

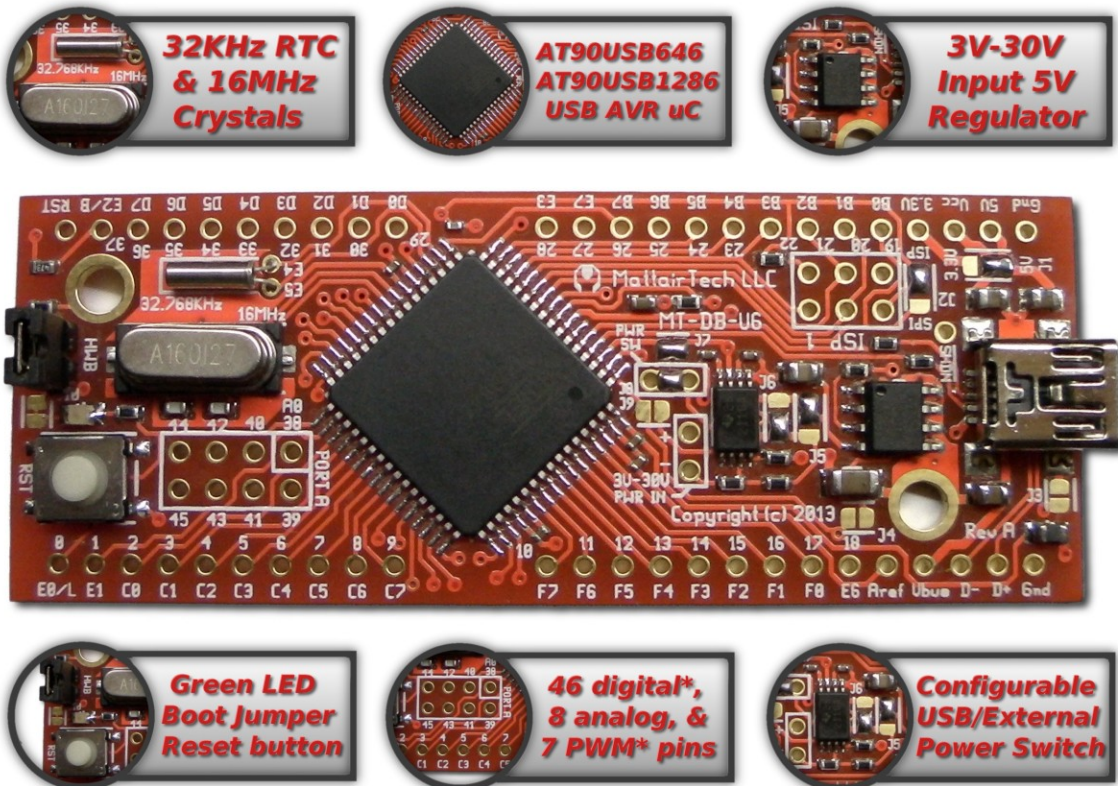
Table of Contents

Overview.....	4
Introduction.....	4
Board Features.....	5
AT90USB646 / AT90USB1286 Features.....	6
MT-DB-U6 Hardware.....	8
Top View / Pinout.....	8
Main Header Pins (Power).....	9
Main Header Pins (Signal).....	10
Inboard Header Pins.....	11
Solder Jumpers.....	12
Onboard 5V, 500mA LDO regulator (3V – 30V input).....	13
Auto Power Source Selection Switch.....	13
32.768KHz RTC crystal / 16MHz main crystal.....	14
HWB Jumper / RESET button / LED.....	14
ISP/SPI Header / JTAG Header.....	15
USB hardware.....	15
Mounting Holes.....	15
Arduino Compatibility (IDE 1.6.7).....	16
What's New.....	16
Summary.....	16
Special Notes.....	17
Pin Configurations.....	17
MT-DB-U6 (AT90USB64/AT90USB128).....	18
MT-DB-U4 (ATmega32U4).....	19
MT-DB-U1/MT-DB-U2 (AT90USB162/ATmega32U2).....	19
Pin Capabilities.....	20
Using Arduino with MattairTech USB boards.....	21
Serial Monitor.....	21
Updated Tone.cpp.....	21
Detailed Memory Usage Output After Compilation.....	22
USB Technical Notes.....	22
Installation.....	23
Driver Installation.....	23
Windows.....	23
Linux.....	24
OS X.....	24
AVR Core Installation.....	24
Uploading the First Sketch.....	25
USB CDC Bootloader (Arduino compatible).....	25
Bootloader Firmware Installation Using the Arduino IDE.....	25
Using AVRDUDE Standalone.....	26

Possible Future Additions.....	26
ChangeLog.....	26
License and credits.....	27
CDC Bootloader (Arduino/AVRDUDE).....	28
DFU Bootloader (FLIP/dfu-programmer).....	31
Using Bitlash (Comes Pre-installed).....	35
Schematic.....	37
Fuse and Lock Settings.....	38
Troubleshooting / FAQ.....	38
Support Information.....	38
Legal.....	39
Appendix A: Precautions.....	40
Appendix B: Other MattairTech Products.....	41

Overview

MT-DB-U6 Arduino Compatible USB AVR Development Board



MT-DB-U6-128: 46 digital*, 8 analog, 7 PWM*, 6 INT*, 8 PCINT, TWI, SPI, USART1, 128KB FLASH, 8KB SRAM, 4KB EEPROM
MT-DB-U6-64: 46 digital*, 8 analog, 7 PWM*, 6 INT*, 8 PCINT, TWI, SPI, USART1, 64KB FLASH, 4KB SRAM, 2KB EEPROM

Introduction

The MT-DB-U6 is a development board for the Atmel AT90USB646 / AT90USB1286 USB AVR microcontroller with Arduino support. Onboard is a 5V, 500mA, low quiescent current, LDO regulator that supports 3V* – 30V DC input voltage (reverse-polarity protected). The board can also be powered via USB (5V Vbus). The power source is automatically selected by the configurable power switch IC which also limits current to 500mA. Also mounted is a mini USB connector, green LED, 16MHz crystal, 32.768KHz RTC crystal, reset button, and an HWB boot jumper. The board has 48 main dual inline header pins with 100 mil pin spacing and 900 mil row spacing which allows for mounting on a perfboard (or barely on a breadboard). There are 8 more pins available on an inboard header (PORT A). The ISP/SPI header can be used with an external programmer, or be reconfigured to allow the MT-DB-U6 itself to be an AVRISP mkII programmer, or to be used as a SPI master or slave. A bootloader comes preinstalled which allows programming of the chip over USB without an external programmer. The PCB measures approximately 2.9" x 1.1" x 0.062" (73mm x 28mm x 1.6mm).

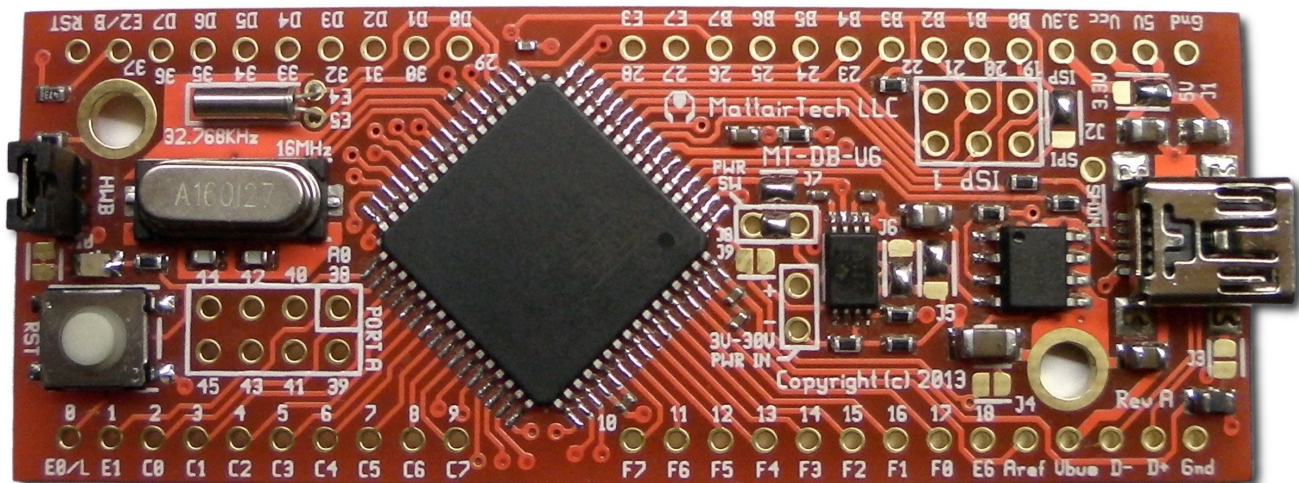
Board Features

- **AT90USB646 / AT90USB1286 USB microcontroller**
 - 64KB/128KB FLASH, 4KB/8KB SRAM, 2KB/4KB EEPROM
 - 8 10-bit ADC channels
 - Serial USART, SPI, and TWI (I2C) communications
 - 4 timers with 14 PWM channels (up to 7 simultaneous)
- **Onboard 5V, 500mA LDO regulator**
 - 3V – 30V DC input (5V regulation with 5.35V – 30V input)
 - low quiescent current (23uA @100uA, 75uA @10mA, 0.7mA @100mA)
 - low dropout (350mV @ 500mA)
 - Reverse polarity protection, reverse leakage protection
 - over-current protection, thermal shutdown, over-voltage shutdown
 - Access to shutdown input (via pin) and low voltage error output (via jumper to E6)
- **Automatic power source selection IC**
 - Seamless automatic switching between USB power and onboard regulator
 - Configurable default source (the source used when both supplies are connected)
 - Current limit set by resistors to 500mA (works better than polyfuse)
 - Low operating current (55uA typical)
 - Inrush current control, thermal protection, reverse and cross-conduction blocking
 - Access to switch status output via jumper to E6 (indicates power source selection)
- **Arduino compatible (now supports IDE 1.6.7 and boards manager)**
- **CDC (Arduino/AVRDUDE) or DFU (FLIP) bootloader preinstalled**
- **Bitlash preinstalled (Arduino command shell)**
- **ISP/SPI header (ISP by default)**
 - As an ISP header, it is used to program the AVR with an external programmer
 - Header can be converted to a SPI (master or slave) header
 - Can be used as an AVRISP mkII programmer to program other AVRs
 - Can mount the MT-SD MicroSD card slot directly to this header
- 16MHz crystal for main clock
- **32.768KHz crystal for real-time counter**
- Green Status LED (can be disconnected)
- Reset button
- Bootloader selection jumper
- Mini USB connector
- USB pins routed to header pins (for panel-mount USB connector)
- Powered by USB or external 3V - 30V power source
- Inductor on analog supply
- All PORT pins routed to headers (E4 and E5 used by RTC crystal)
- 4 main headers are on 0.1" spacing relative to each other (breadboard/perfboard mounting)
- **Two 3mm mounting holes (~5mm pad)**
- High-quality PCB with gold-plated finish
- Measures approx. 2.9" x 1.1" (73mm x 28mm) and 0.062" (1.6mm) thick.

AT90USB646 / AT90USB1286 Features

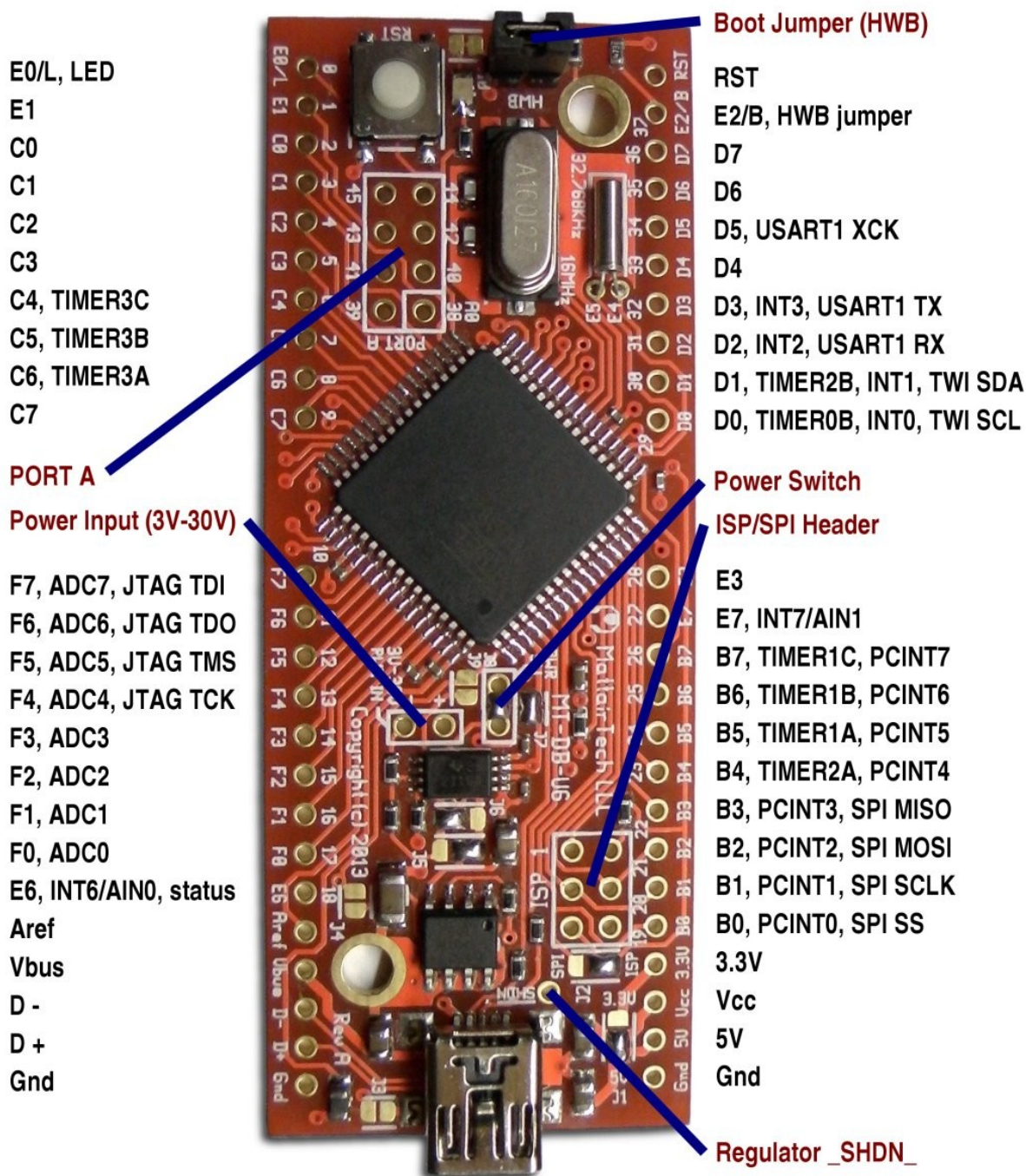
- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
 - 135 Powerful Instructions - Most Single Clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 16 MIPS Throughput at 16 MHz
 - On-Chip 2-cycle Multiplier
- Non-volatile Program and Data Memories
 - 64/128K Bytes of In-System Self-Programmable Flash
- Endurance: 100,000 Write/Erase Cycles
 - Optional Boot Code Section with Independent Lock Bits
- USB Bootloader programmed by default in the Factory
- In-System Programming by On-chip Boot Program hardware activated after reset
- True Read-While-Write Operation
- All supplied parts are preprogrammed with a default USB bootloader
 - 2K/4K (64K/128K Flash version) Bytes EEPROM
- Endurance: 100,000 Write/Erase Cycles
 - 4K/8K (64K/128K Flash version) Bytes Internal SRAM
 - Up to 64K Bytes Optional External Memory Space
 - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 compliant) Interface
 - Boundary-scan Capabilities According to the JTAG Standard
 - Extensive On-chip Debug Support
 - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- USB 2.0 Full-speed/Low-speed Device and On-The-Go Module
 - Complies fully with:
 - Universal Serial Bus Specification REV 2.0
 - On-The-Go Supplement to the USB 2.0 Specification Rev 1.0
 - Supports data transfer rates up to 12 Mbit/s and 1.5 Mbit/s
- USB Full-speed/Low Speed Device Module with Interrupt on Transfer Completion
 - Endpoint 0 for Control Transfers : up to 64-bytes
 - 6 Programmable Endpoints with IN or Out Directions and with Bulk, Interrupt or Isochronous Transfers
 - Configurable Endpoints size up to 256 bytes in double bank mode
 - Fully independent 832 bytes USB DPRAM for endpoint memory allocation
 - Suspend/Resume Interrupts
 - Power-on Reset and USB Bus Reset
 - 48 MHz PLL for Full-speed Bus Operation
 - USB Bus Disconnection on Microcontroller Request
- USB OTG Reduced Host :
 - Supports Host Negotiation Protocol (HNP) and Session Request Protocol (SRP) for OTG dual-role devices
 - Provide Status and control signals for software implementation of HNP and SRP
 - Provides programmable times required for HNP and SRP
- Peripheral Features
 - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
 - Two 16-bit Timer/Counter with Separate Prescaler, Compare- and Capture Mode
 - Real Time Counter with Separate Oscillator
 - Four 8-bit PWM Channels
 - Six PWM Channels with Programmable Resolution from 2 to 16 Bits
 - Output Compare Modulator
 - 8-channels, 10-bit ADC
 - Programmable Serial USART
 - Master/Slave SPI Serial Interface

- Byte Oriented 2-wire Serial Interface
- Programmable Watchdog Timer with Separate On-chip Oscillator
- On-chip Analog Comparator
- Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
 - Power-on Reset and Programmable Brown-out Detection
 - Internal Calibrated Oscillator
 - External and Internal Interrupt Sources
 - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
 - 48 Programmable I/O Lines
 - 64-lead TQFP and 64-lead QFN
- Operating Voltages
 - 2.7 - 5.5V
- Operating temperature
 - Industrial (-40°C to +85°C)
- Maximum Frequency
 - 8 MHz at 2.7V - Industrial range
 - 16 MHz at 4.5V - Industrial range



MT-DB-U6 Hardware

Top View / Pinout



Main Header Pins (Power)

Pin	Description
Gnd (2)	Ground
Vbus	5V output from USB Vbus. This pin is connected to the Vbus pin of the microcontroller. There is a 10uF capacitor on this line, as well as a 200Kohm pulldown resistor. It is also routed to the default power selection jumpers (J5 and J6) to be used as one of the inputs of the power source selection IC.
5V	5V output from the power source selection IC. This will come from either USB Vbus or the onboard regulator. There is a 10uF capacitor on this line. This pin is connected to UVcc of the microcontroller through jumper J7 (connected by default). It is also routed to the Vcc selection jumper (J1), which is set by default to 5V, thus connecting Vcc and 5V.
Vcc	Vcc output from voltage selection jumper (J1). There is a 10uF capacitor on this line. This pin is connected to the Vcc and AVcc (through an inductor) pins on the microcontroller, as well as the ISP header Vcc pin, and the reset and boot jumper (HWB) pullups. Use J1 to connect Vcc to 5V (default setting) when using an onboard power source (USB Vbus or the regulator). The Vcc pin can also be used as an input. Leave J1 unconnected (not to 5V or to 3.3V) and supply power from an external source to the Vcc pin.
3.3V	3.3V output from the microcontrollers internal 3.3V regulator. This pin is connected to Ucap on the microcontroller. There is a 1uF capacitor on this line. This should not be used to power Vcc using jumper J1. However, 3.3V can be supplied externally on the Vcc pin. In this case, you can also set J1 to connect Vcc to 3.3V, disable the microcontrollers internal 3.3V regulator, and disconnect J7.
Aref	Voltage input. This is the reference voltage used by the ADC in the microcontroller. DO NOT connect if using an internal reference.

Main Header Pins (Signal)

Pin	Description
RST	Connects to reset pin of microcontroller as well as the reset button. A 10K pullup resistor and 100nF capacitor are connected to this pin. By default, jumper J2 is set so that RST is connected to the ISP header as well.
E2 / B	This pin is connected to the boot jumper (HWB). The jumper is connected to ground through a 249 ohm resistor. This line has a 47Kohm pullup resistor to Vcc. If the jumper is installed, the bootloader is run, otherwise, the user application is run. This pin can function as a normal GPIO pin at all other times. The 249 ohm resistor provides short-circuit protection in case the pin is used as an output and the jumper is installed.
E0 / L	The green status LED is connected to this pin. The LED is connected to ground through a 249 ohm resistor and jumper J10. Drive the pin high to turn on the LED. Disconnect the LED from the pin by cutting J10.
E6	This pin can be connected to the status output pin (STAT) of the power source selection IC by connecting J9. STAT is an open-drain output that is Hi-Z if the IN2 switch is on (onboard regulator with default J5 and J6 settings). STAT pulls low in the IN1 switch in on (USB with default J5 and J6 settings). Alternatively, pin E6 can be connected to the error flag output pin (FLG) of the onboard regulator by connecting J4. FLG is an open-drain output that drives low when the regulator output is out of regulation due to insufficient input voltage or excessive load.
B0 - B7	Port B. Pins B1 – B3 are also routed to the ISP header. Optionally, B0 can be routed to the ISP header by setting jumper J2 to the SPI position.
C0 - C7	Port C.
D0 - D7	Port D.
E1, E3, E7	Port E.
F0 - F7	Port F. These are generally used as analog pins. JTAG is on pins F4 – F7. Because JTAG is enabled by default, be sure to disable JTAG before using these pins with the ADC or as GPIO. The MattairTech CDC and DFU bootloaders will do this for you.
D-, D+	These are connected to D- and D+ pins of the USB connector. These pins, along with the adjacent Vbus and Ground pins can be used to connect a panel-mount USB connector.

Inboard Header Pins

<i>Pin</i>	<i>Description</i>
E4 / TOSC1	This pin is connected to the on-board 32.768KHz crystal. The crystal must be removed to access the pin.
E5 / TOSC2	This pin is connected to the on-board 32.768KHz crystal. The crystal must be removed to access the pin.
A0 – A7, Port A header	Port A.
ISP Header	This header can be connected to an external ISP programmer to program the microcontroller FLASH, EEPROM, and fuses. Additionally, the header can be used for SPI communications by setting jumper J2 to SPI, which will route pin B0 to the ISP header. Pin B0 can be used as a chip select (or as a slave select if in slave mode). The MT-DB-U6 can itself become an AVRISP mkII compatible programmer with J2 set to SPI and with the appropriate firmware installed (see website).
Regulator Input (+ / -)	Positive and negative (ground) terminals for connection to an external power source or battery. Reverse-polarity protected. Voltage can be from 3V to 30V DC. Note that below about 5.35V, the regulator will fall out of regulation and the low-voltage error output of the regulator will drive low.
Power Switch	This PCB footprint combines a 2-pin header and solder jumper. Shorting this jumper (the default) turns on the output of the power source selection IC (and thus the 5V rail). This can be used as a power switch. When off, total power consumption should be about 25uA – 40uA if using the regulator, or less than 1uA with USB. Note that for the lowest possible power consumption when using the regulator, use the <code>_SHDN_</code> pin.
<code>_SHDN_</code>	This pin is normally pulled to the level of the regulator input through a 200Kohm resistor. Bring this pin low to disable the regulator and reduce power consumption to about 0.1uA. This is useful for battery powered applications that require the lowest possible power consumption.

Solder Jumpers

<i>Pin</i>	<i>Description</i>
J1: Vcc Selection	Use J1 to connect Vcc to 5V (default setting) when using an onboard 5V power source (USB Vbus or the regulator). The Vcc pin can also be used as an input. Leave J1 unconnected (not to 5V or to 3.3V) and supply power from an external source to the Vcc pin. If this external source is 3.3V, you can set J1 to connect Vcc to 3.3V, disable the microcontrollers internal 3.3V regulator, and disconnect J7.
J2: ISP/SPI Selection	The ISP/SPI header can be connected to an external ISP programmer to program the microcontroller FLASH, EEPROM, and fuses (default J2 setting). Additionally, the header can be used for SPI communications by setting jumper J2 to SPI, which will route pin B0 to the ISP header. Pin B0 can be used as a chip select (or as a slave select if in slave mode). The MT-DB-U6 can itself become an AVRISP mkII compatible programmer with J2 set to SPI and with the appropriate firmware installed (see website).
J3: USB Shield Ground	Jumper J3 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, some prefer to have it grounded. An 0603 component may be soldered on the pads.
J4: Regulator Error Output	Pin E6 can be connected to the error flag output pin (FLG) of the onboard regulator by connecting J4. FLG is an open-drain output that drives low when the regulator output is out of regulation due to insufficient input voltage or excessive load.
J5 / J6: Default Power Source	These two jumpers are used together to select the default power source (the source used if both USB and external power are connected). With J6 set to the PORT F side of the PCB and J7 set to the PORT B side (default), the switch IC will select USB power (great for battery-powered applications). With both jumpers in the opposite positions, external power will be used instead (great for data acquisition applications needing both clean power and USB).
J7: UVcc Input	This connects the 5V rail (after the auto switch) to the UVcc pin, which is the input to the AVR internal 3.3V regulator. Disconnect jumper if supplying 3.3V externally or if not using 3.3V (no USB).
J8: Power Switch	This jumper is combined with a 2-pin header footprint. It is soldered by default (power on). Unsolder and install a 2-pin header to use an external power switch (close switch to turn on).
J9: Power selection IC status	Pin E6 can be connected to the status output pin (STAT) of the power switch IC by connecting J9. STAT is an open-drain output that is Hi-Z if the IN2 switch is on (regulator with default J5/J6). STAT pulls low if the IN1 switch is on (USB with default J5/J6).
J10: LED	Cut the trace to disconnect the LED, solder to re-connect

Onboard 5V, 500mA LDO regulator (3V – 30V input)

The 8-pin IC next to the USB connector is the 5V regulator. The input to this regulator is a 2-pin 0.1" header marked PWR IN. The input can be 3V – 30V DC and is reverse-polarity protected. Note that the input should be at least 5.35V to maintain 5V output regulation. The IC can indicate if the voltage is too low on AVR pin E6 (see J4 in table above). The regulator has a low quiescent current (23uA @ 100uA) and is suitable for battery operation. **Note that at higher input voltages, the larger voltage drop will mean higher thermal dissipation for a given amount of current.**

- Micrel MIC5239-5.0
- 5V, 500mA LDO Regulator
- 3V – 30V DC input (5V regulation with 5.35V – 30V input) from 2-pin 0.1" header
- low quiescent current (23uA @100uA, 75uA @10mA, 0.7mA @100mA)
- low dropout (350mV @ 500mA)
- +/- 1% initial output accuracy
- Reverse polarity protection, reverse leakage protection
- over-current protection, thermal shutdown, over-voltage shutdown
- enable input routed to a single-pin header
- J4 can be used to route low voltage error output to microcontroller pin E6
- located near mounting hole, to possibly aid in thermal dissipation
- All capacitors on board are X7R (or NP0 for crystal)
- -40C-125C temperature

Auto Power Source Selection Switch

The small IC between the regulator and the AVR is the power source selection switch. It automatically switches between USB power (5V on Vbus) and external power (from the 5V regulator). If both are connected, the configuration of J5 and J6 determines which source to use (USB by default). See J5 and J6 in the table above for configuration details. The output of this switch can be turned on or off using an external power switch connected to the 2-pin 0.1" header marked PWR SW. Close the switch to turn the output on. By default, J8, which is combined with the header footprint, is soldered, and thus the output is always on. The IC features a current limit which is set to 500mA (suitable for USB). The IC can indicate which power source is selected on AVR pin E6 (see J9 in table above).

- Texas Instruments TPS2112A
- Seamless automatic switching between USB power and onboard regulator
- Configurable default source (the source used when both supplies are connected)
- Current limit set by resistors to 500mA (works better than polyfuse)
- 120mOhm typical switch resistance
- Low operating current (55uA typical)
- Inrush current control, thermal protection, reverse and cross-conduction blocking
- External power switch can be used (see J8 in table above)
- Access to switch status output via jumper to E6 (indicates power source selection)
- TPS2110APWR may be substituted
- -40C-85C temperature

32.768KHz RTC crystal / 16MHz main crystal

The 16MHz crystal is connected to the AVR XTAL1 and XTAL2 pins. The AVR fuses are set by default to use this crystal as the main cpu clock. The crystal is required for full-speed USB operation (low-speed USB can use the internal oscillator). A second 32.768KHz watch crystal is connected to pins E4 (TOSC1) and E5(TOSC2). This can be used as a clock input to the real-time counter. Pins E4 and E5 can be used if the crystal is removed (it is PTH).

HWB Jumper / RESET button / LED

The HWB Jumper is used to select either the bootloader or user application. Note that unless choosing the Atmel bootloader, the hardware HWB function of the AT90USB646 / AT90USB1286 is disabled (HWBE fuse disabled) and the bootloader startup code is always run after reset or power-up (BOOTRST fuse enabled). The bootloader code samples the state of the HWB pin. If the pin is low, the bootloader continues to run, otherwise, the user application is run. The green LED will pulse on and off when the bootloader is running. If the pre-installed demo program is running, it will be lit when USB is connected. Otherwise, the state of the LED is controlled by the user application. If using the Atmel bootloader, the HWBE fuse is enabled and the BOOTRST fuse is disabled. The HWB jumper is read by the mcu hardware after an external reset (not powerup). There is a 47K pullup resistor on the HWB pin (E2/B), which is required by the Atmel bootloader. All of the bootloaders operate at 8 MHz, which is compatible with lower voltages. The user must set the cpu speed to 16MHz in software, if desired and if running at 5V. The HWB jumper has a 249ohm series resistor which will limit current if the pin (E2) is used as an output and drives high with the jumper installed. The reset button has an external 10K pullup resistor.

It is not necessary to remove and replace the jumper when switching between the bootloader and the user application. The jumper can be left on. After FLASH programming, the CDC bootloader will automatically jump to the application. If using a DFU bootloader (including the Atmel bootloader), then you can command FLIP or dfu-programmer to jump to the application. Then, when running the application, the reset button can be pressed to re-enter the bootloader. This is useful when writing and debugging firmware. When the firmware is complete, the jumper can be removed so that future resets will always run the application. If using the CDC bootloader with Arduino, auto-reset is also supported.

ISP/SPI Header / JTAG Header

The ISP/SPI header is configured by default to allow ISP programming using an external programmer (RESET is routed to pin 5). Pin 1 is marked on the board. The ISP/SPI header can be reconfigured so that pin PB0 (SS) is connected to pin 5 rather than RESET. This can be done by switching solder jumper J2, next to the ISP/SPI header, to the “SPI” position. This allows the MT-DB-U6 to be used as a SPI master or slave. The MT-SD MicroSD card can be connected to this header. Additionally, in SPI mode, the board can be used as an AVRISP mkII AVR programmer using Dean Camera's AVRISPMkII software available at <http://www.fourwalledcubicle.com/>. A precompiled hex file will be made available at <http://www.mattairtech.com/> on the MT-DB-U6 product page. Note that when using the ISP/SPI header in this way, Vcc and ground are output to the target board. Therefore, the target board should not be powered itself. You should also verify that it is safe to power the target board through the ISP/SPI connector.

JTAG can be used for programming and debugging. While there is no JTAG header, all JTAG signals are available on the main header pins. Four JTAG signals are shared with ADC pins (F4-F7). JTAG is enabled while running the bootloader, but it is disabled when the user application is run to allow access to the ADC pins (unless using the Atmel bootloader). It can be re-enabled in software.

USB hardware

Onboard is a mini-USB connector, header pins for panel mount USB connector, two 22 ohm series resistors for D+ and D-, and a 200K Vbus pulldown. The pulldown keeps Vbus from floating and thus giving a false Vbus connected signal when Vbus is not present. Jumper J3 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, some prefer to have it grounded on the device side as well, though a ground loop would be formed. An 0603 SMT component may be soldered on the solder jumper pads as well.

Mounting Holes

There are two 3mm mounting holes with ~5mm pads. The mounting hole closest to the regulator can be used to help dissipate heat.

Arduino Compatibility (IDE 1.6.7)

This is a fork of the Arduino AVR core from `arduino/Arduino` (`hardware/arduino/avr/` directory) on GitHub. This will be used to maintain Arduino support for AVR boards including the MattairTech MT-DB-U1, MT-DB-U2, MT-DB-U4, and the MT-DB-U6 (see <https://www.mattairtech.com/>).

This core is intended to be installed using Boards Manager (see below). To update from a previous version, click on MattairTech AVR Boards in Boards Manager, then click Update.

What's New

- Initial release of the 1.6.x compatible AVR core.
- Any combination of CDC, HID, or UART can be used (or no combination), by using the Tools->Communication menu.
- Note that switching between CDC and CDC+HID will require re-selecting the COM port.
- More detailed memory usage at end of compilation (see below).
- Merged in upstream updates.

Summary

Feature	MT-DB-U6	MT-DB-U4	MT-DB-U2	MT-DB-U1
Microcontroller	AT90USB64/AT90US	ATmega32U4, 8-	ATmega32U2, 8-	AT90USB162, 8-
	B128, 8-Bit AVR	Bit AVR	Bit AVR	Bit AVR
Clock Speed	16 MHz	16 MHz	16 MHz	16 MHz
Flash Memory	128 KB			
	(AT90USB128) / 64	32 KB	32 KB	16 KB
	KB (AT90USB64)			
SRAM	8 KB (AT90USB128) /	2.5 KB	1 KB	512 B
	4 KB (AT90USB64)			
EEPROM	4 KB (AT90USB128) /	1 KB	1 KB	512 B
	2 KB (AT90USB64)			
Digital Pins	46*	26	21	21
Analog Input Pins	8 (10-bit)	11* (10-bit)	No analog	No analog
PWM Output Pins	7*	7	4	4
External Interrupts	6* (8 PCINT)*	5 (8 PCINT)*	8 (13 PCINT)*	8 (13 PCINT)*
USB	CDC and HID	CDC and HID	CDC and HID	CDC and HID
UART (Serial)	1	1	1	1
SPI	1	1	1	1
I2C (TWI)	1	1	No I2C	No I2C
Operating Voltage	5V/3.3V	5V/3.3V	5V/3.3V	5V/3.3V

Feature	MT-DB-U6	MT-DB-U4	MT-DB-U2	MT-DB-U1
DC Current per I/O Pin	20 mA	20 mA	20 mA	20 mA

- Only INT pins are supported in this core (PCINT pins are not supported).
- MT-DB-U4: 1 additional analog pin is available by disconnecting the LED (solder jumper on rev B and higher boards)
- MT-DB-U6-64/128: 2 additional digital, 2 additional PWM, or 2 additional INT pins available with RTC crystal removed. Note however, that the RTC crystal holes are smaller and closer together than the header pin holes.

Special Notes

- **Tools->Communications menu**
Currently, the Tools->Communications menu must be used to select the communications configuration. This configuration must match the included libraries. For example, when including the HID and Keyboard libraries, you must select an option that includes HID (ie: CDC_HID_UART). This menu is currently needed to select the USB PID that matches the USB device configuration (needed for Windows). This may become automatic in a future release.
- **Include platform specific libraries**
You may need to manually include platform specific libraries such as SPI.h, Wire.h, and HID.h.
- **EXCEPTION_FOR_57600**
The MattairTech ArduinoCore-avr uses a more accurate baud rate for 57600 than the stock arduino. When using the USART to communicate with another Arduino, define EXCEPTION_FOR_57600.
- **New interrupt mapping**
The MattairTech ArduinoCore-avr has changed interrupt pin mapping from the previous 1.0.5 release. The arduino pin number is now used with attachInterrupt() instead of the interrupt number. See 'Pin Configurations' below.

Pin Configurations

To determine the Arduino pin number, start at the upper-left corner of the board opposite of the USB connector. This is pin 0 (most boards have a 0 printed nearby). The numbering increases in a counter-clockwise direction around the board. Many pins have multiple configurations available. For example, arduino pin 29 (AVR pin D0) on the MT-DB-U6 can be a PWM output (analogWrite), an external interrupt input, digital I/O, or the SCL pin of I2C.

MT-DB-U6 (AT90USB64/AT90USB128)

===== MattairTech MT-DB-U6 (AT90USB64/AT90USB128) =====									
INT/Other	PWM	Analog	Digital		Digital	PWM	INT/Other	Comm	
=====									
LED			0	E0/L	0 0	RST			
			1	E1	HWB	E2/B	37	JUMPER	
			2	C0		D7	36		
			3	C1	0 0	D6	35		
			4	C2	0 0	D5	34		XCK
			5	C3	0 0	D4	33		
6 (TC3C)			6	C4	0 0 *	D3	32	32 (INT3)	TX
7 (TC3B)			7	C5	PORT A	D2	31	31 (INT2)	RX
8 (TC3A)			8	C6		D1	30	30 (TC2B)	SDA
			9	C7		D0	29	29 (TC0B)	SCL
JTAG TDI	10 (ADC7)	10		F7		E3	28		
JTAG TDO	11 (ADC6)	11		F6	PWR SW	E7	27	27 (INT7)	
JTAG TMS	12 (ADC5)	12		F5	0	B7	26	26 (TC1C)	
JTAG TCK	13 (ADC4)	13		F4	- + 0	B6	25	25 (TC1B)	
	14 (ADC3)	14		F3	0 0	B5	24	24 (TC1A)	
	15 (ADC2)	15		F2	PWR IN	B4	23	23 (TC2A)	
	16 (ADC1)	16		F1		B3	22		MISO
	17 (ADC0)	17		F0	* 0 0	B2	21		MOSI
18 (INT6)		18		E6	0 0	B1	20		SCLK
				Aref	0 0	B0	19		SS
				Vbus	ISP	3.3V			
				D-		Vcc			
				D+		5V			
				Gnd	USB	Gnd			

* Pins 38-45 are on the PORT A header. Pins 46 and 47 are the RTC crystal pins E4 and E5 (in use by the RTC by default). With RTC crystal removed, there are 2 additional digital pins (46 and 47), 2 additional PWM pins (TIMER2A on pin 23 and TIMER2B on pin 30), and 2 additional INT pins (INT4 on pin 46 and INT5 on pin 47). All pins can be used with analogRead(). 8 of these pins are actual analog inputs, the rest connect to the internal reference (pin 47) or ground.

MT-DB-U4 (ATmega32U4)

===== MattairTech MT-DB-U4 (ATmega32U4) =====									
INT/Other	PWM	Analog	Digital		Digital	Analog	PWM	INT/other	Comm
=====									
		0 (ADC11)	0	B4	RST				
1 (TC1A)	1 (ADC12)	1	B5	D7/L	25	25 (ADC10)	25 (TC4D)	LED	
2 (TC1B)	2 (ADC13)	2	B6	D6	24	24 (ADC9)			
3 (TC3A)		3	C6	D5	23	23 (REF)			
4 (TC4A)		4	C7	D4	22	22 (ADC8)			
JUMPER			5	E2/B	D3	21		21 (INT3)	TX
				Agnd	D2	20		20 (INT2)	RX
	6 (ADC7)	6	F7	D1	19			19 (INT1)	SDA
	7 (ADC6)	7	F6	D0	18		18 (TC0B)	18 (INT0)	SCL
	8 (ADC5)	8	F5	xtal1					
	9 (ADC4)	9	F4	xtal2					
	10 (ADC1)	10	F1	B7	17		17 (TC1C)		
	11 (ADC0)	11	F0	B3	16				MISO
12 (INT6)	12 (TEMP)	12	E6	B2	15				MOSI
			Aref	B1	14				SCLK
			Avcc	B0	13				SS
			Vbus	3.3V					
			D-	Vcc					
			D+	5V					
			Gnd	USB	Gnd				

* Because of the unusual layout of the ATmega32U4, all pins can be used with analogRead(). 12 of these pins are actual analog inputs (1 used by LED), the rest connect to the internal reference, internal temperature sensor, or ground.

MT-DB-U1/MT-DB-U2 (AT90USB162/ATmega32U2)

===== MattairTech MT-DB-U1/MT-DB-U2 (AT90USB162/ATmega32U2) =====									
Comm	Interrupt	PWM	Digital		Digital	Interrupt	PWM	Comm/other	
=====									
SPI SS			0	B0	RST				
SPI SCLK			1	B1	D7	20	20 (INT7)	JUMPER	
SPI MOSI			2	B2	D6	19	19 (INT6)		
SPI MISO			3	B3	D5	18			
			4	B4	D4	17	17 (INT5)		
			5	B5	D3	16	16 (INT3)	USART1 TX	
			6	B6	D2	15	15 (INT2)	USART1 RX	
	7 (TC1C)	7	B7	D1	14	14 (INT1)			
8 (INT4)		8	C7	D0	13	13 (INT0)	13 (TC0B)	LED	
	9 (TC1A)	9	C6	C2	12				
	10 (TC1B)	10	C5	X1					
		11	C4	X2					
			Vbus	3.3V					
			D-	Vcc					
			D+	5V					
			Gnd	USB	Gnd				

Pin Capabilities

- **Digital: All pins can be used for general purpose I/O**
 - Supports INPUT, OUTPUT, and INPUT_PULLUP.
 - Each pin can source or sink a maximum of 20 mA.
 - Internal pull-up resistors of 20-50 Kohms (disconnected by default).
 - Use the pinMode(), digitalWrite(), and digitalRead() functions.
- **Analog Inputs: 8 pins (MT-DB-U6) or 11 pins (MT-DB-U4) can be configured as ADC analog inputs.**
 - These are available using the analogRead() function.
 - All pins can be used for GPIO and some pins can be used for other digital functions (ie. pwm or serial).
 - Each pin provides 10 bits of resolution (1024 values).
 - Each pin measures from ground to 5.0 volts.
 - The upper end of the measurement range can be changed using the AREF pin and the analogReference() function.
- **PWM: 7 pins (MT-DB-U6, MT-DB-U4) or 4 pins (MT-DB-U2, MT-DB-U1) can be configured as PWM outputs.**
 - Available using the analogWrite() function.
 - Each pin provides 8 bits of resolution (256 values) by default.
- **External Interrupts: Up to 8 pins can be configured with external interrupts.**
 - 6 pins (MT-DB-U6), 5 pins (MT-DB-U4), or 8 pins (MT-DB-U2, MT-DB-U1).
 - Available using the attachInterrupt() function.
- **Serial: 1 pair of pins can be configured for TTL serial I/O.**
 - MT-DB-U6: Serial1: pin 31 (RX) and pin 32 (TX).
 - MT-DB-U4: Serial1: pin 20 (RX) and pin 21 (TX).
 - MT-DB-U2, MT-DB-U1: Serial1: pin 15 (RX) and pin 16 (TX).
- **SPI: 3 or 4 pins can be configured for SPI I/O (SPI).**
 - MT-DB-U6: Pin 21 (MOSI), pin 20 (SCK), pin 22 (MISO), and optionally pin 19 (SS, not currently used).
 - MT-DB-U4: Pin 15 (MOSI), pin 14 (SCK), pin 16 (MISO), and optionally pin 13 (SS, not currently used).
 - MT-DB-U2, MT-DB-U1: Pin 2 (MOSI), pin 1 (SCK), pin 3 (MISO), and optionally pin 0 (SS, not currently used).
 - SPI communication using the SPI library.
- **TWI (I2C): 2 pins can be configured for TWI I/O (Wire).**
 - MT-DB-U6: Pin 30 (SDA) and pin 29 (SCL).
 - MT-DB-U4: Pin 19 (SDA) and pin 18 (SCL).
 - MT-DB-U2, MT-DB-U1: TWI not present
 - TWI communication using the Wire library.
- **LED: One pin can be configured to light the onboard LED (LED_BUILTIN).**
 - Pin 0 (MT-DB-U6), pin 25 (MT-DB-U4), or pin 13 (MT-DB-U2, MT-DB-U1).
 - Bring the pin HIGH to turn the LED on.
- **AREF: One pin can be configured as an AREF analog input.**

- The upper end of the analog measurement range can be changed using the `analogReference()` function.
- **Reset: Bring this line LOW to reset the microcontroller.**

Using Arduino with MattairTech USB boards

Because of the similarities with the Arduino Leonardo, please read <http://arduino.cc/en/Guide/ArduinoLeonardo> first.

Within the Arduino IDE Tools menu, select the appropriate MattairTech board, Frequency/Voltage, Processor, Communications setting, and COM port. There are 2 Frequency/Voltage configurations for each board, 16MHz(5V) and 8MHz(3.3V). You may select 8MHz even if using 5V. When operating at 3.3V, you should select 8MHz. Operating at 16MHz at 3.3V is out of spec, but should work fine at room temperatures. Be sure to select the Communications setting that matches your sketch (by default, this is CDC_ONLY). This is important.

Note that some example sketches indicate the use of pins using the naming convention D2, D3, etc. These are Arduino digital pins, not to be confused with port D pins. Most MattairTech USB AVR boards are printed with both port pin names as well as sequential numbers indicating the Arduino pin number. You may use the 'A' or 'D' prefixes, but they are simply aliased to the arduino pin number (ie: A2 = D2 = 2).

There are several libraries included with Arduino. Some of these need simple changes to work with MattairTech boards. Usually, only pin mappings need to be changed.

Serial Monitor

To print to the Serial Monitor over USB, use 'Serial'. Serial points to SerialUSB (Serial1 is a UART). Unlike most Arduino boards (ie. Uno), USB AVR based boards do not automatically reset when the serial monitor is opened. To see what your sketch outputs to the serial monitor from the beginning, the sketch must wait for the SerialUSB port to open first. Add the following to `setup()`:

```
while (!Serial) ;
```

Remember that if the sketch needs to run without SerialUSB connected, another approach must be used. You can also reset the board manually with the Reset button if you wish to restart your sketch. However, pressing the Reset button will reset the AVR chip, which in turn will reset USB communication. This interruption means that if the serial monitor is open, it will be necessary to close and re-open it to restart communication.

Updated Tone.cpp

Tone.cpp now supports multiple simultaneous tone generation (one tone per timer). The MT-DB-U6 currently supports up to 4 simultaneous tones using timers 3, 1, 2, and 0 if not using the RTC, otherwise, timers 3, 1, and 0 are used for 3 tones. The MT-DB-U4 currently supports up to 3 simultaneous tones using timers 3, 1, and 0. A future release may support a fourth tone from timer 4.

The MT-DB-U2 and MT-DB-U1 support 2 simultaneous tones using timers 1 and 0. Note that timer 0 has a lower accuracy for tone generation because it is 8-bit (timers 3 and 1 are 16-bit). Note also that use of timer 0 temporarily disables the use of `delay()`, which will return to normal operation once the tone stops playing. Thus, timer 0 is set with the lowest priority. For example, if generating DTMF tones on the MT-DB-U4, timers 3 and 1 will be used. However, the MT-DB-U2 and MT-DB-U1 will both use timer 0 for the second tone. If timer 0 is used, `delay()` should not be called while timer 0 is generating a tone. Instead, use `_delay_ms()`, which is included with `avr-libc`.

The DTMF_Demo sketch demonstrates usage of `Tone.cpp` for DTMF generation.

Detailed Memory Usage Output After Compilation

In this release, two programs are run at the end of compilation to provide more detailed memory usage. This is enabled only when verbose messages for compilation is enabled in the IDE Preferences. Just above the normal flash usage message, is the output from the size utility. Above the size utility output is the output from the nm utility. The values on the left are in bytes. The letters stand for: T(t)=.text, D(d)=.data, B(b)=.bss, and everything else (ie: W) resides in flash (in most cases).

USB Technical Notes

- **Note that USB CDC is required for auto-reset into the bootloader to work (otherwise, manually press reset with jumper installed).**

ATmegaxxU4: 832 bytes DPRAM, 1 (control, 64 byte max) + 1 (two banks, 256 byte max) + 5 (two banks, 64 byte max) endpoints
 AT90USBxxx6/7: 832 bytes DPRAM, 1 (control, 64 byte max) + 1 (two banks, 256 byte max) + 5 (two banks, 64 byte max) endpoints

```
// These are used by the core
#define USB_CONTROL_EP_SIZE      16
#define USB_CONTROL_EP_BANKS    1
#define USB_DEFAULT_EP_SIZE     64
#define USB_DEFAULT_EP_BANKS    2
#define USB_CDC_NOTIFICATION_EP_SIZE 16
#define USB_CDC_NOTIFICATION_EP_BANKS 1
#define USB_CDC_DATA_EP_SIZE     64
#define USB_CDC_DATA_EP_BANKS   2

// These can optionally be used by PluggableUSB libraries
#define USB_HID_EP_SIZE          16
#define USB_HID_EP_BANKS        1
#define USB_MIDI_EP_SIZE         64
#define USB_MIDI_EP_BANKS       2
#define USB_MSD_EP_SIZE          64
#define USB_MSD_EP_BANKS        2
```

AT90USBxx2: 176 bytes DPRAM, 8 - 64 byte endpoints, 1 (control) + 2 (one bank) + 2 (two banks) endpoints
 ATmegaxxU2: 176 bytes DPRAM, 8 - 64 byte endpoints, 1 (control) + 2 (one bank) + 2 (two banks) endpoints

```
// These are used by the core
```

```
#define USB_CONTROL_EP_SIZE      16
#define USB_CONTROL_EP_BANKS    1
#define USB_DEFAULT_EP_SIZE     32
#define USB_DEFAULT_EP_BANKS    2
#define USB_CDC_NOTIFICATION_EP_SIZE 16
#define USB_CDC_NOTIFICATION_EP_BANKS 1
#define USB_CDC_DATA_EP_SIZE    32
#define USB_CDC_DATA_EP_BANKS   2

// These can optionally be used by PluggableUSB libraries
#define USB_HID_EP_SIZE          16
#define USB_HID_EP_BANKS        1
#define USB_MIDI_EP_SIZE        32
#define USB_MIDI_EP_BANKS       2
#define USB_MSD_EP_SIZE         32
#define USB_MSD_EP_BANKS        2
```

Installation

Driver Installation

Windows

Prior to core version 1.6.9-mt1, sketches compiled with both CDC and HID USB code by default, thus requiring a CDC driver for the bootloader and a CDC-HID driver for sketches. Now that PluggableUSB is supported, sketches compile with only CDC code by default. Thus, only one driver is needed. Since HID and MIDI are currently supported (and MSD potentially in the future), driver installation will be required for each different combination of USB devices. There are currently four USB composite device combinations that include CDC as well as a CDC only device. Each supported combination has a unique USB VID:PID pair, and these are listed in the .inf file. Once the first device is installed (the CDC only device), future installations *might* be automatic, otherwise, you may direct the installer to the same .inf file. The drivers are signed and support both 32 and 64 bit versions of Windows XP(SP3), Vista, 7, 8, and 10.

1. If you do not already have the CDC bootloader installed, see below.
2. Download https://www.mattairtech.com/software/MattairTech_CDC_Driver_Signed.zip and unzip into any folder.
3. Plug in the board with the jumper installed. The LED should light.
4. Windows will detect the board. Point the installer to the folder from above to install the bootloader driver.
5. If you don't intend on using Arduino, you can skip the rest of this list. See Using AVRDUDE Standalone below.
6. If you do not already have the test firmware installed, see Using AVRDUDE Standalone below.
7. Press the reset button to run the test firmware (blink sketch).
8. Windows will detect the board. Point the installer to the above folder to install the sketch driver (if needed).

9. Continue with AVR Core Installation below.

Linux

1. No driver installation is needed.
2. On some distros, you may need to add your user to the same group as the port (ie: dialout) and/or set udev rules.
3. You MAY have to install and use Arduino as the root user in order to get reliable access to the serial port.
 - This is true even when group permissions are set correctly, and it may fail after previously working.
 - You can also create/modify a udev rule to set permissions on the port so *everyone* can read / write.
4. Continue with AVR Core Installation below.

OS X

UNTESTED

1. No driver installation is needed.
2. Plug in the board. You may get a dialog box asking if you wish to open the "Network Preferences":
 - Click the "Network Preferences..." button, then click "Apply".
 - The board will show up as "Not Configured", but it will work fine.
3. Continue with AVR Core Installation below.

AVR Core Installation

- To update from a previous version, click on MattairTech AVR Boards in Boards Manager, then click Update.
1. The MattairTech AVR Core requires Arduino 1.6.7+.
 2. In the Arduino IDE 1.6.7+, click File->Preferences.
 3. Click the button next to Additional Boards Manager URLs.
 4. Add https://www.mattairtech.com/software/arduino/package_MattairTech_index.json.
 5. Save preferences, then open the Boards Manager.
 6. Install the MattairTech AVR Boards package.
 7. Close Boards Manager, then select your board from Tools->Board.
 8. Select the Frequency/Voltage with the now visible Tools->Frequency/Voltage menu.
 9. Select the processor with the now visible Tools->Processor menu.
 10. Select the communications option with the now visible Tools->Communications menu (must match sketch).
 11. If you do not already have the bootloader or blink sketch installed, see USB CDC Bootloader

below.

12. Plug in the board. The blink sketch should be running.
13. Click Tools->Port and choose the COM port.
14. You can now upload your own sketch.

Uploading the First Sketch

1. In the Arduino IDE 1.6.7 (or above), open File->Examples->01.Basics->Blink.
2. Change the three instances of '13' to 'LED_BUILTIN'.
3. Be sure the correct options are selected in the Tools menu (see AVR Core Installation above).
4. With the board plugged in, select the correct port from Tools->Port.
5. Click the Upload button. After compiling, the sketch should be transferred to the board.
6. Once the bootloader exits, the blink sketch should be running.

USB CDC Bootloader (Arduino compatible)

Each board has several bootloaders available. The CDC bootloader can be used with Arduino. Version 130410 or above is required to support the auto-reset feature. Note that several boards that were shipped after 130410 but before 130626 still have the old bootloader.

The bootloader enters programming mode only if the jumper is installed, except when using Arduino auto-reset or when the FLASH is empty. Even with the jumper installed, programming mode will NOT be entered if the reset was from the watchdog timer, unless the boot key is enabled and the key matches, as is the case with Arduino auto-reset (the Arduino core uses a watchdog reset to enter the bootloader).

The default CDC bootloader has the following compile-time options defined:

```
#define ENABLE_LED_BOOT
#define ENABLE_LED_APPLICATION
#define DISABLE_JTAG_APPLICATION
#define ENABLE_CLKDIV_1_APPLICATION
#define ENABLE_BOOT_KEY
#define ENABLE_RESET_AFTER_PROGRAMMING
#define NO_LOCK_BYTE_WRITE_SUPPORT
```

An alternate version with the above options undefined is available on the website named Bootloader_no_options.hex. Use it if the default options interfere with your application. For example, you may disconnect the LED and use the pin as an analog input.

Bootloader Firmware Installation Using the Arduino IDE

1. If you do not already have the MattairTech AVR core installed, see AVR Core Installation above.
2. Plug a compatible programmer into a USB port, then connect it to the powered AVR board.
3. Select your programmer from Tools->Programmer.
4. Select your board from Tools->Board.

5. Click Tools->Burn Bootloader. Ignore any messages about not supporting shutdown or reset.
6. Continue with driver installation above.

Using AVRDUDE Standalone

AVRDUDE can be used standalone. You can use the version included with Arduino (in arduino-1.6.7/hardware/tools/avr/bin) or download a separate version from <http://download.savannah.gnu.org/releases/avrdude/>.

As an example, AVRDUDE will be used to upload the test firmware (blink sketch):

1. Download firmware from <https://www.mattairtech.com/software/CDC-bootloader-test-firmware.zip> and unzip.
2. If you have not already installed the bootloader driver, see Driver Installation above.
3. Be sure there is a hex file that matches your chip. On the command line (change the hex file to match yours):

```
avrdude -p m32u4 -c avr109 -P usb -U flash:w:"blink.hex"
```

1. On linux, the -P option should be something like /dev/ttyACM0.
2. See http://www.nongnu.org/avrdude/user-manual/avrdude_4.html for details.
3. Press the reset button with the jumper off to load the sketch.
4. When using AVRDUDE standalone, the jumper must be installed before pressing reset to run the bootloader.

Possible Future Additions

- Features for lower power consumption
- MSC (Mass Storage) USB Device Class
- Host mode CDC
- Better OS X support
- PCINT support

ChangeLog

- 1.6.9-mt1:
 - See 'What's New' above.
- 1.0.5.1 fixes the sketch not running when not connected to a USB host (ie: USB charger). Version 1.0.5 fixes several bugs (including BSoD's on Win7-64) and updates the Arduino core files and libraries to 1.0.5. Merged in changes to Arduino 1.0.5 core and examples. Changed a few //s to #s in boards.txt. Fixed blank spaces in board selection list. Eliminate descriptor serial numbers. Use new PID. Fixed Win7-64 BSoD's and code 10's. New inf file to support new PID. Initialize USB (HID and CDC) without needing Serial.begin(). Made USB_WAITFORCONNECT_DISABLED default instead of

USB_WAITFORCONNECT_ENABLED. Added two nop()'s to USBSerial::readRXEndpoint() so switching USB endpoint banks does not result in returning -1 (empty). USBSerial::peek() fixed. Wait for USB_DeviceState_Connected state before continuing. change keyboardmouse demo to use different pins.

- 1.0.4 adds HID keyboard and mouse support, adds auto-reset support, updates LUFA to 130303, updates the Arduino core files and libraries to 1.0.4, updates the bootloaders, and adds support for the new MT-DB-U6.

License and credits

This core has been developed by Arduino LLC. This fork developed by Justin Mattair of MattairTech LLC.

Copyright (c) 2015 Arduino LLC. All right reserved.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

The Bitlash files are Copyright (C) 2008-2012 Bill Roy (bitlash.net)

CDC Bootloader (Arduino/AVRDUDE)

CDC Serial Driver

The CDC Serial 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 https://www.mattairtech.com/software/MattairTech_CDC_Driver_Signed.zip. Note that Windows Vista 64-bit, Windows 7 64-bit and Windows 8 require the signed driver. You may need to rename the file so that it has the inf extension. Next, plug in the board with the jumper removed. Windows will then prompt you for the MattairTech CDC Serial driver. Point the installer to the directory where you downloaded the driver and install, ignoring any warnings. Once the driver is loaded, the device will appear as the MattairTech CDC Serial device using a COM port in the device manager.

If you wish, double-click on the CDC Serial device entry in the device manager to configure the driver. Nothing on the port settings tab needs to be changed. We are using a virtual COM port so the settings are ignored. The baud rate will always be as fast as possible. On the advanced tab, you can adjust the FIFO buffer sizes. If you experience any buffering problems (ie: a delayed response to user input), then change both buffer sizes to 1.

CDC Bootloader

The CDC bootloader uses the AVR109 protocol, and can be used withing the Arduino environment, or directly with AVRDUDE. Version 130410 or above is required to support the auto-reset feature. If using a terminal emulator, you must first disconnect before running the bootloader. The bootloader enters programming mode only if the jumper is installed, even when using Arduino auto-reset. The one exception is when the FLASH is empty. Even with the jumper installed, programming mode will NOT be entered if the reset was from the watchdog timer. The one exception to this is when the boot key is enabled and the key matches. The key will match when the Arduino IDE auto-resets the board to enter bootloader programming mode. The key is needed because the Arduino core part of the firmware, which listens for the IDE auto-reset signal, uses a watchdog reset to enter the bootloader. This way, the user application can make use of the watchdog timer. The bootloader will jump to the user application at the end of FLASH programming. Other operations with AVRDUDE, like writing the EEPROM, will not trigger this. Just press reset to get back to the bootloader (as long as the jumper is installed).

May 2, 2014 UPDATE:

Version 140502:

- Added #define ARDUINO_MODE to AppConfig.h. **This eliminates the requirement for the jumper to be installed.** Arduino users should now always leave the jumper off. If you cannot enter the bootloader (ie: sketch compiled with USB_AUTORESET_DISABLED), you can force the bootloader by installing the jumper (be sure to re-select the COM port).

- Removed `#define ENABLE_CLKDIV_1_APPLICATION` from `AppConfig.h`. Now, the bootloader always runs at the crystal speed (16MHz). See next entry.
- Fixed problem on Linux systems where the LED would sometimes freeze and the USB connection would fail. This was due to the bootloader running at 8MHz. Now it always runs at 16MHz. Note that when operating at 3.3V, the cpu will be overclocked, but it should work fine.
- Fixed problem where AVRDUDE would sometimes freeze at the end of programming. This was due to the bootloader exiting before the last ACK was sent to AVRDUDE. This may have affected other host software as well.
- Increased the time between disconnecting from the USB host and switching to the application.
- Updated LUFA library to version 140302.

Each board has several bootloaders available. The CDC bootloader can be used with Arduino. Version 130410 or above is required to support the auto-reset feature. Note that several boards that were shipped after 130410 but before 130626 still have the old bootloader. It is strongly recommended to use version 140502 or higher when using with Arduino.

The default CDC bootloader has the following compile-time options defined:

```
#define NO_LOCK_BYTE_WRITE_SUPPORT
#define ENABLE_LED_BOOT
#define ENABLE_LED_APPLICATION
#define DISABLE_JTAG_APPLICATION
#define ENABLE_BOOT_KEY
#define ENABLE_RESET_AFTER_PROGRAMMING
#define ARDUINO_MODE
```

An alternate version with the above options undefined is available on the website named `Bootloader_no_options.hex`. Use it if the default options interfere with your application. For example, you may disconnect the LED and use the pin as an analog input.

When using the auto-reset feature of Arduino, the `boards.txt` file must currently list the bootloader directory as `caterina` (the bootloader used on the Leonardo). The actual bootloader is a modified version of the LUFA CDC bootloader by Dean Camera (lufa-lib.org). It resides

in the mtdbxx folder (where xx corresponds to the board you have). So, if you wish to use the Arduino IDE to burn the bootloader, you must temporarily change the appropriate entry in the boards.txt file to point toward the actual bootloader directory. Change it back to caterina when finished to re-enable auto-reset.

Example for Windows:

```
avrdude -p usb1286 -c avr109 -P COM5 -U flash:w:"bitlashdemo_MT-DB-U6.hex"
```

Example for Linux:

```
avrdude -p usb646 -c avr109 -P /dev/ttyACM0 -U flash:w:"bitlashdemo_MT-DB-U6.hex"
```

Arduino environment:

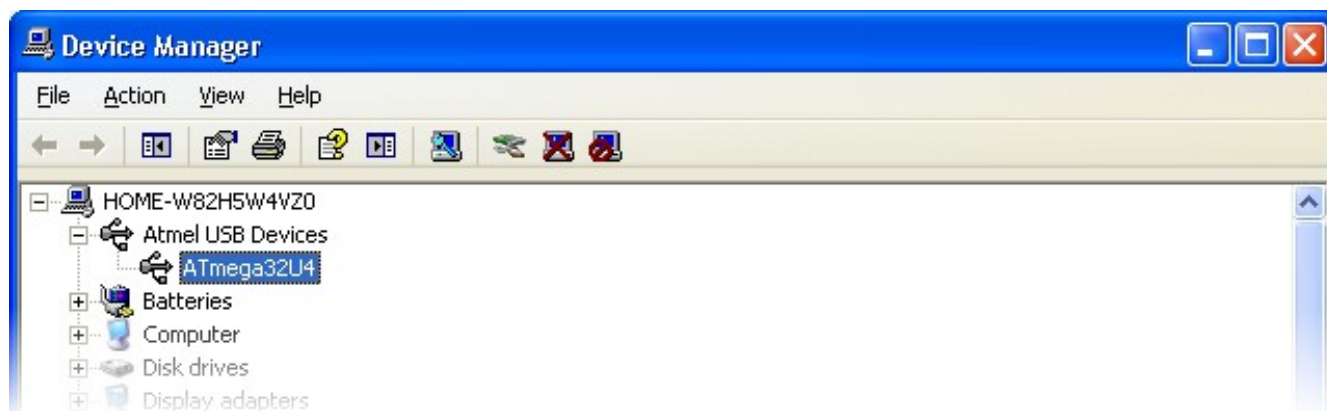
Be sure to select the COM port. Then upload your sketch with the Upload button.

DFU Bootloader (FLIP/dfu-programmer)

Installation

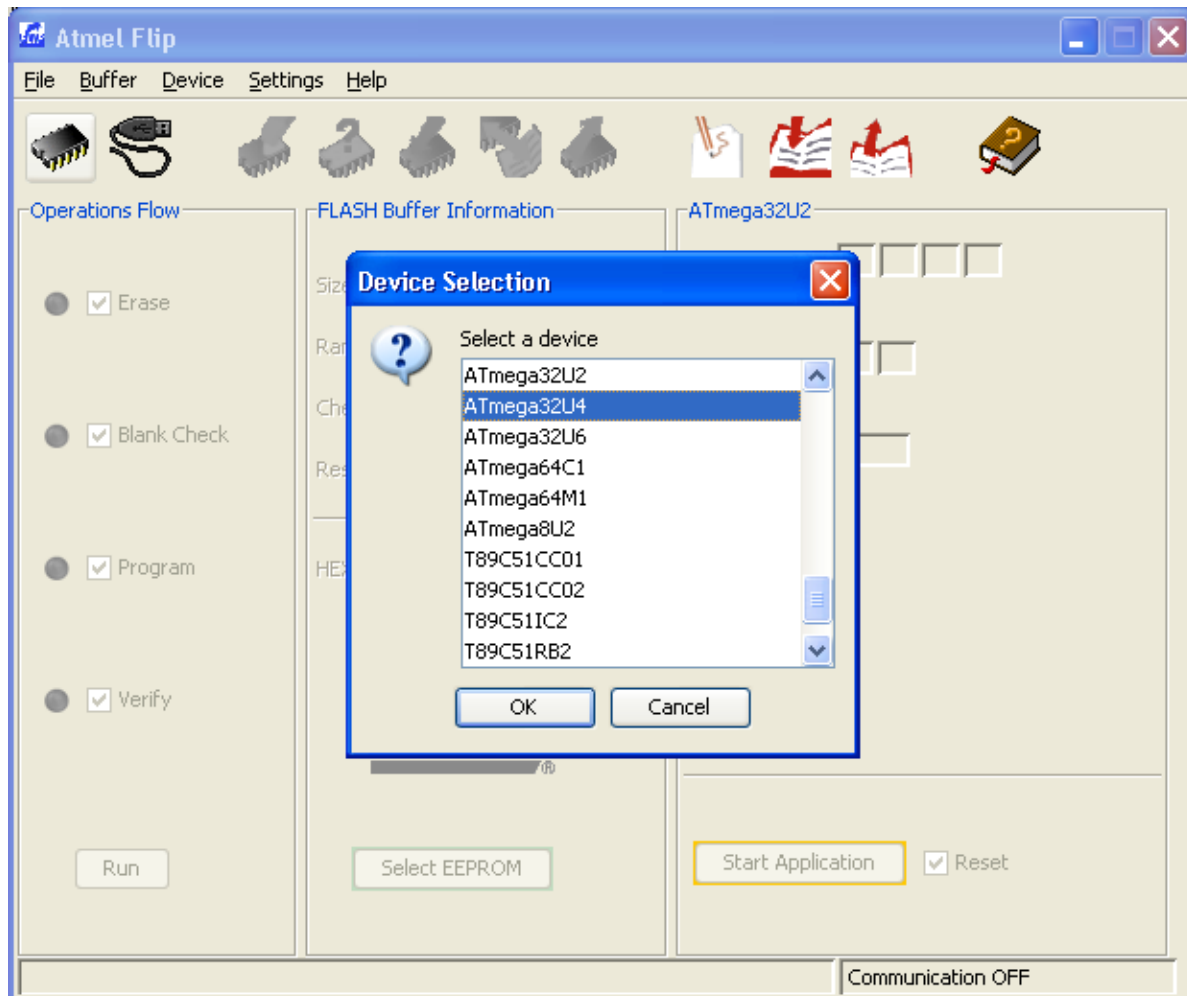
FLIP is a graphical utility used to load firmware into the AVR. FLIP includes the DFU bootloader driver. It supports Windows XP through Windows 7 (32 or 64 bit). Download FLIP 3.4.2 or higher from http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3886 and install.

Once FLIP is installed, the DFU bootloader drivers can be loaded. Install the HWB jumper and power-up the board (or press reset). This will enter the DFU bootloader. The LED should be pulsing. Windows will then prompt you for the 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 under Atmel USB Devices in the device manager (ATmega32U4 shown).

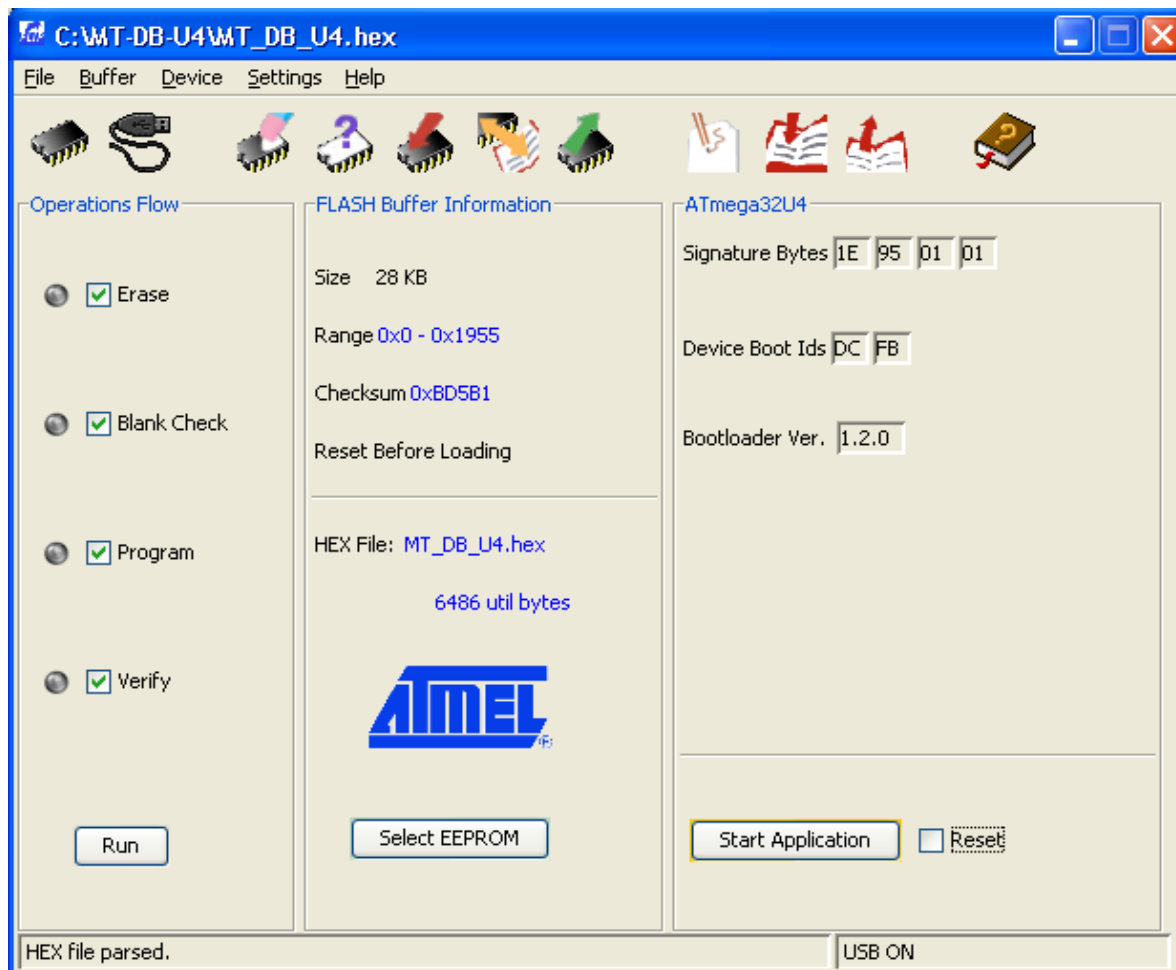


FLIP

Install the HWB jumper and power-up the board (or press reset). This will enter the DFU bootloader. The LED should be pulsing. Now launch the FLIP utility. When it has loaded, click on the chip icon and select the AT90USB646 / AT90USB1286 (ATmega32U4 shown).



Next, click on the USB icon, select USB, then connect. The screen should now show information about the AT90USB646 / AT90USB1286. 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 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 AT90USB646 / AT90USB1286. The screenshot shows details for the ATmega32U4.



You may also 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.

You can run your application without removing the jumper or pressing reset by unchecking the reset box and pressing the "Start Application" button (lower right).

dfu-programmer

dfu-programmer is a command line utility used to program the AT90USB646 / AT90USB1286 that runs under Linux. A DFU driver installation is not required. Download version 0.5.4 or higher from <http://dfu-programmer.sourceforge.net/> . The following commands can be used:

```
dfu-programmer at90usb646 / at90usb1286 erase
```

```
dfu-programmer at90usb646 / at90usb1286 flash-EEPROM YourHex.eep (if applicable)
```

```
dfu-programmer at90usb646 / at90usb1286 flash YourHex.hex
```

```
dfu-programmer at90usb646 / at90usb1286 start (to jump to application section  
without reset)
```

Using Bitlash (Comes Pre-installed)

Bitlash

Bitlash is an open source interpreted language shell and embedded programming environment. The preinstalled Bitlash demo was compiled in the Arduino environment and supports Arduino functions (ie: dw() for digitalWrite()). A terminal emulator (recommended) or the Arduino serial monitor may be used. See the CDC Bootloader section for details on installing the CDC Serial driver.

- * Up to 46 digital input/output (w/ optional pullups)
- * Up to 8 10-bit analog inputs (0V-5V)
- * Up to 7 8-bit PWM outputs (can be used for analog out)
- * Up to 8 external pin interrupts
- * Up to 4 simultaneous frequency outputs
- * Pulse width measurement

The following example demonstrates data acquisition and control capabilities (see comments, which should not be typed in):

```
bitlash here! v2.0RC4 (c)2011 Bill Roy, bitlash.net -type HELP- 1706 bytes free
> pinmode(13,1)      // set pin 13 to be a digital output
> d13=1              // set pin 13 to high
> pinmode(12,0)      // set pin 12 to be a digital input (each pin is by
default already set this way)
> d12=1              // enable internal pullup resistor for pin 12
> x=d12              // read pin 12 and store result in x
> print x            // display value of x
1
> print ar(10)        // read analog input 10, returns a value between 0 and
1023 (0V - 5V), print to display (or you can assign to variable)
784
> pinmode(6,1)       // set pin 6 to be a digital output
> a6=127              // set pin 6 to output a 50% duty cycle PWM (can be used
for analog output), value between 0 and 255
> if (d12==0) {a6=(ar(10)/4);} // set PWM output 6 to be proportional to
analog input 10
```

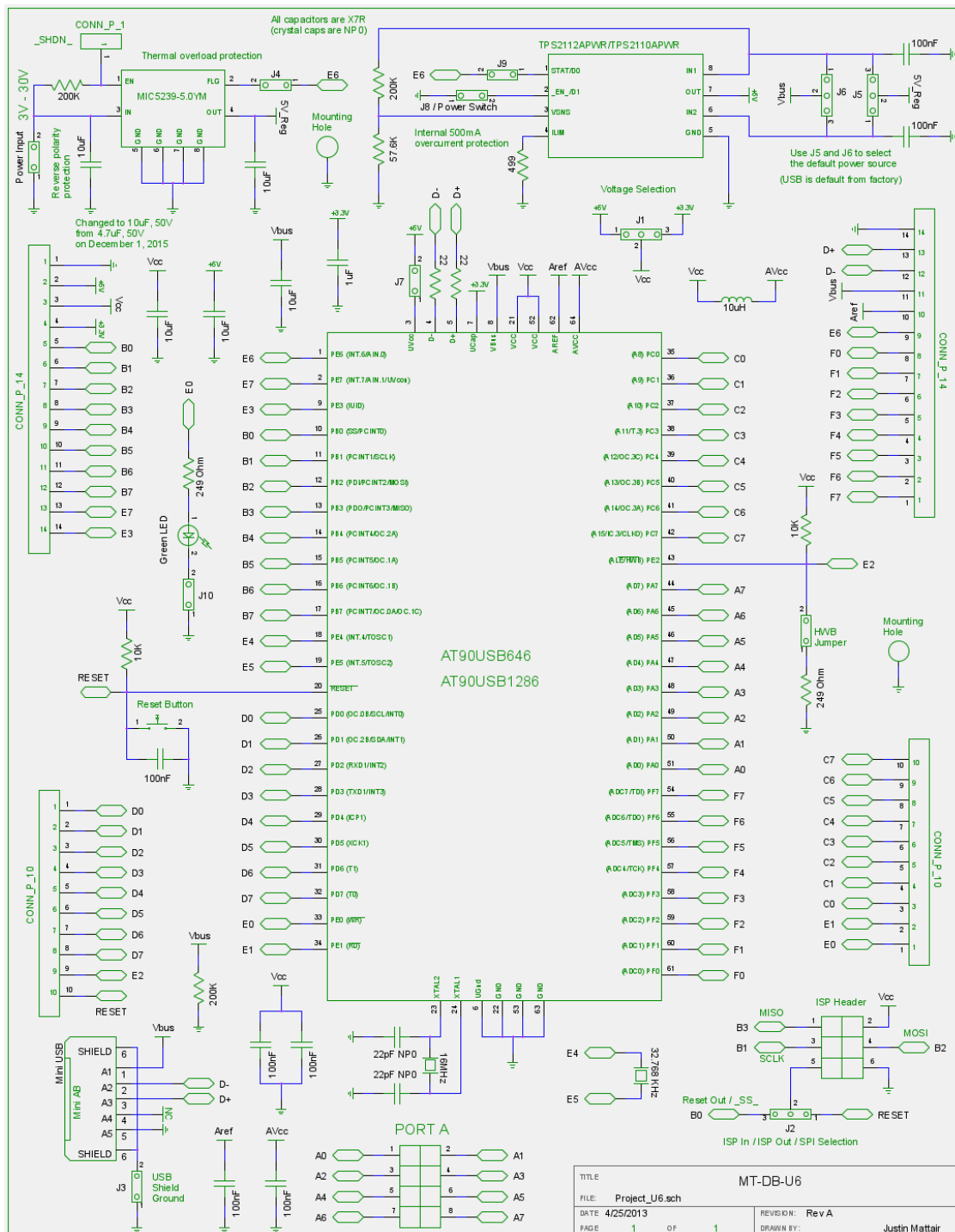
The following example saves three functions to EEPROM. It is then run in the background, pulsing an LED on pin 6 using analog write (PWM):

```
bitlash here! v2.0RC4 (c)2011 Bill Roy, bitlash.net -type HELP- 1706 bytes free
> print free, " bytes free"
1702 bytes free
> pinMode(6,1)
> d6=1
> x=255;d=0;
> function brighter {if (x==255) {d=0;} else { a6=++x; snooze(2);}}
saved
> function dimmer {if (x==0) {d=1;} else {a6--x; snooze(2);}}
saved
> function pulseLED {if (d==0) {dimmer();} else {brighter();}
saved
> ls
function brighter {if (x==255) {d=0;} else { a6=++x; snooze(2);}};
function dimmer {if (x==0) {d=1;} else {a6--x; snooze(2);}};
function pulseled {if (d==0) {dimmer();} else {brighter();}};
> run pulseled
> ps
0: pulseled
> stop 0
>
```

Documentation for Bitlash is available at <http://bitlash.net/wiki/docindex>

Commands	arg else function help if ls peep print ps return rm run stop switch while
Functions	abs ar aw bc beep br bs bw constrain delay dr dw er ew free inb max millis min outb pinmode printf pulsein random shiftout sign snooze

Schematic



Fuse and Lock Settings

The bootloaders were pre-installed with the following commands (ATmega1286 CDC bootloader shown):

```
avrdude -p usb1286 -c avrisp2 -P usb -e  
avrdude -p usb1286 -c avrisp2 -P usb -U lfuse:w:0x7f:m -U hfuse:w:0x9a:m -U efuse:w:0xfb:m  
avrdude -p usb1286 -c avrisp2 -P usb -B 4 -U flash:w:"Bootloader.hex"
```

The Bitlash program was pre-installed with the following command (ATmega1286 CDC bootloader shown):

```
avrdude -p usb1286 -c avr109 -P /dev/ttyACM0 -U flash:w:"bitlashdemo_MT-DB-U6-128.hex"
```

The lockbits are not set with the CDC bootloader. They ARE set with the Atmel DFU bootloader.

Troubleshooting / FAQ

- Nothing yet

Support Information

Please check the MattairTech website (<http://www.MattairTech.com/>) for firmware and software 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. A support forum is planned. Support for AVR's in general can be found at AVRfreaks (<http://www.avrfreaks.net/>). There, I monitor the forums section as the user physicist.

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

Acknowledgments

Thanks to Dean Camera (<http://www.fourwalledcubicle.com/>) for his excellent LUFA library, CDC bootloader, DFU bootloader, and AVRISP mkII clone programmer. Thanks to the members of AVRfreaks (<http://www.avrfreaks.net/>) for their support. Finally, thanks to Atmel for creating a great product, the AVR microcontroller.

Legal

Copyright Notices

Portions of this code are copyright (c) 2009-2013 Justin Mattair (www.mattairtech.com)
This code uses the LUFA USB library Copyright (C) 2013, Dean Camera (www.fourwalledcubicle.com)
and distributed under a modified MIT license (see files).
The CDC and DFU bootloaders are modified versions from LUFA.
The Arduino core files are copyright (c) 2005-2013 David A. Mellis (www.arduino.cc),
copyright (c) 2004-2010 Hernando Barragan (wiring.org.co),
Copyright 2011-2013, Paul Stoffregen, paul@pjrc.com,
copyright (c) 2006 Nicholas Zambetti,
and copyright (c) 2009 Brett Hagman.
They were modified by Justin Mattair and retain the LGPL 2.1 license (see files).
The Bitlash files are Copyright (C) 2008-2012 Bill Roy (bitlash.net)
They were modified by Justin Mattair and retain the original BSD style license (see files).
Portions of this code are copyright © 2003-2012, Atmel Corporation (<http://www.atmel.com/>)

Software Warranty 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

This development board/kit is intended for use for FURTHER ENGINEERING, DEVELOPMENT, DEMONSTRATION, OR EVALUATION PURPOSES ONLY. It is not a finished product, and may not (yet) comply with some or any technical or legal requirements that are applicable to finished products, including, without limitation, directives regarding electromagnetic compatibility, recycling (WEEE), FCC, CE, or UL (except as may be otherwise noted on the board/kit). MattairTech LLC supplied this board/kit AS IS, without any warranties, with all faults, at the buyer's and further users' sole risk. The user assumes all responsibility and liability for proper and safe handling of the goods. Further, the user indemnifies MattairTech LLC from all claims arising from the handling or use of the goods. Due to the open construction of the product, it is the user's responsibility to take any and all appropriate precautions with regard to electrostatic discharge and any other technical or legal concerns.

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.

Appendix A: Precautions

CAUTION

Do not change power configuration while unit is powered.
Do not short 5V, Vbus, 3.3V, Avcc, or ground to each other.
When connecting Aref externally, connect to a voltage source \leq Vcc and be sure that the internal reference is disabled.

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.

CAUTION

Normally, power is supplied from Vin or Vbus.
However, it is possible to disconnect the regulator and supply an externally regulated voltage on the 3.3V, 5V, and/or Vcc pins. When doing this, care must be taken to limit inrush current on these pins due to the low ESR of the ceramic capacitors. Failure to do so may cause damaging inductive voltage spikes due to any wire inductance (ie: benchtop power supply leads). Inrush current is normally controlled by the PTC fuse, which has a small series resistance.


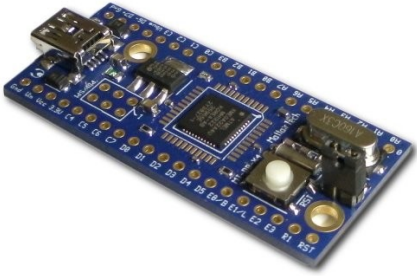
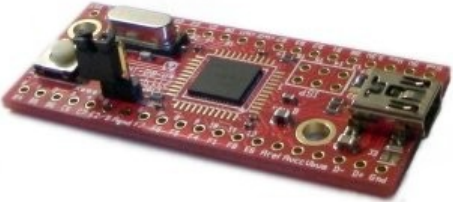
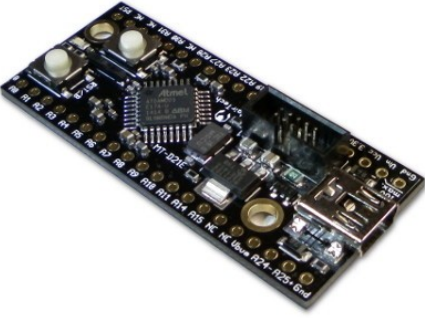
CAUTION

At higher input voltages to the regulator, the larger voltage drop will mean higher thermal dissipation for a given amount of current.
Be sure to limit current consumption to prevent excessive heat.
The regulator will enter thermal shutdown if it gets too hot.

CAUTION

The MT-DB-U6 contains static sensitive components.
Use the usual ESD procedures when handling.

Appendix B: Other MattairTech Products

 <p>AVR Programmer and Multitool</p> <p>ZeptoProg II</p>	<p>ZeptoProg II AVRISP mkII Programmer</p> <ul style="list-style-type: none"> • AVRISPmkII compatible AVR Programmer ■ Supports all AVR's with ISP, PDI, or TPI ■ Optional 5V output via headers to target board, with standard jumper and polyfuse ■ 4-channel Logic Analyzer ■ GPIO / PWM / frequency input & output ■ SPI Interface • Serial Bridge • Target board voltage of 2V to 5.5V via level-shifted pins on two main headers
	<p>MT-DB-X4 USB AVR XMEGA board</p> <ul style="list-style-type: none"> • ATxmega128A4U USB XMEGA AVR • 128KB FLASH, 8KB SRAM, 2KB EEPROM • 3.3V LDO regulator (low quiescent current) • 16MHz and 32.768KHz crystals • LED, boot jumper, PDI header • Reset button, mounting holes • USB DFU bootloader preinstalled
	<p>MT-DB-U4 USB AVR development board</p> <ul style="list-style-type: none"> • ATmega32U4 USB AVR • 32KB FLASH, 2.5KB SRAM, 1KB EEPROM • 16MHz crystal, LED, USB powered • boot jumper, Reset button, mounting holes • Arduino compatible • CDC or DFU bootloader
	<p>MT-D21E USB ARM Cortex M0+ board</p> <ul style="list-style-type: none"> • ATSAMD21E17A or ATSAMD21E18A (32-pin) • 128KB/256KB FLASH, 16KB/32KB SRAM • Onboard 3.3V, 250mA LDO regulator (2uA quiescent) • 16MHz and 32.768KHz crystals • USB connector (power by USB or external up to 15V) • Blue LED, 10-pin Cortex header, 2 buttons, I2C pullups • USB Mass Storage Bootloader (no programmer required)