Pololu A-Star 32U4 User's Guide

1. Overview

The Pololu A-Star 32U4 microcontroller boards are general-purpose programmable modules based on Atmel's ATmega32U4 AVR microcontroller, which has 32 KB of flash program memory, 2.5 KB of RAM, and built-in USB functionality. Each A-Star (abbreviated A*) adds onboard components and connectors that support the microcontroller and make it easier to use. The boards feature USB interfaces and ship with a preloaded Arduino-compatible bootloader, and we provide a software add-on that enables them to be easily programmed from the Arduino environment. The following sections of this user's guide discuss each A-Star in more detail.

A **USB A to Micro-B cable** [http://www.pololu.com/product/2072] (not included) is required to connect an A-Star 32U4 to a computer.

Features

- Programmable 16 MHz Atmel ATmega32U4 AVR microcontroller
  - 32 KB flash (4 KB used by bootloader, leaving 28 KB available for user program by default)
  - 2.5 KB SRAM
  - 1 KB EEPROM
  - Native full-speed USB (12 Mbps)
- Preloaded with Arduino-compatible bootloader
- Can be powered from USB or external source regulated to 5 V by onboard regulator
- Reverse-voltage protection on external power input
- 6-pin ISP header for use with an **external programmer** [http://www.pololu.com/product/1300]

A-Star comparison table
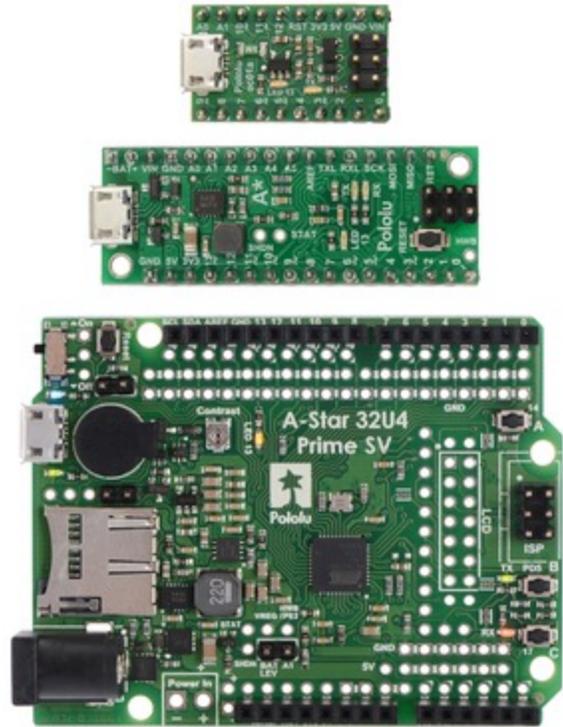
Size comparison of the Pololu A-Star 32U4 family: A-Star 32U4 Micro (top), A-Star 32U4 Mini SV (middle), and A-Star 32U4 Prime SV (bottom).

	A-Star 32U4 Micro	A-Star 32U4 Mini ULV	A-Star 32U4 Mini LV	A-Star 32U4 Mini SV	A-Star 32U4 Prime LV	A-Star 32U4 Prime SV
Microcontroller:	ATmega32U4	ATmega32U4			ATmega32U4	
User I/O lines:	18	26			26	
PWM outputs:	7	7			7	
Analog inputs:	8	12			12	
Ground access points:	2	4			43	
User LEDs:	2	3			3	
User pushbuttons:	—	—			3	
Reset button:		✓			✓	
Power switch:					✓	
Buzzer option:					✓	
microSD option:					✓	
LCD option:					✓	
Operating voltage:	5.5 V to 15 V	0.5 V to 5.5 V	2.7 V to 11.8 V	5 V to 36 V	2.7 V to 11.8 V	5 V to 36 V
Regulator type:	linear	switching step-up	switching step-up/step-down	switching step-down	switching step-up/step-down	switching step-down
Regulated current:	100 mA[1]	500 mA[1]	1 A[1]	500 mA[1]	1 A[1]	1 A[1]
Dimensions:	1" × 0.6"	1.9" × 0.7"			2.8" × 2.1"	
Weight:	1.3 g[2]	3.4 g[2]			13 g to 33 g	

1 These values are rough approximations for comparison purposes. Available current depends on input voltage, current consumed by the board, ambient conditions, and regulator topology. See product documentation and performance graphs for details.

2 Without included optional headers.

1.1. Supported operating systems

The A-Star 32U4 boards can be programmed using any operating system that supports the Arduino environment. We have tested the A-Stars, our Arduino software add-on, and the Arduino IDE on Microsoft Windows 8.1, 8, 7, Vista, XP (with Service Pack 3), Linux, and Mac OS X.

2. Contacting Pololu

We would be delighted to hear from you about any of your projects and about your experience with the Pololu A-Stars. You can **contact us** [http://www.pololu.com/contact] directly or post on our **forum** [http://forum.pololu.com/]. Tell us what we did well, what we could improve, what you would like to see in the future, or anything else you would like to say!
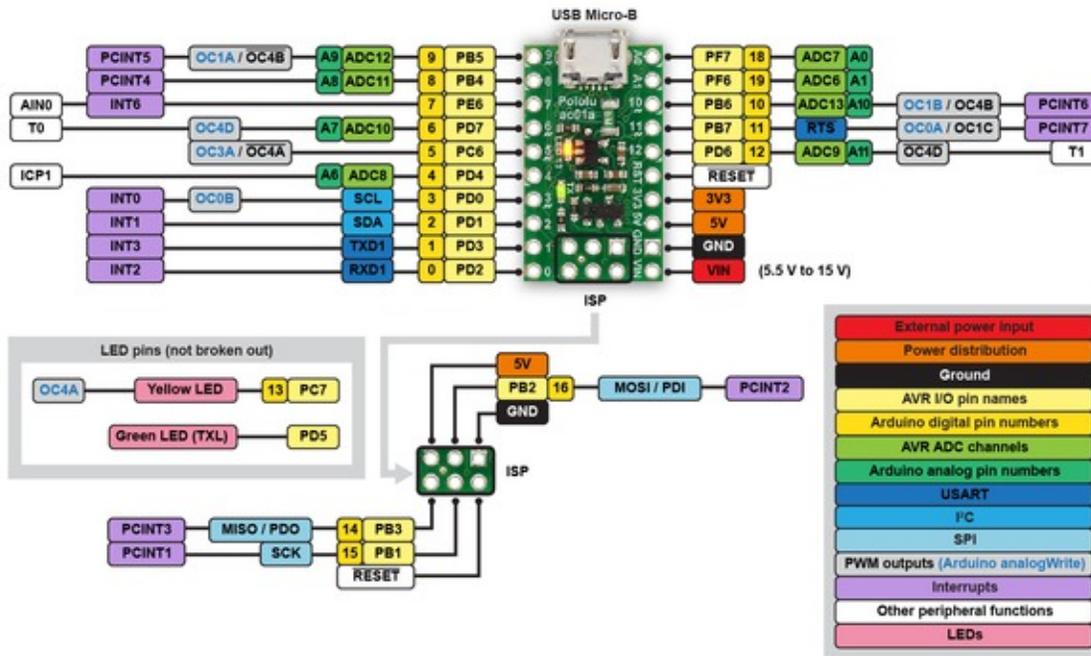
3. A-Star 32U4 Micro
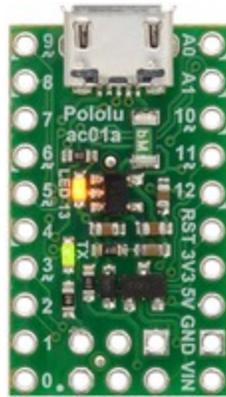
A-Star 32U4 Micro, top view.

3.1. A-Star 32U4 Micro pinout and components

Pinout

The diagram above identifies the I/O and power pins on the A-Star 32U4 Micro; it is also available as a **printable PDF [http://www.pololu.com/file/download/a-star-32u4-micro-pinout.pdf?file_id=0J796]** (409k pdf). For more information about the ATmega32U4 microcontroller on this board, see **Atmel's ATmega32U4 documentation [http://www.atmel.com/ devices/atmega32u4.aspx]**.

Printed on the A* circuit board are indicators that you can use to quickly identify each pin's capabilities: a triangle next to the pin means it can be used as an analog input, and a square wave symbol under the pin number means it can be used as a PWM output.

LEDs

The A-Star 32U4 Micro has two indicator LEDs.

The **yellow** LED is connected to Arduino pin 13, or PC7. You can drive this pin **high** in a user program to turn this LED on. The **A-Star 32U4 Bootloader [http://www.pololu.com/docs/0J61/9]** fades this LED on and off while it is waiting for a sketch to be loaded.

The **green** LED is connected to PD5 and lights when the pin is driven **low**. While the board is running the A-Star 32U4 Bootloader or a program compiled in the Arduino environment, it will flash this LED when it is transmitting data via the USB connection.

Connectors

The A-Star 32U4 includes a USB Micro-B connector that can be used to connect to a computer's USB port via a **USB A to Micro-B cable [http://www.pololu.com/product/2072]** (not included). The USB connection can be used to transmit and receive data from the computer, and a preloaded USB bootloader makes it possible to program the board over USB. The USB connection can also provide power to the A-Star.

The board also has a 6-pin ISP header that allows it to be programmed with an external programmer, such as our **USB AVR programmer [http://www.pololu.com/product/1300]**. Pin 1 of the header is indicated with a small white dot and has an octagonal shape. Three of the pins on this header can be used as an SPI interface or as general-purpose digital I/O, as shown in the pinout diagram. In the Arduino environment, you can refer to these three pins using either their pin numbers or the names of their SPI functions (which are defined as aliases); for example, `digitalRead(15)` and `digitalRead(SCK)` are equivalent.

Power

The A-Star 32U4 Micro can either be powered directly from the USB 5 V supply or from a separate source on the **VIN** pin. The board features a power selection circuit that allows both USB and VIN to be connected at the same time; if this is done, the A-Star will draw power from VIN.

USB power input: The A-Star can be powered from the USB 5 V bus voltage (VBUS) if it is connected to a USB cable. It will draw power from USB only if VIN is disconnected. A resettable PTC fuse on VBUS makes it less likely for the A-Star (and the connected computer or other device) to be damaged if too much current is drawn from the USB connection.

VIN power input: The A-Star can be powered from VIN if you connect a 5.5 V to 15 V power supply (such as a battery or wall power adapter) to the VIN and GND pins, with the positive terminal connected to VIN.

> When powering the A-Star 32U4 Micro from VIN, a minimum voltage of 5.5 V is required to ensure that the board's 5 V supply is stable. Even if power is being provided to the A-Star via USB, connecting a voltage higher than 0 V but lower than 5.5 V to VIN is not recommended, as this can interfere with the power selection circuit and cause the 5 V line to drop (potentially triggering a brown-out reset).

**5V power output:** This pin provides access to the board's 5 V supply, which comes from either the USB 5 V bus voltage or a low-dropout (LDO) regulator on VIN, depending on which power source is connected. The regulator can supply up to 100 mA, although some of this is used by the board itself (typically about 25 mA) or used to provide current for the GPIO pins or 3.3 V power output (see below).

3V3 power output: This pin gives access to the output of the internal 3.3 V regulator inside the ATmega32U4. The microcontroller uses this regulated voltage for USB signaling, but up to about 50 mA is available for powering external circuits or devices.

When the A-Star 32U4 Micro is being powered through VIN, the sum of the 5V output current, 3V3 output current, GPIO output current, and current used by the board itself should not exceed the 100 mA that the regulator can provide.

3.2. A-Star 32U4 Micro schematic and dimensions
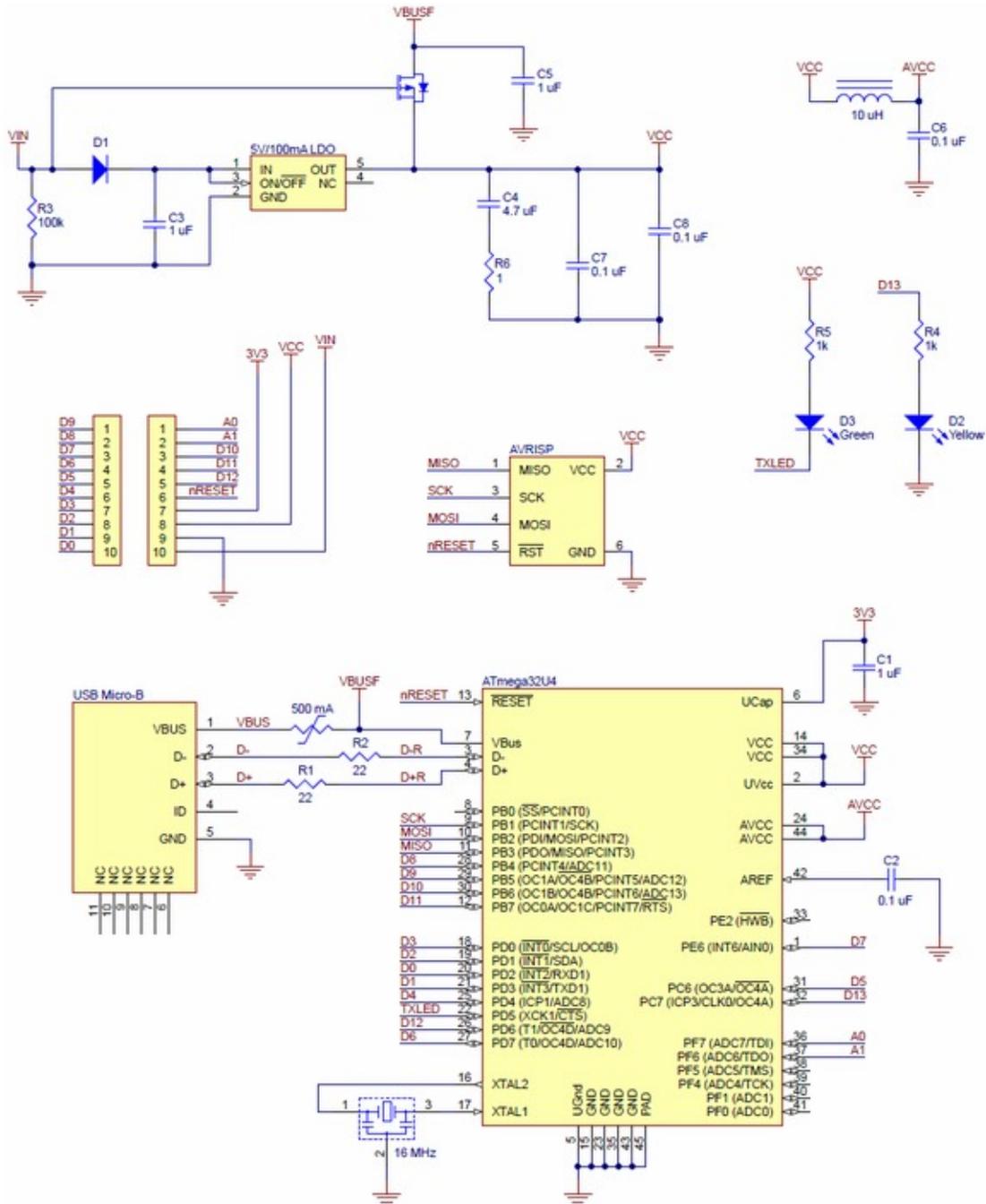
Schematic diagram

Pololu A-Star 32U4 Micro schematic diagram.

This schematic is also available as a PDF: **A-Star 32U4 Micro schematic diagram** **[http://www.pololu.com/file/download/ pololu-a-star-32u4-micro-schematic-diagram.pdf?file_id=0J742]** (253k pdf).

Dimension diagram

A dimension diagram of the A-Star 32U4 Micro is available as a PDF: **A-Star 32U4 Micro dimension diagram** **[http://www.pololu.com/file/download/pololu-a-star-32u4-micro-dimension-diagram.pdf?file_id=0J747]** (255k pdf).
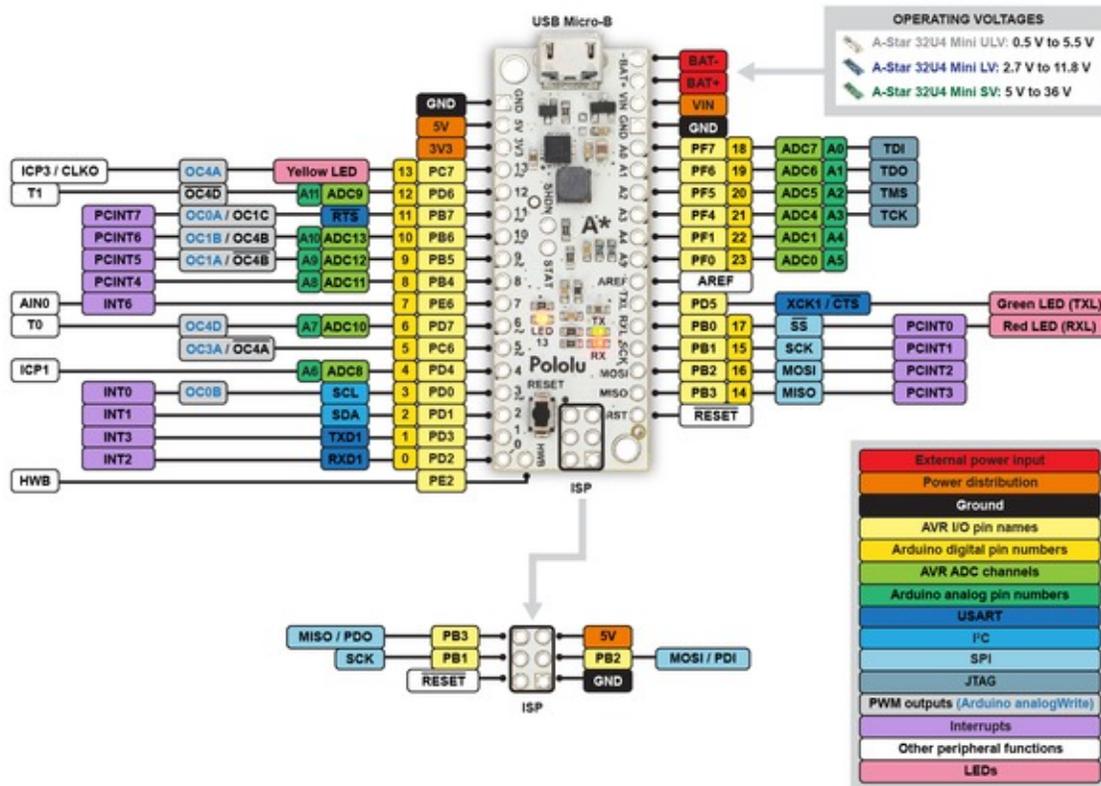
4. A-Star 32U4 Mini
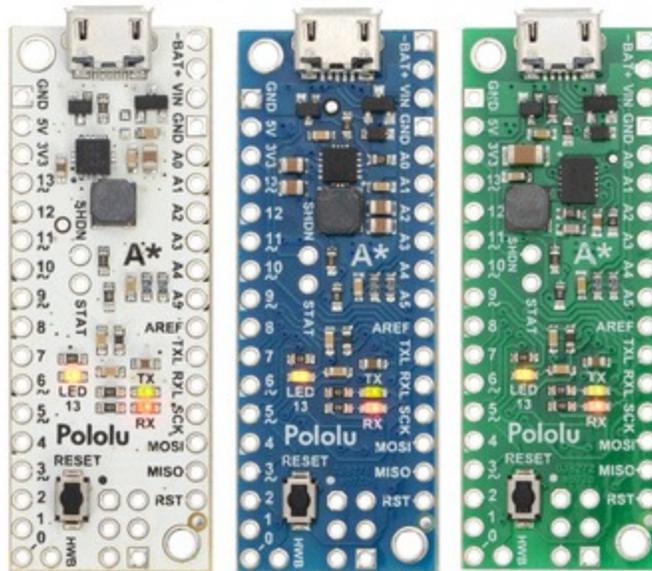
A-Star 32U4 Mini ULV, LV, and SV.

4.1. A-Star 32U4 Mini pinout and components

Pinout

This diagram identifies the I/O and power pins on the A-Star 32U4 Mini (ULV, LV, and SV versions); it is also available as a **printable PDF** [http://www.pololu.com/file/download/a-star-32u4-mini-pinout.pdf?file_id=0J784] (269k pdf). For more information about the ATmega32U4 microcontroller and its peripherals, see Atmel's ATmega32U4 documentation.

Printed on the A* circuit board are indicators that you can use to quickly identify each pin's capabilities: a triangle next to the pin means it can be used as an analog input, and a square wave symbol under the pin number means it can be used as a PWM output.

LEDs

The A-Star 32U4 Mini has three indicator LEDs.

The yellow LED is connected to Arduino pin 13, or PC7. You can drive this pin **high** in a user program to turn this LED on. The **A-Star 32U4 Bootloader** [http://www.pololu.com/docs/0J61/9] fades this LED on and off while it is waiting for a sketch to be loaded.

The green LED is connected to the pin labeled TXL, or PD5, and lights when the pin is driven **low**. While the board is running the A-Star 32U4 Bootloader or a program compiled in the Arduino environment, it will flash this LED when it is *transmitting* data via the USB connection.

The red LED is connected to the pin labeled RXL (usable as Arduino pin 17), or PB0, and lights when the pin is driven **low**. While the board is running the A-Star 32U4 Bootloader or a program compiled in the Arduino environment, it will flash this LED when it is *receiving* data via the USB connection.

Connectors

The A-Star 32U4 includes a USB Micro-B connector that can be used to connect to a computer's USB port via a **USB A to Micro-B cable** [http://www.pololu.com/product/2072] (not included). The USB connection can be used to transmit and receive data from the computer, and a preloaded USB bootloader makes it possible to program the board over USB. The USB connection can also provide power to the A-Star.

The board also has a 6-pin ISP header that allows it to be programmed with an external programmer, such as our **USB AVR programmer** [http://www.pololu.com/product/1300]. Pin 1 of the header is indicated with a small dot and has an octagonal shape.

Power

The A-Star 32U4 Mini can either be powered directly from the USB 5 V supply or from an external voltage source, which is regulated to 5 V by its onboard switching regulator.

The board's power selection circuit uses the **TPS2113A power multiplexer** [http://www.pololu.com/product/2596] from Texas Instruments to choose whether its 5 V supply is sourced from USB or an external supply via the regulator, allowing both sources to be connected at the same time and enabling the A-Star to safely and seamlessly transition between them. The TPS2113A is configured to select external power unless the regulator output falls below about 4.5 V. If this happens, it will select the higher of the two sources, which will typically be the USB 5 V bus voltage if the A* is connected to USB. The currently selected source is indicated by the **STAT** pin in the middle of the board; this pin is an open-drain output that is low if the external power source is selected and high-impedance if the USB supply is selected. The current limit of the TPS2113A is set to about 1.9 A. For more information about the power multiplexer, see the **TPS2113A datasheet** [http://www.pololu.com/file/download/tps2113a.pdf?file_id=0J771] (1MB pdf).

In some situations, it might be undesirable for the A-Star 32U4 Mini to draw power from an external source when it is connected to USB. If this is the case, the regulator can be disabled by driving the regulator shutdown pin, **SHDN**,

high; this shuts down the regulator and causes the power mux to fall back to USB power. For example, this could allow a battery-powered device to turn off the regulator and avoid draining its battery while it is connected to a computer.

The input voltage range of the regulator depends on the particular version of the A-Star 32U4 Mini:

- **ULV**: 0.5 V to 5.5 V (see **Section 4.2** for regulator details)
- **LV**: 2.7 V to 11.8 V (see **Section 4.3** for regulator details)
- **SV**: 5 V to 36 V (see **Section 4.4** for regulator details)

Reverse-protected power inputs: The BAT+ and BAT- pins are power inputs with reverse-voltage protection. These are the recommended pins to use when connecting an external power supply because they allow the A-Star's reverse-voltage protection circuit to help prevent it from being damaged by accidentally-reversed power connections.
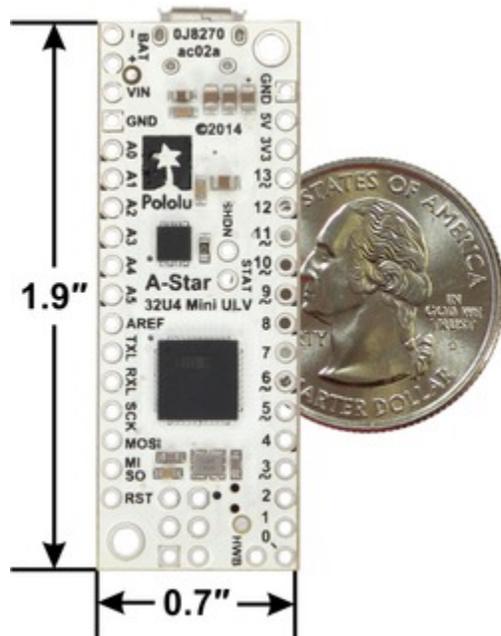
VIN power output (or alternative input)**:** When power is supplied through the BAT pins, the VIN pin can be used as an output to supply reverse-protected power to other devices. Alternatively, the external supply can be connected directly between VIN and GND, bypassing the reverse-voltage protection.

5V power output: This pin provides access to the board's 5 V supply, which comes from either the USB 5 V bus voltage or the onboard switching regulator, depending on which power sources are connected and enabled. Note that some of the available current on the 5 V line is used by the board itself (typically about 30 mA) or used to provide current for the GPIO pins or 3.3 V power output (see below).

3V3 power output: This pin gives access to the output of the internal 3.3 V regulator inside the ATmega32U4. The microcontroller uses this regulated voltage for USB signaling, but up to about 50 mA is available for powering external circuits or devices.

When the A-Star 32U4 Mini is being powered through VIN, the sum of the 5V output current, 3V3 output current, GPIO output current, and current used by the board itself should not exceed the maximum current that the regulator can provide; see the following sections for details about the regulator on each version.

4.2. A-Star 32U4 Mini ULV regulator

The **A-Star 32U4 Mini ULV** can be powered from a **0.5 V to 5.5 V** external source. The input voltage is regulated to 5 V by a TPS61202 switching step-up (boost) converter from Texas Instruments. (We also make a **standalone regulator [http://www.pololu.com/product/2562]** based on this integrated circuit.)

The regulator's low minimum input voltage makes it possible to power this A* with 1 to 3 NiMH, NiCd, or alkaline cells or from a single lithium cell. Unlike standard boost regulators, the TPS61202 also features a linear down-regulation mode that is automatically enabled when the input voltage exceeds 5 V, allowing it to handle input voltages as high as 5.5 V.

As shown in the left graph below, the ULV's switching regulator has an efficiency – defined as (Power out)/(Power in) – of 70% to 90% for most combinations of input voltage and load.
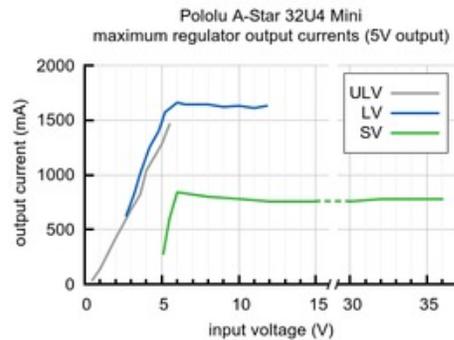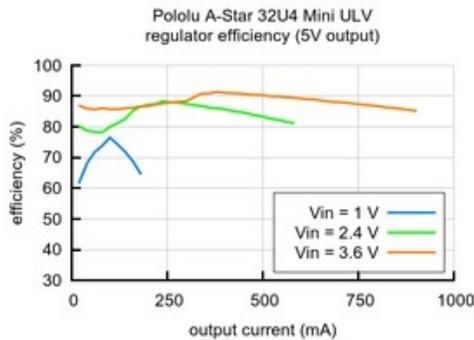
A-Star 32U4 Mini ULV, bottom view with dimensions.

The A-Star's components, including the microcontroller and LEDs, draw 30 mA to 40 mA in typical applications. The rest of the regulator's achievable output current, which depends on input voltage as well as ambient conditions, can be used to power other devices. The right graph above shows output currents at which the voltage regulator's over-temperature protection typically kicks in after a few seconds. These currents represent the limit of the regulator's capability and cannot be sustained for long periods; a good estimate for the maximum continuous regulator output current is 60% to 70% of the values shown in the graph.
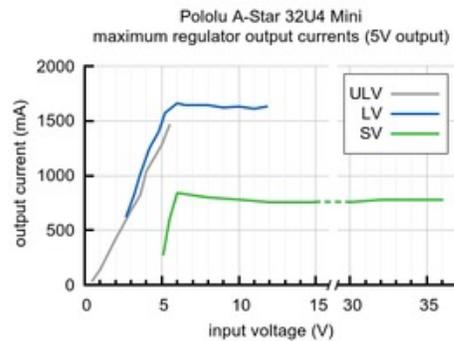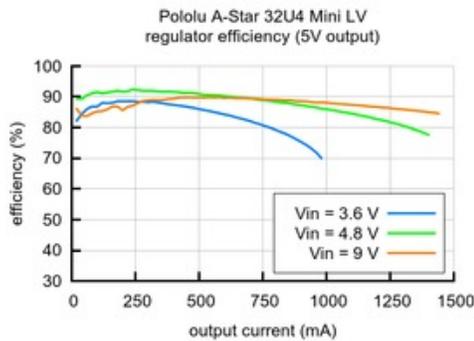
4.3. A-Star 32U4 Mini LV regulator
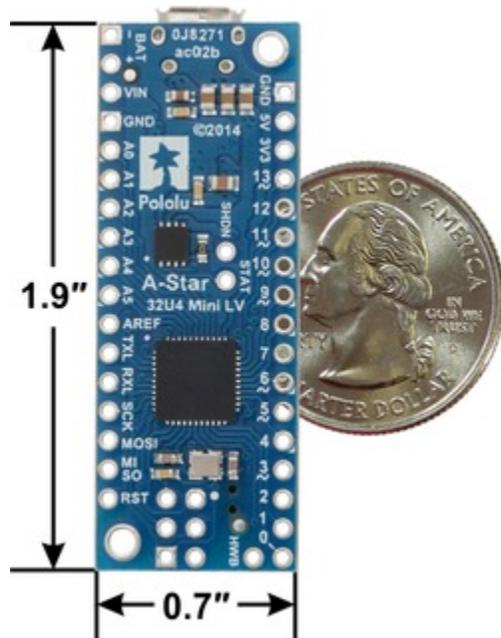
A-Star 32U4 Mini LV, bottom view with dimensions.

The **A-Star 32U4 Mini LV** can be powered from a **2.7 V to 11.8 V** external source. The input voltage is regulated to 5 V by a TPS63061 switching step-up/step-down (buck-boost) converter from Texas Instruments. (We also make a **standalone regulator** [http://www.pololu.com/product/2123] based on this integrated circuit.)

The regulator's flexibility in input voltage is especially well-suited for battery-powered applications in which the battery voltage begins above 5 V and drops below 5 V as the battery discharges. Without the typical restriction on the battery voltage staying above 5 V throughout its life, a wider range of battery types can be considered. For example:
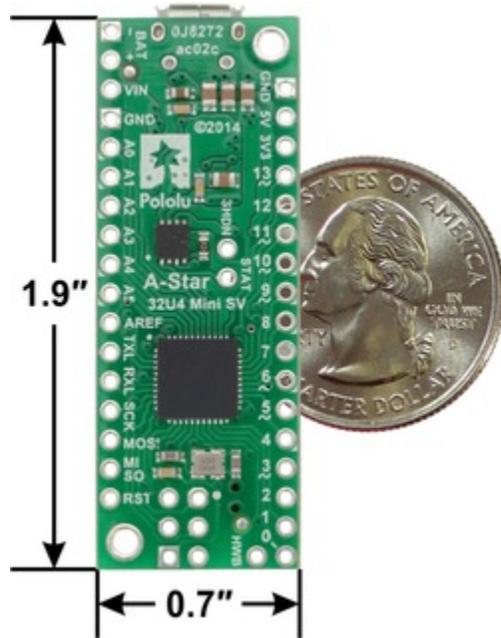
• A 4-cell battery holder, which might have a 6 V output with fresh alkalines or a 4.0 V output with partially discharged NiMH cells, can be used to power this A*.

• A disposable 9 V battery powering the board can be discharged to under 3 V instead of cutting out at 6 V, as with typical linear or step-down regulators.

As shown in the left graph below, the LV's switching regulator has an efficiency – defined as (Power out)/(Power in) – of 80% to 90% for most combinations of input voltage and load.

The A-Star's components, including the microcontroller and LEDs, draw 30 mA to 40 mA in typical applications. The rest of the regulator's achievable output current, which depends on input voltage as well as ambient conditions, can be used to power other devices. The right graph above shows output currents at which the voltage regulator's over-temperature protection typically kicks in after a few seconds. These currents represent the limit of the regulator's capability and cannot be sustained for long periods; a good estimate for the maximum continuous regulator output current is 60% to 70% of the values shown in the graph.

4.4. A-Star 32U4 Mini SV regulator

The **A-Star 32U4 Mini SV** can be powered from a **5 V to 36 V** external source. The input voltage is regulated to 5 V by a 500 mA ISL85415 switching step-down (buck) converter from Intersil. (We also make a **standalone regulator** [http://www.pololu.com/product/2843] based on this integrated circuit.)

As shown in the left graph below, the SV's switching regulator has an efficiency – defined as (Power out)/(Power in) – of 80% to 95% for most combinations of input voltage and load.

A-Star 32U4 Mini SV, bottom view with dimensions.

The A-Star's components, including the microcontroller and LEDs, draw 30 mA to 40 mA in typical applications. The rest of the regulator's achievable output current, which depends on input voltage as well as ambient conditions, can be used to power other devices. The right graph above shows output currents at which the voltage regulator's over-temperature protection typically kicks in after a few seconds. These currents represent the limit of the regulator's capability and cannot be sustained for long periods; a good estimate for the maximum continuous regulator o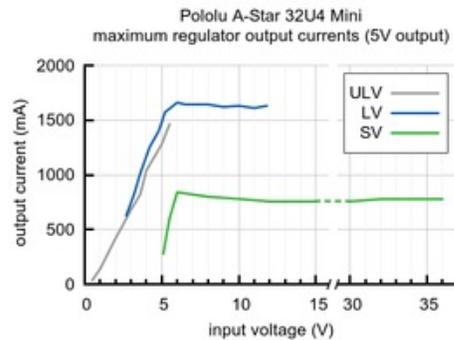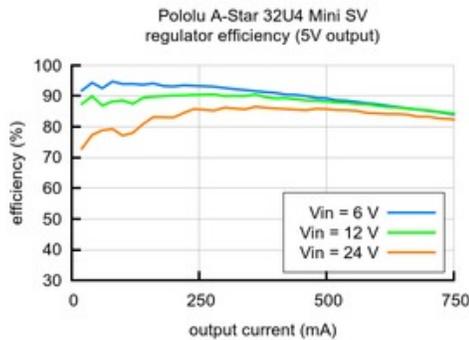utput current is 60% to 70% of the values shown in the graph or 500 mA, whichever is lower (the regulator is not rated for operation above 500 mA).

4.5. A-Star 32U4 Mini schematic and dimensions

Schematic diagram

The schematic diagram for the A-Star 32U4 Minis is available as a PDF: **A-Star 32U4 Mini schematic diagram** [http://www.pololu.com/file/download/a-star-32u4-mini-schematic.pdf?file_id=0J780] (451k pdf).

Dimension diagrams

Dimension diagrams for the A-Star 32U4 Minis are available as PDFs:

- **A-Star 32U4 Mini ULV dimension diagram** [http://www.pololu.com/file/download/ac02a-dimensions.pdf?file_id=0J781] (274k pdf)

- **A-Star 32U4 Mini LV dimension diagram** [http://www.pololu.com/file/download/ac02b-dimensions.pdf?file_id=0J782] (292k pdf)

- **A-Star 32U4 Mini SV dimension diagram** [http://www.pololu.com/file/download/ac02c-dimensions.pdf?file_id=0J783] (291k pdf)

5. A-Star 32U4 Prime
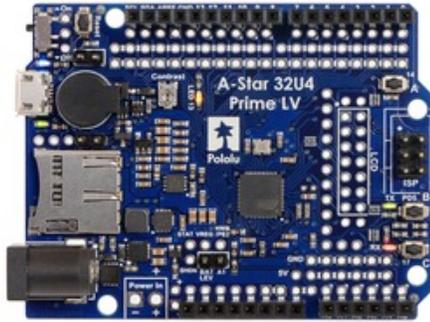
A-Star 32U4 Prime LV microSD.
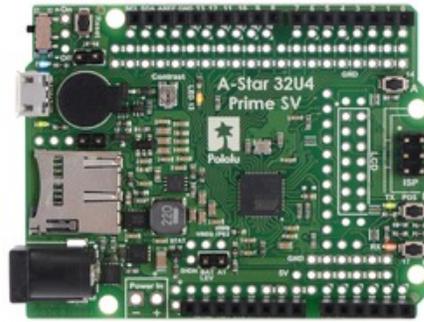
A-Star 32U4 Prime SV microSD.

5.1. A-Star 32U4 Prime configurations

The A-Star 32U4 Prime LV (blue solder mask, 2.7 V to 11.8 V operating voltage range) and SV (green solder mask, 5 V to 36 V operating voltage range) are each available in five different versions, allowing you to choose the right amount of customizability and features for your application. (The pictures here show the LV board, but they also reflect the corresponding SV configurations.)

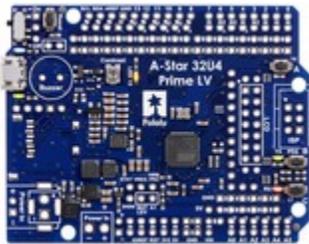**A-Star 32U4 Prime LV (SMT Components Only)**
A-Star 32U4 Prime SV (SMT Components Only)

This version comes with all surface-mount components soldered to the board (including the USB connector, pushbuttons, and power switch), except for the microSD card connector and level shifters, which are *not* populated. It comes with no headers, DC barrel jack, or buzzer.
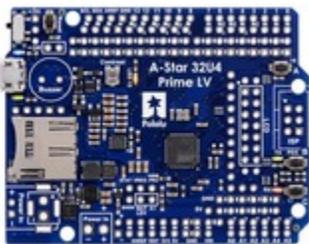
A-Star 32U4 Prime LV microSD (SMT Components Only)
A-Star 32U4 Prime SV microSD (SMT Components Only)

This version adds the microSD card connector and level shifters, but is otherwise identical to the preceding version.
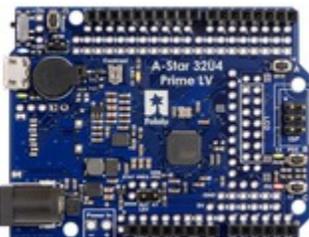
A-Star 32U4 Prime LV
A-Star 32U4 Prime SV

This version comes with all surface-mount components except for the microSD card connector and level shifters. These through-hole parts are also soldered in:

• four standard Arduino female headers (one 1×6, two 1×8, one 1×10) on the outermost pins, matching an Arduino Leonardo or Uno R3

• 6-pin ISP header (2×3 male)

• DC barrel jack

• buzzer

• 1×2 low-profile male header for buzzer jumper (shorting block included)

- 1×2 low-profile male header for battery level jumper (shorting block included)
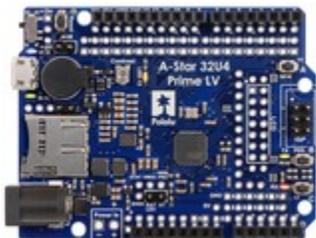
A-Star 32U4 Prime LV microSD
A-Star 32U4 Prime SV microSD

This version adds the microSD card connector and level shifters, as well as a third 1×2 low-profile male header and shorting block for the microSD CS jumper, but is otherwise identical to the preceding version.

A-Star 32U4 Prime LV microSD with LCD
A-Star 32U4 Prime SV microSD with LCD

This version is the same as the preceding version, but it comes with a **8×2 character LCD** [http://www.pololu.com/product/356] (with 2×7 female header soldered), and the following parts are additionally soldered to the board:

- LCD connector (2×7 male header)
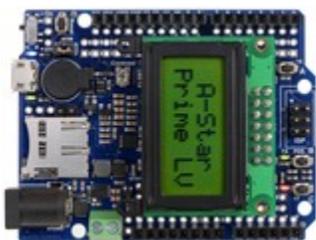- 3.5 mm screw terminal block

For users wishing to populate some or all of the through-hole parts on an SMT-only version of the A-Star 32U4 Prime (or to add more functionality to another version), we offer an **A-Star Prime accessory pack** [http://www.pololu.com/product/3110] that contains these items:

- four standard Arduino **female headers** [http://www.pololu.com/category/50/0.100-in-2.54-mm-female-headers] (one 1×6, two 1×8, one 1×10)
- four 1×25 **male header strips** [http://www.pololu.com/product/965]
- 2×3 male header (ISP)
- 2×7 male header (LCD)
- **2×7 female header** [http://www.pololu.com/product/1027] (LCD)
- **1×2 female header** [http://www.pololu.com/product/1012]
- 1×10 low-profile male header strip
- three low-profile shorting blocks
- buzzer
- **DC barrel jack** [http://www.pololu.com/product/1139]
- **2-pin 3.5 mm screw terminal block** [http://www.pololu.com/product/2444]

The **8×2 character LCD** [http://www.pololu.com/product/356] is also available separately.

> **Note:** The A-Star 32U4 Prime does not come with a microSD card. The microSD connector is an optional feature that can be used to increase the data storage available to the board, but it is not required.

5.2. A-Star 32U4 Prime pinout and components
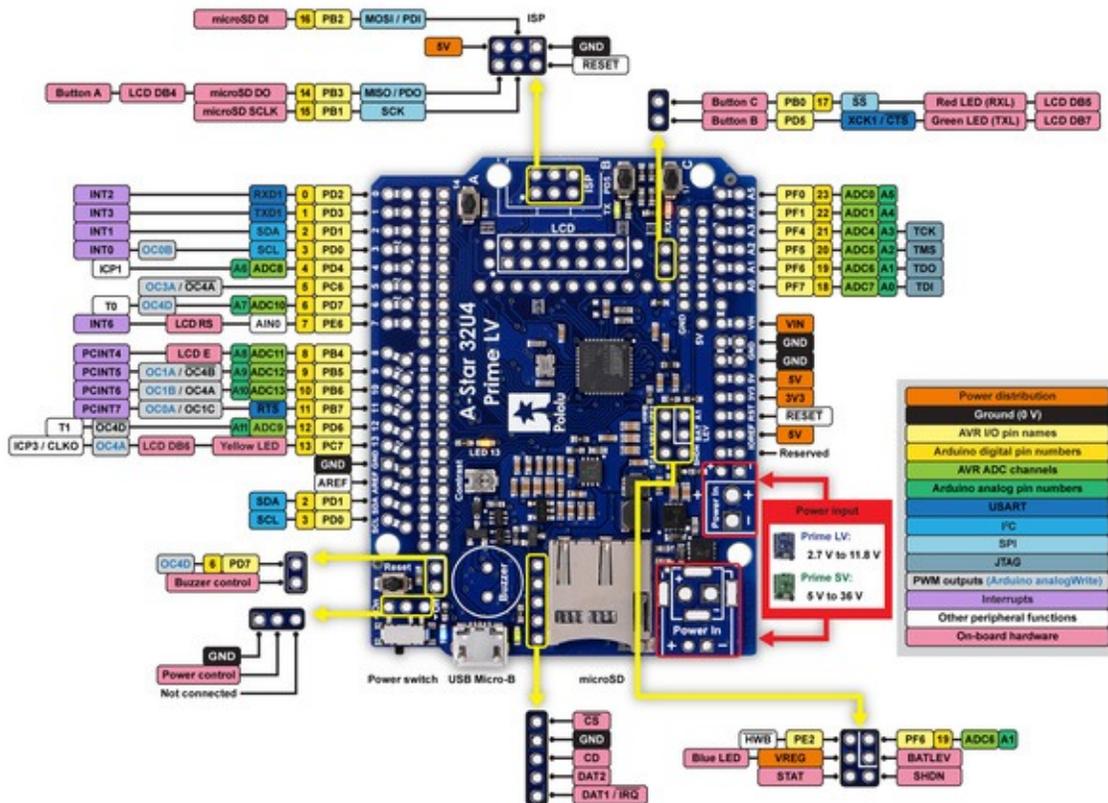
Pinout

This diagram identifies the I/O and power pins on the A-Star 32U4 Prime (LV and SV versions); it is also available (along with the power distribution diagram below) as a **printable PDF** [http://www.pololu.com/file/download/ a_star_32u4_prime_pinout_power.pdf?file_id=0J847] (1MB pdf). For more information about the ATmega32U4 microcontroller and its peripherals, see Atmel's ATmega32U4 documentation.

The outermost rows of pins of the A-Star 32U4 Prime correspond to the pins on an Arduino Leonardo, and each is duplicated on a second inner row for more convenient access. Printed on the A* circuit board are indicators that you can use to quickly identify each I/O pin's capabilities: a triangle by the inner through hole means the pin can be used as an analog input, and a square wave symbol next to the hole pair means the pin can be used as a PWM output.

Additional rows of power and ground pins also run along most of the I/O pins to allow easy connections to devices like sensors and servos. The innermost pin is ground, and the second-innermost is power, which is not connected by default. The power pins are split into three buses: one bus for pins A0–A4 along the bottom, and two buses along the top, split between digital pins 7 and 8. You can configure each bus separately by using a jumper to connect it to a voltage source of your choice. (Access points for 5V and VIN are adjacent to the A0–A4 bus, so it can be conveniently connected to either supply with a shorting block.)
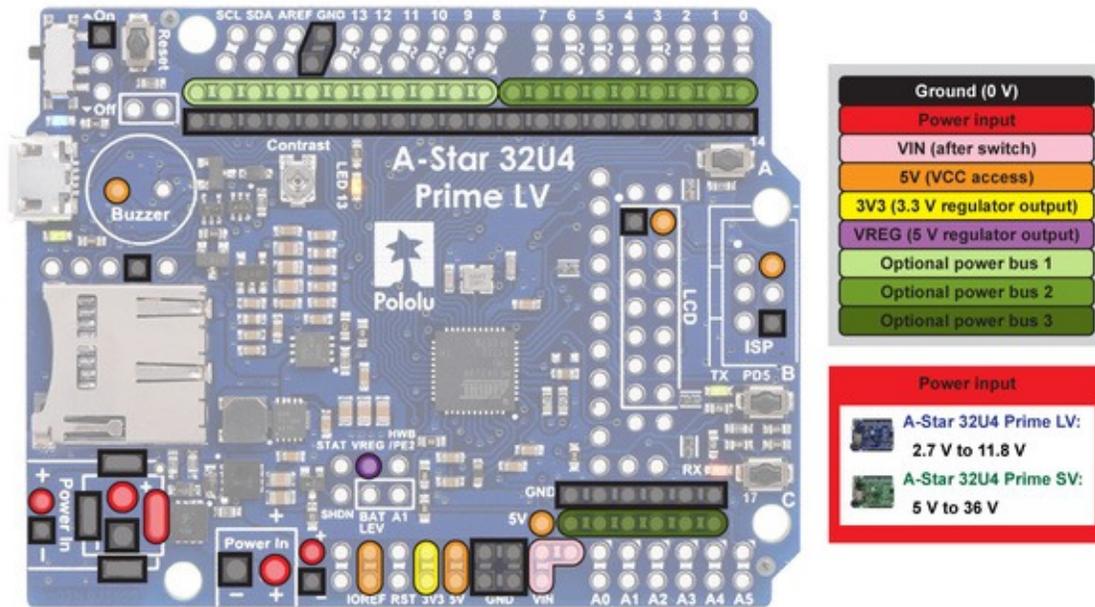
Compatibility with Arduino shields and accessories

The A-Star 32U4 Prime matches the Arduino Leonardo and the Arduino Uno R3 in the shape of its circuit board and the arrangement of its pins. Furthermore, it uses the same ATmega32U4 microcontroller as the Leonardo, running at the same voltage and frequency, so the A* should generally work with any shield or accessory that is compatible with the Leonardo (including our **Zumo Robot for Arduino** [http://www.pololu.com/product/2510]).

LEDs

The A-Star 32U4 Prime has five indicator LEDs.

A **blue** power LED located next to the power switch indicates when the onboard regulator is active (an external power supply is connected and the switch is turned on).

A **green** power LED next to the USB connector indicates when the USB bus voltage (VBUS) is present.

A **yellow** user LED is connected to Arduino pin 13, or PC7. You can drive this pin **high** in a user program to turn this LED on. The **A-Star 32U4 Bootloader** [http://www.pololu.com/docs/0J61/9] fades this LED on and off while it is waiting for a sketch to be loaded.

A **green** user LED is connected to PD5 and lights when the pin is driven **low**. While the board is running the A-Star 32U4 Bootloader or a program compiled in the Arduino environment, it will flash this LED when it is *transmitting* data via the USB connection.

A **red** user LED is connected to Arduino pin 17, or PB0, and lights when the pin is driven **low**. While the board is running the A-Star 32U4 Bootloader or a program compiled in the Arduino environment, it will flash this LED when it is *receiving* data via the USB connection.

The AStar32U4Prime library contains functions that make it easier to control the three user LEDs. All three user LED control lines are also LCD data lines, so you will see them flicker when you update the LCD. The green and red user LEDs also share I/O lines with pushbuttons (see below).

Pushbuttons

The A-Star 32U4 Prime has four pushbuttons: a **reset button** next to the power switch and **three user pushbuttons** located along the right edge of the board. The user pushbuttons, labeled A, B, and C, are on Arduino pin 14 (PB3), PD5, and Arduino pin 17 (PB0), respectively. Pressing one of these buttons pulls the associated I/O pin to ground through a resistor.

The three buttons' I/O lines are also used for other purposes: pin 14 is MISO on the SPI interface, PD5 and pin 17 control the green and red user LEDs, and all three pins are LCD data lines. Although these uses require the pins to be driven by the AVR (or SPI slave devices in the case of MISO), resistors in the button circuits ensure that the A-Star will not be damaged even if the corresponding buttons are pressed at the same time, nor will SPI or LCD communications be disrupted. The functions in the AStar32U4Prime library take care of configuring the pins, reading and debouncing the buttons, and restoring the pins to their original states.

Note that button A cannot be used if the microSD $\overline{CS}$ pin is tied to ground with a jumper (see *microSD card connector and level shifters* below).

Buzzer

The assembled versions of the A-Star 32U4 Prime come with a buzzer that can be used to generate simple sounds and music. The buzzer is not present on the SMT-only versions, but the buzzer driver circuit is still populated, allowing you to solder in your own buzzer or speaker. The stock buzzer is available as part of the **A-Star 32U4 Prime accessory pack [http://www.pololu.com/product/3110]**.

A through-hole jumper next to the buzzer provides a way to connect the buzzer input to digital pin 6 (which also serves as OC4D, a hardware PWM output from the AVR's 10-bit Timer4). If you alternate between driving the buzzer pin high and low at a given frequency, the buzzer will produce sound at that frequency. You can play notes and music with the buzzer using functions in the AStar32U4PrimeBuzzer library.

Buzzer and $\overline{CS}$-GND jumpers on the A-Star 32U4 Prime.

microSD card connector and level shifters

Some versions of the A-Star 32U4 Prime include an onboard microSD card connector that enables the microcontroller to read from and write to microSD memory cards. The card socket is connected to the SPI interface on the ATmega32U4 through level-shifting circuits, allowing the 5 V microcontroller to safely communicate with standard 3.3 V SD cards. DI, DO, and SCLK on the card are connected to MOSI, MISO, and SCK on the AVR, respectively. The **Arduino SD library [http://arduino.cc/en/Reference/SD]** can be used to access the file system on an inserted microSD card.

A fourth pin, $\overline{CS}$ (chip select), must be held low to enable communication with the microSD card in SPI mode. This can be done by connecting one of the microcontroller's I/O lines to $\overline{CS}$ (the SD library examples use digital pin 4 by default) and driving the pin low to select the microSD card. The $\overline{CS}$ line also controls whether the output of the DO level shifter is enabled: when $\overline{CS}$ is high, the level shifter places its output into a high-impedance state, allowing the MISO line to be driven by other SPI slave devices, used as an LCD control line, or read as a pushbutton input.

Alternatively, you can tie $\overline{CS}$ low more easily by shorting it to the adjacent ground pin, causing the microSD card to always be selected and the DO level shifter output to always be enabled. This allows you to avoid using an extra I/O line and making an additional connection, but the DO level shifter output will override the pushbutton on the same pin
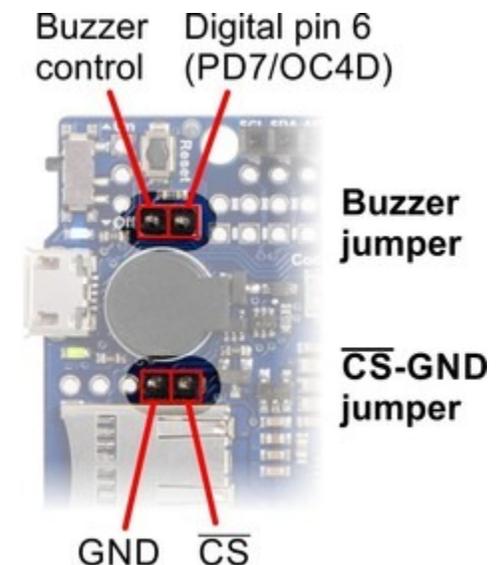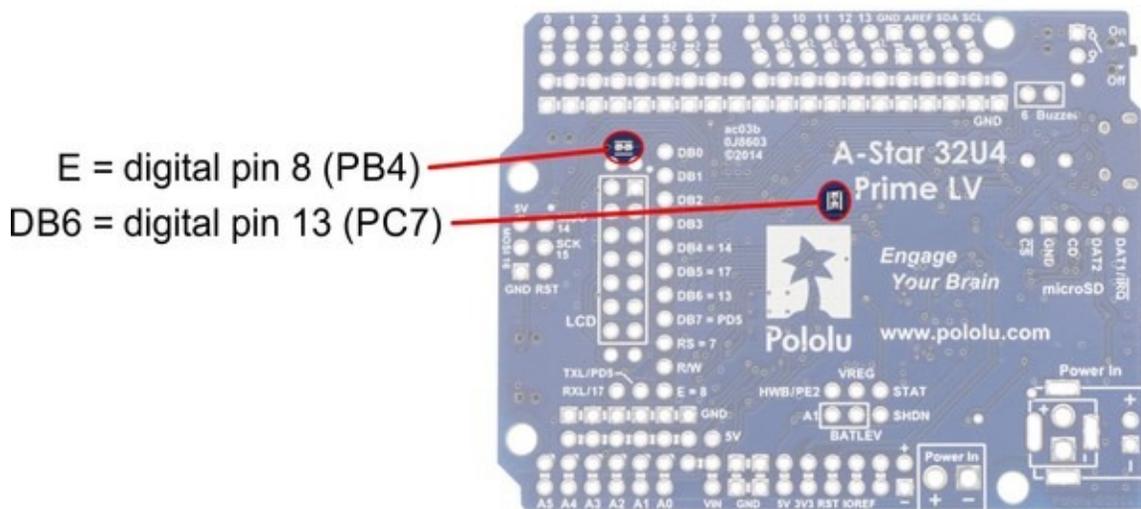
and prevent any presses on button A from being detected. You should remove the $\overline{\text{CS}}$-GND jumper when no microSD card is present, as asserting $\overline{\text{CS}}$ with no microSD card inserted can put the DO level shifter output into an undefined state.

The CD (card detect) pin next to the connector can be used to detect the presence of a microSD card: it is floating when a card is inserted and shorted to ground when a card is not present. Two additional SD data lines (DAT1 and DAT2), which are unused in SPI mode, are also broken out for advanced uses.

LCD

The A-Star 32U4 Prime has a mounting location for a 2×7 header where you can connect a character LCD with the common **HD44780 parallel interface** [http://www.pololu.com/file/download/DMC50448N-AAE-AD.pdf?file_id=0J71] (109k pdf). The A* is optionally available with a male header installed here and an **8×2 character LCD** [http://www.pololu.com/product/356] (with corresponding female header) included; on other versions, you can add your own display using the connectors of your choice. A larger LCD can be connected with a **ribbon cable** [http://www.pololu.com/product/973] and optionally a **shrouded box header** [http://www.pololu.com/product/898].

The LCD control lines are broken out to a column of through holes next to the LCD connector, labeled on the back side of the board. By default, some of these are connected to I/O lines from the ATmega32U4 to allow control of the LCD in 4-bit mode, but you can remap the connections by cutting the surface-mount jumpers indicated in the picture below and making new connections between I/O lines and LCD control pins.

RS = digital pin 7 (PE6)
DB4 = digital pin 14 (PB3)
DB7 = PD5
DB5 = digital pin 17 (PB0)

E = digital pin 8 (PB4)
DB6 = digital pin 13 (PC7)
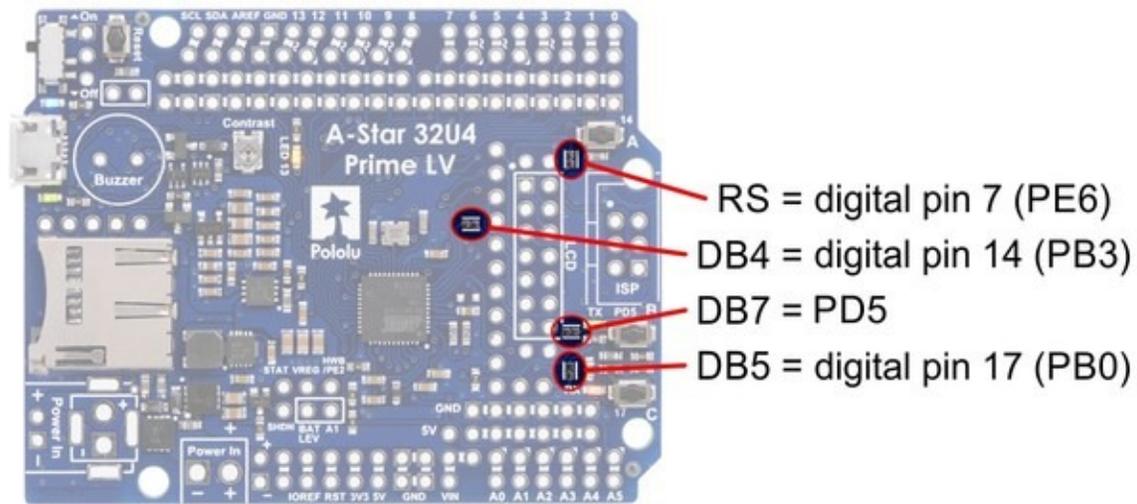
The AStar32U4PrimeLCD library provides functions to display data on a connected LCD. It is designed to gracefully handle alternate use of the LCD data lines by only changing pin states when needed for an LCD command, after which it will restore them to their previous states. This allows the LCD data lines to be used for other functions (such as pushbutton inputs and LED drivers).

Other connectors

The A-Star 32U4 includes a USB Micro-B connector that can be used to connect to a computer's USB port via a **USB A to Micro-B cable** [http://www.pololu.com/product/2072] (not included). The USB connection can be used to transmit and receive data from the computer, and a preloaded USB bootloader makes it possible to program the board over USB. The USB connection can also provide power to the A-Star.

The board also has a 6-pin ISP header that allows it to be programmed with an external programmer, such as our **USB AVR programmer** [http://www.pololu.com/product/1300]. Pin 1 of the header is indicated with a small dot and has an octagonal shape. Three of the pins on this header can be used as an SPI interface or as general-purpose digital I/O, as labeled on the back side of the board and shown in the pinout diagram. In the Arduino environment, you can refer to these three pins using either their pin numbers or the names of their SPI functions (which are defined as aliases); for

example, `digitalRead(15)` and `digitalRead(SCK)` are equivalent. If you have an SMT-only version of the A-Star, it is possible to solder a **shrouded box header** [http://www.pololu.com/product/854] to this location; the outlines on the board indicate the correct connector orientation.

Power

The A-Star 32U4 Prime can either be powered directly from the USB 5 V supply or from an external voltage source, which is regulated to 5 V by its onboard switching regulator. The slide switch on A* controls whether the external source is connected to the input of the regulator, providing a convenient way to switch off external power to the A-Star without unplugging any connections. The adjacent set of three pins provides a place to connect your own power switch: to enable external power, connect the middle pin to ground (accessible through the upper pin).

> When the A-Star is connected to a computer via USB, it will receive 5V power even when the power switch is off. This can be useful if you want to upload or test a program without drawing external power (to avoid draining a battery pack, for example).

The board's power selection circuit uses the **TPS2113A power multiplexer** [http://www.pololu.com/product/2596] from Texas Instruments to choose whether its 5 V supply is sourced from USB or an external supply via the regulator, allowing both sources to be connected at the same time and enabling the A-Star to safely and seamlessly transition between them. The TPS2113A is configured to select external power unless the regulator output falls below about 4.5 V. If this happens, it will select the higher of the two sources, which will typically be the USB 5 V bus voltage if the A* is connected to USB. The currently selected source is indicated by the **STAT** pin in the middle of the board; this pin is an open-drain output that is low if the external power source is selected and high-impedance if the USB supply is selected. The current limit of the TPS2113A is set to about 1.9 A. For more information about the power multiplexer, see the **TPS2113A datasheet** [http://www.pololu.com/file/download/tps2113a.pdf?file_id=0J771] (1MB pdf).

In some situations, it might be undesirable for the A-Star 32U4 Prime to draw power from an external source when it is connected to USB, even if the power switch is left on. If this is the case, the regulator can be disabled by driving the regulator shutdown pin, **SHDN**, high; this shuts down the regulator and causes the power mux to fall back to USB power. For example, this could allow a battery-powered system to automatically turn off the regulator while it is connected to a computer.

The input voltage range of the regulator depends on the particular version of the A-Star 32U4 Prime:

- **LV**: 2.7 V to 11.8 V (see **Section 5.3** for regulator details)
- **SV**: 5 V to 36 V (see **Section 5.4** for regulator details)

Reverse-protected and switched power inputs: The connection points labeled "Power In" are power inputs with reverse-voltage protection that are controlled by the power switch. These are the recommended pins to use when connecting an external power supply because they allow the A-Star's reverse-voltage protection circuit to help prevent it from being damaged by accidentally-reversed power connections, and they enable the A-Star's power switch to disconnect the supply.

A number of different connectors can be soldered to the board and used to provide power, including a **DC barrel jack** [http://www.pololu.com/product/1139] or a **3.5 mm screw terminal block** [http://www.pololu.com/product/2444] (the assembled versions of the board come with one or both of these populated).

> **Warning:** You must never connect different power sources to multiple Power In locations at the same time, as doing so will create a short between the supplies.

VIN power output (or alternative input)**:** When power is supplied through the Power In pins, the VIN pin can be used as an output to supply reverse-protected and switched power to other devices. Alternatively, the external supply can be connected directly between VIN and GND, bypassing the reverse-voltage protection and power switch.

5V power output: This pin provides access to the board's 5 V supply, which comes from either the USB 5 V bus voltage or the onboard switching regulator, depending on which power sources are connected and enabled. Note that some of the available current on the 5 V line is used by the board itself (typically at least 30 mA, but potentially much more if peripherals like the buzzer are active) or used to provide current for the GPIO pins or 3.3 V regulator (see below).

3V3 power output: This pin gives access to the output of the onboard 3.3 V linear regulator. The microSD card, if present, draws power from the 3.3 V line; the remainder (up to a few hundred milliamps) is available for powering external circuits or devices.

When the A-Star 32U4 Prime is being powered through Power In, the sum of the 5V output current, 3V3 output current, GPIO output current, and current used by the board itself should not exceed the maximum current that the switching regulator can provide.

In a battery-powered application, it might be useful for the A-Star to monitor the battery's voltage level. The **BATLEV** pin provides access to a voltage divider that outputs a fraction of the VIN voltage (one-third on the LV, one-eighth on the SV), and this voltage can be read by connecting it to the adjacent analog pin 1 (A1) (or another analog input). The `readBatteryMillivoltsLV()` and `readBatteryMillivoltsSV()` functions in the AStar32U4Prime library can be used to determine the battery voltage from this reading.

Battery level jumper on the A-Star 32U4 Prime.
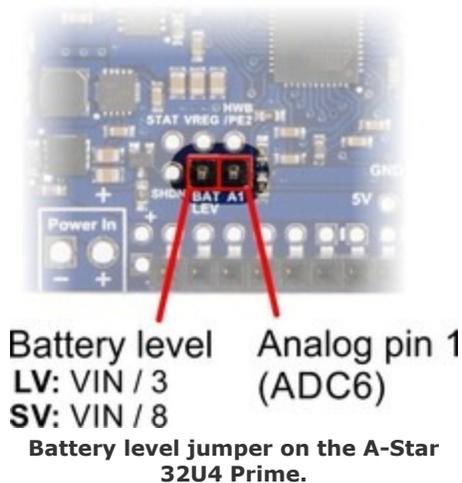
5.3. A-Star 32U4 Prime LV regulator
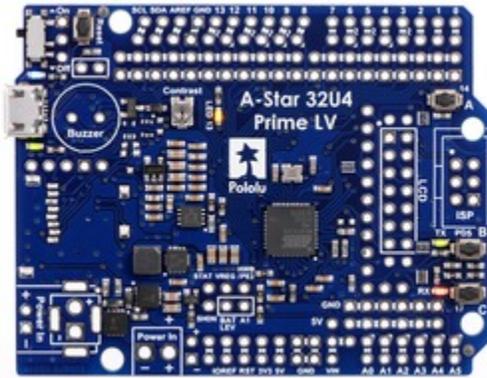
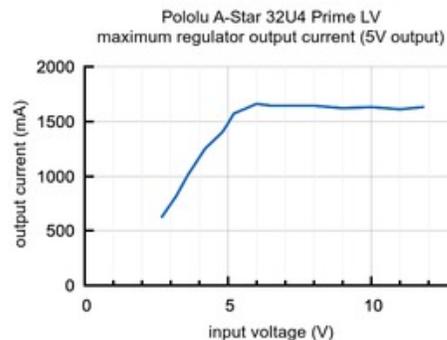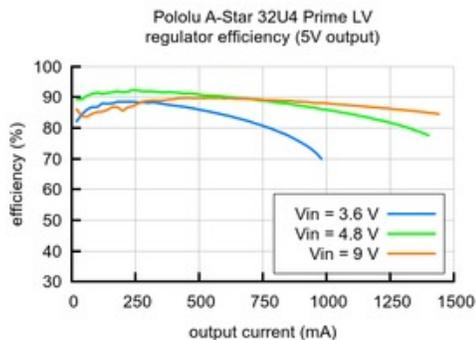The **A-Star 32U4 Prime LV** can be powered from a **2.7 V to 11.8 V** external source. The input voltage is regulated to 5 V by a TPS63061 switching step-up/step-down (buck-boost) converter from Texas Instruments. (We also make a **standalone regulator [http://www.pololu.com/product/2123]** based on this integrated circuit.)

The regulator's flexibility in input voltage is especially well-suited for battery-powered applications in which the battery voltage begins above 5 V and drops below 5 V as the battery discharges. Without the typical restriction on the battery voltage staying above 5 V throughout its life, a wider range of battery types can be considered. For example:

• A 4-cell battery holder, which might have a 6 V output with fresh alkalines or a 4.0 V output with partially discharged NiMH cells, can be used to power this A*.

• A disposable 9 V battery powering the board can be discharged to under 3 V instead of cutting out at 6 V, as with typical linear or step-down regulators.

As shown in the left graph below, the LV's switching regulator has an efficiency – defined as (Power out)/(Power in) – of 80% to 90% for most combinations of input voltage and load.

The A-Star's components, including the microcontroller and LEDs, draw 30 mA to 40 mA in typical applications (without the buzzer, microSD card, or an LCD). The rest of the regulator's achievable output current, which depends on input voltage as well as ambient conditions, can be used to power other devices. The right graph above shows output currents at which the voltage regulator's over-temperature protection typically kicks in after a few seconds. These currents represent the limit of the regulator's capability and cannot be sustained for long periods; under typical operating conditions, a safe limit for the maximum continuous regulator output current is 60% to 70% of the values shown in the graph.

5.4. A-Star 32U4 Prime SV regulator
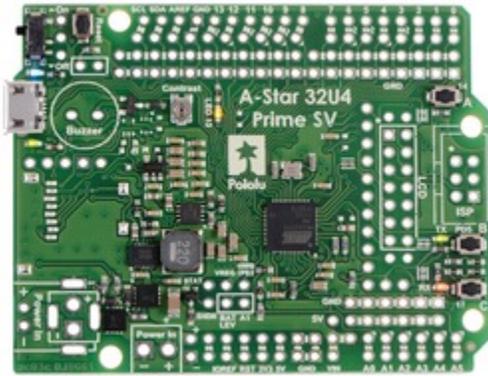
The **A-Star 32U4 Prime SV** can be powered from a **5 V to 36 V** external source. The input voltage is regulated to 5 V by a 1 A ISL85410 switching step-down (buck) converter from Intersil. (We also make a **standalone regulator** [http://www.pololu.com/product/2831] based on this integrated circuit.)

As shown in the left graph below, the SV's switching regulator has an efficiency – defined as (Power out)/(Power in) – of 80% to 95% for most combinations of input voltage and load.
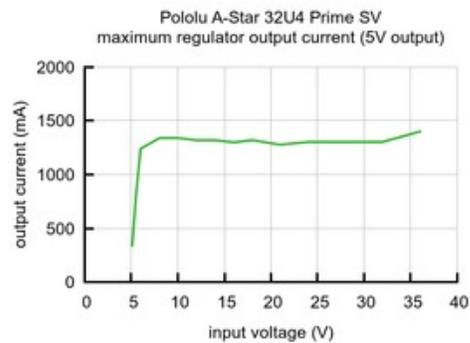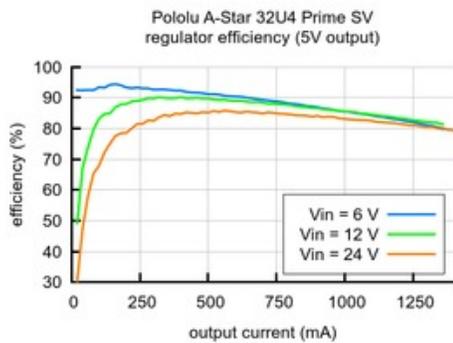
The A-Star's components, including the microcontroller and LEDs, draw 30 mA to 40 mA in typical applications (without the buzzer, microSD card, or an LCD). The rest of the regulator's achievable output current, which depends on input voltage as well as ambient conditions, can be used to power other devices. The right graph above shows the output currents where the regulator's output voltage drops below 4.75 V. These currents are close to the limits of the regulator's capability and generally cannot be sustained for long periods; under typical operating conditions, a safe limit for the maximum continuous regulator output current is approximately 1 A.

The dropout voltage of a step-down regulator is defined as the minimum amount by which the input voltage much exceed the regulator's target output voltage in order to assure the target output can be achieved. As can be seen in the graph below, the dropout voltage of the Prime SV's regulator increases approximately linearly with the output current. For light loads where the dropout voltage is small, the board can operate down to 5 V. However, for larger loads, the dropout voltage should be taken into consideration when selecting a power supply; operating above 6 V will ensure the full output current is available.
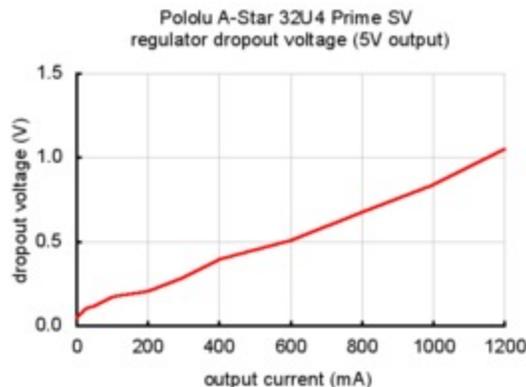
**Typical dropout voltage of the 5 V regulator
on the A-Star 32U4 Prime SV.**

Note: Although the ISL85410 is rated for a maximum operating input voltage of 36 V, it is *not* appropriate to power the Prime SV with a 36 V battery, as battery voltages can be much higher than nominal voltages when they are charged. The maximum nominal battery voltage we recommend is 24 V, and if you approach that limit, you should take extra precautions to prevent LC voltage spikes from damaging the board (see **this application note** [http://www.pololu.com/docs/0J16] for more information). A good practice is to ensure that the A-Star's power switch is off before connecting it to a voltage source.

5.5. A-Star 32U4 Prime schematic

Schematic diagram

The schematic diagrams for the A-Star 32U4 Primes are available as PDFs:

- **A-Star 32U4 Prime LV schematic diagram** [http://www.pololu.com/file/download/a-star_32u4_prime_lv_schematic_diagram.pdf?file_id=0J846] (1MB pdf)
- **A-Star 32U4 Prime SV schematic diagram** [http://www.pololu.com/file/download/a-star_32u4_prime_sv_schematic_diagram.pdf?file_id=0J856] (1MB pdf)

5.6. A-Star 32U4 Prime demo program

The **A-Star 32U4 Prime LV microSD with LCD** [http://www.pololu.com/product/3109] and **A-Star 32U4 Prime SV microSD with LCD** [http://www.pololu.com/product/3115] each come preloaded with a program that demonstrates many of its features and allows you to test that it is working correctly, and this demo program will run when you first turn on the board.

The demo program uses the standard A-Star 32U4 Prime LCD pins, buzzer pin, button pins, microSD pins, and pin 4. To avoid damage or improper operation, if you have shields or other electronics connected, make sure they do not use those pins in a conflicting way.

When the demo program starts, you should see the words "A-Star Prime" and then "Demo Program" appear. If you do not see any text on the LCD, you may need to adjust the contrast potentiometer. The demo program is not usable without an LCD. If a buzzer is soldered in and the buzzer shorting block is installed, you should also hear a beep when the demo program starts.

After the program has started, press the B button to proceed to the main menu. Press C or A to scroll forward or backward through the menu, and press B to make a selection or to exit one of the demos. There are six demos accessible from the menu:

- **Keyboard:** This demo makes the A-Star act as a two-key keyboard. If the A-Star is connected to a computer via USB, you can press button A or button C to send the corresponding key presses to the computer.

- **LEDs:** Blinks the red, green, and yellow user LEDs.

- **Music:** Plays an adaptation of J. S. Bach's Fugue in D Minor for microcontroller and piezo, while scrolling a text display. This demonstrates the ability for the A-Star to play music in the background.

- **Power:** Displays information about the board's power sources. The top line displays the measured voltage on VIN, in millivolts. For this measurement to work, a jumper should be installed to connect A1 to BATLEV. The board should also be powered from one of the connection points labeled "Power In", and the power switch must be in the "On" position, or else the reading will be 0 mV. The bottom line displays "USB=Y" if power from the USB connector is detected and "USB=N" if it is not detected.

- **SD card:** This demo communicates with the microSD card to get the size of the card's main partition and then displays it on the LCD, in units of megabytes. To use this part of the program, you will need to install a jumper between GND and $\overline{CS}$. (You will also need to have a version of the board with a microSD card socket and you will need to insert a formatted microSD card into the socket.) While the $\overline{CS}$-GND jumper is installed, button A will not work. Also, if the $\overline{CS}$-GND jumper is installed and the microSD card is not inserted, the program might detect spurious presses on button A. To avoid these issues, you can connect $\overline{CS}$ to pin 4 using a **male-female jumper wire** [http://www.pololu.com/category/67/male-female-premium-jumper-wires] instead of installing the jumper.

- **Serial:** This demo uses the A-Star's virtual USB serial port to send and receive data from the computer. You can connect to the A-Star's serial port using a standard terminal program or by using the Serial Monitor feature from the Tools menu in the Arduino IDE. Any characters you send to the A-Star will be displayed on the LCD, and information about the received characters will be sent back to the computer. If you press the A or C buttons, a message will be sent to the computer saying they were pressed.

The A-Star 32U4 Prime versions with included LCDs also come with the buzzer and battery voltage sensing jumpers installed by default and the contrast potentiometer adjusted properly, so the LCD, buzzer, and voltage sensing should work by default.

The source code of the demo program is included with the AStar32U4Prime library, so you can compile it and load it onto any A-Star 32U4 Prime using the Arduino IDE.
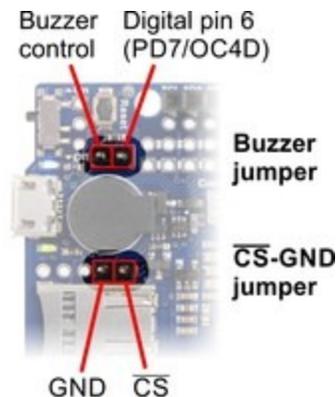
Buzzer and $\overline{CS}$-GND jumpers on the A-Star 32U4 Prime.
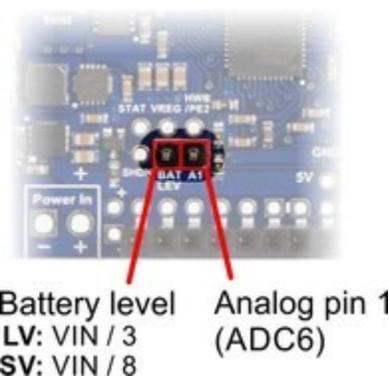
Battery level jumper on the A-Star 32U4 Prime.

6. Getting started

6.1. Installing Windows drivers

> If you use Windows XP, you will need to have either **Service Pack 3** [http://www.microsoft.com/downloads/details.aspx?FamilyId=68C48DAD-BC34-40BE-8D85-6BB4F56F5110] or Hotfix KB918365 installed before installing the A-Star drivers. Some users who installed the hotfix have reported problems which were solved by upgrading to Service Pack 3, so we recommend Service Pack 3 over the hotfix.

Before you connect your Pololu A-Star 32U4 to a computer running Microsoft Windows, you must install its drivers:

1. Download the **A-Star Software and Drivers** [http://www.pololu.com/file/download/a-star-1.2.0.zip?file_id=0J743] (153k zip) and extract the ZIP file to a temporary folder on your computer. These files are also available from the **A-Star repository on GitHub** [https://github.com/pololu/a-star].

2. Open the "drivers" folder. Right-click on "a-star.inf" and select "Install".
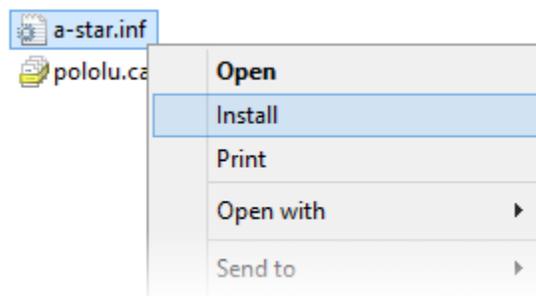
3. Windows will ask you whether you want to install the drivers. Click "Install" (Windows 8, 7, and Vista) or "Continue Anyway" (Windows XP).
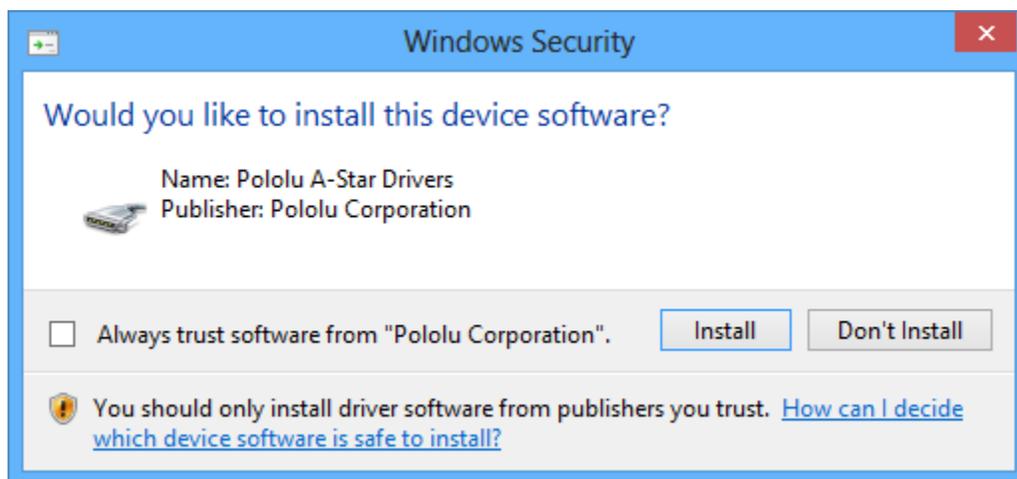
4. Windows will not tell you when the installation is complete, but it should be done after a few seconds.

Windows 8, Windows 7, and Windows Vista users: After installing the drivers, your computer should automatically recognize the A-Star when you connect it via USB. No further action from you is required. However, the first time

you connect an A-Star to your computer, Windows will take several seconds to recognize the device and configure itself properly. The first time you program the A-Star, Windows will again take several seconds to recognize the A-Star's USB bootloader, and this could cause the programming operation to fail the first time. Also, Windows will need to re-recognize the A-Star and the bootloader if you connect the board to another USB port that it has not been connected to before.

Windows XP users: After installing the drivers, you will need to follow steps 5–9 for each new A-Star you connect to your computer. You will also need to follow these steps the first time you attempt to program the A-Star in order to make Windows recognize the bootloader, and when you connect the A-Star to a different USB port that it has not been connected to before.

5. Connect the A-Star to your computer's USB port.

6. When the "Found New Hardware Wizard" is displayed, select "No, not this time" and click "Next".

7. On the second screen of the "Found New Hardware Wizard", select "Install the software automatically" and click "Next".

8. Windows XP will warn you again that the driver has not been tested by Microsoft and recommend that you stop the installation. Click "Continue Anyway".

9. When you have finished the "Found New Hardware Wizard", click "Finish".

6.2. Programming using the Arduino IDE

The Pololu A-Star 32U4 boards can be programmed from the popular Arduino integrated development environment (IDE). The Arduino IDE is a cross-platform, open source application that integrates a C++ code editor, the GNU C++ compiler, and a program upload utility. To get started programming your A-Star with the Arduino IDE, follow these steps:

1. Download the Arduino IDE from the **Arduino Download page** [http://arduino.cc/en/Main/Software] and install it. You can download either the 1.0.x version or the 1.5.x version; the A-Stars support both.

2. If you have not done so already, download the **A-Star Software and Drivers** [http://www.pololu.com/file/download/a-star-1.2.0.zip?file_id=0J743] (153k zip). These files are also available from the **A-Star repository on GitHub** [https://github.com/pololu/a-star]. Copy the "pololu" folder from the downloaded "add-on" folder into the **[sketchbook location]/hardware** folder. You can see the sketchbook location in the Arduino IDE Preferences dialog, which is available from the File menu. For a typical user, the sketchbook location will be a folder named Arduino inside the Documents folder. For a typical user running a recent version of Windows, the "pololu" folder should be installed in this location:

Programming the A-Star 32U4 from the Arduino IDE.

C:\Users\<username>\Documents\Arduino\hardware\pololu

If the "Arduino" or "hardware" directories do not exist yet, you will need to create them.

3. Close the Arduino IDE if it is open, and then restart it.

4. In the **Tools > Board** menu, select the "Pololu A-Star 32U4" entry. If you do not see the A-Star listed in the Board menu, then the add-on is probably not installed correctly. Try doing step 2 again and restarting the Arduino IDE.
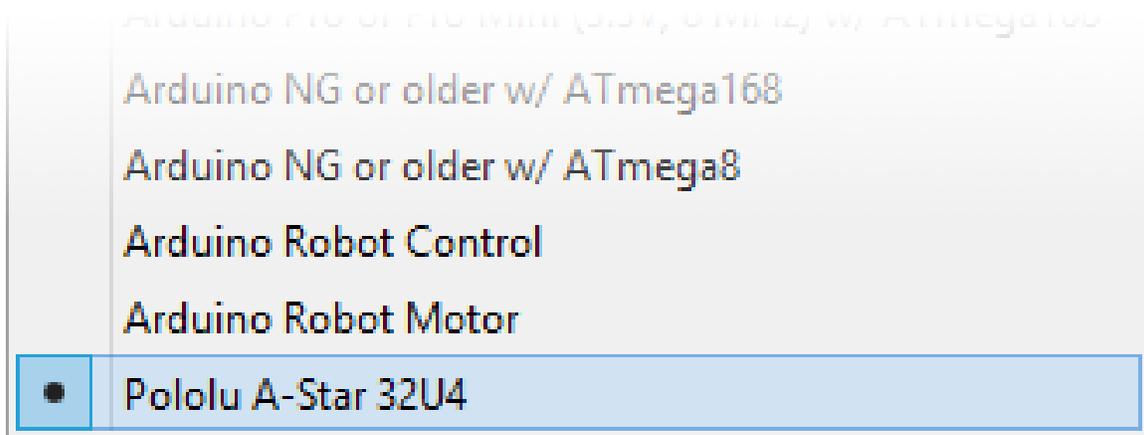
Selecting the Pololu A-Star 32U4 in the Boards menu.

5. In the **Tools > Port** menu, select the port for the A-Star. On Windows you can determine what COM port the A-Star is assigned to by looking at the "Ports (COM & LPT)" section of the Device Manager. On Linux, the port name will begin with "/dev/ttyACM". On Mac OS X, the port name will begin with "/dev/tty.usbmodem".
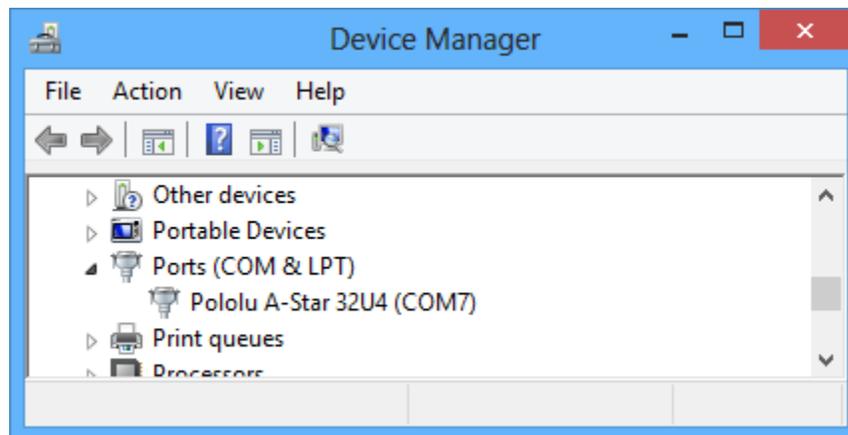
Windows 8 device manager showing the A-Star's virtual COM port.

6. Open up the "Blink" Arduino example, which can be found under **File > Examples > 01.Basics > Blink**. The code in this example will make the A-Star blink its yellow LED. When you select the Blink example, a new Arduino IDE window will open up. It is OK to close the first window.
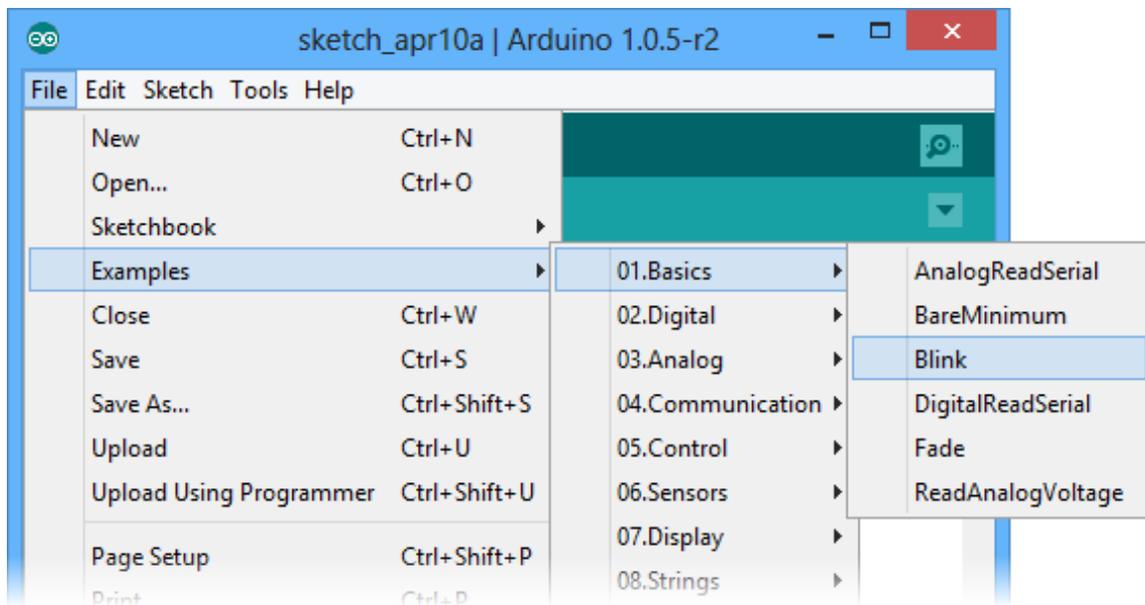
Selecting the Blink example in the Arduino IDE.

7. Press the "Upload" button to compile the sketch and upload it to the A-Star. If everything goes correctly, you will see the message "Done uploading" appear near the bottom of the window. If you are using Windows and you have not previously programmed an A-Star on this USB port, then Windows might take several seconds to recognize the A-Star bootloader. The bootloader times out after 8 seconds and goes back to running the sketch, so the upload might fail if Windows does not recognize it quickly enough. If this happens, try again. If you are using Windows XP and have not programmed an A-Star on this USB port, you will have to go through the Found New Hardware Wizard again as described in **Section 6.1**, but the second time you try to upload it should work. If the Arduino IDE has trouble connecting to the port or using it, try unplugging the A-Star, closing any programs that might be using the serial port, restarting the Arduino IDE, and then plugging the A-Star back in.
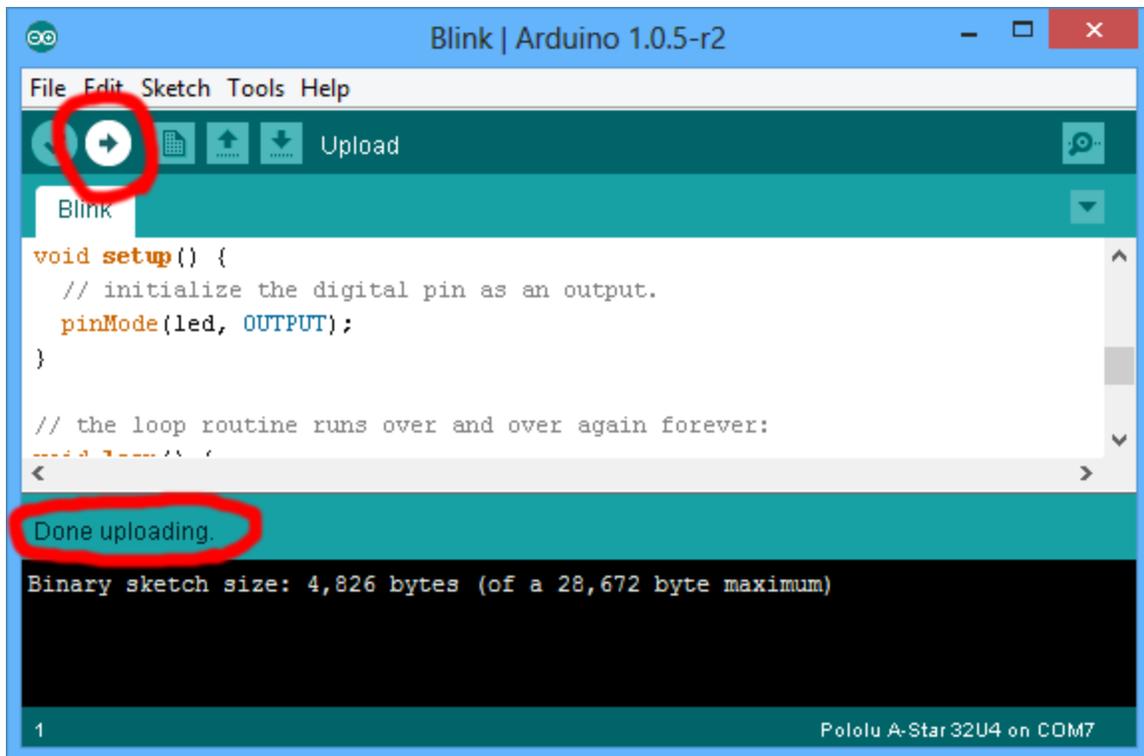
Uploading a sketch to the A-Star using the Arduino IDE.

8. If you uploaded the Blink sketch, then the A-Star's yellow LED should be blinking once every two seconds. However, we ship the A-Stars with that same example already programmed onto it, so you might not be convinced that anything has changed. Try changing the delay values in the sketch to something else and uploading again to see if you can change the speed of the LED.

> The A-Star 32U4 boards are similar enough to the Arduino Leonardo that you do not actually have to install the add-on. If you want to, you can just select the "Arduino Leonardo" board in the Arduino IDE. Note that if you upload a sketch to the A-Star this way, your computer will then recognize it as a Leonardo (for example, its entry in the Windows Device Manager will display "Arduino Leonardo").

After you succeed in programming the A-Star from the Arduino IDE, there are many resources you can use to learn more:

• The Arduino IDE has many **examples [http://arduino.cc/en/Tutorial/HomePage]** that can run on A-Stars.

• The Arduino website has a **Language Reference [http://arduino.cc/en/Reference/HomePage]**, a wiki called the **The Arduino Playground [http://playground.arduino.cc/]**, and other resources.

• The A-Star 32U4 boards are similar to the **Arduino Leonardo [http://www.pololu.com/product/2192]** and **Arduino Micro [http://www.pololu.com/product/2188]**, so you can search the Internet for relevant projects that use one of those boards.

• **Section 11** lists many more resources.

6.3. Programming using avr-gcc and AVRDUDE

This section explains how to program the A-Star boards using the avr-gcc toolchain and AVRDUDE. This section is intended for advanced users who do not want to use the Arduino IDE as described in **Section 6.2**.

Getting the prerequisites

If you are using Windows, we recommend downloading **WinAVR [http://winavr.sourceforge.net/]**, which contains the avr-gcc toolchain and a command-line utility called **AVRDUDE [http://www.nongnu.org/avrdude/]** that can be used to upload programs to the A-Star bootloader. If the version of GNU Make that comes with WinAVR crashes on your computer, we recommend using the **Pololu version of GNU Make [https://github.com/pololu/make/releases]**.

If you are using Mac OS X, we recommend downloading the **CrossPack for AVR Development [http://www.obdev.at/products/crosspack]**.

If you are using Linux, you will need to install avr-gcc, avr-libc, and AVRDUDE. Ubuntu users can get the required software by running:

```
sudo apt-get install gcc-avr avr-libc avrdude
```

After you have installed the prerequisites, open a command prompt and try running these commands to make sure all the required utilities are available:

```
avr-gcc -v
avr-objcopy -V
make -v
avrdude
```

If any of those commands fail, make sure the desired executable is installed on your computer and make sure that it is in a directory listed in your PATH environment variable.

Compiling an example program

Copy the following code to a file named "main.c":

```
#define F_CPU 16000000
#include <avr/io.h>
#include <util/delay.h>

int main()
{
  DDRC |= (1 << DDC7);     // Make pin 13 be an output.
  while(1)
  {
    PORTC |= (1 << PORTC7);   // Turn the LED on.
    _delay_ms(500);
    PORTC &= ~(1 << PORTC7);  // Turn the LED off.
    _delay_ms(500);
  }
}
```

In the same folder, create a file named "Makefile" with the following contents:

```
PORT=\\\\.\\USBSER000
MCU=atmega32u4
CFLAGS=-g -Wall -mcall-prologues -mmcu=$(MCU) -Os
LDFLAGS=-Wl,-gc-sections -Wl,-relax
CC=avr-gcc
TARGET=main
OBJECT_FILES=main.o
```

```
all: $(TARGET).hex

clean:
        rm -f *.o *.hex *.obj *.hex

%.hex: %.obj
        avr-objcopy -R .eeprom -O ihex $< $@

%.obj: $(OBJECT_FILES)
        $(CC) $(CFLAGS) $(OBJECT_FILES) $(LDFLAGS) -o $@

program: $(TARGET).hex
        avrdude -p $(MCU) -c avr109 -P $(PORT) -U flash:w:$(TARGET).hex
```

Make sure that the PORT variable in the Makefile is set to the name of the A-Star's virtual serial port. In Windows, `\\\\.\\USBSER000` should work if the A-Star is the only USB device connected that is using the usbser.sys driver, but you can change it to be the actual name of the COM port (e.g. `COM13`).

In a command prompt, navigate to the directory with the Makefile and main.c. If you run the command `make`, the code should get compiled and produce a file named "main.hex".

Programming

To program the A-Star, you will need to get it into bootloader mode first. One way to do this is to reset the AVR twice within 750 ms. The A-Star 32U4 Mini boards have a reset button that can be used to reset the board. On any A-Star board, a pushbutton can be connected between the GND and RST pins to serve as a reset button, or you can use a wire. Once the A-Star is in bootloader mode, quickly run the command `make program` to program the A-Star. If you wait longer than 8 seconds, the A-Star bootloader will exit and the AVR will go back to running the user program.

7. A-Star Arduino libraries

The A-Star can be programmed directly from the Arduino IDE as described in **Section 6.2**.

To help interface with the on-board hardware on the A-Star 32U4 Prime LV, we provide the **AStar32U4Prime library**. The library makes it easy to display information on the LCD, play sounds and music from the buzzer, use the three pushbuttons, control all three user LEDs, and read the battery voltage level. The **AStar32U4Prime library documentation [https://pololu.github.io/a-star-32u4-prime/]** provides detailed information about the library, and the library comes with several example sketches.

The instructions below explain how to install the AStar32U4Prime library and start using it:

1. If you have not done so already, download the **A-Star Software and Drivers [http://www.pololu.com/file/ download/a-star-1.2.0.zip?file_id=0J743]** (153k zip). These files are also available from the **A-Star repository on GitHub [https://github.com/pololu/a-star]**. Extract the downloaded ZIP file and find the "AStar32U4Prime" folder, which is inside the "libraries" folder. Copy the downloaded "AStar32U4Prime" folder into the "libraries" subdirectory inside your Arduino sketchbook directory. You can see the sketchbook location in the Arduino IDE Preferences dialog, which is available from the File menu. For a typical user, the sketchbook location will be a folder named Arduino inside the Documents folder. For a typical user running a recent version of Windows, the "AStar32U4Prime" folder should be installed in this location:

C:\Users\<username>\Documents\Arduino\libraries\AStar32U4Prime

If the "Arduino" or "libraries" directories do not exist yet, you will need to create them.

2. Close the Arduino IDE if it is open, and then restart it.

3. In the **Files > Examples** menu, locate the entry for the AStar32U4Prime library. This is where you can find several example sketches that show how to use the library. If you do not see AStar32U4Prime in your Examples menu, then the library was probably not installed correclty. Try doing step 1 again and restarting the Arduino IDE.

After you install the AStar32U4Prime library, you can learn more about it by trying the included example sketches and by reading the **AStar32U4Prime library documentation [https://pololu.github.io/a-star-32u4-prime/]**.
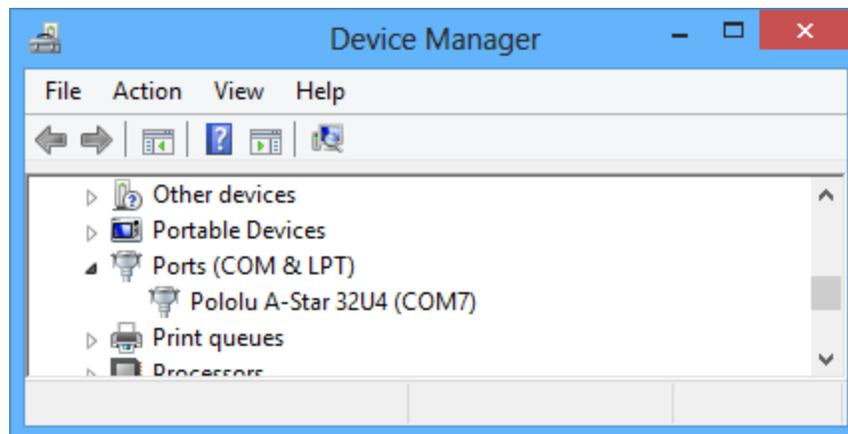
8. The A-Star 32U4 USB interface

The A-Star 32U4 boards are based on a single AVR ATmega32U4 microcontroller that runs the user program and also handles the USB connection to the computer. The AVR has a full-speed USB transceiver built into it and can be programmed to present almost any type of USB device interface to the computer.

USB is an asymmetric system that consists of a single "host" connected to multiple "devices". The host is typically a personal computer. The ATmega32U4 can only act as a USB device, so an A-Star cannot be connected to other USB devices; it can only be connected to a host such as your computer.

Programming an A-Star using the Arduino IDE as described in **Section 6.2** will automatically configure it as a composite device with a single virtual serial port and a human interface device (HID) with a keyboard and mouse. Since we preload an example Arduino sketch on each A-Star before it ships, this is the default USB configuration of the A-Star.

On a Windows computer, you can see the virtual serial port by going to your computer's Device Manager and expanding the "Ports (COM & LPT)" list. You should see a COM port labeled "Pololu A-Star 32U4". In parentheses after the name, you will see the name of the port (e.g. "COM3" or "COM4"). Windows will assign a different COM port number to the A-Star depending on what USB port you plug it into and whether it is in bootloader mode or not. If you need to change the COM port number assigned to the A-Star, you can do so using the Device Manager. Double-click on the COM port to open its properties dialog, and click the "Advanced…" button in the "Port Settings" tab. From this dialog you can change the COM port assigned to the A-Star.

Windows 8 device manager showing the A-Star's virtual COM port.

On a Windows computer, you can see the rest of the A-Star's USB interface by going to the Device Manager, selecting **View > Devices by connection**, and then expanding entries until you find the "Pololu A-Star 32U4" COM port. Near it, you should see the parent composite device and the human interface device.
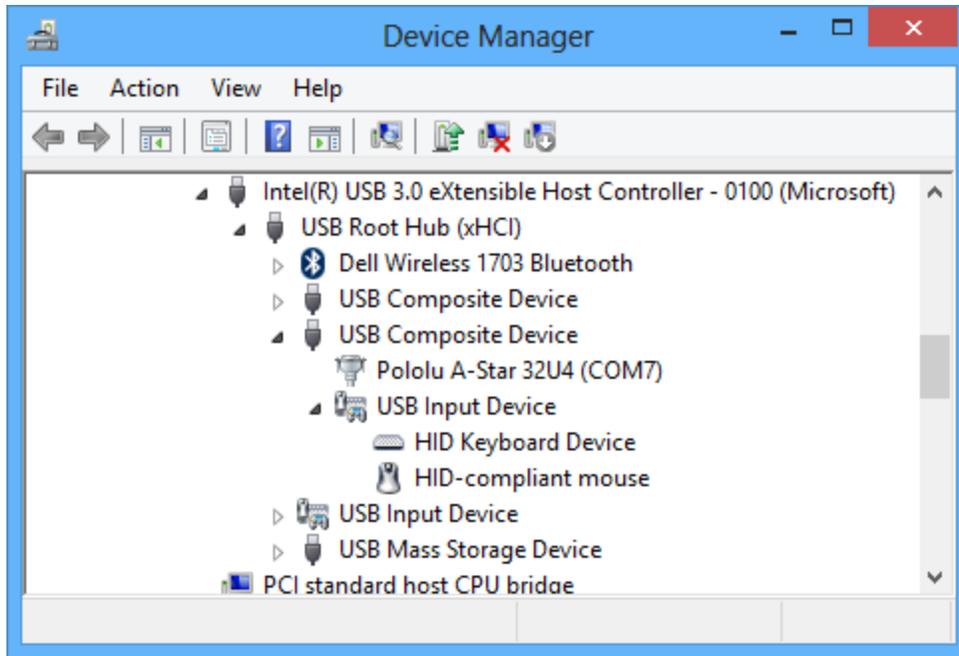
The Windows 8 Device Manager in "Devices by connection" mode, showing that the A-Star is a composite device.

On a Linux computer, you can see details about the A-Star's USB interface by running `lsusb -v -d 1ffb:` in a Terminal. The virtual serial port can be found by running `ls /dev/ttyACM*` in a Terminal.

On a Mac OS X computer, the virtual serial port can be found by running `ls /dev/tty.usbmodem*` in a Terminal.

You can send and receive bytes from an A-Star's virtual serial port using any terminal program that supports serial ports. Some examples are the Serial Monitor in Arduino IDE, the **Pololu Serial Transmitter Utility** [http://www.pololu.com/docs/0J23], **Br@y Terminal** [http://sites.google.com/site/terminalbpp/], **PuTTY** [http://www.chiark.greenend.org.uk/~sgtatham/putty/], **TeraTerm** [http://ttssh2.sourceforge.jp/], **Kermit** [http://www.columbia.edu/kermit/ck80.html], and **GNU Screen** [http://www.gnu.org/software/screen/]. Many computer programming environments also support sending and receiving bytes from a serial port.

9. The A-Star 32U4 Bootloader

The A-Star 32U4 boards come with a USB bootloader that can be used in conjunction with the Arduino IDE or AVRDUDE to load new programs onto an A-Star. This section documents some technical details of the bootloader for advanced users who want to better understand how it works. If you just want to get started using your A-Star, it is fine to skip this section.

The A-Star 32U4 Bootloader, which is used on all A-Star 32U4 boards, is based on the **Caterina bootloader** **[https://github.com/arduino/Arduino/tree/master/hardware/arduino/bootloaders/caterina]**, which is the bootloader used on the **Arduino Leonardo [http://www.pololu.com/product/2192]**, **Arduino Micro [http://www.pololu.com/product/2188]** and several other ATmega32U4 boards. The bootloader is open source and its **source code [https://github.com/pololu/a-star/tree/master/bootloaders/caterina]** is available on GitHub. The bootloader occupies the upper four kilobytes of the ATmega32U4's program memory, leaving 28 KB for the user program. The bootloader's USB interface consists of a single virtual serial port that accepts the programming commands defined in **AVR109 [http://www.atmel.com/images/doc1644.pdf]**. The bootloader always runs first immediately after the AVR is reset.

Startup logic

The main difference between the A-Star 32U4 Bootloader and Caterina is in the startup logic. This is the part of the bootloader that runs immediately after the AVR is reset, and it decides whether to run the user program or run the rest of the bootloader. The startup logic of the Caterina bootloader is designed so that when the RST line goes low, the bootloader will run. This means that if you want to restart your program using the RST line, it will take 8 seconds before the bootloader times out waiting for an upload and the sketch starts.

The A-Star 32U4 Bootloader has different startup logic that allows you to use the RST line to reset the board with a smaller delay. If the RST line goes low once, the user program will run after a 750 ms delay. If the RST line goes low twice within 750 ms, then the bootloader will run. (This behavior is the same as on boards like SparkFun's Pro Micro.)

The start-up logic of the A-Star 32U4 Bootloader is shown in the flowchart below:
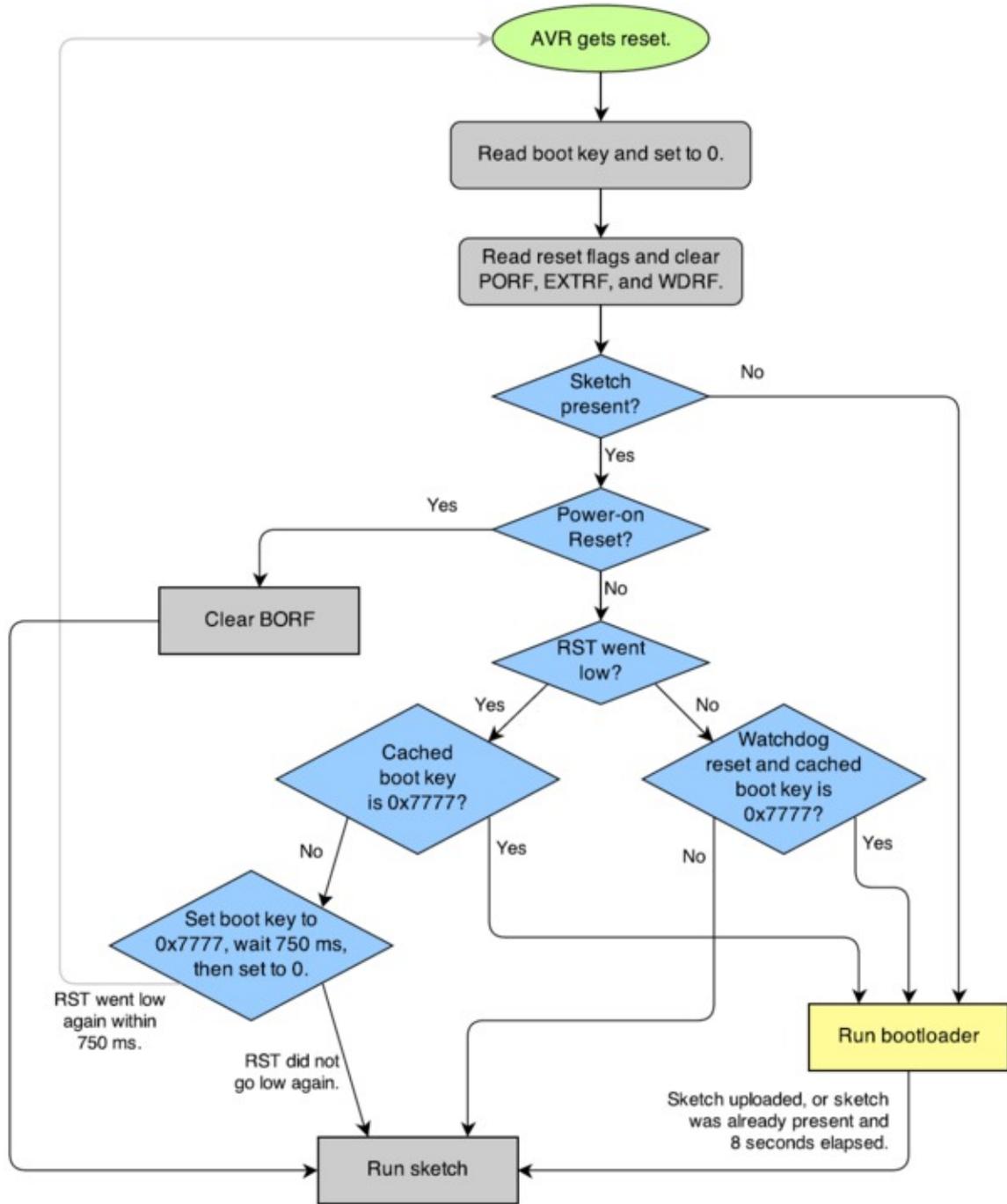
The startup logic for the A-Star 32U4 bootloader.

Brown-out detection

Unlike many other ATmega32U4 boards, the A-Stars have brown-out detection enabled. The brown-out threshold is 4.3 V, and if the voltage on VCC goes below this then the AVR will reset. The bootloader was designed so that the user program can detect brown-out resets. To do so, check to see if the BORF bit in the MCUSR register is set, and then clear it later. Here is some example code you could put in your `setup` function for detecting brown-out resets:

```
pinMode(13, OUTPUT);
if (MCUSR & (1 << BORF))
{
  // A brownout reset occurred.  Blink the LED
  // quickly for 2 seconds.
  for(uint8_t i = 0; i < 10; i++)
  {
    digitalWrite(13, HIGH);
    delay(100);
    digitalWrite(13, LOW);
    delay(100);
  }
}
MCUSR = 0;
```

10. Reviving an unresponsive A-Star

In order to load a new program onto your A-Star, you will need to get it into bootloader mode and send programming commands to it over its virtual serial port using appropriate software. If you are programming the A-Star from the Arduino IDE, the sketch loaded onto the A-Star will generally support a special USB command for putting the A-Star mode in bootloader mode, and the Arduino IDE sends that command automatically when you click the Upload button. However, you might find yourself in a situation where the A-Star is unresponsive and that method will not work. This can happen for two reasons:

- You accidentally loaded a malfunctioning program onto the A-Star that is incapable of responding to the special USB command. For example, your program might be stuck in an infinite loop with interrupts disabled.

- You loaded a program which uses a non-standard type of USB interface or no USB interface.

The following sections provide different procedures you can use to revive your A-Star.

10.1. Reviving using the Arduino IDE

This section explains two special methods for programming an A-Star using the Arduino IDE in case your usual method of programming is not working. These instructions were developed for the Arduino IDE versions 1.0.5-r2 and 1.5.6-r2, and they might need to be modified for future versions.

Reset button

If you have an A-Star 32U4 Micro, you should connect a **momentary pushbutton** [http://www.pololu.com/product/1400] between the GND and RST pins to serve as a reset button. The A-Star 32U4 Mini and A-Star 32U4 Prime boards have a reset button you can use. Alternatively, you can use a wire to temporarily connect GND and RST together instead of using a reset button.

Resetting the board twice within 750 ms makes the board go into bootloader mode. The bootloader will exit after 8 seconds and try to run the sketch again if it has not started receiving programming commands. To revive the A-Star, you need to make sure you start sending it programming commands before the 8-second period is over.

In bootloader mode, the yellow LED (the one labeled *LED 13*) fades in and out. It is useful to look at this LED so you can know what mode the microcontroller is in. Also, we recommend enabling verbose output during upload using the Arduino IDE's "Preferences" dialog. Looking at the LED and looking at the verbose output during the following procedures will help you understand what is going on.

The uploading-before-bootloader method (Arduino 1.0.x only)

The goal of the uploading-before-bootloader method is to select a non-existent serial port in the Arduino IDE and then make sure the Arduino IDE enters the uploading phase before the microcontroller goes into bootloader mode. This method does not work on the Arduino IDE 1.5.6-r2 and later because those versions of the IDE give a fatal error message if the selected serial port is not present at the beginning of the uploading phase (e.g. "Board at COM7 is not available.").

1. Connect the A-Star to your computer via USB.

2. In the "Tools" menu, open the "Board" sub-menu, and select "Pololu A-Star 32U4".

3. In the "Tools" menu, open the "Port" sub-menu, and check to see if any ports are selected. If the "Port" menu is grayed out or no ports in it are selected, that is good, and you can skip to step 6.

4. Reset the board twice to get the board into bootloader mode. While the board is in bootloader mode, quickly select the new serial port that corresponds to the bootloader in the "Port" menu.

5. After 8 seconds, the bootloader will exit and attempt to run the sketch again. Wait for the bootloader to exit. Verify that either the "Port" menu is grayed out or no ports in it are selected.

6. Click the Upload button. The Arduino IDE will compile your sketch and start uploading it.

7. As soon as the large status bar near the bottom of the IDE says "Uploading…", press reset the board twice to get into bootloader mode.

The Arduino IDE will stay in the uploading phase for 10 seconds, waiting for a new serial port to appear. Once the serial port of the bootloader appears, the Arduino IDE will connect to it and send programming commands.

The bootloader-before-uploading method

The goal of the bootloader-before-uploading method is to select the bootloader's virtual serial port in the Arduino IDE and then make sure the board is in bootloader mode at the time when the Arduino IDE enters the uploading phase.

1. Connect the A-Star to your computer via USB.

2. In the "Tools" menu, open the "Board" sub-menu and check to see if the "Pololu A-Star 32U4 (bootloader port)" entry is visible. If this entry is visible, you can skip to step 6.

3. If you are using a 1.0.x version of the Arduino IDE, open the file **[sketchbook location]/hardware/pololu/boards.txt** using a text editor. If you are using a 1.5.x version of the Arduino IDE, open the file **[sketchbook location]/hardware/pololu/avr/boards.txt** using a text editor. You can see the sketchbook location in the Arduino IDE preferences dialog. The file you are looking for is part of the A-Star add-on.

4. In the boards.txt file that you opened, find the lines at the bottom of the file that start with `#a-star32U4bp`. Uncomment each of those lines by deleting the "#" character, and then save the file.

5. Close the Arduino IDE and restart it.

6. In the "Tools" menu, open the "Board" sub-menu and select "Pololu A-Star 32U4 (bootloader port)". This entry is configured so that the Arduino IDE will send programming commands directly to selected serial port, instead of trying to send a special USB command to the port to get it into bootloader mode and then waiting for the new port to appear. By selecting this entry, the timing of the programming process below becomes easier, especially on Windows.

7. Prepare the computer to show you a list of its virtual serial ports. If you are using Windows, this means you should open the Device Manager. If you are on Linux or Mac OS X, this means you should open a Terminal and type the command `ls /dev/tty*` but do not press enter until the board is in bootloader mode in the next step.

8. Reset the board twice to get the board into bootloader mode. While it is in bootloader mode, quickly look at the list of serial ports provided by your operating system in order to determine what port the bootloader is assigned to.

9. Reset the board twice to get the board into bootloader mode again. While the board is in bootloader mode, quickly select the serial port of the bootloader in the Arduino IDE. The port can be selected in the "Port" sub-menu under "Tools".

10. In the Arduino IDE, click the "Verify" button to compile your sketch. This could make the timing easier during the next step.

11. Press the reset button twice to get the board into bootloader mode again. As soon as you see the yellow LED fading in and out, press the Upload button.

The Arduino IDE will compile your sketch and then upload it to the selected serial port.

If the compilation of the sketch takes longer than 8 seconds, then this procedure will fail because the bootloader will time out and start trying to run the malfunctioning sketch again. If that happens, try the procedure again using a simpler sketch such as the Blink example that can be found under **File > Examples > 01.Basics > Blink**.

10.2. Reviving using AVRDUDE

This section explains a special method for reviving an A-Star using the command-line utility **AVRDUDE** **[http://www.nongnu.org/avrdude/]** in case your usual method of programming is not working. AVRDUDE stands for "AVR Downloader/UploaDEr", and it is compatible with the A-Star bootloader.

If you have an A-Star 32U4 Micro, you should connect a **momentary pushbutton [http://www.pololu.com/product/1400]** between the GND and RST pins to serve as a reset button. The A-Star 32U4 Mini and A-Star 32U4 Prime boards have a reset button you can use. Alternatively, you can use a wire to temporarily connect GND and RST together instead of using a reset button.

1. Connect the A-Star to your computer via USB.

2. Prepare the computer to show you a list of its virtual serial ports. If you are using Windows, this means you should open the Device Manager. If you are on Linux or Mac OS X, this means you should open a Terminal and type the command `ls /dev/tty*` but do not press enter until the board is in bootloader mode in the next step.

3. Press the reset button twice within 750 ms to make the AVR go into bootloader mode. You should see the yellow LED fading in and out when the AVR is in bootloader mode. While it is in bootloader mode, quickly look at the list of serial ports provided by your operating system in order to determine what port the bootloader is assigned to.

4. Type the following command in your terminal and replace COM4 with the name of the bootloader's serial port, but do not press enter yet. This command will erase the malfunctioning program on the A-Star but preserve the bootloader.

    ```
    avrdude -c avr109 -p atmega32U4 -P COM4 -e
    ```

5. Press the reset button twice within 750 ms to make the AVR go into bootloader mode.

6. Quickly run the command you typed previously. The command needs to be run within 8 seconds of starting the bootloader, or else the bootloader will exit and try to run the malfunctioning program again.

By following the instructions above, the malfunctioning program on the A-Star will be erased and the A-Star will stay in bootloader mode indefinitely. You can now load another program onto it using the Arduino IDE or AVRDUDE.

11. Related Resources

To learn more about using the Pololu A-Star boards, see the following list of resources:

- The Arduino IDE has many **examples [http://arduino.cc/en/Tutorial/HomePage]** that can run on the A-Stars.

- The Arduino website has a **Language Reference [http://arduino.cc/en/Reference/HomePage]**, a wiki called the **The Arduino Playground [http://playground.arduino.cc/]**, and other resources.

- The A-Star baords are similar to the **Arduino Leonardo [http://www.pololu.com/product/2192]** and **Arduino Micro [http://www.pololu.com/product/2188]**, so you can search the Internet for relevant projects that use one of those boards.

- **Atmel's ATmega32U4 documentation [http://www.atmel.com/devices/atmega32u4.aspx]** has the ATmega32U4 datasheet and many related documents.

- **AVR Libc Home Page [http://www.nongnu.org/avr-libc/]**: this page documents the standard library of functions that you can use with GNU C and C++ compilers for the AVR.

- **AStar32U4Prime library documentation [https://pololu.github.io/a-star-32u4-prime/]**

- **LUFA – the Lightweight USB Framework for AVRs [http://www.fourwalledcubicle.com/LUFA.php]**

- **WinAVR [http://winavr.sourceforge.net/]**

- **Atmel Studio 6 [http://www.atmel.com/microsite/atmel_studio6/]**

- **AVRDUDE [http://www.nongnu.org/avrdude/]**

- **AVR Freaks [http://www.avrfreaks.net/]**

Finally, we would like to hear your comments and questions on the **A-Star section of the Pololu Robotics Forum [http://forum.pololu.com/viewforum.php?f=10]**!