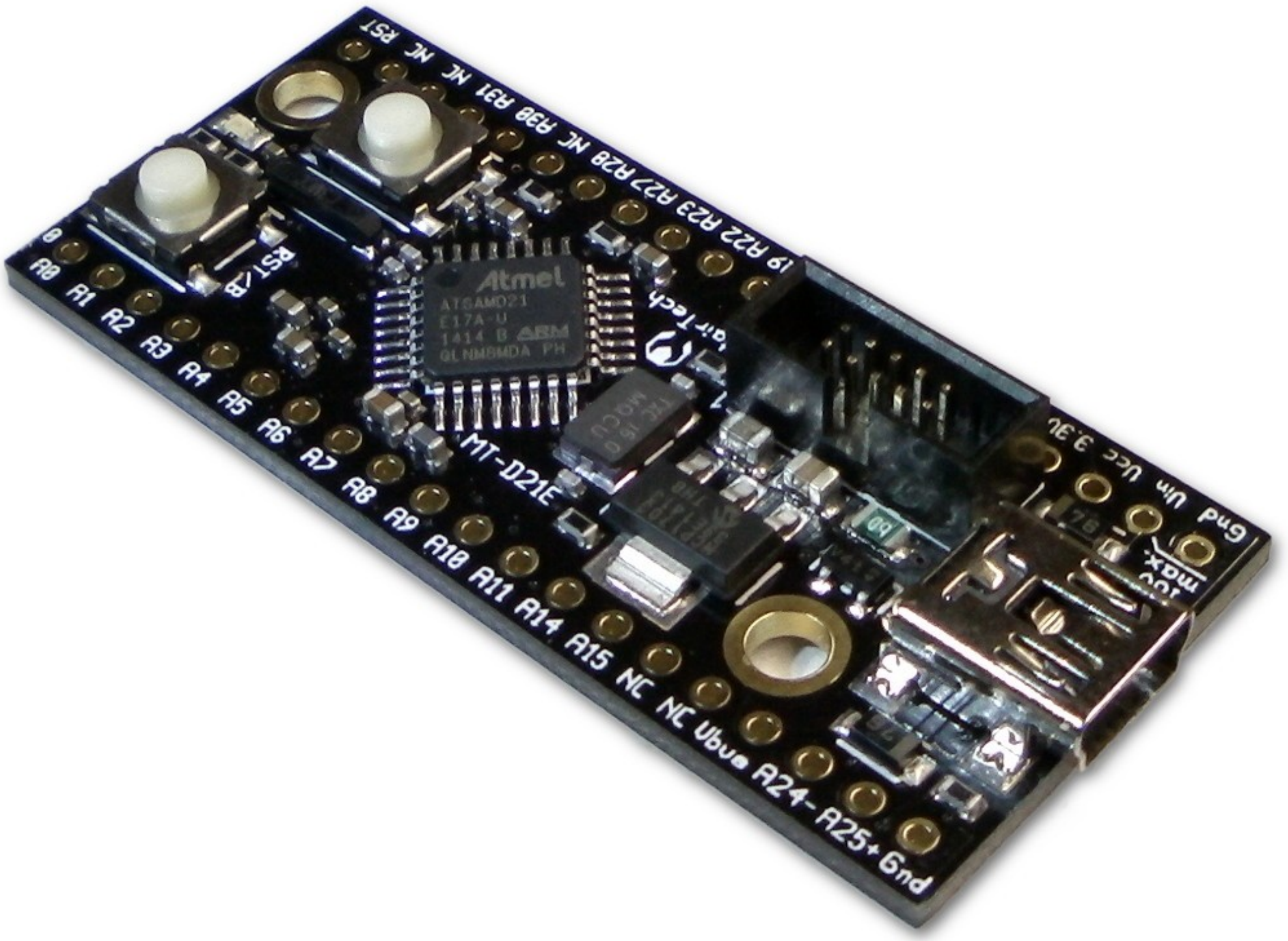
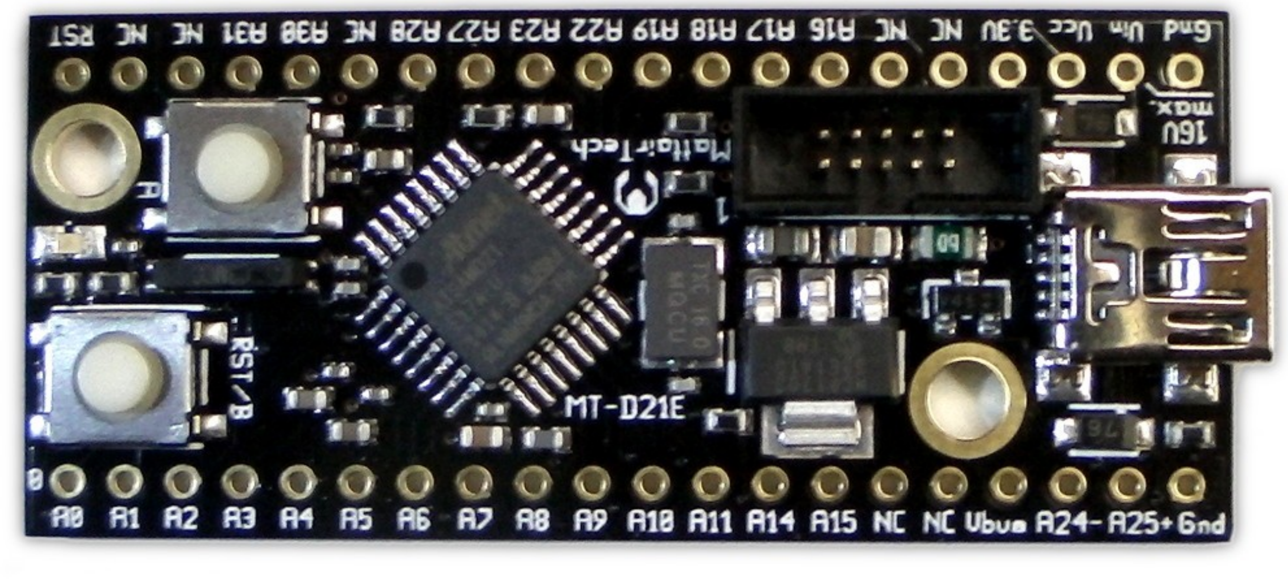
MT-D21E User Guide



Revision A

Table of Contents

Overview



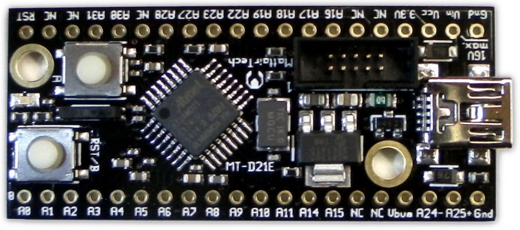
Introduction

The MT-D21E is a development board for the 32-pin Atmel SAM D21E ARM Cortex M0+ USB microcontroller. It can be powered from USB or from the Vin pin. Two schottky diodes facilitate simple switching (and reverse-polarity protection) between the two power sources. This voltage is regulated to 3.3V by the onboard 250mA, extremely low quiescent current (2uA) LDO regulator that supports up to 16V DC input voltage. Overcurrent protection is provided by a 180mA hold (400mA trip) PTC resettable fuse. Also mounted is a mini USB connector, blue LED, 16MHz crystal, 32.768KHz crystal, and two buttons. A USB CDC bootloader (Atmel SAM-BA) can be pre-installed for device programming without an external programmer. It is compatible with Arduino, and core files are provided to support Arduino 1.6.5+. A USB Mass Storage Class bootloader can optionally be installed for device programming (see caveats). The Cortex debug header (10-pin, 50-mil) can be used with an external debugger/programmer. The board has 40 main dual inline header pins with 100 mil pin spacing and 700 mil row spacing which allows for mounting on a breadboard or perfboard. There are 2 3mm mounting holes. The PCB measures approx. 2.1” x 0.9” x 0.062” (52mm x 23mm x 1.6mm).

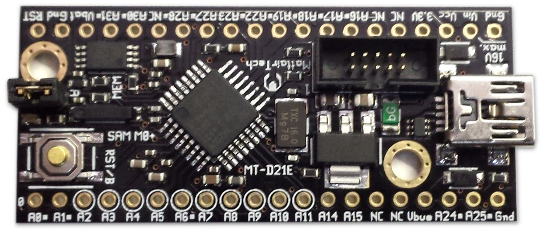
Board Features

- **Atmel SAM D21E 32-pin ARM Cortex M0+ microcontroller**
  - ATSAMD21E17A (128KB) or ATSAMD21E18A (256KB)
  - Up to 48MHz
  - 128KB or 256KB in-system self-programmable Flash
  - 16KB or 32KB SRAM Memory
- **Onboard 3.3V, 250mA LDO regulator**
  - up to 16V DC input
  - extremely low quiescent current (2.0uA typical)
  - low dropout (525mV typical @ 250mA, 725mV max. @ 250mA)
  - 0.4% output tolerance typical
  - Over-current and over-temperature protection
- **Simple power source switching**
  - 2 schottky barrier diodes (Vbus and Vin)
  - Low voltage drop
  - Reverse-polarity protection
- **PTC resettable fuse (180mA hold / 400mA trip)**
- **Cortex Debug Header (10-pin, 50-mil)**
  - Can be used for device programming and debugging
- **16MHz crystal (can use PLL for up to 48MHz cpu clock)**
- **32.768KHz crystal**
- **Blue Status LED (can be disconnected)**
- **Button A for general use (pin A27) with debouncing**
- **Button B configurable for reset or general use (pin A31) with debouncing**
- **Two 4.7Kohm resistors can be connected to pins A16 and A17 for use with I2C**
- **USB SAM-BA (CDC) bootloader (optional)**
  - Arduino compatible (use the Arduino IDE to upload)
  - Bossa command line utility (Windows, Linux, limited OS X)
- **Arduino 1.6.5+ compatible core (1.6.6 support now available)**
- **USB Mass Storage Device (MSD) bootloader (optional, see caveats)**
- Mini USB connector
- ESD protection on USB D+ and D- lines
- USB pins routed to header pins (for panel-mount USB connector)
- Powered by USB or external power source (up to 16V) on Vin
- Ferrite bead and 2 capacitors on analog supply
- Two capacitors each can be enabled for pins A3 and/or A4 for use with external references
- **19 solder jumpers on PCB bottom for configuration flexibility**
- All PORT pins routed to headers
- 2 main headers are on 0.1” spacing (breadboard/perfboard mounting)
- **Two 3mm mounting holes (~5mm pad)**
- High-quality PCB with gold-plated finish
- Measures approx. 2.1” x 0.9” (52mm x 23mm) and 0.062” (1.6mm) thick.

Board Revisions



Revision A *(until March 14, 2017)*
MT-D21E_User_Guide.pdf (this manual)



Revision B *(released March 15, 2017)*
MT-D21E_revB_User_Guide.pdf

Revision B includes the following changes:

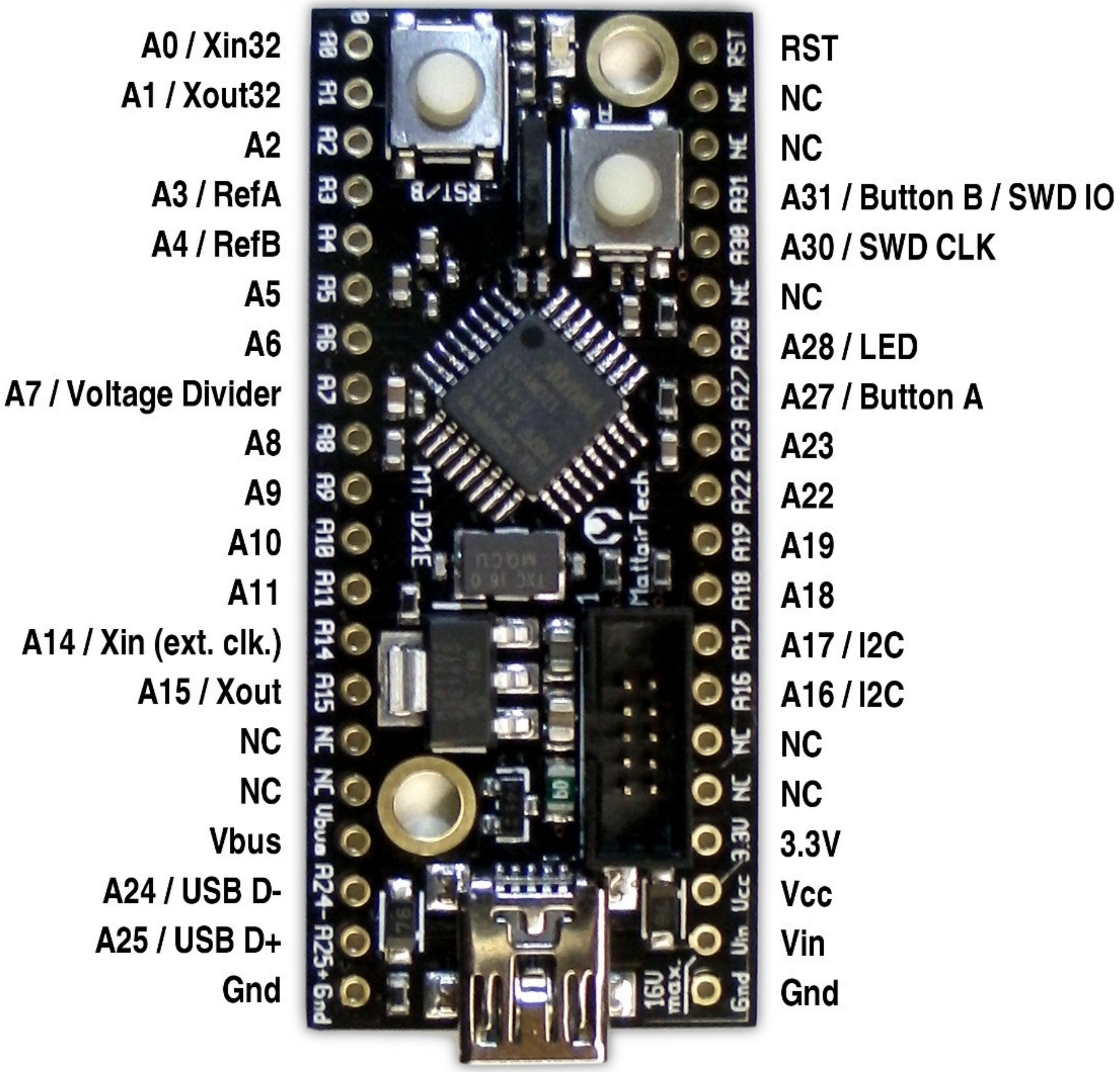
- Add support for more MCU options (SAMD, SAML, or SAMC)
  - With the SAML, an inductor is installed to support the buck converter
  - 5V from USB Vbus can be connected to Vcc to support SAMC at 5V (through 2 jumpers)
- Serial memory device can be installed (128K SRAM, 512KB FLASH, or 64KB EEPROM)
  - Using SPI at up to 12MHz using pins A18, A19, A22 (A27 for CS)
- Added Vbat and Gnd pins (coin cell), can connect to SRAM and/or Vcc (advanced)
- Button A now jumper A to make room for memory device, used as CS for memory as well
- Moved the blue LED to pin A6 (arduino pin 6), changed resistor to 1Kohm (600uA @ 3.3V)
- Physically larger PTC fuse (0805) with higher trip (500mA) and hold (250mA)
- Changed ESD device to protect Vbus in addition to D+ and D-, Vbus can be unpowered

ATSAMD21ExxA Features

- **Processor**
  - ARM Cortex-M0+ CPU running at up to 48MHz
    - Single-cycle hardware multiplier
    - Micro Trace Buffer
- **Memories**
  - 32/64/128/256KB in-system self-programmable Flash
  - 4/8/16/32KB SRAM Memory
- **System**
  - Power-on reset (POR) and brown-out detection (BOD)
  - Internal and external clock options with 48MHz Digital Frequency Locked Loop (DFLL48M) and 48MHz to 96MHz Fractional Digital Phase Locked Loop (FDPLL96M)
  - External Interrupt Controller (EIC) / One non-maskable interrupt
  - 16 external interrupts
  - Two-pin Serial Wire Debug (SWD) programming, test and debugging interface
- **Low Power**
  - Idle and standby sleep modes
  - SleepWalking peripherals
- **Peripherals**
  - 12-channel Direct Memory Access Controller (DMAC)
  - 12-channel Event System
  - **Up to five 16-bit Timer/Counters (TC), configurable as either:**
    - One 16-bit TC with compare/capture channels
    - One 8-bit TC with compare/capture channels
    - One 32-bit TC with compare/capture channels, by using two TCs
  - **Three 24-bit Timer/Counters for Control (TCC), with extended functions:**
    - Up to four compare channels with optional complementary output
    - Generation of synchronized pulse width modulation (PWM) pattern across port pins
    - Deterministic fault protection, fast decay and configurable dead-time (complementary outputs)
    - Dithering that increase resolution with up to 5 bit and reduce quantization error
  - **32-bit Real Time Counter (RTC) with clock/calendar function**
  - Watchdog Timer (WDT)
  - CRC-32 generator
  - **One full-speed (12Mbps) Universal Serial Bus (USB) 2.0 interface**
    - Embedded host and device function
    - Eight endpoints
  - **Up to six Serial Communication Interfaces (SERCOM), each configurable to operate as either:**
    - USART with full-duplex and single-wire half-duplex configuration
    - I_2C up to 3.4MHz
    - SPI
    - LIN slave
  - One two-channel Inter-IC Sound (I_2S) interface
  - **One 12-bit, 350ksps Analog-to-Digital Converter (ADC) with up to 20 channels**
    - Differential and single-ended input
    - 1/2x to 16x programmable gain stage
    - Automatic offset and gain error compensation
    - Oversampling and decimation in hardware to support 13-, 14-, 15- or 16-bit resolution
  - **10-bit, 350ksps Digital-to-Analog Converter (DAC)**
  - Two Analog Comparators (AC) with window compare function
  - Peripheral Touch Controller (PTC)
- **I/O**
  - Up to 52 programmable I/O pins

MT-D21E Hardware

Top View / Pinout



```
============================= MattairTech MT-D21E (ATsamd21eXXa) ========================
Other  INT     PWM   Digital  Analog                     Digital  PWM      INT      Other
=========================================================================================

                                      -------------------
Xin32                                | A0            RST |                          Reset
Xout32                               | A1            NC  |
DAC                     2   2(ADC0)  | A2            NC  |
REF                     3   3(ADC1)  | A3            A31 | 31  31(TCC11) 31(INT11) SWDIO*
     4(INT4)            4   4(ADC4)  | A4            A30 | 30  30(TCC10) 30(INT10) SWDCLK
     5(INT5)            5   5(ADC5)  | A5            NC  |
                        6   6(ADC6)  | A6            A28 | 28            28(INT8)   LED
VDIV                    7   7(ADC7)  | A7            A27 | 27            27(INT15)  BTNA
    8(INTNMI) 8(TCC00)  8   8(ADC16) | A8            A23 | 23  23(TC41)  23(INT7)   SS
    9(INT9)   9(TCC01)  9   9(ADC17) | A9            A22 | 22  22(TC40)  22(INT6)   MISO
TX1          10(TCC02)  10  10(ADC18)| A10           A19 | 19            19(INT3)   SCK
RX1          11(TCC03)  11  11(ADC19)| A11           A18 | 18            18(INT2)   MOSI
TX2 14(INT14) 14(TC30)  14           | A14           A17 | 17  17(TCC21)  17(INT1)  SCL
RX2           15(TC31)  15           | A15           A16 | 16  16(TCC20)  16(INT0)  SDA
                                     | NC            NC  |
                                     | NC            NC  |
                                     | Vbus          3.3V|   * Button B available on 31
USB D-                               | A24-  _____   Vcc |
USB D+                               | A25+ |     |  Vin |
                                     | Gnd  | USB |  Gnd |
                                      -------------------
```

Main Header Pins (Power)

Pin	Description
Gnd (2)	Ground
Vbus	Vbus is connected directly to the Vbus pin (5V) of the USB connector. It is routed through a schottky diode and through J6 to the regulator input circuitry, which includes a 4.7uF capacitor. Vbus voltage can be measured on pin A1 by connecting J12 and setting J3 toward the Vbus side of the board. J12 will complete the circuit for a resistor divider consisting of a 200Kohm (top) and a 20Kohm resistor (bottom), and J3 connects to Vbus. The resistor divider will pull pin A7 to near ground level when Vbus is disconnected. Because of a small leakage current from the schottky diode, a small voltage should be interpreted as USB disconnected.
Vin	Vin is the external power input pin. Up to 16V can be connected. It is routed through a schottky diode to the regulator input circuitry, which includes a 4.7uF, 25V capacitor. The schottky diode can be shorted with J1, eliminating the voltage drop across the diode, which can be useful for battery applications. Note that when the diode is shorted, reverse-polarity protection is disabled, and J6 should be disconnected to prevent Vbus current from flowing into Vin. Vin voltage can be measured on pin A1 by connecting J12 and setting J3 toward the Vin side of the board. J12 will complete the circuit for a resistor divider consisting of a 200Kohm (top) and a 20Kohm resistor (bottom), and J3 connects to Vbus. The resistor divider will pull pin A7 to near ground level when Vbus is disconnected. Because of a small leakage current from the schottky diode, a small voltage should be interpreted as Vin disconnected.

Vcc	This pin is connected to the Vcc and VccAna (through a ferrite bead) pins on the microcontroller, the Cortex debug header Vcc pin, the reset pullup, and the TWI pullup resistors. Vcc is connected to 3.3V through J5, which in turn is connected to the output of the onboard regulator. The Vcc pin can also be used as an input. Disconnect J5 to supply power from an external source to the Vcc pin.
3.3V	3.3V is connected to the output of the onboard 3.3V regulator. There is a 10uF capacitor on the output. 3.3V is normally connected to Vcc through J5.

CAUTION
Higher regulator input voltages mean larger voltage drops and thus higher thermal dissipation for a given amount of current. Be sure to limit current consumption to prevent excessive heat when using higher voltages and/or currents. The regulator will enter thermal shutdown if it gets too hot. All capacitors are X7R, X7S, or NP0, so they can deal with the higher temperatures of the regulator. Note that the PTC fuse is located near the regulator, so high temperatures will lower the PTC trip and hold currents.

Main Header Pins (Signal)

Pin	Description
A0, A1 (Xin32, Xout32)	These can be used for analog or digital functions. Alternatively, jumpers J16 and J17 can be set to route A0 and A1 to the 32.768KHz crystal.
A2, A5, A6	These can be used for analog or digital functions. Pin A2 can be used as a DAC output.
A3, A4	These can be used for analog or digital functions. Alternatively, jumpers J19 and/or J18 can be set to enable both a 100nF capacitor and a 1uF capacitor so that the pin can be used with an external voltage reference.
A7 / Voltage Divider	This can be used for analog or digital functions. Additionally, this pin can be connected to the voltage divider for measurement of Vin or Vbus by setting J3 and J12 appropriately.
A8 - A11	These can be used for analog or digital functions.
A14, A15 (Xin, Xout)	These can be used for digital functions. Pin A14 can be used with an external clock. Alternatively, jumpers J10 and J11 can be set to route A15 and A14 to the 16MHz crystal.
A24-, A25+ (USB D- and D+)	These can be used for digital functions. By default, these pins are also connected to pins D- and D+ of the USB connector through jumpers J7 and J4. These header pins, along with the adjacent Vbus and Ground pins can be used for a panel-mount USB connector.

A16, A17 (I2C)	These can be used for digital functions. Additionally, jumpers J8 and J9 can be enabled, which will connect two 4.7Kohm pullup resistors for use with I2C.
A18, A19, A22, A23	These can be used for digital functions.
A27 / Button A	This can be used for digital functions. By default, this pin is connected to Button A through jumper J13. This button is debounced using a 249ohm resistor and a 100nF capacitor. The pin is brought to ground when the button is pressed. The button is used for bootloader entry (optional) or can be for general purpose use.
A28 / LED	This can be used for digital functions. By default, this pin is connected to a blue LED through jumper J14 and a 499ohm resistor. The LED circuit should consume around 1mA. Drive the pin high to turn on the LED.
A30 / SWD CLK	This can be used for digital functions. Additionally, this pin is connected to the Cortex debug header where it is used as SWD CLK.
A31 / Button B / SWD IO	This can be used for digital functions. Additionally, the pin is routed to the Cortex debug header where it is used as SWD IO. Alternatively, this pin can be connected to Button B through jumper J15 (note that this button can also be used for RST). This button is debounced using a 249ohm resistor and a 100nF capacitor. The pin is brought to ground when the button is pressed. The button can be can be for general purpose use.
RST	RST connects to the reset pin of the microcontroller, to Button B through jumper J15 (note that this button can also be for general purpose use; see A31 above), and to the Cortex debug header. A 10K pullup resistor is connected as well.
NC	These pins are not connected to anything. There are 7 pins marked NC.
Cortex Debug Header	This 10-pin, 50-mil header can be connected to an external programmer/debugger.

Solder Jumpers

Jumper	Description
J1: Vin diode disable	Vin is the external power input pin. Up to 16V can be connected. It is routed through a schottky diode to the regulator input circuitry. The schottky diode can be shorted by closing J1, eliminating the voltage drop across the diode, which can be useful for battery applications. Note that when the diode is shorted, reverse-polarity protection is disabled, and J6 should be opened to prevent Vbus current from flowing into Vin.
J2: USB Shield Ground	Jumper J2 can be closed 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. Bear in mind that the USB shield will then act as an antenna. To avoid this, an 0603 component and an 0402 (ie: 1Mohm resistor and 4.5nF capacitor) may be soldered on the pads.
J3: Voltage divider input	Vin or Vbus voltage can be measured on pin A7 by closing J12 and setting J3 toward either the Vin (for Vin measurement) or the Vbus (for Vbus measurement) side of the board (refer to printing on top side of pcb). J12 will complete the circuit for a resistor divider consisting of a 200Kohm (top) and a 20Kohm resistor (bottom), and J3 connects to Vin or Vbus. The resistor divider will pull pin A7 to near ground level when Vbus is disconnected. Because of a small leakage current from the schottky diode, a small voltage should be interpreted as Vin/Vbus disconnected.
J4: USB D+ / Pin A25	Microcontroller pins A24 and A25 are connected to header pins A24- and A25+. By default, these pins are also connected to pins D- and D+ of the USB connector through jumpers J7 and J4. The header pins, along with the adjacent Vbus and Ground pins can be used for a panel-mount USB connector.
J5: Vcc – 3.3V	This connects the 3.3V regulator output rail to Vcc. Open J5 if supplying a regulated voltage (3.6V or less) externally on the Vcc pin.
J6: Vbus Power	This routes Vbus to the regulator input circuitry. There are two schottky diodes, one for Vin and one for Vbus. They facilitate automatic power switching between these two sources. If only external power will be used (Vin), open J6. This will prevent Vbus power from being used when a USB cable is plugged in for communications.
J7: USB D- / Pin A24	See J4.
J8: I2C pullup resistor	Close J8 to connect pin A16 through a 4.7Kohm resistor to Vcc for use with I2C.
J9: I2C pullup resistor	Close J9 to connect pin A17 through a 4.7Kohm resistor to Vcc for use with I2C.
J10: 16MHz crystal selection	J10 and J11 determine whether microcontroller pins A15 and A14 connect to header pins A15 and A14 or to the 16MHz crystal. When the SAM-BA bootloader is installed, the header pins are connected. If the MSD bootloader or no bootloader are installed, the alternate position is used by default(16MHz crystal connected). Note that the MSD bootloader does not use an external crystal, as it uses USB clock recovery (DFLL tuned using the USB SOF signal).
J11: 16MHz crystal selection	See J10. The image below shows connection to the header pins.
J12: Voltage divider	See J3.

enable	
J13: Button A enable	This jumper connects to Button A to pin A27. This button is debounced using a 249ohm resistor and a 100nF capacitor. The pin is brought to ground when the button is pressed. The button is used for bootloader entry (optional) or can be for general purpose use.
J14: LED enable	This jumper connects pin A28 to a blue LED through a 499ohm resistor. The LED circuit should consume around 1mA. Drive the pin high to turn on the LED.
J15: Button B function selection	This jumper connects button B to either the RST pin, which is the default as shown in the image below, or to pin A31 for general purpose use. This button is debounced using a 249ohm resistor and a 100nF capacitor. The pin is brought to ground when the button is pressed. Note that pin A31 is also used by the Cortex debug header (SWD IO). The button can be completely disconnected by removing solder from all three pads.
J16: 32.768KHz crystal selection	J16 and J17 determine whether microcontroller pins A0 and A1 connect to header pins A0 and A1 or to the 32.768KHz crystal. When the SAM-BA bootloader is installed, they are routed to the crystal. If the MSD bootloader or no bootloader is installed, they are routed to the header pins by default. Note that the MSD bootloader does not use an external crystal, as it uses USB clock recovery (DFLL tuned using the USB SOF signal).
J17: 32.768KHz crystal selection	See J16.
J18: VREF capacitors	When using pin A4 as VREF, close J18 to enable both a 100nF capacitor and a 1uF capacitor from A4 to ground.
J19: VREF capacitors	When using pin A3 as VREF, close J19 to enable both a 100nF capacitor and a 1uF capacitor from A3 to ground.

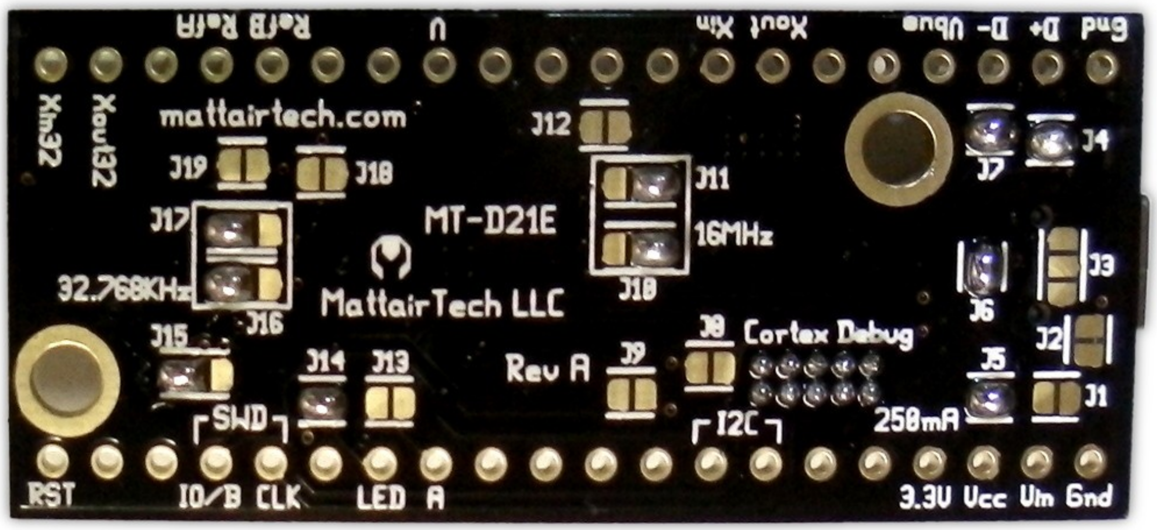


Image note: The solder jumper configuration shows BOTH crystals disconnected.

MattairTech Arduino SAM M0+ Core

Please visit https://github.com/mattairtech/ArduinoCore-samd for updated documentation and information on the new 1.6.8-beta release with support for OS X and many updates.

This is a fork from arduino/ArduinoCore-samd on GitHub. This will be used to maintain Arduino support for SAM M0+ boards including the MattairTech MT-D21E and the MT-D11 (see https://www.mattairtech.com/). It adds support for new devices like the L21, C21, and D11. It also adds new clock sources, like a high speed crystal or internal oscillator.

This core is intended to be installed using Boards Manager (see below). To update from a previous version, click on MattairTech SAM M0+ Boards in Boards Manager, then click Update.

Differences from Arduino in Versioning: The MattairTech version number does not correspond to either the IDE version or to the upstream ArduinoCore-samd version. See the CHANGELOG for details on which upstream commits have been merged in to the MattairTech core.

What's New Beta (1.6.8-beta)

See Beta Builds section for installation instructions.

1.6.8-beta-b1:

- Fixed auto-reset not working on some versions of Windows
- Documentation updates

1.6.8-beta-b0:

- Added L21 and C21 support. Improved D11D and D11C support.
  - Use Tools->Microcontroller menu to select mcu.
- Both the core and bootloader have added support for:
  - external high-speed crystal (400KHz - 32MHz) using PLL
  - external 32.768KHz crystal using PLL
  - internal oscillator with USB calibration using DFLL
  - internal oscillator using DFLL in open-loop mode (or 48MHz RC oscillator with C21)
  - PLL_FRACTIONAL_ENABLED and PLL_FAST_STARTUP options
  - The clock source is selectable in the Tools->Clock Source menu
- New Tools->Serial Config menu for selecting different combinations of serial peripherals
- New Tools->Bootloader Size menu allows selection of bootloader size
- New Tools->USB Config menu simplifies USB configuration compared to previous core

- Updated variant.cpp table format for future CCL and GCLK use. See VARIANT_COMPLIANCE_CHANGELOG.
- Updated bootloader.
- Updated bossac upload tool (fixed support for SAML and SAMC)
- New CMSIS-Atmel package (this is different than from Arduino)
- Merged in all changes from upstream through SAMD CORE 1.6.14 (April 2017)

What's New Release (1.6.6)

This is out of date, use the beta for now.

- **1.6.6-mt3:**
  - Fixes compilation with CDC_UART and CDC_ONLY settings
- **1.6.6-mt2:**
  - Changes the default Communication setting to CDC_UART (from CDC_HID_UART)
- **1.6.6-mt1:**
  - New documentation section 'Special Notes'. Please read!
  - Updated ASCII pinouts to be more readable and less ambiguous.
  - Updated the Signed driver for Windows (extras directory) (see CHANGELOG for details)
  - Merged in changes from upstream (see CHANGELOG for details)
  - Fix warnings about deprecated recipe.ar.pattern
  - Merged in changes from upstream SAMD CORE 1.6.2 2015.11.03 (see CHANGELOG for details)

Features Summary

Feature	21J (64 pin)	21G (48 pin)	21E (32 pin)	D11 (24, 20, or 14 pin)
Board Variants	New board coming June, Generic 21J	Arduino Zero, Arduino M0, Generic 21G	MT-D21E, Generic 21E	MT-D11, Generic D11D14AM, Generic D11D14AS, Generic D11C14A
Processor	48 MHz 32-bit ARM Cortex M0+	48 MHz 32-bit ARM Cortex M0+	48 MHz 32-bit ARM Cortex M0+	48 MHz 32-bit ARM Cortex M0+
Flash Memory	Up to 256KB, L21 & C21 have RWW	Up to 256KB, L21 & C21 have RWW	Up to 256KB, L21 & C21 have RWW	16 KB (4KB used by bootloader)
SRAM	Up to 32KB (plus <=8KB LPSRAM on L21)	Up to 32KB (plus <=8KB LPSRAM on L21)	Up to 32KB (plus <=8KB LPSRAM on L21)	4 KB
Digital Pins	52 (51 for L21)	38 (37 for L21)	26 (25 for L21)	24-pin: 21, 20-pin: 17, 14-pin: 11
Analog Inputs	20 channels, 12-bit	14 channels, 12-bit	10 channels, 12-bit	24-pin: 10, 20-pin: 8, 14-pin: 5 (12-bit)
Analog Outputs	One 10-bit (two 12-bit on L21)	One 10-bit (two 12-bit on L21)	One 10-bit (two 12-bit on L21)	One 10-bit
PWM Outputs	18	14	14	8 (6 for 14-pin)
Interrupts	16	16	16	8 (7 for 14-pin)
USB	Full Speed Device and Host (not C21)	Full Speed Device and Host (not C21)	Full Speed Device and Host (not C21)	Full Speed Device
SERCOM	6	6	4	3 (2 for 14-pin)
UART (Serial)	Up to 3	Up to 3	Up to 2	Up to 2
SPI	Up to 3	Up to 2	1	1
I2C (WIRE)	Up to 3	Up to 2	1	1
I2S	D21 only	D21 only	D21 only	Not present
Voltage	1.62V-3.63V (2.7V-5.5V for the C21)	1.62V-3.63V (2.7V-5.5V for the C21)	1.62V-3.63V (2.7V-5.5V for the C21)	1.62V-3.63V
I/O Pin Current	D21: 7mA, L21: 5mA, C21: 6mA@5V	D21: 7mA, L21: 5mA, C21: 6mA@5V	D21: 7mA, L21: 5mA, C21: 6mA@5V	7 mA

Board Variants

Pin configuration and peripheral assignment information is now in the README.md for each board variant. README.md also now includes technical information on the new PinDescription table format.

- MattairTech MT-D21E Rev B (SAMx21Exxx)
- MattairTech MT-D21E Rev A (SAMD21ExxA)
- MattairTech MT-D11 (SAMD11D14AM)
- MattairTech Generic D11C14A
- MattairTech x21J based board (coming June)
- MattairTech Generic D11D14AS (coming soon)
- MattairTech Generic D11D14AM (coming soon)
- MattairTech Generic x21E (coming soon)
- MattairTech Generic x21G (coming soon)
- MattairTech Generic x21J (coming soon)
- Arduino Zero (arduino.cc)
- Arduino M0 (arduino.org)

Pin Configurations

MT-D21E rev A

```
============================= MattairTech MT-D21E (ATsamD21EXXA) ========================
Other  COM    PWM   Analog  INT  Arduino*             Arduino*  INT   PWM     COM   Other
=========================================================================================
                                      -------------------
Xin32                                | A0            RST |                           BOOT
Xout32                               | A1            NC  |
DAC                   *            2 | A2            NC  |
REFA                  *            3 | A3            A31 | 31    *   TCC11         SWDIO*
REFB                  *      *     4 | A4            A30 | 30    *   TCC10         SWDCLK
                      *      *     5 | A5            NC  |
                      *            6 | A6            A28 | 28    *                   LED
VM                    *            7 | A7            A27 | 27    *                   BTNA
             TCC00    *     NMI    8 | A8            A23 | 23    *   TC41     SS
             TCC01    *      *     9 | A9            A22 | 22    *   TC40     MISO
       TX1   TCC02    *           10 | A10           A19 | 19    *            SCK
       RX1   TCC03    *           11 | A11           A18 | 18    *            MOSI
       TX2   TC30            *    14 | A14           A17 | 17    *   TCC21    SCL
       RX2   TC31                 15 | A15           A16 | 16    *   TCC20    SDA
                                     | NC            NC  |
                                     | NC            NC  |
                                     | Vbus          3.3V|     * Button B available on 31.
USB D-       TC50                    | A24-  _____   Vcc |
USB D+       TC51                    | A25+ |     |  Vin |
                                     | Gnd  | USB |  Gnd |
                                      -------------------

* Most pins can be used for more than one function. The port pin number printed
  on the board is also used in Arduino (but without the 'A') for all of the supported
  functions (ie: digitalRead(), analogRead(), analogWrite(), attachInterrupt(), etc.).
* When USB CDC is enabled, Serial refers to SerialUSB, otherwise it refers to Serial1.
* Leave pin A30 floating (or use external pullup) during reset.
* Tone available on TC5. DO NOT connect voltages higher than 3.3V!
```

Tools Menu Additions

Depending on the board variant, different menu options will appear in the Tools menu.

Microcontroller Menu

This menu will appear with boards that have multiple microcontroller options.

Clock Source Menu

There are up to four clock source choices, depending on board variant and microcontroller. They are:

- 32KHZ_CRYSTAL (default)
- HIGH_SPEED_CRYSTAL
- INTERNAL_OSCILLATOR
- INTERNAL_USB_CALIBRATED_OSCILLATOR

See Clock Source section for more information.

Bootloader Size Menu

With the D21, L21, and C21, the bootloader size can be configured as:

- 8KB_BOOTLOADER (default)
- 16KB_BOOTLOADER
- NO_BOOTLOADER

With the D11, the bootloader size can be configured as:

- 4KB_BOOTLOADER (default)
- NO_BOOTLOADER

Choose NO_BOOTLOADER if not using a bootloader (an external programmer will be used for sketch upload).

Serial Config Menu

This menu is used to select different combinations of serial peripherals. This is useful especially for the D11, which has a reduced pin count and number of SERCOMs. It can also be used to reduce FLASH and SRAM usage by selecting fewer UART peripherals, which are instantiated in the core, rather than only when including a library (like SPI and WIRE). Most board variants support two UART as an option.

USB Config Menu

This menu will appear with all microcontrollers except the C21, which does not have USB. The options are:

- CDC_ONLY (default)
- CDC_HID
- WITH_CDC

- HID_ONLY
- WITHOUT_CDC
- USB_DISABLED

Choose an option that best matches your code and library usage. Each option results in a different USB PID. Choose an option with CDC if you want auto-reset to function, or the serial monitor over USB. If CDC is not enabled, Serial will refer to Serial1 instead of SerialUSB. These options can be used to optimize FLASH and SRAM usage by allowing CDC to be disabled (or USB completely disabled).

Clock Source

There are up to four clock source choices, depending on board variant and microcontroller. They are:

- 32KHZ_CRYSTAL (default)
- HIGH_SPEED_CRYSTAL
- INTERNAL_OSCILLATOR
- INTERNAL_USB_CALIBRATED_OSCILLATOR

External 32.768KHz Crystal

The PLL will be used with the 32.768KHz crystal. PLL_FRACTIONAL_ENABLED can be defined, which will result in a more accurate 48MHz output frequency at the expense of increased jitter.

External High-Speed Crystal

HS_CRYSTAL_FREQUENCY_HERTZ must be defined with the external crystal frequency in Hertz. The crystal frequency must be between 400000Hz and 32000000Hz. The PLL will be used. PLL_FRACTIONAL_ENABLED can be defined, which will result in a more accurate 48MHz output frequency at the expense of increased jitter. If PLL_FAST_STARTUP is defined, the crystal will be divided down to 1MHz - 2MHz, rather than 32KHz - 64KHz, before being multiplied by the PLL. This will result in a faster lock time for the PLL, however, it will also result in a less accurate PLL output frequency if the crystal is not divisible (without remainder) by 1MHz. In this case, define PLL_FRACTIONAL_ENABLED as well. By default, PLL_FAST_STARTUP is disabled. PLL_FAST_STARTUP is also useful for USB host mode applications. See datasheet USB electrical characteristics. The crystal frequency must be at least 1000000Hz when PLL_FAST_STARTUP is defined.

Internal Oscillator

The DFLL will be used in open-loop mode, except with the C21 which lacks a DFLL, so the internal 48MHz RC oscillator is used instead. NVM_SW_CALIB_DFLL48M_FINE_VAL is the fine calibration value for DFLL open-loop mode. The coarse calibration value is loaded from NVM OTP (factory calibration values).

Internal Oscillator with USB Calibration

This is available for the D21, D11, or L21. It will also use the DFLL in open-loop mode, except when connected to a USB port with data lines (and not suspended), then it will calibrate against the USB SOF signal. NVM_SW_CALIB_DFLL48M_FINE_VAL is the fine calibration value for DFLL open-loop mode. The coarse calibration value is loaded from NVM OTP (factory calibration values).

Clock Generators

0. MAIN (mcu)
1. XOSC (high speed crystal)
2. OSCULP32K (initialized at reset for WDT on D21 and D11)
3. OSC_HS (the reset default internal RC oscillator is put here at 8MHz, except with C21)

Analog Reference

TODO: more info

- **D21 / D11**
  - AR_INTERNAL1V0
  - AR_INTERNAL_INTVCC0
  - AR_INTERNAL_INTVCC1
  - AR_EXTERNAL_REFA
  - AR_EXTERNAL_REFB
  - AR_DEFAULT (this also uses 1/2 gain on each input)
- **L21**
  - AR_INTREF
  - AR_INTERNAL_INTVCC0
  - AR_INTERNAL_INTVCC1
  - AR_EXTERNAL_REFA
  - AR_EXTERNAL_REFB
  - AR_INTERNAL_INTVCC2
  - AR_INTREF_1V0
  - AR_INTREF_1V1
  - AR_INTREF_1V2
  - AR_INTREF_1V25
  - AR_INTREF_2V0
  - AR_INTREF_2V2
  - AR_INTREF_2V4
  - AR_INTREF_2V5
  - AR_DEFAULT = AR_INTERNAL_INTVCC2
  - AR_INTERNAL1V0 = AR_INTREF (Default INTREF for SAML is 1.0V)
- **C21**
  - AR_INTREF
  - AR_INTERNAL_INTVCC0
  - AR_INTERNAL_INTVCC1
  - AR_EXTERNAL_REFA

    - AR_EXTERNAL_DAC
    - AR_INTERNAL_INTVCC2
    - AR_INTREF_1V024
    - AR_INTREF_2V048
    - AR_INTREF_4V096
    - AR_DEFAULT = AR_INTERNAL_INTVCC2
    - AR_INTERNAL1V0 = AR_INTREF (Default INTREF for SAMC is 1.024V)
- **Common**
    - AR_INTERNAL = AR_INTERNAL_INTVCC0
    - AR_INTERNAL2V23 = AR_INTERNAL_INTVCC0 (2.23V only when Vcc = 3.3V)
    - AR_INTERNAL1V65 = AR_INTERNAL_INTVCC1 (1.65V only when Vcc = 3.3V)
    - AR_EXTERNAL = AR_EXTERNAL_REFA

Chip Specific Notes

SAMD21

- When USB is disabled, pullups will be enabled on PA24 and PA24 to avoid excessive current consumption (<1mA) due to floating pins. Note that it is not necessary to enable pull resistors on any other pins that are floating. Errata: Disable pull resistors on PA24 and PA25 manually before switching to a peripheral.

SAML21

- There are two DACs, DAC0 and DAC1. Both are supported. Because changing the configuration of one DAC requires disabling both, there will be about a 40us period when the second DAC is disabled. Most of this time is due to an errata that requires a delay of at least 30us when turning off the DAC while refresh is on. The L21 DACs have a refresh setting which must be enabled in this core.
- The analog reference has additional options on the L21 and C21. See Analog Reference section.
- On the L21, SERCOM5 is in a low power domain. The Fm+ and HS modes of I2C (wire) are not supported.
- The SAML and SAMC have double-buffered TCs, which are supported in the core.
- The CHANGE and RISING interrupt modes on pin A31 do not seem to work properly on the L21.
- The L21 has two performance levels that affect power consumption. During powerup, the L21 starts at the lowest performance level (PL0). The startup code changes to the highest performance level (PL2) in order to support 48MHz and USB (among other things).
- Two Flash Wait States are inserted for the L21 and C21 (the D21/D11 use one wait state).

SAMC21

- There are two SAR ADCs. Both are supported. The PinDescription table determines the peripheral instance and pin mapping.

- The analog reference has additional options on the L21 and C21. See Analog Reference section.
- The SAML and SAMC have double-buffered TCs, which are supported in the core.
- Two Flash Wait States are inserted for the L21 and C21 (the D21/D11 use one wait state).
- The C21 requires internal pull resistors to be activated on floating pins to minimize power consumption (not needed on D21/D11 or L21).
- The C21 uses the minimum sampling time so that rail-to-rail and offset compensation works. Offset compensation adds 3 ADC clock cycles, so the total is 4 clock cycles. The D21, D11, and L21 use the maximum sampling time.

SAMD11

- The D11D has three SERCOM. The D11C has two sercom (no sercom2).
- When USB is disabled, pullups will be enabled on PA24 and PA24 to avoid excessive current consumption (<1mA) due to floating pins. Note that it is not necessary to enable pull resistors on any other pins that are floating. Errata: Disable pull resistors on PA24 and PA25 manually before switching to a peripheral.

Reducing SRAM/FLASH Usage on the D11

TODO

Differences Between MattairTech and Arduino Cores

- TODO
- Table summarizing which core files are modified and by how much
- Changes due to adding/changing features vs porting to new chip

Random Notes (TODO)

- TONE: TC5 does not exist on the D11. Using TC2 instead (TC1 on the D11C14 as TC2 is not routed to pins). It will conflict with the 2 associated TC analogWrite() pins.
- D21: Enables wakeup capability on pin in case being used during sleep (WAKEUP always enabled on SAML and SAMC)
- All pins (digital and analog) setup in STARTUP mode (enable INEN and set default pull direction to pullup (pullup will not be enabled))
- INEN enabled for both input and output (but not analog)
- pinPeripheral now handles disabling the DAC (if active). Note that on the L21, the DAC output would interfere with other peripherals if left enabled, even if the anaolog peripheral is not selected.
- Pull resistors enabled only if pin attributes allow and only if pin is not configured as output.
- Pull direction (pullup or pulldown) is now set with pinMode only (defaults to pullup if pinMode never called).

Serial Monitor

To print to the Serial Monitor over USB, use 'Serial'. Serial points to SerialUSB (Serial1 and Serial2 are UARTs). Unlike most Arduino boards (ie. Uno), SAMD 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 SAMD 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.

When USB CDC is not enabled, Serial will instead refer to Serial1, which is the first UART.

Code Size and RAM Usage (1.6.5-mt2)

TODO: Update this. Maybe just for D11 and move to D11 Chip Specific Notes.

Sketch and Configuration	MT-D21E (Code + RAM)	MT-D11 (Code + RAM)
Blink (CDC + HID + UART)	7564 + 1524	7452 + 1424
Blink (CDC + UART)	6588 + 1496	6484 + 1396
Blink (CDC Only)	5248 + 1304	5192 + 1300
Blink (UART Only)	3828 + 336	3716 + 236
Blink (No USB or UART)	2472 + 144	2416 + 140
Datalogger (No USB or UART)	10340 + 948	10260 + 944

- 180 bytes of flash can be saved on the MT-D11 by using PIN_MAP_COMPACT (see 'New PinDescription Table' below).
- Datalogger compiled without USB or UART support, but with SPI and SD (with FAT filesystem) support. Serial output was disabled.
- Note that USB CDC is required for auto-reset into the bootloader to work (otherwise, manually press reset twice in quick succession).
- USB uses primarily 3 buffers totaling 1024 bytes. The UART uses a 96 byte buffer. The

banzai() function (used for auto-reset) resides in RAM and uses 72 bytes.

- Any combination of CDC, HID, or UART can be used (or no combination), by using the Tools->Communication menu.

Detailed Memory Usage Output After Compilation

The flash used message at the end of compilation is not correct. The number shown represents the .text segment only. However, Flash usage = .text + .data segments (RAM usage = .data + .bss segments). In this release, two programs are run at the end of compilation to provide more detailed memory usage. To enable this output, go to File->Preferences and beside "Show verbose output during:", check "compilation".

Just above the normal flash usage message, is the output from the size utility. However, this output is also incorrect, as it shows .text+.data in the .text field, but 0 in the .data field. However, the .text field does show the total flash used. The .data field can be determined by subtracting the value from the normal flash usage message (.text) from the value in the .text field (.text+.data). The .bss field is correct.

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).

Installation

Driver Installation

Windows

Prior to core version 1.6.6-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. Note that the Windows 10 generic CDC drivers work as well.

1. If you do not already have the SAM-BA bootloader installed, see below.
2. Download https://www.mattairtech.com/software/MattairTech_CDC_Driver_Signed.zip and unzip into any folder. Note that the Windows 10 generic CDC drivers work as well.
3. Plug in the board. The LED should fade when the bootloader is running (or blink if the test sketch is running).
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 Bossac Standalone below.
6. If you do not already have the test firmware installed (comes preinstalled), see Using Bossac Standalone below.
7. Press the reset button to run the test firmware (if needed). The LED will blink.
8. Windows will detect the board. Point the installer to the above folder to install the sketch driver (if needed).
9. Continue with SAM M0+ Core Installation below.

Linux

0. No driver installation is needed.
1. On some distros, you may need to add your user to the same group as the port (ie: dialout) or set udev rules (See the file https://github.com/mattairtech/ArduinoCore-samd/tree/master/drivers/99-mattairtech-USB-CDC.rules).
2. 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.

3. If you are running modemmanager (ie: Ubuntu), disable it, or use the udev rules file above.
4. Continue with SAM M0+ Core Installation below.

OS X

1. OS X support currently in beta (see below), the following instructions are only for 1.6.6-mtX and below.
2. Only the 256 KB chip variants work with the OS X version of the upload tool, bossac.
3. First, you will need to open boards.txt and change mattairtech_mt_d21e_bl8k.upload.tool to equal arduino:bossac.
4. Open platform.txt and change tools.bossac.path to equal{runtime.tools.bossac-1.6.1-arduino.path}.
5. No driver installation is needed.
6. 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.
7. Continue with SAM M0+ Core Installation below.

SAM M0+ Core Installation

- To update from a previous version, click on MattairTech SAM M0+ Boards in Boards Manager, then click Update.

1. The MattairTech SAM M0+ Core requires Arduino 1.6.7 or above (including 1.8.x).
2. In the Arduino IDE, 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 Arduino SAM M0+ Boards package. Use version 1.6.2 or higher.
7. Install the MattairTech SAM M0+ Boards package.
8. Close Boards Manager, then click Tools->Board->MattairTech MT-D21E (or MT-D11).
9. Select the MCU with the now visible Tools->Microcontroller menu (if present).
10. If you do not already have the bootloader or blink sketch installed, see SAM-BA USB CDC Bootloader below.
11. Plug in the board. The blink sketch should be running.
12. Click Tools->Port and choose the COM port. Note that the board indicated may not match the chosen board*
13. You can now upload your own sketch.

** Currently, with MattairTech boards, USB PIDs are shared across boards (but they are different based on Tools->USB Config). This will result in Tools->Port showing "MattairTech MT-D21E (rev B)" for all MattairTech boards.*

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.

Beta Builds

Periodically, a beta is released for testing.

The beta builds are available through Boards Manager. If you want to install them:

1. Open the **Preferences** of the Arduino IDE.
2. Add this URL https://www.mattairtech.com/software/arduino/beta/package_MattairTech_index.json in the **Additional Boards Manager URLs** field, and click OK.
3. Open the **Boards Manager** (menu Tools->Board->Board Manager...)
4. Install **MattairTech SAM M0+ Boards - Beta build**
5. Select one of the boards under **MattairTech SAM M0+ Beta Build XX** in Tools->Board menu
6. Compile/Upload as usual

The Arduino IDE will notify the user if an update to the beta is available, which can then be installed automatically. Alternatively, if a particular beta is needed, replace the url in step 2 with: https://www.mattairtech.com/software/arduino/beta/package_MattairTech_sam_m0p-${VERSION}-beta-b${BUILD_NUMBER}_index.json where ${VERSION} and ${BUILD_NUMBER} match the beta name as shown in the CHANGELOG (ie: package_MattairTech_sam_m0p-1.6.7-beta-b0_index.json). In this case, the IDE will not notify the user of updates.

New PinDescription Table

Technical information on the new PinDescription table format is now in the README.md that accompanies each board variant. See board variants above.

Note that a new column (GCLKCCL) was added for 1.6.8-beta-b0.

MATTAIRTECH_ARDUINO_SAMD_VARIANT_COMPLIANCE in variant.h is used to track versions. If using board variant files with the old format, the new core will still read the table the old way, losing any new features introduced by the new column. Additionally, new definitions have been added for L21 and C21 support.

Each pin can have multiple functions.

The PinDescription table describes how each of the pins can be used by the Arduino core. Each pin can have multiple functions (ie: ADC input, digital output, PWM, communications, etc.), and the PinDescription table configures which functions can be used for each pin. This table is mainly accessed by the pinPeripheral function in wiring_private.c, which is used to attach a pin to a particular peripheral function. The communications drivers (ie: SPI, I2C, and UART), analogRead(), analogWrite(), analogReference(), attachInterrupt(), and pinMode() all call pinPeripheral() to verify that the pin can perform the function requested, and to configure the pin for that function. Most of the contents of pinMode() are now in pinPeripheral().

Pin Mapping

There are different ways that pins can be mapped. Typically, there is no relation between the arduino pin number used, and the actual port pin designator. Thus, the pcb must be printed with the arduino numbering, otherwise, if the port pin is printed, a cross reference table is needed to find the arduino pin number. However, this results in the least amount of space used by the table. Another method, used by default by the MT-D21E and MT-D11, maps Arduino pin numbers to the actual port pin number (ie: Arduino pin 28 = Port A28). This works well when there is only one port (or if the PORTB pins are used for onboard functions and not broken out). PIO_NOT_A_PIN entries must be added for pins that are used for other purposes or for pins that do not exist (especially the D11), so some FLASH space may be wasted. For an example of both types, see variant.cpp from the MT-D11 variant.

See Board Variants above for more technical information on the PinDescription table.

See WVariant.h for the definitions used in the table.

Possible Future Additions/Changes

- Timer library is currently under development (like TimerOne, plus input capture, plus ??)
- OS X support currently in beta testing
- Reduce SRAM usage by USB endpoint buffers by only allocating endpoints actually used (D11 especially)
- Drivers for MT-D21E optional memory devices (SRAM, FLASH, EEPROM)
- USB Host mode CDC ACM (partially complete; BSD-like license?)
- Features for lower power consumption (library?)
- Reliability and security enhancements
- Enhanced SD card library
- Optional use of single on-board LED as USB activity LED
- MSC (Mass Storage) USB Device Class
- Polyphonic tone
- Wired-AND, Wired-OR for port pins

- High-speed port pin access (IOBUS)
- Libraries for some hardware I plan on using:
    - TFT LCD (CFAF128128B-0145T)
    - Motor controller
    - IR decoder
    - I2S DAC/AMP and I2S MEMS microphone
    - Battery management IC
    - XBee/Xbee Pro devices
    - RS485
    - Several I2C (Wire) sensor devices:
        - Accelerometer/magnetometer (LSM303CTR)
        - Barometer/altimeter (LPS22HBTR)
        - Humidity/temperature
        - Light/color sensor

ChangeLog

The Changelog has moved to a separate file named CHANGELOG. The most recent changes are still in the 'What's New' section above.

Troubleshooting

- **Tools->Port shows wrong board**
    - Currently, with MattairTech boards, USB PIDs are shared across boards (but they are different based on Tools->USB Config). This will result in Tools->Port showing "MattairTech MT-D21E (rev B)" for all MattairTech boards.
- **Tools->USB Config menu**
    - Currently, the Tools->USB Config menu (was Tools->Communications) 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 (all options except CDC_ONLY or USB_DISABLED). This menu is currently needed to select the USB PID that matches the USB device configuration (needed for some versions of Windows). It is also used to control if CDC support is compiled (CDC is always enabled in the stock Arduino core). Auto reset requires CDC to be enabled.
        - Be sure that the Tools->Communications menu matches the sketch and libraries you are compiling.
        - Different combinations of USB devices will result in different COM port assingments in Windows.

- **Incude platform specific libraries**
  - You may need to manually include platform specific libraries such as SPI.h, Wire.h, and HID.h.
- **Errors when compiling, uploading, or burning the bootloader**
  - Be sure to install the Arduino samd core before installing the MattairTech sam m0+ core. If you have problems upgrading the IDE to 1.6.6, you may need to uninstall both the Arduino and MattairTech cores, then re-install in the proper order. Use Arduino core 1.6.2 or above.
- **On Linux, disable modem manager (Ubuntu)**
- **Do not perform a manual auto-reset** (using a terminal program to change baud to 1200)
- **Boards Manager must be opened twice to see some updates** (only applies to some old IDE versions)

SAM-BA USB CDC Bootloader (Arduino Compatible)

The SAM-BA bootloader has both a CDC USB interface, and a UART interface. It is compatible with the Arduino IDE, or it can be used with the Bossac tool standalone. With Arduino, auto-reset is supported (automatically runs the bootloader while the sketch is running) as well as automatic return from reset. The SAM-BA bootloader described here adds to the Arduino version, which in turn is based on the bootloader from Atmel. The Arduino version added several features, including three new commands (Arduino Extended Capabilities) that increase upload speed. The bootloader normally requires 8 KB FLASH, however, a 4 KB version can be used for the D11 chips.

Bossac is a command line utility for uploading firmware to SAM-BA bootloaders. It runs on Windows. Linux, and OS X. It is used by Arduino to upload firmware to SAM and SAM M0+ boards. The version described here adds to the Arduino version (https://github.com/shumatech/BOSSA, Arduino branch), which in turn is a fork from the original Bossa (http://www.shumatech.com/web/products/bossa). It adds support for more SAM M0+ chips (D21, L21, C21, and D11).

Note that only the Arduino or Mattairtech versions of bossac are currently supported for SAM M0+ chips. Neither the stock bossac (or Bossa) nor the Atmel SAM-BA upload tool will work.

Arduino Extended Capabilities:

- X: Erase the flash memory starting from ADDR to the end of flash.
- Y: Write the content of a buffer in SRAM into flash memory.
- Z: Calculate the CRC for a given area of memory.

The bootloader can be started by:

- Tapping reset twice in quick succession (BOOT_DOUBLE_TAP).
- Holding down button A (BOOT_LOAD_PIN) while powering up.
- Clicking 'Upload Sketch' in the Arduino IDE, which will automatically start the bootloader (when CDC is enabled).
- If the application (sketch) area is blank, the bootloader will run.

Otherwise, it jumps to application and starts execution from there. The LED will light during bootloader execution. Note that the 4KB bootloader does not support the Arduino Extended Capabilities. However, BOOT_DOUBLE_TAP does fit into the SAMD11 4KB bootloader.

When the Arduino IDE initiates the bootloader, the following procedure is used:

1. The IDE opens and closes the USB serial port at a baud rate of 1200bps. This triggers a “soft erase” procedure.
2. The first row of application section flash memory is erased by the MCU. If it is interrupted for any reason, the erase procedure will likely fail.
3. The board is reset. The bootloader (which always runs first) detects the blank flah row, so bootloader operation resumes.
4. Opening and closing the port at a baud rate other than 1200bps will not erase or reset the SAM M0+.

See bootloaders/zero/README.md for more technical information on the bootloader.

Bootloader Firmware Installation

Bootloader Installation Using the Arduino IDE

1. If you do not already have the MattairTech SAM M0+ core installed, see SAM M0+ Core Installation above.
2. Plug in the SAM M0+ board. The bootloader must be running to (press reset twice within 500ms).
3. Plug an Atmel ICE into USB, then connect it to the powered SAM M0+ board. A green LED should light on the Atmel ICE.
4. Click Tools->Programmer->Atmel ICE.
5. Click Tools->Board->MattairTech MT-D21E (or whichever board you are using).
6. Click Tools->Microcontroller and select your MCU (if menu present).
7. Click Tools->Burn Bootloader. Ignore any messages about not supporting shutdown or reset.
8. Continue with driver installation above.

A running sketch may interfere with the bootloader installation process. Be sure you are running the existing bootloader or using a blank chip.

Bootloader Installation Using Another Tool (ie: Atmel Studio, openocd)

1. Download the bootloader from https://www.mattairtech.com/software/arduino/SAM-BA-bootloaders-zero-mattairtech.zip.
2. Unzip to any directory. Be sure that a bootloader is available for your particular MCU.
3. Follow the procedures for your upload tool to upload the firmware.
   - Perform a chip erase first. Be sure no BOOTPROT bits are set.
   - Install the binary file to 0x00000000 of the FLASH.
   - You can optionally set the BOOTPROT bits to 8KB (or 4KB for the MT-D11). The Arduino installation method does not set these.
   - You can optionally set the EEPROM bits or anything else. The Arduino installation method uses factory defaults.
4. Continue with driver installation above.

Bootloader Binaries

The bootloaders/zero/binaries directory contains all of the SAM-BAm0+ bootloaders built by the build_all_bootloaders.sh script.

MattairTech Boards

MattairTech boards are all configured with only one interface: SAM_BA_USBCDC_ONLY (except C21, which uses SAM_BA_UART_ONLY). CLOCKCONFIG_CLOCK_SOURCE is set to CLOCKCONFIG_INTERNAL_USB (CLOCKCONFIG_INTERNAL for the C21). Only the main LED is defined. BOOT_LOAD_PIN is not defined, but BOOT_DOUBLE_TAP_ENABLED is.

Arduino/Genuino Boards

Arduino/Genuino boards are all configured with both interfaces. CLOCKCONFIG_CLOCK_SOURCE is set to CLOCKCONFIG_32768HZ_CRYSTAL. All LEDs that are installed for each board are defined (and some have LED_POLARITY_LOW_ON set). BOOT_LOAD_PIN is not defined, but BOOT_DOUBLE_TAP_ENABLED is.

Generic Boards

The generic boards are all configured to minimize external hardware requirements. Only one interface is enabled: SAM_BA_USBCDC_ONLY (except C21, which uses SAM_BA_UART_ONLY). CLOCKCONFIG_CLOCK_SOURCE is set to CLOCKCONFIG_INTERNAL_USB (CLOCKCONFIG_INTERNAL for the C21), so no crystal is required. No LEDs are defined. BOOT_LOAD_PIN is not defined, but BOOT_DOUBLE_TAP_ENABLED is, since it uses the reset pin.

Using Bossac Standalone

TODO: Update https://www.mattairtech.com/software/SAM-BA-bootloader-test-firmware.zip with new chips (L21 and C21).

When using Bossac standalone, you will need to ensure that your application starts at 0x00002000 for 8 KB bootloaders, and 0x00001000 for 4 KB bootloaders. This is because the bootloader resides at 0x00000000. This can be accomplished by passing the following flag to the linker (typically LDFLAGS in your makefile; adjust for your bootloader size):

```
-Wl,--section-start=.text=0x2000
```

You can also use a linker script. See the MattairTech SAM M0+ package for examples. Be sure to generate and use a binary file. Many makefiles are set up to generate an elf, hex, and bin already.

Download Bossac from:

- https://www.mattairtech.com/software/arduino/bossac-1.7.0-mattairtech-1-mingw32.tar.gz (Windows 32 bit and 64 bit)
- https://www.mattairtech.com/software/arduino/bossac-1.7.0-mattairtech-1-x86_64-linux-gnu.tar.gz (Linux 64 bit)
- https://www.mattairtech.com/software/arduino/bossac-1.7.0-mattairtech-1-i686-linux-gnu.tar.gz (Linux 32 bit)
- https://www.mattairtech.com/software/arduino/bossac-1.7.0-mattairtech-1-x86_64-apple-darwin.tar.gz (OS X 64 bit)

Linux 64 bit users can also download Bossa (GUI) and bossash (shell) from:

- https://www.mattairtech.com/software/arduino/Bossa-1.7.0-mattairtech-1-x86_64-linux-gnu.tar.gz (Linux 64 bit)

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

1. Download firmware from https://www.mattairtech.com/software/SAM-BA-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 binary that matches your chip. On the command line (change the binary to match yours):

```
bossac.exe -d --port=COM5 -U true -i -e -w -v Blink_Demo_ATSAMD21E18A.bin -R
```

4. On Linux --port might be /dev/ttyACM0. If the device is not found, remove the --port argument for auto-detection.
5. See http://manpages.ubuntu.com/manpages/vivid/man1/bossac.1.html for details.
6. The board should reset automatically and the sketch should be running.

USB Mass Storage Bootloader

Source code and binaries available at https://github.com/mattairtech/SAMD-MSD-Bootloader.

A USB Mass Storage Class device (MSC or MSD) bootloader can be optionally installed. This will allow programming of the FLASH without an external programmer. Additionally, no special software is required on the host computer. The bootloader occupies the first 16KB of FLASH, leaving the rest for the user firmware. The BOOTPROT fuse bits (2:0) are set 0x01, which will protect the first 16KB of FLASH from internal or external programming (from 0x00000000 to 0x00004000). Note that the MSD bootloader does not use an external crystal, as it uses USB clock recovery (DFLL tuned using the USB SOF signal).

Special Requirements when Compiling Software

- Because the user firmware will begin executing at FLASH byte address 0x00004000, you must pass the following flag to the linker (typically LDFLAGS in your makefile):

  ```
  -Wl,--section-start=.text=0x4000
  ```

- Be sure to generate a binary file. Most makefiles are set up to generate an elf, hex, and bin already. You will need the bin file.
- You will need to rename the binary file to FLASH.BIN.

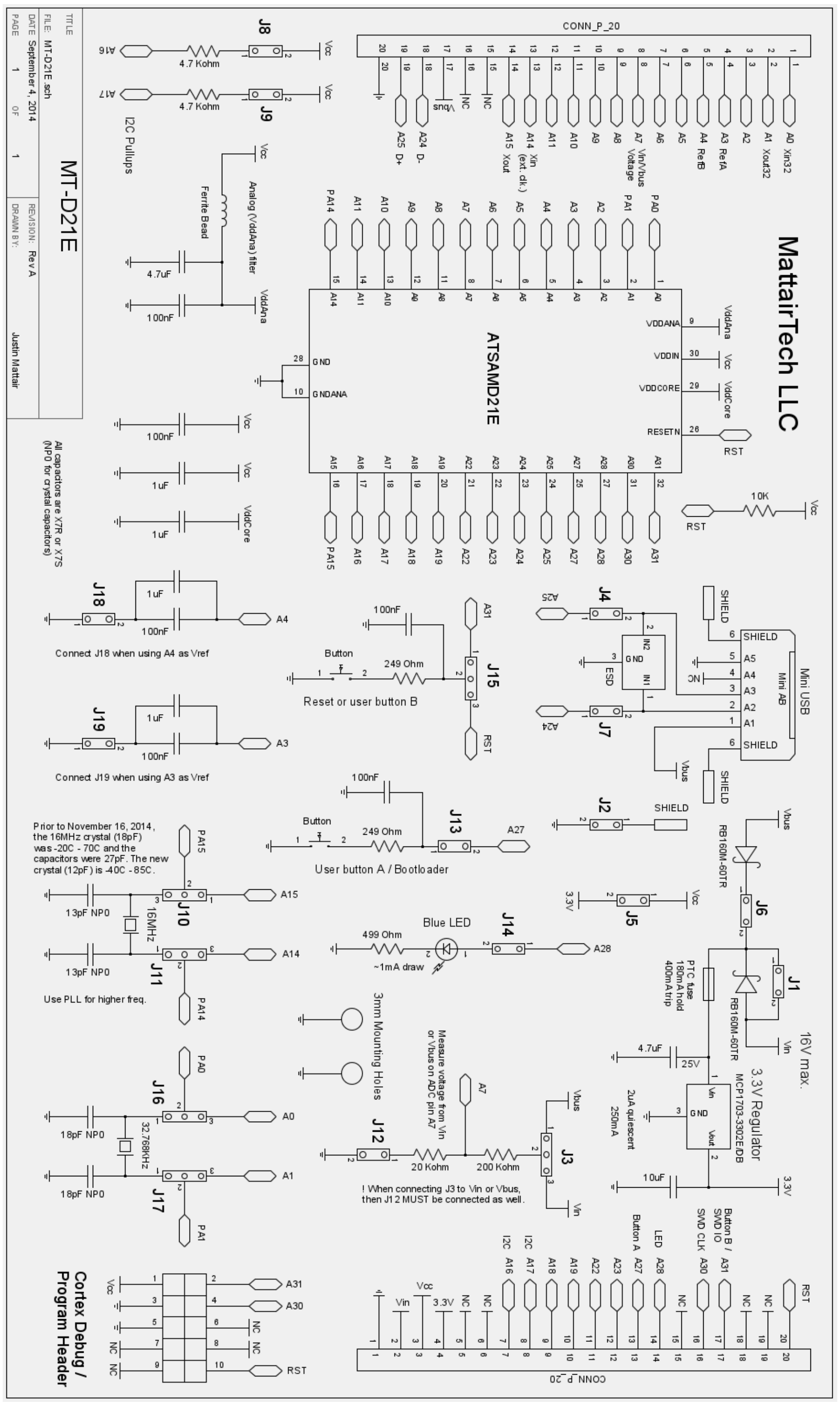
Entering the bootloader and programming the firmware

- Enter the bootloader by pressing button A while powering up the board from USB. Or, hold button A while pressing and releasing button B (if configured as RST). Button A must be connected to pin A27 via solder jumper J13 (this will already be soldered if you ordered the bootloader option). Note that when no user firmware is installed, the bootloader will not automatically run, so you must always use the bootloader button. When the bootloader is run for the first time, the host operating system may take a small amount of time to install drivers. Drivers are already included with the OS, so there is nothing more to download. Once loaded, the LED will begin blinking at 2Hz.
- Mount the “FLASH disk” if it is not mounted automatically. The only file on the entire volume will be FLASH.BIN. This file represents the entire FLASH contents and will always exist. The file date will always be the same upon mounting (2/14/1989). You can read this file simply by copying it to your hard drive. It will include the installed firmware plus 0xFF for the remainder of the file (up to the end of the FLASH).
- Program the FLASH by copying your new FLASH.BIN over the existing copy on the “FLASH disk”. On Windows, you can do this with a file manager. On OS-X (and possibly Linux), you will need to use the cp command, which should already be present. Open up a console (Terminal on OS-X) and type (adjust for your system):

  ```
  cp FLASH.BIN '/run/media/cygnus/MT-D21E MSD'
  ```

- Be sure to unmount the volume before running your new firmware, so that any disk caches are flushed.
- To run your firmware, simply reset or cycle power without pressing button A.
- Technical notes: The startup portion of the bootloader will run prior to executing your firmware. This startup code will enable the button A pullup resistor, wait 8ms for the debouncing capacitor to charge, then test the state of the button. If it is not pressed, the user firmware will be executed as follows:
    - The stack pointer location will be rebased to 0x00004000
    - The interrupt vector table will be rebased to (0x00004000 & SCB_VTOR_TBLOFF_Msk)
    - A jump will be perfomed to the user firmware reset vector.

Schematic



Fuse and Lock Settings

With SAM-BA (CDC) Bootloader

Both the bootloader and the blink sketch were pre-installed by using the Arduino IDE. Neither the region lock bits or the security bit is set. The fuses are left at default settings.

With Mass Storage Bootloader

The Mass Storage Bootloader was pre-installed with the following commands (ATSAMD21E17A shown):

```
atprogram -t atmelice -i SWD -d atsamd21e17a -cl 500khz program -c --verify -f
c:\msd_bootloader_128_flash.hex
atprogram -t atmelice -i SWD -d atsamd21e17a -cl 500khz write -fs -o 0x00804000 --values f9
```

Neither the region lock bits or the security bit is set. The three BOOTPROT fuse bits (2:0) are set to 0x01 (16KB). The blink program (compiled with an offset of 0x00004000) was then installed using the Mass Storage Bootloader.

Without Bootloader

The Blink program was pre-installed with the following commands (ATSAMD21E17A shown):

```
atprogram -t atmelice -i SWD -d atsamd21e17a -cl 500khz program -c --verify -f
c:\MT_D21E_Blink_128_no_offset_flash.hex
```

Neither the region lock bits or the security bit is set. The fuses are left at default settings.

Blink Demo

With SAM-BA (CDC) Bootloader

The blink sketch comes pre-installed using the Arduino IDE.

With MSD Bootloader or Without Bootloader

A demo program comes pre-installed. It simply blinks the LED at 1Hz using an internal clock source. The hex files can be found on the MT-D21E product page at https://www.mattairtech.com/. The blink demo was compiled using the Atmel Standalone Toolchain for Linux. It makes use of Atmel Software Framework (ASF) so it is rather large for a blink program. I can send the source upon request. I will post source if I ever recompile a simpler version that does not depend on ASF.

Troubleshooting / FAQ

- On October 6, 2014, the old PTC fuse was replaced with a 180mA hold (400mA trip) 16V PTC fuse.
- Prior to November 16, 2014, the 16MHz crystal (18pF) was -20C - 70C and the capacitors were 27pF. The new crystal (12pF) is -40C – 85C and the new capacitors are 13pF.
- Prior to January 31, 2016, there was a documentation error regarding J16 and J17, the solder jumpers associated with the 32.768KHz crystal. The image of the PCB bottom shows BOTH crystals disconnected (pins routed to the main headers). The text incorrectly indicated that J16 and J17 were set such that the 32.768KHz crystal was connected. I wrote the text with the intent to use an image with the 32.768KHz connected (SAM-BA bootloader configuration), but I used the wrong image. Note that if the 32.768KHz crystal is disconnected, the crystal oscillator circuitry may still run at around 32KHz, but it will be unreliable and very inaccurate (USB may work intermittently).

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 Atmel ARM in general can be found at http://www.at91.com/.

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

Legal

Copyright / Licenses

Arduino core files:

This core has been developed by Arduino LLC in collaboration with Atmel.
This fork developed by Justin Mattair of MattairTech LLC.

Bootloader files:

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.

Appendix A: Precautions

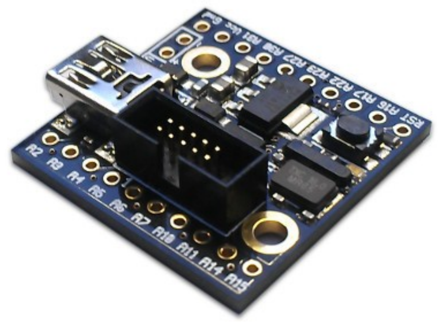
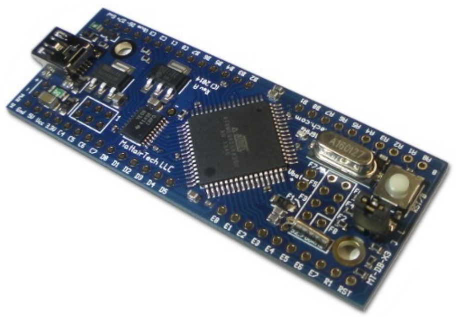
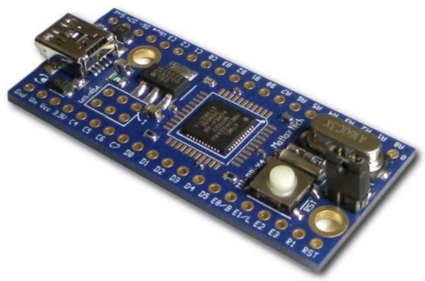
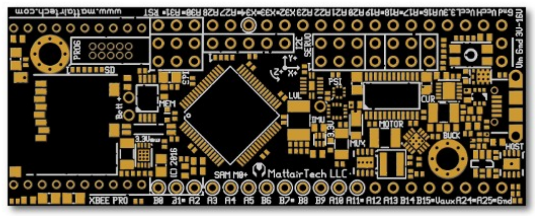
CAUTION
Do not change power configuration, or solder any jumper while unit is powered. Do not short Vin, Vbus, 3.3V, or ground to each other (ie: solder jumpers on bottom shorting on clipped lead).

CAUTION
Higher regulator input voltages mean larger voltage drops and thus higher thermal dissipation for a given amount of current. Be sure to limit current consumption to prevent excessive heat when using higher voltages and/or currents. The regulator will enter thermal shutdown if it gets too hot. All capacitors are X7R, X7S, or NP0, so they can deal with the higher temperatures of the regulator. Note that the PTC fuse is located near the regulator, so high temperatures will lower the PTC trip and hold currents.

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 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
The MT-D21E contains static sensitive components. Use the usual ESD procedures when handling.

Appendix B: Other MattairTech Products

 ●	**MT-D11 USB ARM Cortex M0+ board** ● ATSAMD11D14AM (24-pin) ● 16KB FLASH, 4KB SRAM ● Onboard 3.3V, 250mA LDO regulator (2uA quiescent) ● 16MHz and 32.768KHz crystals ● USB connector (power by USB or external up to 16V) ● Blue LED, 10-pin Cortex header, button, I2C pullups ● USB CDC Bootloader (no programmer required) ● Arduino 1.6.5+ support (core and bootloader)
 ●	**MT-DB-X3 USB AVR XMEGA board** ● XMEGA A3U, A3BU, C3, and D3 (64-pin) ● 32KB - 384KB FLASH, 4KB – 32KB SRAM ● 3.3V 250mA regulator (2uA quiescent current) ● Optional 5V 500mA regulator (23uA quiescent current) ● Optional auto-direction sensing level shifter ● 16MHz and 32.768KHz crystals, optional coin cell holder ● LED, boot jumper, PDI header, button, TWI pullups ● USB DFU bootloader preinstalled (except D variant)
 ●	**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
● 	**D21J / L21J / C21J dev board coming soon!** ● Choice of SAMD, SAML, or SAMC chips ● XBEE / XBEE Pro socket (supports LTE modems) ● MicroSD card slot / device and host mode USB ● Motor controller / relay driver, up to 2.4A, 2.7V-16V ● 2A buck converter with input up to 16V, power mux ● Two 3.3V linear regulators (250mA and 1.5A w/enable) ● SPI Serial memory (128KB SRAM with battery backup support on board, 64KB EEPROM, or 512KB FLASH) ● 3-axis accelerometer / 3-axis gyroscope, pressure sensor / temperature sensor, current / voltage measure