application (like a terminal) over USB. On the PC side, the MT-X1 will appear as a virtual COM port. The MT-X1 simply relays bytes between the PC and XMEGA. On the XMEGA, the link is used like a normal serial port. When in synchronous mode, the USB AVR is the master, so the XCK pin is enabled as an output. The XMEGA must enable its clock pin as an input and be configured as a slave. Note that when configuring the speed to be manual, it is possible to set the speed higher than 2MHz. The maximum speed supported is 2MHz. Also note that RX is prioritized over TX and LED handling. Use the following equations to determine the value to enter into the baud rate register when manually setting the speed.

Baud Rate Register Value (Manual Speed)

Async 1X	Async 2X	Synchronous
$UBRR = \frac{f_{osc}}{16 * BAUD} - 1$	$UBRR = \frac{f_{osc}}{8 * BAUD} - 1$	$UBRR = \frac{f_{osc}}{2 * BAUD} - 1$
$BAUD = \frac{f_{osc}}{16 * (UBRR + 1)}$	$BAUD = \frac{f_{osc}}{8 * (UBRR + 1)}$	$BAUD = \frac{f_{osc}}{2 * (UBRR + 1)}$

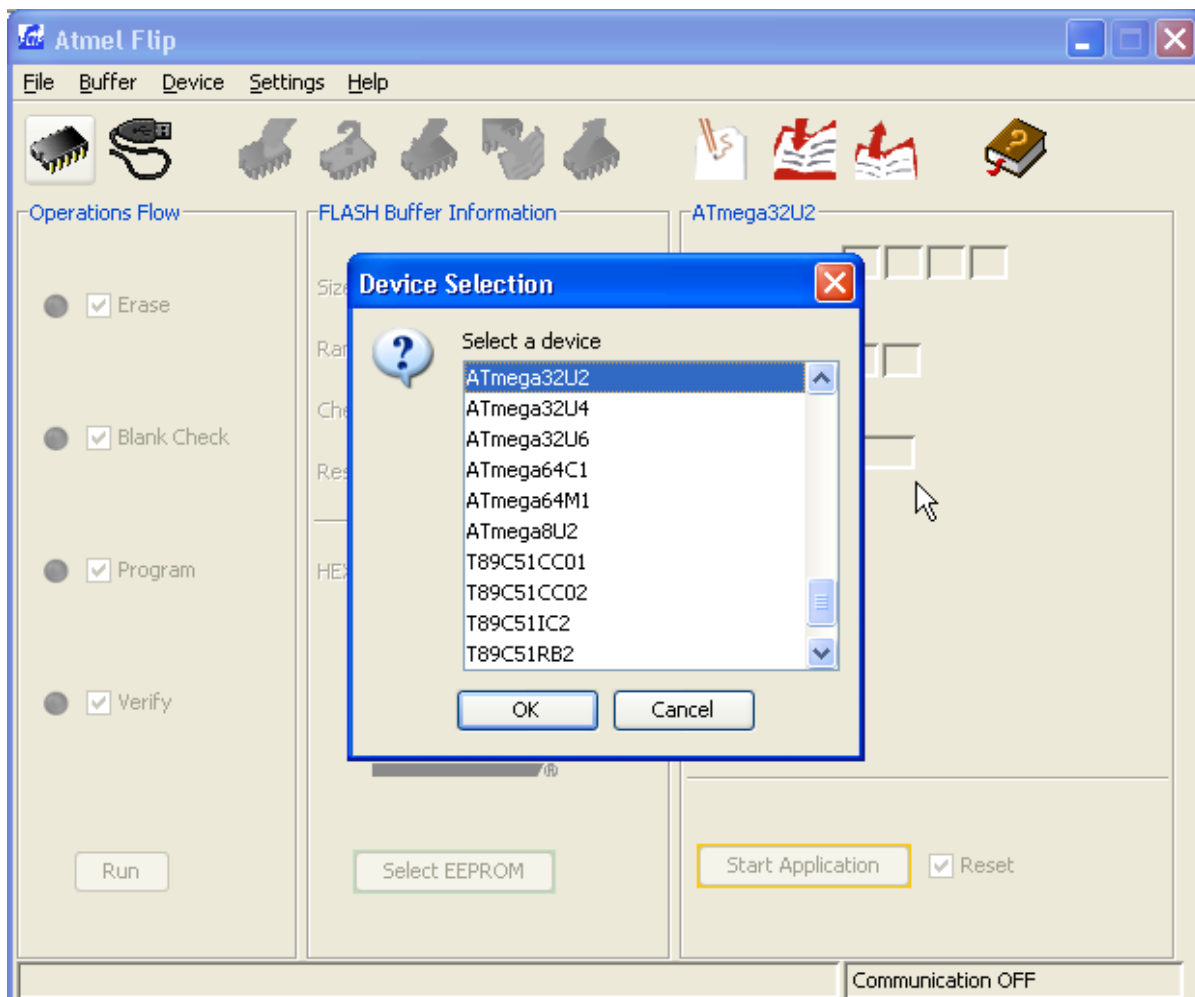
where $f_{osc} = 8000000$

DFU Bootloader

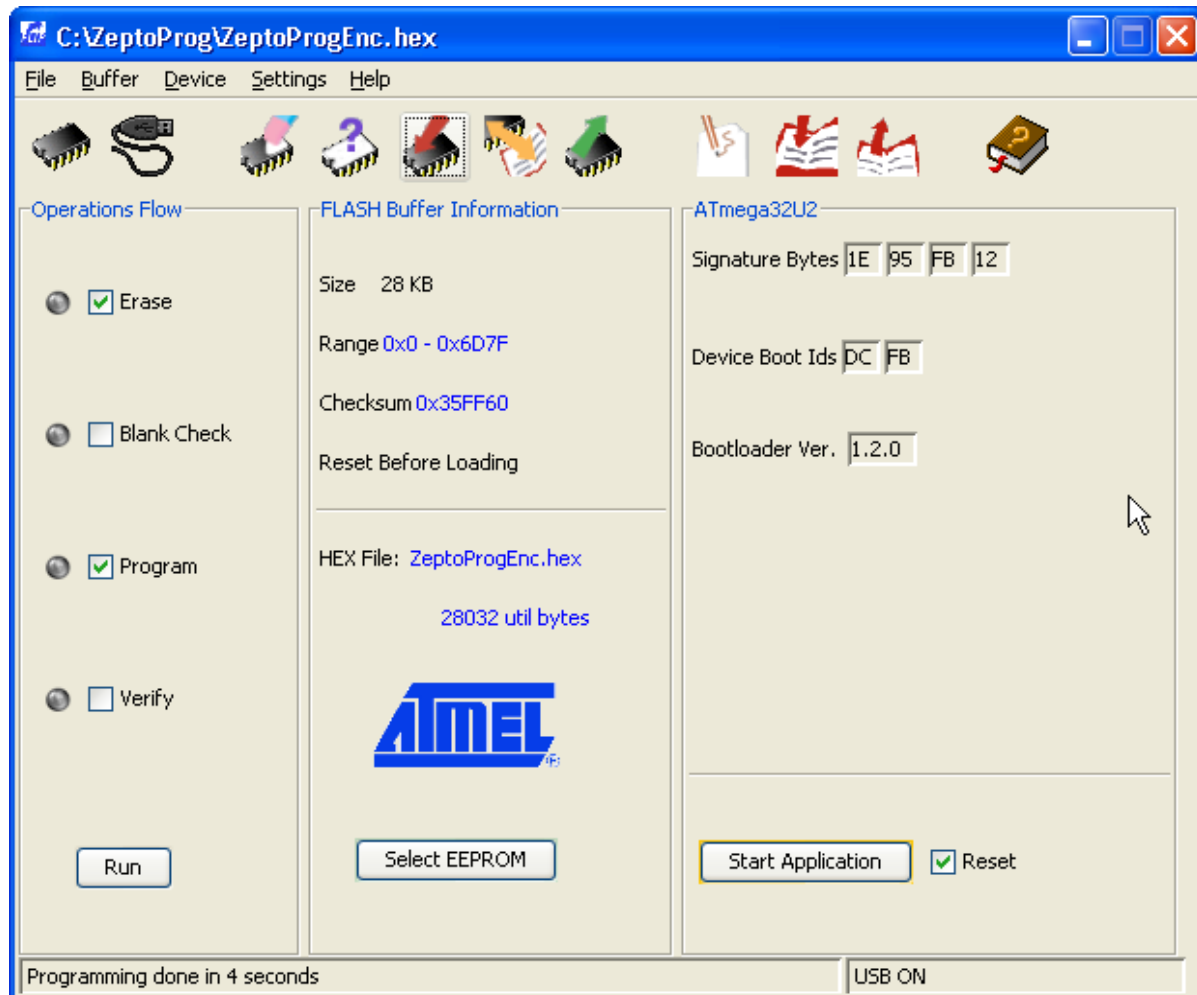
The MT-X1 firmware will be updated periodically to add new features and fix bugs. These updates will be available on the MattairTech website. The updates may include just a hex file (for programming flash), or both a hex file and eep file (for programming both flash and EEPROM).

FLIP

Plug in the MT-X1 with jumper JMP installed and while holding down the PROG button. This will enter the DFU bootloader. LED_STS should be on and LED_PWR should be off. Now launch the FLIP utility. When it has loaded, click on the chip icon and select the AT90USB162.



Next, click on the USB icon, select USB, then connect. The screen should now show information about the AT90USB162. Click on the File menu, and open the appropriate hex file. More information will appear about the program. Be sure that erase is checked. The MT-X1 firmware cannot be loaded unless the flash is erased first. Program must be checked. Verify should also be checked. Now click on the Run button in the lower-left of the screen, and the firmware will be quickly loaded onto the MT-X1. If you encounter problems, then you will need to unplug the MT-X1, disconnect FLIP, and start over making certain that the above settings are observed.



You may also need to program the EEPROM. If so, click on Select EEPROM at the bottom. Then, click on the File menu and open the appropriate eep file. You will have to change the file filter to allow you to see the eep file. Note that eep files are just hex files but with the eep extension instead of hex. More information will appear about the file when selected. Both Program and Verify should be checked. Click run to program the EEPROM.

dfu-programmer

TODO

dfu-programmer at90usb162 erase

dfu-programmer at90usb162 flash-EEPROM MT-X1-110111.eep (if applicable)

dfu-programmer at90usb162 flash MT-X1-110111.hex

XMEGA Demo Program

TODO

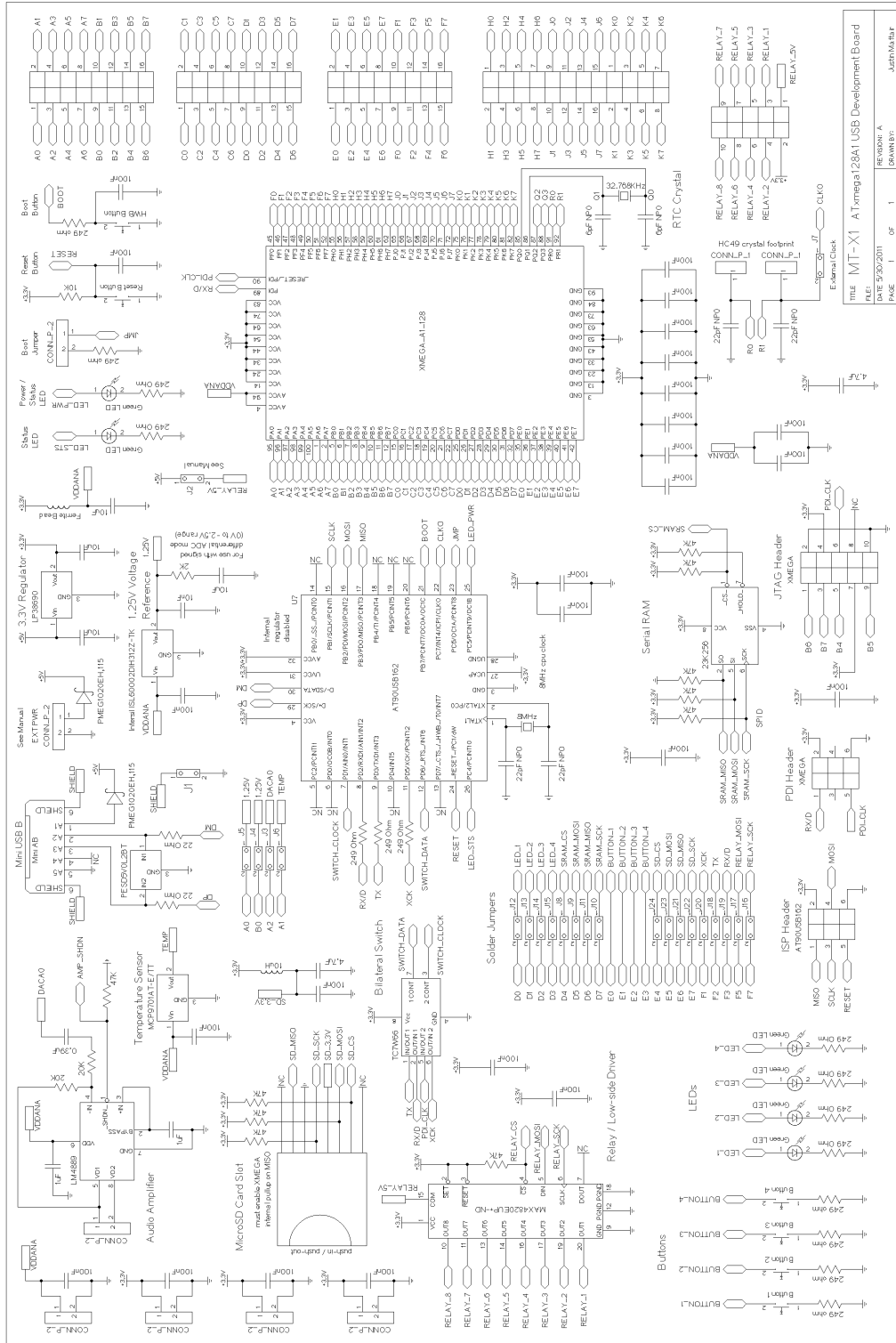
XMEGA uses serial bridge to display menu system on ANSI compatible terminal emulator screen. Remove the jumper and boot the board without pressing the button (remember that the terminal must be disconnected while resetting). Press any key, and the main demo menu will appear. Now, all interaction is with the XMEGA via the USB serial bridge.

MicroSD Card Demo: The SD card demo makes use of the FatFS module from ChaN. FAT12, FAT16, and FAT32 are supported. Press 'h' for a help menu.

Audio Demo: You can load wav files from the SD card and play them over an ~8 ohm speaker connected to the audio amplifier. You will need to use an audio program (like sox), to encode music or sound to 8-bit or 16-bit (recommended), 44.1KHz or less, mono uncompressed PCM. Then, in the audio demo, select the file chooser and pick a file. You can use the slider to adjust volume.

When using the sleep demo, current measurements can be made using the external power header. Disconnect J2 to allow voltages above 5.5V. Supply 6.0V – 7.5V to this header so that current is drawn from this connector rather than from USB. Running the demo will put the XMEGA to sleep with the RTC running and waking the cpu every second to update the time. Most current consumption will then be from the USB AVR. Now unplug USB. The current consumption will drop further. Currently, ~75uA is consumed by the board in this state (without MicroSD card installed). Not all low power features are used, so this number can be lowered further. But note that a minimum load of 100uA should be present on the regulator output. The measured current at the external connector includes the regulator ground current, which will add to the 100uA minimum load.

Schematic



MT-X1 A Xmega128A1 USB Development Board
DATE: 5/30/2011
PAGE: 1 OF 1
REVISION: A
DRAWN BY: JUSTIN MARR

About

Contact Information

Justin Mattair
MattairTech LLC
PO Box 1079
Heppner, OR 97836 USA
541-626-1531
justin@mattair.net
<http://www.mattairtech.com/>

Support Information

Please check the MattairTech website (<http://www.MattairTech.com/>) for firmware updates. Email me if you have any feature requests, suggestions, or if you have found a bug. If you need support, please contact me (email is best). You can also find support information at the MattairTech website. Support for AVRs in general can be found at AVRfreaks (<http://www.avrfreaks.net/>). There, I monitor the forums section as the user physicist.

Precautions

CAUTION
TODO

CAUTION
The MT-X1 contains static sensitive components. Use the usual ESD procedures when handling.

CAUTION
Improper fuse settings may result in an unusable AVR. Be certain that you know the effects of changing the fuses, that you understand the convention used for describing the state of the fuses (programmed = 0), and that you are using an appropriate programming speed before attempting to change fuse settings.

Legal Notices

Copyright Notices

Copyright © 2009-2011, Justin Mattair (<http://www.mattairtech.com/>)
Copyright © 2009-2011, Dean Camera (<http://www.fourwalledcubicle.com/>)
Copyright © 2009, CHaN (http://elm-chan.org/fsw/ff/00index_e.html)
Copyright © 2003-2011, Atmel Corporation (<http://www.atmel.com/>)
Copyright © 2010, Peter Kwan (serial demo)

Software Disclaimer

The author disclaim all warranties with regard to this software, including all implied warranties of merchantability and fitness. In no event shall the author be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

Hardware Disclaimer

The product described in this document is subject to continuous development and improvements. All particulars of the product and its use contained in this document are given by MattairTech LLC in good faith. However all warranties implied or expressed including but not limited to implied warranties of merchantability or fitness for particular purpose are excluded.

This document is intended only to assist the reader in the use of the product. MattairTech LLC shall not be liable for any loss or damage arising from the use of any information in this document or any error or omission in such information or any incorrect use of the product.

Trademarks

AVR® is a registered trademark of Atmel Corporation.
All other trademarks are the property of their respective owners.

Acknowledgments

Thanks to Dean Camera (<http://www.fourwalledcubicle.com/>) for his excellent LUFA library and DFU bootloader, which are used in the MT-X1 firmware. Thanks to the members of AVRfreaks (<http://www.avrfreaks.net/>) for their support. Finally, thanks to Atmel for creating a great product.