

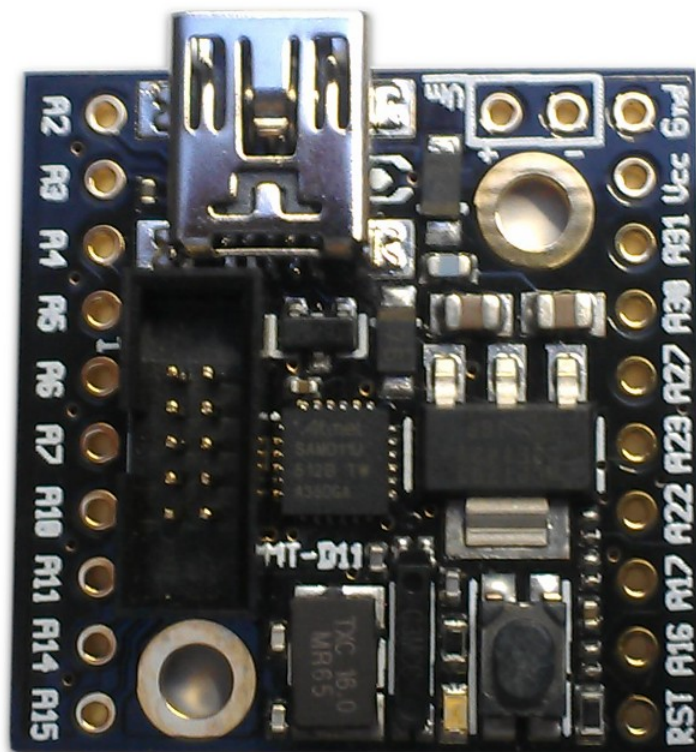
## Table of Contents

<b>Overview.....</b>	<b>4</b>
Introduction.....	4
Board Features.....	5
ATSAMD11D14A Features.....	6
<b>MT-D11 Hardware.....</b>	<b>7</b>
Top View / Pinout.....	7
Main Header Pins (Power).....	8
Main Header Pins (Signal).....	9
Solder Jumpers.....	10
<b>MattairTech Arduino SAM M0+ Core.....</b>	<b>11</b>
What's New Beta (1.6.8-beta).....	11
What's New Release (1.6.6).....	12
Features Summary.....	13
Board Variants.....	14
Pin Configurations.....	15
Tools Menu Additions.....	16
Microcontroller Menu.....	16
Clock Source Menu.....	16
Bootloader Size Menu.....	16
Serial Config Menu.....	16
USB Config Menu.....	16
Clock Source.....	17
External 32.768KHz Crystal.....	17
External High-Speed Crystal.....	17
Internal Oscillator.....	17
Internal Oscillator with USB Calibration.....	17
Clock Generators.....	18
Analog Reference.....	18
Chip Specific Notes.....	19
SAMD21.....	19
SAML21.....	19
SAMC21.....	19
SAMD11.....	20
Reducing SRAM/FLASH Usage on the D11.....	20
Differences Between MattairTech and Arduino Cores.....	20
Random Notes (TODO).....	20
Serial Monitor.....	21
Installation.....	23
Driver Installation.....	23
Windows.....	23
Linux.....	23
OS X.....	24

---

SAM M0+ Core Installation.....	24
Uploading the First Sketch.....	25
Beta Builds.....	25
New PinDescription Table.....	25
Possible Future Additions/Changes.....	26
ChangeLog.....	27
Troubleshooting.....	27
<b>SAM-BA USB CDC Bootloader (Arduino Compatible).....</b>	<b>29</b>
Bootloader Firmware Installation.....	30
Bootloader Installation Using the Arduino IDE.....	30
Bootloader Installation Using Another Tool (ie: Atmel Studio, openocd).....	30
Bootloader Binaries.....	30
MattairTech Boards.....	30
Arduino/Genuino Boards.....	31
Generic Boards.....	31
Using Bossac Standalone.....	31
<b>Schematic.....</b>	<b>33</b>
<b>Troubleshooting / FAQ.....</b>	<b>34</b>
<b>Support Information.....</b>	<b>34</b>
<b>Legal.....</b>	<b>35</b>
<b>Appendix A: Precautions.....</b>	<b>38</b>
<b>Appendix B: Other MattairTech Products.....</b>	<b>39</b>

## Overview



## Introduction

The MT-D11 is a development board for the 24-pin Atmel SAMD11D14AM 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 a button. A USB CDC bootloader (Atmel SAM-BA) is 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+. The Cortex debug header (10-pin, 50-mil) can be used with an external debugger/programmer. The board has 20 main dual inline header pins with 100 mil pin spacing and 800 mil row spacing which allows for mounting on a breadboard or perfboard. There are 2 3mm mounting holes. The PCB measures approx. 1.02" x 1.02" x 0.062" (26mm x 26mm x 1.6mm).

### ***Board Features***

- **Atmel SAM D11 32-bit ARM Cortex M0+ microcontroller**
  - SAMD11D14AM
  - Up to 48MHz
  - 16KB in-system self-programmable Flash
  - 4KB 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 configurable for reset or general use (pin A15) with debouncing**
  - Use as reset (default), with bootloader entry via reset double-tap
  - Use as a general purpose button, with bootloader entry by holding during powerup
- **Two 4.7Kohm resistors can be connected to pins A22 and A23 for use with I2C**
- **USB SAM-BA (CDC) bootloader**
  - 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)**
- Mini USB connector
- ESD protection on USB D+ and D- lines
- Powered by USB or external power source (up to 16V) on Vin
- 100nF capacitor can be enabled for pin A3 for use with external reference
- **9 solder jumpers on PCB bottom for configuration flexibility**
- All PORT pins routed to headers (except crystal pins A8 and A9)
- 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. 1.02" x 1.02" (26mm x 26mm) and 0.062" (1.6mm) thick.

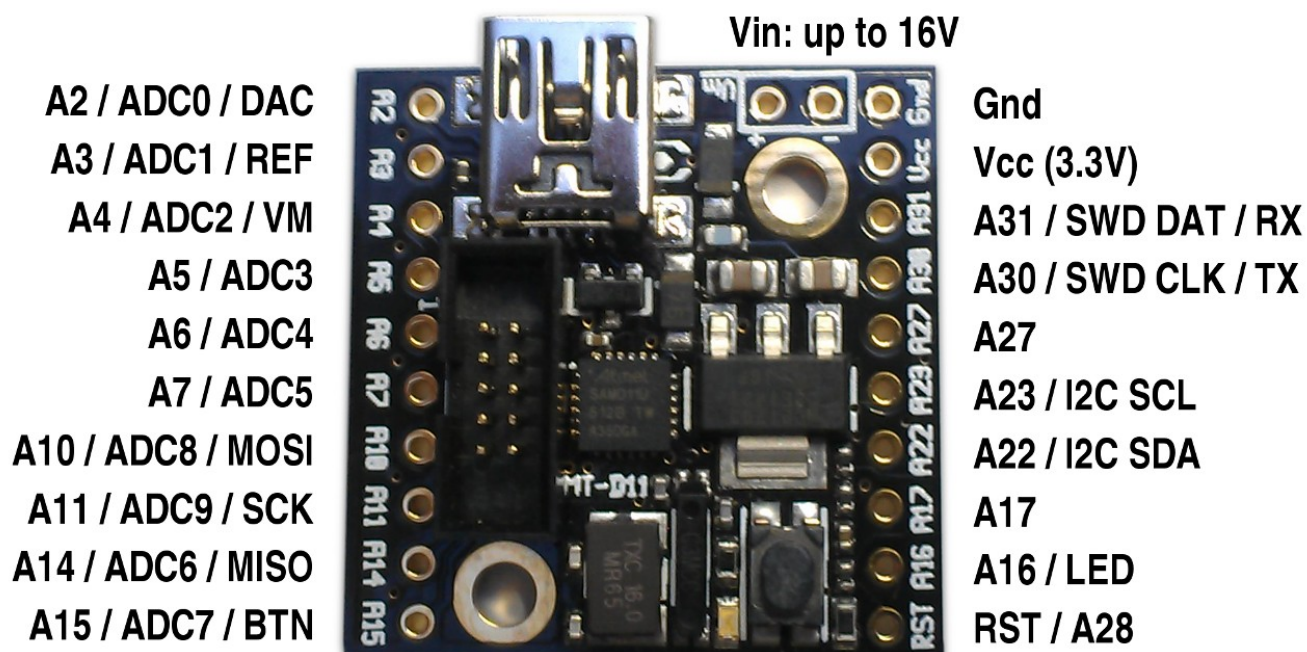
### ***ATSAMD11D14A Features***

- Processor
  - ARM Cortex-M0+ CPU running at up to 48MHz
    - Single-cycle hardware multiplier
    - Micro Trace Buffer
- Memories
  - 16KB in-system self-programmable Flash
  - 4KB 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)
  - 8 external interrupts
  - One non-maskable interrupt
  - Two-pin Serial Wire Debug (SWD) programming, test and debugging interface
- Low Power
  - Idle and standby sleep modes
  - SleepWalking peripherals
- Peripherals
  - 6-channel Direct Memory Access Controller (DMAC)
  - 6-channel Event System
  - Two 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
  - One 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 between complementary output
    - 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 device function
    - Eight endpoints
    - Can run from internal RC oscillator
  - Up to three Serial Communication Interfaces (SERCOM), each configurable to operate as either:
    - USART with full-duplex and single-wire half-duplex configuration
    - I2C Bus up to 3.4MHz
    - SMBUS/PMBUS
    - SPI
    - LIN slave
  - 12-bit, 350ksps Analog-to-Digital Converter (ADC) with up to 10 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)
    - Up to 72-channel capacitive touch and proximity sensing
- I/O
  - Up to 22 programmable I/O pins
- Packages
  - 24-pin QFN
  - 20-pin SOIC
  - 14-pin SOIC
- Operating Voltage
  - 1.62V – 3.63V



## MT-D11 Hardware

### Top View / Pinout



===== MattairTech MT-D11 (ATsamD11D14AM) =====										
Other	COM	PWM	Analog	INT	Arduino*	Arduino*	INT	PWM	COM	Other
=====										
DAC			*		A2	USB	Gnd			
REF			*		A3		Vcc			
	TCC00		*	*	A4	-----	A31	31	*	TC21
	TCC01		*	*	A5		A30	30		TC20
	TCC02		*		A6		A27	27	*	
	TCC03		*		A7		A23	23		
MOSI / TX2			*	*	A10		A22	22	*	
SCK / RX2			*		A11		A17	17		TC11
MISO			*	NMI	A14		A16	16	*	TC10
BTN	SS		*	*	A15		RST			
										LED
										BOOT

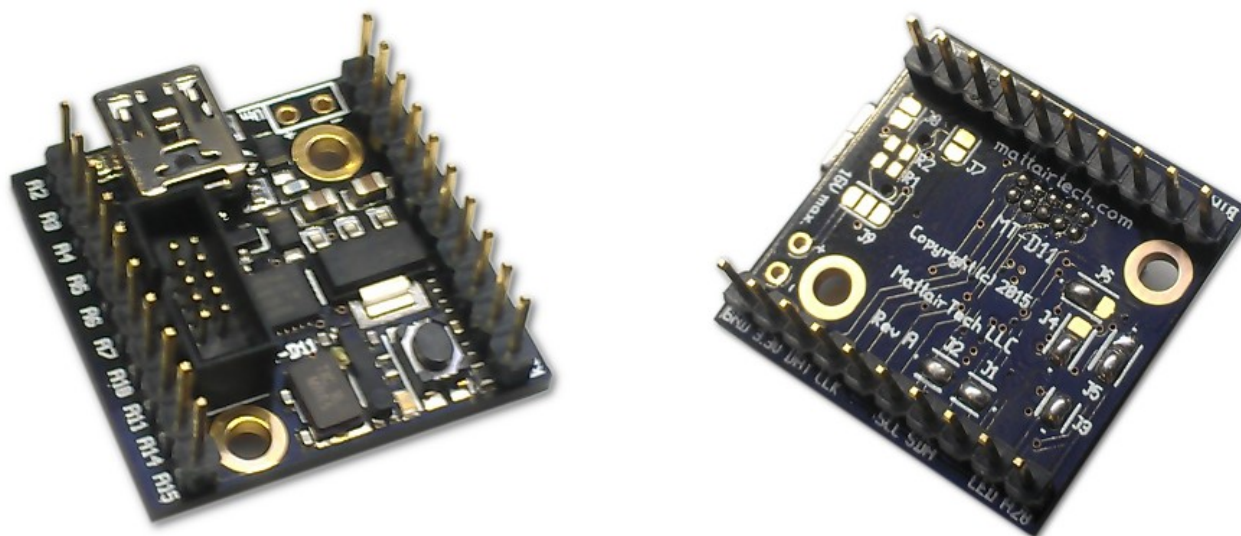
- \* Most pins can be used for more than one function. When using `PIN_MAP_STANDARD`, the port pin number printed on the board is also used in Arduino (but without the 'A') for all of the supported functions (ie: `digitalRead()`, `analogRead()`, `analogWrite()`, etc.). When using `PIN_MAP_COMPACT`, the Arduino numbering is sequential starting from 0 at the top left pin (A2). `PIN_MAP_COMPACT` uses less RAM.
- \* 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 TC2. DO NOT connect voltages higher than 3.3V!

**Main Header Pins (Power)**

Pin	Description
Gnd (2)	Ground
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. Vin voltage can be measured on pin A4 by mounting two 0603 resistors on pads R1 and R2 and setting J9 toward the edge of the board. Setting J9 toward the interior of the board will connect the resistor divider to Vbus instead. The resistor divider will pull pin A4 to near ground level when Vin/Vbus is disconnected. Because of a small leakage current from the schottky diode, a small voltage should be interpreted as Vin/Vbus disconnected. Vbus is connected directly to the Vbus pin (5V) of the USB connector. It is routed through a schottky diode to the regulator input circuitry.
Vcc	This pin is connected to the output of the onboard 3.3V regulator. There is a 10uF capacitor on this output. It also connects to the Vcc pin on the microcontroller, the Cortex debug header Vcc pin, the reset pullup, and the TWI pullup resistors.

**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, 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)**

<b>Pin</b>	<b>Description</b>
<b>A2 / Vout</b>	This can be used for ADC, DAC, or GPIO.
<b>A3 / REF</b>	This can be used for ADC or GPIO. Additionally, jumper J7 can be set to enable a 100nF capacitor so that the pin can be used with an external voltage reference.
<b>A4 / VM</b>	This can be used for ADC, Sercom, Timer, or GPIO. Additionally, it can be connected to a voltage divider (not mounted) to measure either Vin or Vbus. See Vin above.
<b>A5, A6, A7</b>	These can be used for ADC, Sercom, Timer, or GPIO.
<b>A10, A11, A14</b>	These can be used for ADC, Sercom, Timer, or GPIO. In Arduino, these can be used as SPI pins.
<b>A15 / Button</b>	This can be used for ADC, Sercom, Timer, or GPIO. This pin can optionally be connected to the Button through jumper J5. 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 used for bootloader entry by pressing twice in quick succession when connected to reset, or by holding during powerup when connected to pin A15. When connected to pin A15, the button can be used for general purposes.
<b>A16 / LED</b>	This can be used for Sercom, Timer, or GPIO. By default, this pin is connected to a blue LED through jumper J3 and a 499ohm resistor. The LED circuit consumes around 1mA. Drive the pin high to turn on the LED.
<b>A17</b>	This can be used for Sercom, Timer, or GPIO.
<b>A22 / SDA, A23 / SCL</b>	These can be used for Sercom, Timer, or GPIO. By default, jumpers J1 and J2 are enabled, which connects two 4.7Kohm pullup resistors for use with I2C. In Arduino, these can be used as Wire pins.
<b>A27</b>	This can be used for GPIO.
<b>A30 / CLK</b>	This can be used for Sercom, Timer, or GPIO. Additionally, this pin is connected to the Cortex debug header where it is used as SWD CLK.
<b>A31 / DAT</b>	This can be used for Sercom, Timer, or GPIO. Additionally, the pin is routed to the Cortex debug header where it is used as SWD IO.
<b>RST / A28</b>	RST connects to the reset pin of the microcontroller, optionally to the Button through jumper J5 (note that this button can also be for general purpose use; see A15 above), and to the Cortex debug header. A 10K pullup resistor is connected as well.
<b>Cortex Debug Header</b>	This 10-pin, 50-mil header can be connected to an external programmer/debugger.

### Solder Jumpers

<i>Jumper</i>	<i>Description</i>
<b>J1: I2C pullup resistor</b>	Close J1 to connect pin A22 (SDA) through a 4.7Kohm resistor to Vcc for use with I2C.
<b>J2: I2C pullup resistor</b>	Close J2 to connect pin A23 (SCL) through a 4.7Kohm resistor to Vcc for use with I2C.
<b>J3: LED enable</b>	This jumper connects pin A16 to a blue LED through a 499ohm resistor. The LED circuit consumes around 1mA. Drive the pin high to turn on the LED.
<b>J4: Crystal selection</b>	J4 and J6 determine whether microcontroller pins A9 and A8 (respectively) connect to the 32.768KHz crystal (default), or to the 16MHz crystal. The dot on the silkscreen indicates 32.768KHz.
<b>J5: Button function selection</b>	This jumper connects button BTN to either the RST pin (default) or to pin A15 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. The button can be used for bootloader entry by pressing twice in quick succession when connected to reset, or by holding during powerup when connected to pin A15. When connected to pin A15, the button can be used for general purposes. The button can be completely disconnected by removing solder from all three pads. The dot on the silkscreen indicates connection to pin A15. Note that SPI SS is connected to A15 as well. When using SPI, SS must be kept high, so the button cannot be used in this case (or reassign the SS pin in code).
<b>J6: Crystal selection</b>	J4 and J6 determine whether microcontroller pins A9 and A8 (respectively) connect to the 32.768KHz crystal (default) or to the 16MHz crystal. The dot on the silkscreen indicates 32.768KHz.
<b>J7: REF capacitor</b>	When using pin A3 as REF, close J7 to enable a 100nF capacitor from A3 to ground.
<b>J8: USB Shield Ground</b>	This can be closed to connect the USB shield to ground. The USB specification calls for the 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.
<b>J9: Voltage divider input</b>	Vin voltage can be measured on pin A4 by mounting two 0603 resistors on pads R1 and R2 and setting J9 toward the edge of the board. Setting J9 toward the interior of the board will connect the resistor divider to Vbus instead. The resistor divider will pull pin A4 to near ground level when Vin/Vbus is disconnected. Because of a small leakage current from the schottky diode, a small voltage should be interpreted as Vin/Vbus 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-D11

===== MattairTech MT-D11 (ATsamD11D14AM) =====												
Other	COM	PWM	Analog	INT	Arduino*			Arduino*	INT	PWM	COM	Other
=====												
DAC			*		A2	USB	Gnd					
REF			*		A3		Vcc					
		TCC00	*	*	A4	-----	A31	31	*	TC21	RX1	SWDIO
		TCC01	*	*	A5		A30	30		TC20	TX1	SWDCLK
		TCC02	*		A6		A27	27	*			
		TCC03	*		A7		A23	23			SCL	
	MOSI / TX2		*	*	A10		A22	22	*		SDA	
	SCK / RX2		*		A11		A17	17		TC11		
	MISO		*	NMI	A14		A16	16	*	TC10		LED
BTN	SS		*	*	A15		RST					BOOT
=====												

\* Most pins can be used for more than one function. When using PIN\_MAP\_STANDARD, 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(), etc.). When using PIN\_MAP\_COMPACT, the Arduino numbering is sequential starting from 0 at the top left pin (A2). PIN\_MAP\_COMPACT uses less RAM.

\* 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 TC2. 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 PA25 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 PA25 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 analog 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.

<b>Sketch and Configuration</b>	<b>MT-D21E (Code + RAM)</b>	<b>MT-D11 (Code + RAM)</b>
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](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](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 assignments in Windows.

- **Include 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 flash 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](http://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](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](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](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_ATSAM21E18A.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.

## Schematic



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

Justin Mattair  
MattairTech LLC  
PO Box 1079  
Heppner, OR 97836 USA  
[justin@mattair.net](mailto: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.

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

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

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

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

#### Bootloader files:

*Portions of this code are copyright (c) 2009-2015 Justin Mattair ([www.mattairtech.com](http://www.mattairtech.com))*

*Portions of this code are copyright © 2003-2014, Atmel Corporation (<http://www.atmel.com/>):*

```
/**
 * \file
 *
 * \brief User Interface
 *
 * Copyright (c) 2014 Atmel Corporation. All rights reserved.
 *
 * \asf_license_start
 *
 * \page License
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 *
 * 1. Redistributions of source code must retain the above copyright notice,
 *    this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright notice,
 *    this list of conditions and the following disclaimer in the documentation
```



```

*   and/or other materials provided with the distribution.
*
* 3. The name of Atmel may not be used to endorse or promote products derived
*   from this software without specific prior written permission.
*
* 4. This software may only be redistributed and used in connection with an
*   Atmel microcontroller product.
*
* THIS SOFTWARE IS PROVIDED BY ATMEL "AS IS" AND ANY EXPRESS OR IMPLIED
* WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
* MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT ARE
* EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR
* ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN
* ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
* POSSIBILITY OF SUCH DAMAGE.
*
* \asf_license_stop
*
*/

```

Portions of this code are Copyright (C) 2009-2012 ARM Limited. All rights reserved.

```

* @note
* Copyright (C) 2009-2012 ARM Limited. All rights reserved.
*
* @par
* ARM Limited (ARM) is supplying this software for use with Cortex-M
* processor based microcontrollers. This file can be freely distributed
* within development tools that are supporting such ARM based processors.
*
* @par
* THIS SOFTWARE IS PROVIDED "AS IS". NO WARRANTIES, WHETHER EXPRESS, IMPLIED
* OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF
* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE.
* ARM SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL, OR
* CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.

```

ATSAMD11D14A Features (page 5) taken from Atmel datasheet.

### ***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, Vcc, 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, 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

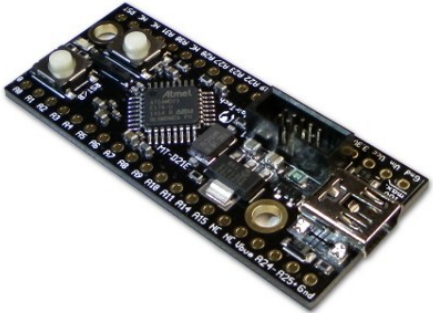
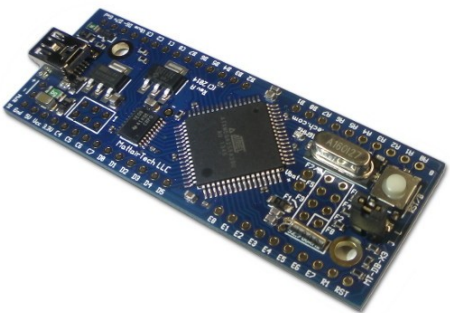
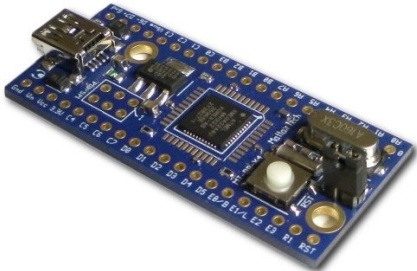
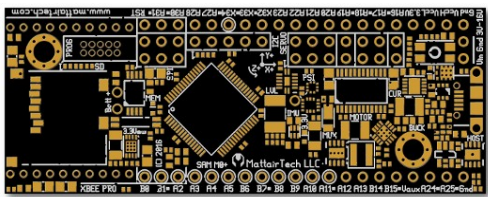
Normally, power is supplied from Vin or Vbus.

However, it is possible to disconnect the regulator and supply an externally regulated voltage on the Vcc pin. 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-D11 contains static sensitive components.  
Use the usual ESD procedures when handling.

## Appendix B: Other MattairTech Products

	<p><b>MT-D21E USB ARM Cortex M0+ board</b></p> <ul style="list-style-type: none"> <li>• ATSAM21E17A or ATSAM21E18A (32-pin)</li> <li>• 128KB/256KB FLASH, 16KB/32KB SRAM</li> <li>• Onboard 3.3V, 250mA LDO regulator (2uA quiescent)</li> <li>• 16MHz and 32.768KHz crystals</li> <li>• USB connector (power by USB or external up to 16V)</li> <li>• Blue LED, 10-pin Cortex header, 2 buttons, I2C pullups</li> <li>• USB CDC or MSD Bootloader (no programmer required)</li> <li>• Arduino 1.6.5+ support (core and bootloader)</li> </ul>
	<p><b>MT-DB-X3 USB AVR XMEGA board</b></p> <ul style="list-style-type: none"> <li>• XMEGA A3U, A3BU, C3, and D3 (64-pin)</li> <li>• 32KB - 384KB FLASH, 4KB - 32KB SRAM</li> <li>• 3.3V 250mA regulator (2uA quiescent current)</li> <li>• Optional 5V 500mA regulator (23uA quiescent current)</li> <li>• Optional auto-direction sensing level shifter</li> <li>• 16MHz and 32.768KHz crystals, optional coin cell holder</li> <li>• LED, boot jumper, PDI header, button, TWI pullups</li> <li>• USB DFU bootloader preinstalled (except D variant)</li> </ul>
	<p><b>MT-DB-X4 USB AVR XMEGA board</b></p> <ul style="list-style-type: none"> <li>• ATxmega128A4U USB XMEGA AVR</li> <li>• 128KB FLASH, 8KB SRAM, 2KB EEPROM</li> <li>• 3.3V LDO regulator (low quiescent current)</li> <li>• 16MHz and 32.768KHz crystals</li> <li>• LED, boot jumper, PDI header</li> <li>• Reset button, mounting holes</li> <li>• USB DFU bootloader preinstalled</li> </ul>
	<p><b>D21J / L21J / C21J dev board coming soon!</b></p> <ul style="list-style-type: none"> <li>• Choice of SAMD, SAML, or SAMC chips</li> <li>• XBEE / XBEE Pro socket (supports LTE modems)</li> <li>• MicroSD card slot / device and host mode USB</li> <li>• Motor controller / relay driver, up to 2.4A, 2.7V-16V</li> <li>• 2A buck converter with input up to 16V, power mux</li> <li>• Two 3.3V linear regulators (250mA and 1.5A w/enable)</li> <li>• SPI Serial memory (128KB SRAM with battery backup support on board, 64KB EEPROM, or 512KB FLASH)</li> <li>• 3-axis accelerometer / 3-axis gyroscope, pressure sensor / temperature sensor, current / voltage measure</li> </ul>