MattairTech

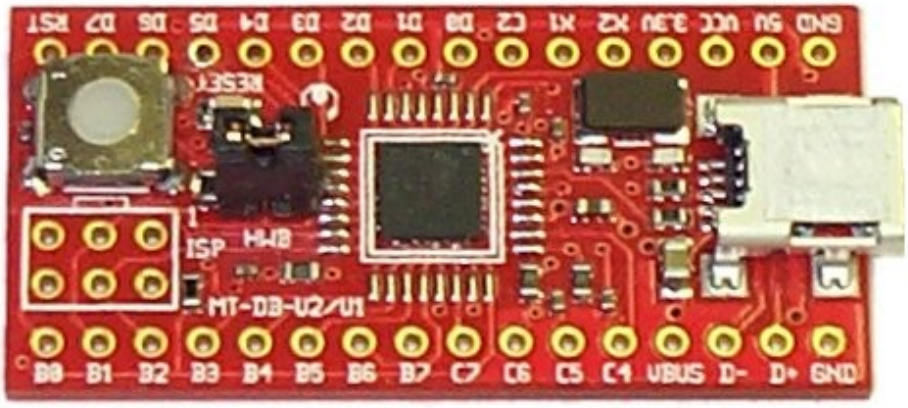


Table of Contents

Overview

Introduction

The MT-DB-U1 is a development board for the Atmel AT90usb162 USB microcontroller. The AT90usb162 contains 16 KB of flash, 512 bytes SRAM, 512 bytes EEPROM, 22 programmable IO pins, and a USB device controller. The board has 32 pins in a dual inline configuration with 100 mil pin spacing and 600 mil row spacing which allows for easy mounting on a breadboard. It includes a mini USB connector, status LED, 16MHz crystal, reset button, HWB boot jumper, 3 PWM filters, and ISP header pads. A bootloader comes preinstalled which allows programming of the chip over USB without an external programmer. The ISP header can be used with an external programmer for in-system programming. This header can be reconfigured to allow the MT-DB-U1 itself to be an ISP programmer, or to be used as a SPI master or slave. The board can be powered at 5V via USB, at 3.3V via the AT90usb162 internal regulator and USB, or it can be powered externally (3V – 3.6V or 4.0V – 5.5V). All programmable IO pins are routed to headers, including those used by on-board hardware. The chip can be clocked externally, and the board is compatible with HV programming. The USB connections are also routed to header pins, which allows for panel-mount USB connectors. The PCB is high-quality with ENIG (gold-plated) finish, red soldermask, and white screenprinting showing the pinout. It measures approximately 1.7” x 0.8” (42mm x 20mm) and is 0.062” (1.6mm) thick.

MT-DB-U1 Features

- AT90usb162 USB microcontroller
  - 16KB FLASH, 512 bytes SRAM, 512 bytes EEPROM
  - USB device controller, Serial USART, and SPI communications
  - 2 timers with 5 PWM channels
- Arduino compatible (now supports IDE 1.6.7 and boards manager)
- CDC (Arduino/AVRDUDE) or DFU (FLIP) bootloader preinstalled
- ISP header pads (program chip using external programmer)
- 16MHz crystal
- Green Status LED
- 3 RC PWM filters on each output compare pin of 16-bit timer 1 (can be disconnected)
- Reset button
- Bootloader selection jumper
- Mini USB connector
- Can be powered via USB at 5V (Vbus) or 3.3V (AT90usb162 internal regulator)
- Can be powered externally at 3V to 3.6V or 4V to 5.5V
- All programmable IO pins routed to headers (including those used by on-board hardware)
- USB pins routed to header pins (for panel-mount USB connector)
- High-quality PCB with gold-plated finish and red soldermask
- DIL-32 board, standard 0.1" pin spacing. Can be mounted on a breadboard
- Measures approx. 1.7” x 0.8” (42mm x 20mm) and 0.062” (1.6mm) thick.

AT90usb162 Features

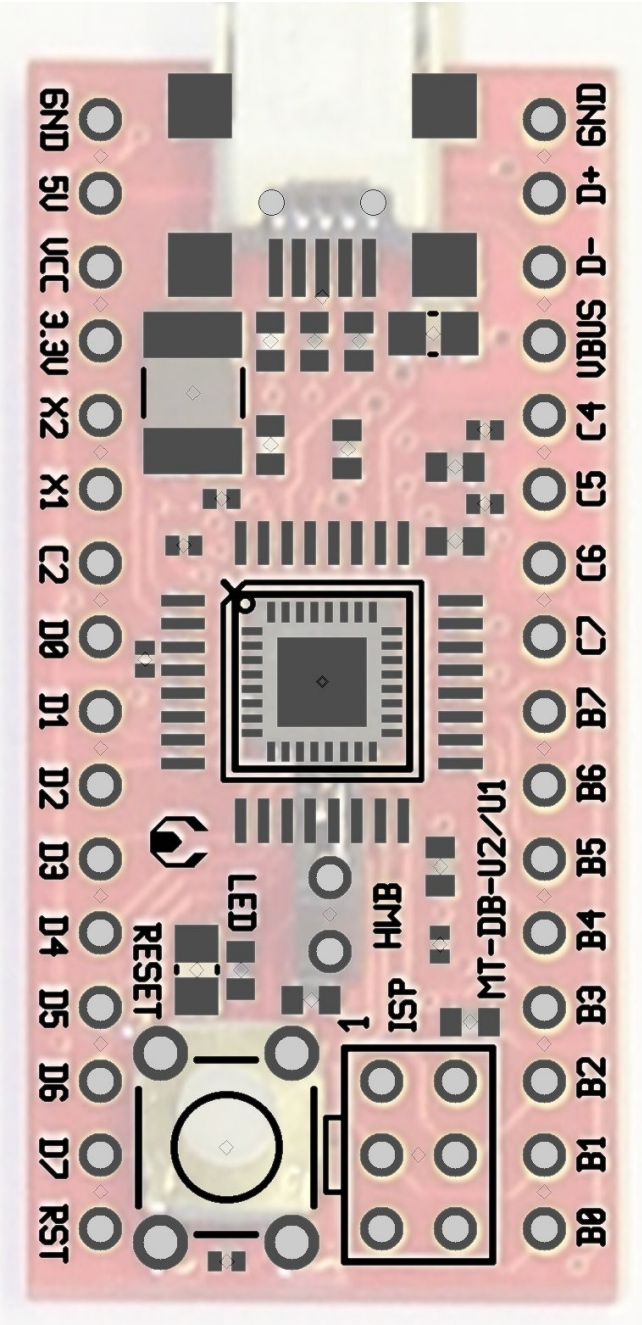
- **High Performance, Low Power AVR® 8-Bit Microcontroller**
- **Advanced RISC Architecture**
  - 125 Powerful Instructions – Most Single Clock Cycle
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 Mhz
- **Non-volatile Program and Data Memories**
  - 16K Bytes of In-System Self-Programmable Flash
  - 512 Bytes EEPROM
  - 512 Bytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/ 100,000 EEPROM
  - Data retention: 20 years at 85°C/ 100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by on-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
- **USB 2.0 Full-speed Device with Interrupt on Transfer Completion**
  - Complies fully with Universal Serial Bus Specification REV 2.0
  - 48 MHz PLL for Full-speed Bus Operation: 12 Mbit/s data rate
  - Fully independant 176 bytes USB DPRAM for endpoint memory
  - Endpoint 0 for Control Transfers: from 8 up to 64-bytes
  - 4 Programmable Endpoints:
    - IN or Out Directions
    - Bulk, Interrupt and IsochronousTransfers
    - Programmable maximum packet size from 8 to 64 bytes
    - Programmable single or double buffer
  - Suspend/Resume Interrupts
  - Microcontroller reset on USB Bus Reset without detach
  - USB Bus Disconnection on Microcontroller Request
- **Peripheral Features**
  - One 8-bit Timer/Counters with Separate Prescaler and Compare
    (two 8-bit PWM channels)
  - One 16-bit Timer/Counter with Prescaler, Compare and Capture
    (three 8-bit PWM channels)
  - USART with SPI master mode and hardware flow control (RTS/CTS)
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- **On Chip Debug Interface (debugWIRE)**
- **Special Microcontroller Features**
  - Power-On Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - 5 Sleep Modes: Idle, Powersave, Powerdown, Stby., and Ext. Stby.
- **I/O and Packages**
  - 22 Programmable I/O Lines

QFN32 (5x5mm) / TQFP32 packages

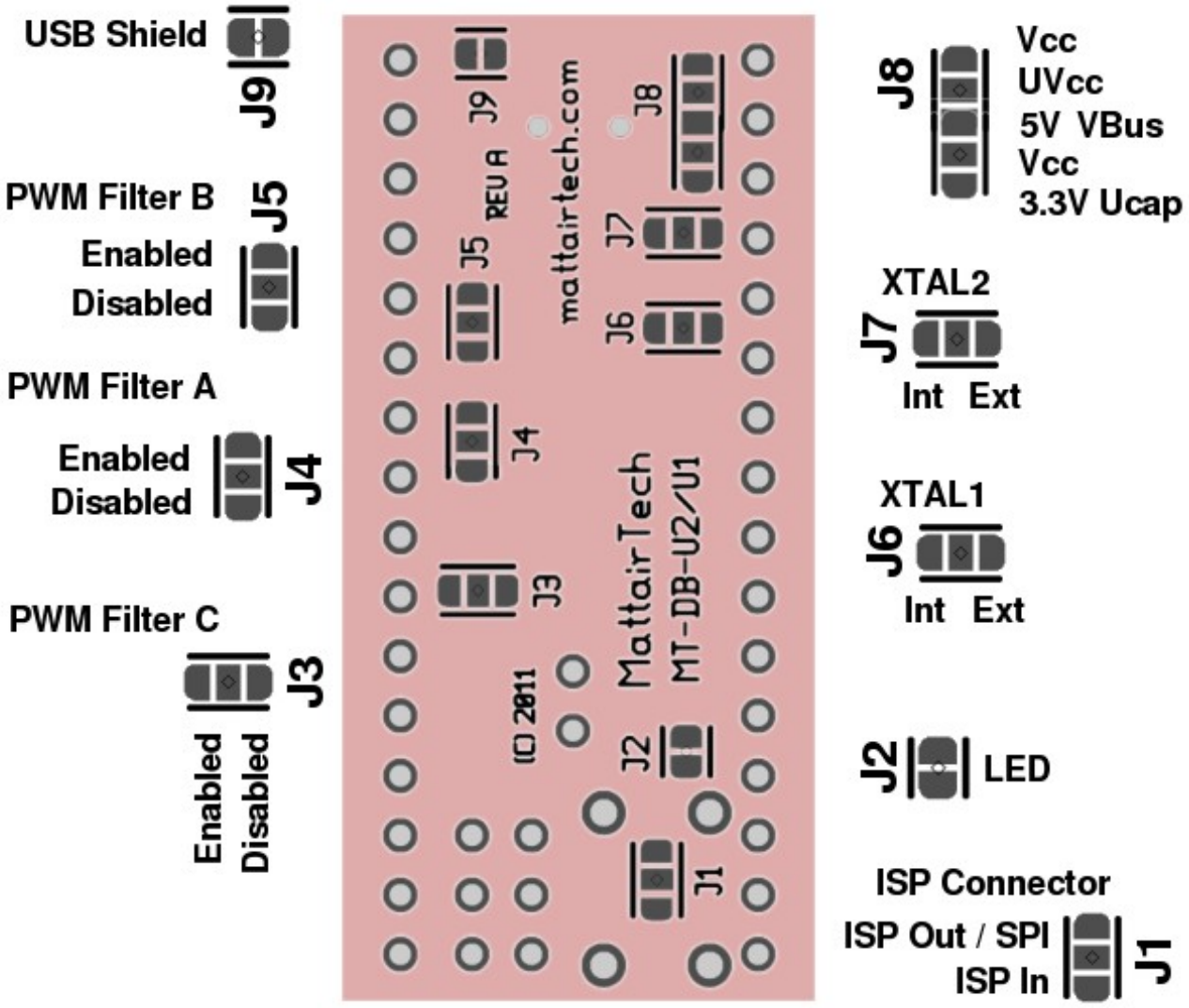
- **Operating Voltages**
  2.7 – 5.5V
- **Operating temperature**
  Industrial (-40°C to +85°C)
- **Maximum Frequency**
  8 MHz at 2.7V - Industrial range
  16 MHz at 4.5V - Industrial range

MT-DB-U1 Hardware

Layout / Header Pins



Solder Jumpers



Pin Descriptions

Pin	Description
Gnd	Ground
5V, Vbus	5V output from USB Vbus. Vbus pin and 5V pin are tied together.
Vcc	Voltage input pin. Use solder jumper J8 to configure. This pin is connected to the Vcc and AVcc pins on the microcontroller, as well as the ISP header and reset pullup. See Power Configuration Section.
3.3V	3.3V output from the microcontroller internal 3.3V regulator. This pin is connected to Ucap on the microcontroller.
X2 (C0)	This pin can be connected to the XTAL2 pin of the microcontroller using jumper J7. This is useful if pin C0 is used as GPIO (if external clock is used). This pin is disconnected by default (onboard crystal is used).
X1	This pin can be connected to the XTAL1 pin of the microcontroller using jumper J7. This is useful if an external clock is used. This is also useful for HV programming or recovery from incorrectly set fuses. This pin is disconnected by default (onboard crystal is used).
C2	GPIO pin (Port C) Consult datasheet for functionality.
D0 - D7	GPIO pins (Port D) Consult datasheet for functionality.
D0 / LED	The green status LED is connected to this pin when solder jumper J2 is set. The LED is connected to ground through a 240 ohm resistor. The user application is free to use this LED. Drive the pin high to turn on the LED.
D7 / HWB	This pin is connected to the bootloader jumper (HWB). The jumper is connected to ground through a 240 ohm resistor. The pin is sampled after all reset sources, including power-up. If the pin is low (HWB jumper installed), then the bootloader is run. If the pin is high (HWB jumper removed), then the user application is run. This pin functions as a normal GPIO pin at all other times. The 240 ohm resistor provides short-circuit protection in case the pin is used as an output and the jumper is installed.
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. If jumper J1 is set to ISP In, then RST is also connected to pin 5 of the ISP header.
B0 - B6	GPIO pins (Port B) Consult datasheet for functionality.
B7	PWM filter C output (OC.1C) or pin B7, depending on jumper configuration.
C4, C7	GPIO pins (Port C) Consult datasheet for functionality.
C5	PWM filter B output (OC.1B) or pin C5, depending on jumper configuration.
C6	PWM filter A output (OC.1A) or pin C6, depending on jumper configuration.

D+, D-	USB data pins. Can be used for panel-mount connectors.

PWM Filters

There are three PWM filters, which can be used to smooth out a PWM square wave into an analog voltage (with some ripple). The outputs of these filters can be connected to header pins C6, C5, and B7 using solder jumpers J4, J5, and J3 respectively (factory default). The filters consist of a 1K resistor and 100nF capacitor. The cutoff frequency is:

$$f_{3db} = 1/(2\pi RC) = 1/(2\pi 1\text{K} * 100\text{nF}) = 1/(2\pi 0.0001) = 1600 Hz$$

This is a first-order low-pass filter that can output levels from 0V to Vcc. All three filter inputs are connected to the 16-bit timer 1 output compare pins (OC.1A, OC.1B, and OC.1C). The filter inputs are always connected to the microcontroller. Therefore, if the solder jumpers are configured to connect the microcontroller pins directly to the header pins, there will be some loading on the pin (1K resistor in series with a 100nF capacitor to ground).

Clock Source

By default, a 16 MHz crystal is installed and connected to the XTAL pins of the AT90usb162. This 16 Mhz clock must be divided by 2 in software if the Vcc voltage is less than 4.5V. If an external clock is is used, solder jumper J6 can be switched to connect the microcontroller pin directly to header pin rather than the onboard crystal. An external clock signal can then be applied to pin X1 (XTAL1). This will also free up microcontroller pin C0 (XTAL2), which can be configured in the microcontroller as a GPIO pin and routed to header pin X2 by switching solder jumper J7.

HWB Jumper / RESET button / LED

The HWB Jumper is used to select either the bootloader or user application. The pin is sampled after reset or power-up. Note that the hardware HWB function of the AT90usb162 is disabled. That is, the HWBE fuse is disabled. The bootloader startup code is always run after reset or power-up (BOOTRST fuse is set). The code samples the state of the HWB pin. If the pin is low, the bootloader continues to run. If the pin is high, the user application is run. The green LED will pulse on and off using a continuously changing PWM period when the DFU bootloader is running. If the preinstalled demo program is running, it will be lit when USB is connected. Otherwise, the state of the LED is controlled by the user application. The bootloader always runs at 8 MHz, which is compatible with lower voltages.The user may set the cpu speed to 16MHz in software, if running at 5V.

Jumper	*Mode*	*Driver*
No	User Program	CDC (COM port) (*optional*)
Yes	Bootloader Program	CDC Serial/DFU Bootloader

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 the DFU 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.

The pins associated with the LED, jumper, and reset button are all routed to header pins. The LED can be disconnected by unsoldering jumper J2. The jumper is connected to pin D7 on one side and to ground through a 240 ohm resistor on the other side. There is a 10K pullup on the reset line.

ISP Header

The ISP header is configured by default to allow ISP programming using an external programmer. That is, RESET is routed to pin 5. Pin 1 is marked on the board (it is the pin closest to the chip). The ISP 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 J1. This allows the MT-DB-U1 to be used as an AVRISPmkII programmer itself, 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-U1 product page. Note that when using the ISP 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 connector. Another use for the ISP header configured with SS on pin 5 is to make use of SPI, either as a master or slave. SPI can also be used on the normal DIL headers. When using the ISP header, you may need to remove the HWB jumper to allow the ISP connector to fit.

Power Configuration

The MT-DB-U1 can be powered in a variety of ways by utilizing solder jumper J8 located on the bottom of the board. By default, the board is configured to be powered via USB, with Vcc at 5V, and the microcontroller internal 3.3V regulator enabled and powering only the USB pads. The microcontroller clock is configured at boot to run at 8MHz. The following lists some of the configurations possible. Code may have to be re-compiled when switching configurations (ie: to change cpu clock speed and/or internal regulator power).

Power Configuration	Jumper J8	Regulator	Max CPU
USB bus powered – 5V Vcc (default)	**3.3V Vcc = 5V = UVcc Vcc**	Enable	16 MHz
USB Bus powered – 3.3V Vcc	**3.3V = Vcc 5V = UVcc Vcc**	Enable	8 MHz
Externally powered – 4.0 to 5.5V Vcc	**3.3V Vcc 5V UVcc = Vcc**	Enable	16 MHz
Externally powered – 3.0 to 3.6V Vcc	**3.3V = Vcc 5V UVcc = Vcc**	Disable	8 MHz

WARNING

Care must be taken when configuring the solder jumpers.
It is possible to cause permanent damage to the device or the power supply by improperly setting the jumpers.
Do not change any jumpers while the unit is powered.
When using the microcontrollers internal regulator to power itself, be sure not to exceed the regulator maximum current output.

USB Bus Powered -- 5V

By default, the MT-DB-U1 is configured for 5V from the USB connector (Vbus). In this configuration, solder jumper J8 is set such that Vcc and UVcc are connected to Vbus (5V). The AVR internal 3.3V regulator must be enabled (default setting). This will supply 3.3V to the USB pads and 3.3V header pin.

USB Bus Powered – 3.3V

The internal 3.3V regulator can be used to supply the AVR itself with 3.3V. In this configuration, solder jumper J8 is set such that Vcc is connected to 3.3V and UVcc is connected to Vbus (5V). The AVR internal 3.3V regulator must be enabled (default setting). This will supply 3.3V to the AVR itself,

the USB pads, and the 3.3V header pin. Take care not to exceed the datasheet maximum current output of the internal regulator. Note that at 3.3V, the AVR should be set to run at 8MHz or less. This can be done in software using the prescaler.

Externally Powered – 4.0V to 5.5V

In this configuration, solder jumper J8 is set such that UVcc is connected to Vcc. Vcc is then supplied externally with 4.0V to 5.5V on the Vcc header pin. The 5V pin still outputs 5V when the USB cable is plugged in. The AVR internal 3.3V regulator must be enabled (default setting). This will supply 3.3V to the USB pads and 3.3V header pin. Note that when using a voltage less than 4.5V, the AVR should be set to run at 8MHz or less. This can be done in software using the prescaler.

Externally Powered – 3.0V to 3.6V

In this configuration, solder jumper J8 is set such that both UVcc and 3.3V are connected to Vcc. Vcc is then supplied externally with 3.0V to 3.6V on the Vcc header pin. The internal 3.3V regulator must be disabled in software. Note that the regulator is always enabled after reset or powerup, and is on when the bootloader is running. It is the responsibility of the user application to disable the regulator. The 5V pin still outputs 5V when the USB cable is plugged in. In this configuration, the AVR should be set to run at 8MHz or less. This can be done in software using the prescaler.

USB Shield

Jumper J9 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.

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/AT90USB128, 8-Bit AVR	ATmega32U4, 8-Bit AVR	ATmega32U2, 8-Bit AVR	AT90USB162, 8-Bit AVR
Clock Speed	16 MHz	16 MHz	16 MHz	16 MHz
Flash Memory	128 KB (AT90USB128) / 64 KB (AT90USB64)	32 KB	32 KB	16 KB
SRAM	8 KB (AT90USB128) / 4 KB (AT90USB64)	2.5 KB	1 KB	512 B
EEPROM	4 KB (AT90USB128) / 2 KB (AT90USB64)	1 KB	1 KB	512 B
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.

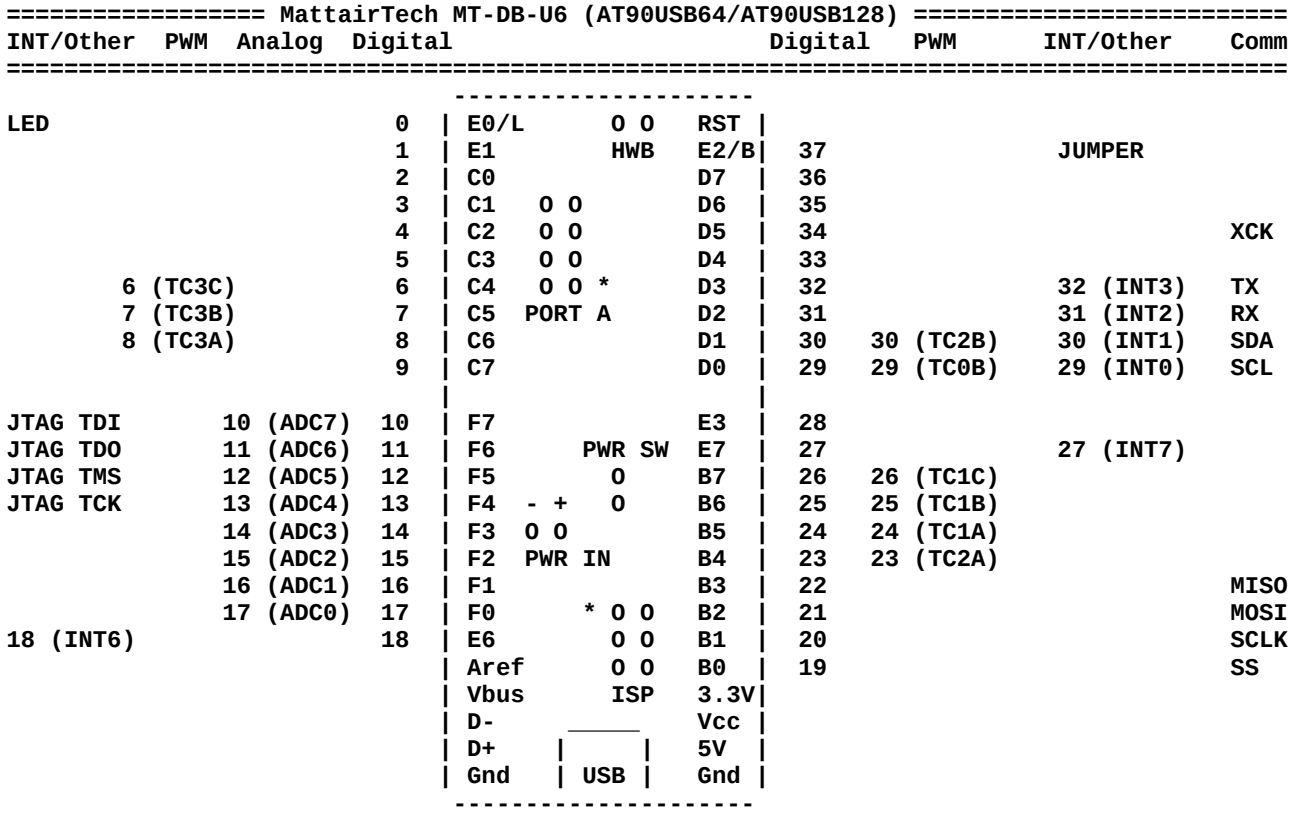
Special Notes

- **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)



* 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.

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 (note that several boards that were shipped before 130626 still have the old bootloader). 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).

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.

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 m32u4 -c avr109 -P COM5 -U flash:w:"bitlashdemo_MT-DB-U4.hex"
```

Example for Linux:

```
avrdude -p m32u4 -c avr109 -P /dev/ttyACM0 -U flash:w:"bitlashdemo_MT-DB-U4.hex"
```

Arduino environment:

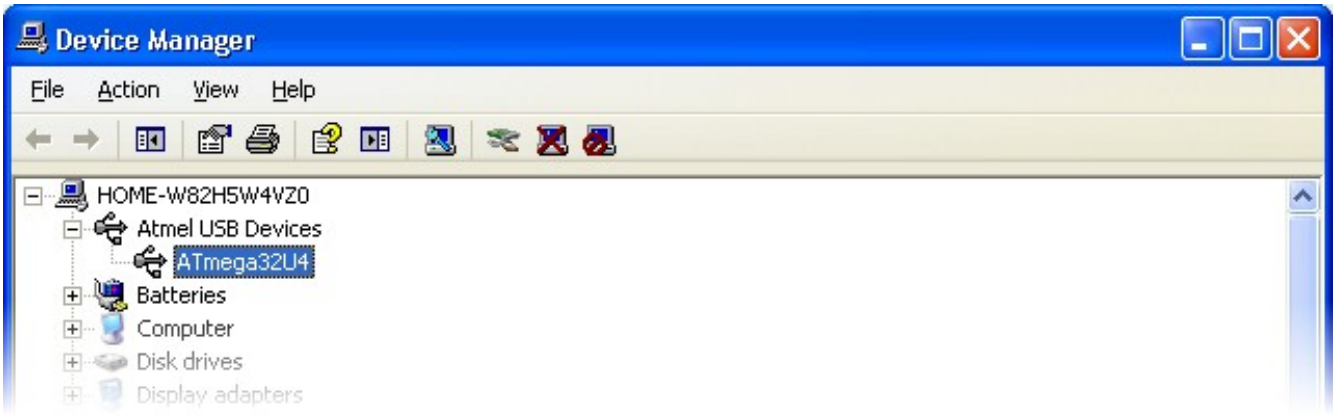
Be sure to select the COM port. Then upload your sketch with the Upload button.

DFU Bootloader (FLIP/dfu-programmer)

Installation

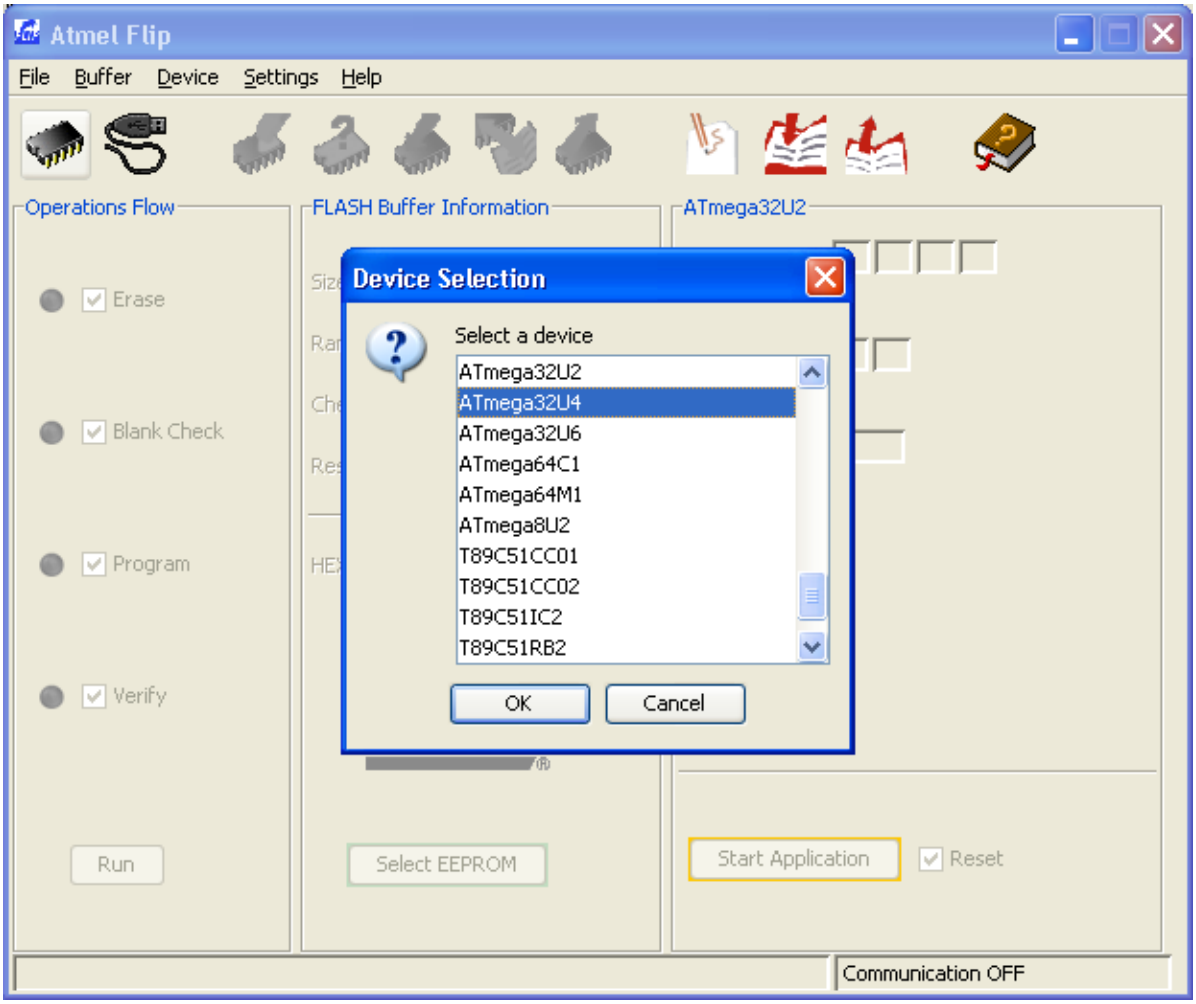
FLIP is a graphical utility used to load firmware into the AT90usb162. FLIP includes the DFU bootloader driver. It supports Windows XP through Windows 7 (32 or 64 bit). Download FLIP 3.4.2 or higher from http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3886 and install.

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 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.

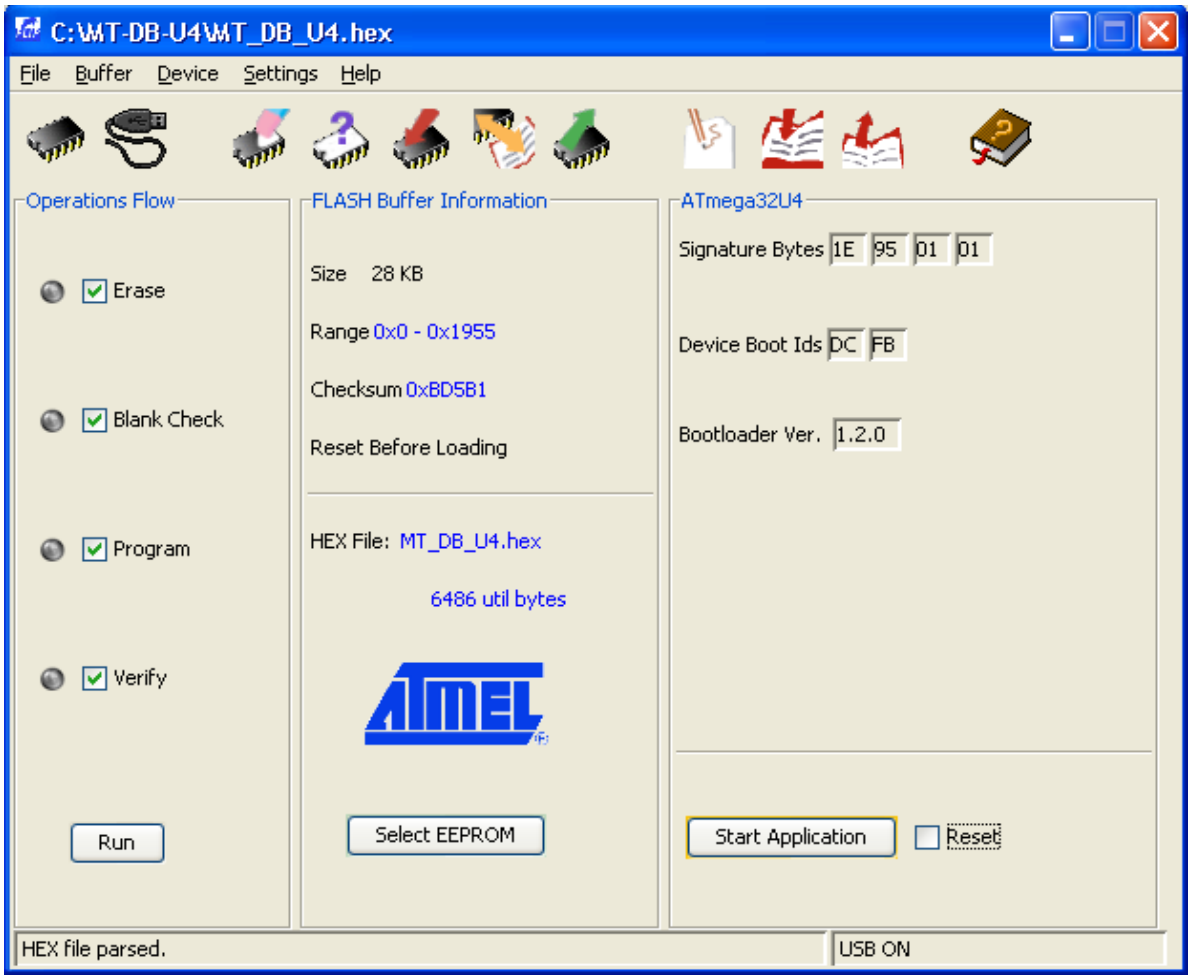


FLIP

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 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 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 AT90usb162.



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 AT90usb162 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 AT90usb162 erase

dfu-programmer AT90usb162 flash-eeprom YourHex.eep (if applicable)

dfu-programmer AT90usb162 flash YourHex.hex

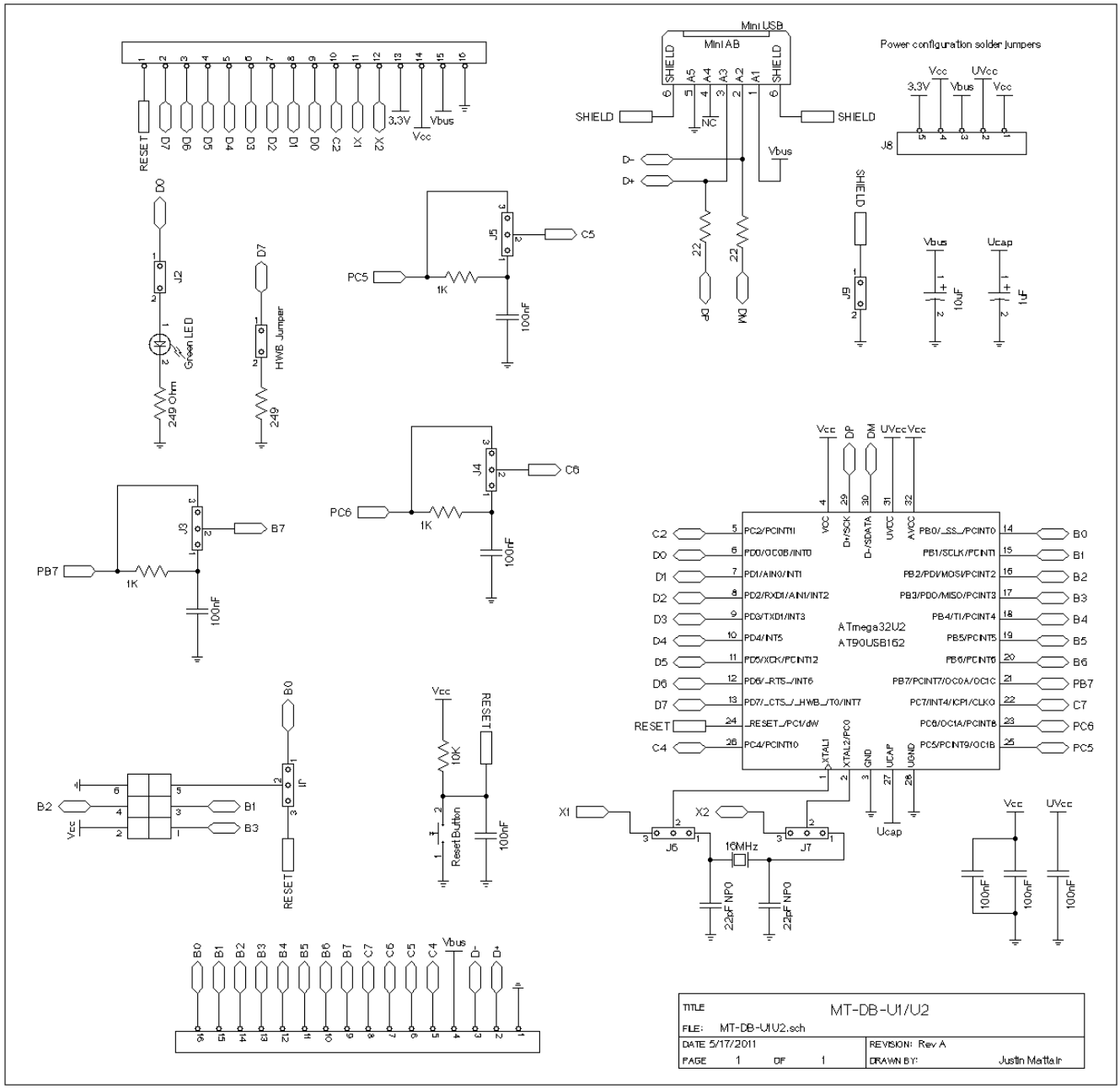
dfu-programmer AT90usb162 start (to jump to application section without reset)

Running Demo Program

The demo program makes use the use of the MT-DB-U1 as a CDC device (virtual COM port). This is one of the most common ways to connect to a PC over USB. It uses Dean Camera's open-source LUFA USB library available at http://www.fourwalledcubicle.com/. The LUFA download includes many examples that can be easily compiled for the AT90USB162.

See the CDC Bootloader section for details on installing the CDC Serial driver. The old demo requires an ANSI terminal to allow drawing of the menu system. If you see garbage on the terminal screen, click on the configuration icon and change the emulation to ANSI (or ANSIW). After connecting, a message that reads “Press any Key” is printed periodically. If you do not see this message, just press any key to continue.

Schematic


Mini USB
Mini AB
SHIELD
NC
Power configuration solder jumpers
Vcc
UVcc
3.3V
Vbus
J8
RESET
D0
D1
D2
D3
D4
D5
D6
D7
X1
X2
Vcc
J2
Green LED
249 Ohm
HWB Jumper
249
J5
C5
PC5
1K
100nF
D-
D+
22
DP
DM
J9
Ucap
10uF
1uF
J3
B7
PB7
J4
C6
PC6
ATmega32U2
AT90USB162
PC2/PCINT11
PD0/OC0B/INT0
PD1/AIN0/INT1
PD2/RXD1/AIN1/INT2
PD3/TXD1/INT3
PD4/_INT5
PD5/XCK/PCINT12
PD6/_RTS_/INT6
PD7/_CTS_/_HWB_/T0/INT7
RESET/PC1/dW
PC4/PCINT10
PB0/_SS_/PCINT0
PB1/SCLK/PCINT1
PB2/PDI/MOSI/PCINT2
PB3/PDO/MISO/PCINT3
PB4/T1/PCINT4
PB5/PCINT5
PB6/PCINT6
PB7/PCINT7/OC0A/OC1C
PC7/INT4/ICP1/CLKO
PC6/OC1A/PCINT8
PC5/PCINT9/OC1B
XTAL1
XTAL2/PC0
GND
UCAP
UGND
B0
B1
B2
B3
B4
B5
B6
B7
C7
C6
C5
C4
C2
RESET
X1
X2
J6
J7
16MHz
22pF NPO
J1
10K
Reset Button
TITLE
MT-DB-U1/U2
FILE: MT-DB-U1U2.sch
DATE 5/17/2011
REVISION: Rev A
PAGE 1 OF 1
DRAWN BY: Justin Mattair

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 AVRs 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 and bootloaders. 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

Software Warranty Disclaimer

Hardware Disclaimer

Trademarks

Appendix A: Precautions

WARNING

Care must be taken when configuring the solder jumpers.
It is possible to cause permanent damage to the device or the power supply by improperly setting the jumpers.
Do not change any jumpers while the unit is powered.
When using the microcontrollers internal regulator to power itself, be sure not to exceed the regulator maximum current output.

CAUTION

The MT-DB-U1 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.

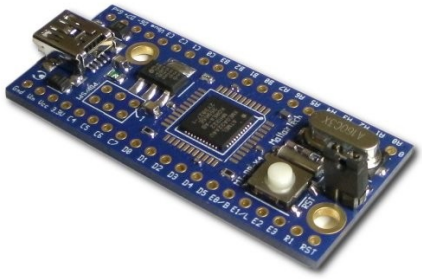
Appendix B: Other MattairTech Products

ZeptoProg II AVRISP mkII Programmer

- AVRISPmkII compatible AVR Programmer
- Supports all AVRs with ISP, PDI, or TPI
- Optional 5V output via headers to target board, with standard jumper and PTC fuse
- 4-channel Logic Analyzer
- Serial bridge / pattern generator / SPI interface
- GPIO / PWM / frequency input & output
- Atmel Studio / AVRDUDE support
- Target board voltage of 2V to 5.5V via level-shifted pins on two main headers

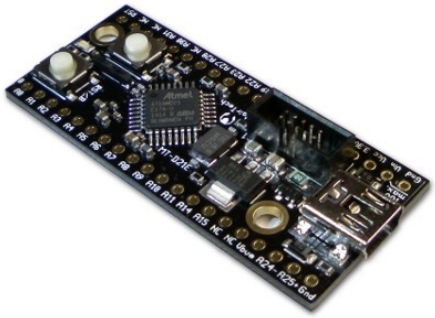
MT-DB-U6 USB AVR development board

- AT90USB646 / AT90USB1286 USB AVR
- 64KB/128KB FLASH, 4KB/8KB SRAM
- 5V, 500mA LDO regulator (3V-30V input)
- Auto power source selection IC (USB/External)
- 16MHz and 32.768KHz crystals
- Arduino compatible
- CDC or DFU bootloader



MT-DB-X4 USB AVR XMEGA board

- 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



MT-D21E USB ARM Cortex M0+ board

- 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)