I'm not a robot

 Microcontroller board Arduino UnoArduino Uno R3 SMD board with ATmega328P MCU in SMD packageDeveloperarduino.ccManufacturerManyTypeSingle-board microcontroller[1]AvailabilityUno R4 webpageOperating systemNone, with bootloader (default), FreeRTOSCPU Atmel AVR (8-bit) ARM Cortex-M0+ (32-bit) ARM Cortex-M3 (32-bit) Intel Quark (x86) (32-bit) MemorySRAMStorageFlash, EEPROMWebsitearduino.cc The Arduino Uno is a series of open-source microcontroller board based on a diverse range of microcontrollers (MCU). It was initially developed and released by Arduino company in 2010.[2][3] The microcontroller board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[1] The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.[4] It can be powered by a USB cable or a barrel connector that accepts voltages between 7 and 20 volts, such as a rectangular 9-volt battery. It has the same microcontroller as the Arduino Nano board, and the same headers as the Leonardo board.[5][6] The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available. The word "uno" means "one" in Italian and was chosen to mark a major redesign of the Arduino hardware and software.[7] The Uno board was the successor of the Duemilanove release and was the 9th version in a series of USB-based Arduino boards.[8] Version 1.0 of the Arduino IDE for the Arduino Uno board has now evolved to newer releases.[4] The ATmega328 on the board comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.[3] While the Uno communicates using the original STK500 protocol,[1] it differs from all preceding boards in that it does not use a FTDI USB-to-UART serial chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.[9] Arduino RS232 Serial board - a predecessor with ATmega8 MCU The Arduino project started at the Interaction Design Institute Ivrea (IDII) in Ivrea, Italy. At that time, the students used a BASIC Stamp microcontroller, at a cost that was a considerable expense for many students. In 2003, Hernando Barragán created the development platform Wiring as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas, who are known for work on the Processing language. The project goal was to create simple, low-cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a printed circuit board (PCB) with an ATmega168 microcontroller, an IDE based on Processing, and library functions to easily program the microcontroller.[10] In 2003, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, added support for the cheaper ATmega8 microcontroller to Wiring. But instead of continuing the work on Wiring, they forked the project and renamed it Arduino. Early Arduino boards used the FTDI USB-to-UART serial chip and an ATmega168.[10] The Uno differed from all preceding boards by featuring the ATmega328P microcontroller and an ATmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. In June 2023, Arduino released two new flavors of the Uno; R4 Minima and R4 Wifi. These mark a departure from previous boards as they use Renesas RA4M1 ARM Cortex M4 microcontroller, and the R4 Wifi a Espressif ESP32-S3-MINI co-processor. These versions are form factor, pin and power compatible with version R1 to R3, so should be largely be able to be drop in replacements.[11] Arduino Uno R3 board with AVR-based ATmega328P MCU in DIP-28 package Microcontroller (MCU):[12] IC: Microchip ATmega328P (8-bit AVR core) Clock Speed: 16 MHz on Uno board, though IC is capable of 20 MHz maximum at 5 Volts Flash memory: 32 KB, of which 0.5 KB used by the bootloader SRAM: 2 KB EEPROM: 1 KB USART peripherals: 1 (Arduino software default configures USART as a 8N1 UART) SPI peripherals: 1 I²C peripherals: 1 Operating Voltage: 5 Volts Digital I/O Pins: 14 PWM Pins: 6 (Pin # 3, 5, 6, 9, 10 and 11)[13] Analog Input Pins: 6 DC Current per I/O Pin: 20 mA DC Current for 3.3V Pin: 50 mA Size: 68.6 mm x 53.4 mm Weight: 25 g ICSP Header: Yes Power Sources: USB connector. USB bus specification has a voltage range of 4.75 to 5.25 volts. The official Uno boards have a USB-B connector, but 3rd party boards may have a miniUSB / microUSB / USB-C connector. 5.5mm/2.1mm barrel jack connector. Official Uno boards support 6 to 20 volts, though 7 to 12 volts is recommended. The maximum voltage for 3rd party Uno boards varies between board manufactures because various voltage regulators are used, each having a different maximum input rating. Power into this connector is routed through a series diode before connecting to VIN to protect against accidental reverse voltage situations. VIN pin on shield header. It has a similar voltage range of the barrel jack. Since this pin doesn't have reverse voltage protection, power can be injected or pulled from this pin. When supplying power into VIN pin, an external series diode is required in case barrel jack is used. When board is powered by barrel jack, power can be pulled out of this pin.[14] Arduino Uno R4 WiFi with ARM-based R7FA4M1AB MCU in 64pin SMD package Two Uno R4 boards are available: Uno R4 Minima and Uno R4 WiFi. The latter has a WiFi coprocessor and LED matrix, but the Minima doesn't. Common features on both Uno R4 Minima[15] and Uno R4 WiFi[16] boards: Microcontroller (MCU):[17] IC: Renesas R7FA4M1AB (32-bit ARM Cortex-M4F core with single-precision FPU) Clock Speed: 48 MHz Flash memory: 256 KB + bootrom SRAM: 32 KB (16 KB ECC) (16 KB parity) EEPROM: 8 KB (data flash) USART peripherals: 4 SPI peripherals: 2 I²C peripherals: 2 Operating Voltage: 5 Volts USB-C connector. Barrel jack connector and VIN pin on shield header supports up to a maximum of 24 volts DC. Additional features only available on the Uno R4 Minima board:[15] SWD programming connector. This is a 10-pin 5x2 1.27mm header for connecting the microcontroller (R7FA4M1AB) to an external SWD (serial wire debug) programming / debugging device. Additional features only available on the Uno R4 WiFi board:[16] WiFi coprocessor - 240 MHz Espressif ESP32-S3-MINI (IEEE802.11 b/g/n WiFi and Bluetooth 5 LE) and a 6-pin 3x2 2.54mm header for external programming. 12x8 LED matrix - it is driven by 11 GPIO pins using a charlieplexing scheme. Qwiic I²C connector. This 4-pin 1.00mm JST SH connector provides external connection to a 3.3 volt I²C bus. Don't attach 5 volt I²C devices directly to this connector.[18] RTC battery header pin (VRTC). This pin connects an external battery to the RTC (real-time clock) inside the microcontroller (R7FA4M1AB) to keep clock running when board is powered down. Connect this pin to positive side of 1.6 to 3.6 volt battery and negative side of battery to ground header pin (GND), such as a 3 volt lithium coin battery.[17] Remote-Off header pin (OFF). This pin disables the 5 volt buck switching voltage regulator (SL854102) when powered by the barrel jack or VIN header pin. Connect this pin to ground header pin (GND) to disable this voltage regulator. Header pinout of the Arduino Uno board LED: There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it is off. VIN: The input voltage to the Arduino/Genuino board when it is using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. 5V: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board. 3V3: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA. GND: Ground pins. IOREF: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source, or enable voltage translators on the outputs to work with the 5V or 3.3V. Reset: Typically used to add a reset button to shields that block the one on the board.[9] Each of the 14 digital pins and 6 analog pins on the Uno can be used as an input or output, under software control (using pinMode(), digitalWrite(), and digitalRead() functions). They operate at 5 volts. Each pin can provide or receive 20 mA as the recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50K ohm. A maximum of 40mA must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labeled A0 through A5; each provides 10 bits of resolution (i.e. 1024 different values). By default, they measure from ground to 5 volts, though it is possible to change the upper end of the range using the AREF pin and the analogReference() function.[9] In addition, some pins have specialized functions: Serial / UART: pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL serial chip. External interrupts: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. PWM (pulse-width modulation): pins 3, 5, 6, 9, 10, and 11. Can provide 8-bit PWM output with the analogWrite() function. SPI (Serial Peripheral Interface): pins 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK). These pins support SPI communication using the SPI library. TWI (two-wire interface) / I²C: pin SDA (A4) and pin SCL (A5). Support TWI communication using the Wire library. AREF (analog reference): Reference voltage for the analog inputs.[9] The Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows serial communication on any of the Uno's digital pins.[9] Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by the software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.[9] This setup has other implications. When the Uno is connected to a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened.[9] The following table compares official Arduino boards, and has a similar layout as a table in the Arduino Nano article. The table is split with a dark bar into two high-level microcontroller groups: 8-bit AVR cores (upper group), and 32-bit ARM Cortex-M cores (lower group). Though 3rd-party boards have similar board names it doesn't automatically mean they are 100% identical to official Arduino boards. 3rd-party boards often have a different voltage regulator / different USB-to-UART chip / different color solder mask, and some have a different USB connector or additional features, too. [19] BoardName& Part# BoardSizeGroup BoardCommun-ication MCUPart#& Pins MCUI/OVoltage MCUCore MCUClock MCUFlash MCUSRAM MCUEEPROM MCUUSART& UART MCUSPI MCUI²C MCUOther BusPeripherals MCU Timers32/24/16/8/WD/RT/RC MCUADC& DAC MCUEngines Uno R3,[20]A000066,[9]Uno R3 SMD,[21]A000073[22] Uno USB-B ATmega328P,[12]28 pin DIP,32 pin SMD 5V(1.8-5.5V) 8bit AVR 16 MHz* 32 KB 2 KB 1 KB 1, 0 1 1 None 0, 0, 1, 2,WD 10bit,None None Uno WiFi R2,[23]ABX00021[24] Uno USB-B,WiFi,Bluetooth ATmega4809,[25]48 pin 5V(1.8-5.5V) 8bit AVR 16 MHz* 48 KB 6 KB 0.25 KB 4, 0 1 1 None 0, 0, 5, 0,WD, RT 10bit,None None Leonardo,[26]A000057[27] Uno USB-Micro-B ATmega32U4,[28]44 pin 5V(2.7-5.5V) 8bit AVR 16 MHz 32 KB 2.5 KB 1 KB 1, 0 1 1 USB-FS 0, 0, 2, 1,WD, 10bit 10bit,None None Mega 2560 R3,[29]A000067[30] Mega USB-B ATmega2560,[31]100 pin 5V(4.5-5.5V) 8bit AVR 16 MHz 256 KB 8 KB 4 KB 4, 0 1 1 None 0, 0, 4, 2,WD 10bit,None None Uno R4 Minima,[15]ABX00080,[32]Uno R4 WiFi,[16]ABX00087,[33] Uno USB-C,WiFi* R7FA4M1AB,[17]64 pin 5V(1.6-5.5V) 32bit ARMCortex-M4F(FPU) 48 MHz 256 KB+ bootrom 32 KB(ECC)(parity) None+ 8 KBdata flash 4, 0 2 2 USB-FS,CAN-A/B 2, 0, 8, 0,WD, RC,24bit SysTick 14bit,12bit DMA x4,CRC, RNG,Crypto, Touch,LCD Zero,[34]ABX00003[35] Uno USB-Micro-Bx2 ATSAMD21G18,[36]48 pin 3.3V(1.62-3.63V) 32bit ARMCortex-M0+ 48 MHz 256 KB 32 KB None 6, 0 None None USB-FS,I²S 0, 4, 5, 0,WD, RC,24bit SysTick 12bit,10bit DMA x12,CRC32, Touch Due,[37]A000062[38] Mega USB-Micro-Bx2 ATSAM3X8E,[39]144 pin 3.3V(1.62-3.6V) 32bit ARMCortex-M3 84 MHz 512 KB+ bootrom 96 KB None 4, 1 1 2 USB-HS,CAN-A/B x2,I²S, SD 3, 0, 8, 0,WD, RT, RC,24bit SysTick 12bit,12bit x2 DMA x8,RNG GIGA R1 WiFi,[40]ABX00063[41] Mega USB-C,USB-A,WiFi,Bluetooth STM32H747XI,[42]240 pin 3.3V(1.62-3.6V) 32bit ARMCortex-M7FCortex-M4F(dual core)(FPU) 480 MHz(M7F),240 MHz(M4F) 2048 KB+ bootrom 1056 KB(ECC) None 4, 5 6 4 USB-HS & FS,CAN-A/B/FD x2,I²S x4, SD x2,S/PDIF x4, CEC,SWP, QSPI 2, 0, 18, 0,WD, RC,24bit SysTick 16bit x3,12bit x2 DMA x4,CRC, RNG,Graphics Arduino Leonardo board with ATmega32U4 MCU Arduino Due board with ATSAM3X8E MCU Table notes Board Size Group column - Simplified board dimension size grouping: Uno means similar size as Arduino Uno R3 and Duemilanove (predecessor) boards, Mega means similar size as the longer Arduino Mega 2560 R3 and Mega (predecessor) boards. This table has a similar layout as a table in the Arduino Nano article. MCU Part# / Pins column - MCU means microcontroller. All MCU information in this table was sourced from official datasheets in this column. The pin count is useful to determine the quantity of internal MCU features that are available. All MCU hardware features may not be available at the shield header pins because the MCU IC package has more pins than the shield header pins on the Arduino board (*). MCU I/O Voltage column - Microcontrollers on official Arduino boards are powered at a fixed voltage of either 3.3 or 5 volts, though some 3rd party boards have a voltage selection switch. The voltage rating of the microcontroller is stated inside parenthesis, though Arduino boards don't support this full range. MCU Clock column - MHz means 106 Hertz. The ATmega328P MPU and ATmega4809 MCU are rated for a maximum of 20 MHz, but the Uno R3 and Uno WiFi R2 boards both operate at 16 MHz. The following Arduino boards have a 32.768 kHz crystal too: Uno WiFi R2, Zero, Due, GIGA R1 WiFi. The Uno R4 Minima has SMD footprints for a 32.768KHz crystal and two capacitors, but aren't installed. MCU memory columns - KB means 1024 bytes, MB means 10242 bytes. The R7FA4M1AB MCU (Uno R4 boards) contains data flash memory instead of EEPROM memory. MCU SRAM column - SRAM size doesn't include caches or peripheral buffers. ECC means SRAM has error correction code checking, Par means SRAM has parity checking. MCU USART/UART column - USARTs are software configurable to be a: UART / SPI / other peripherals (varies across MCUs). MCU Other Bus Peripherals column - For USB bus, "FS" means Full Speed (12 Mbps max), "HS" means High Speed (480 Mbps max). For CAN bus, "A" means CAN 2.0A, "B" means CAN 2.0B, "FD" means CAN-FD. Some buses require additional external circuitry to operate. MCU Timers column - The numbers in this column are the total number of each timer bit width, for example, the ATmega328P has one 16-bit timer and two 8-bit timers. "WD" means Watchdog timer, "RT" means Real Time Counter/Timer, "RC" means Real Time Clock (sec/min/hr). The 24-bit SysTick timer(s) inside the ARM cores aren't included in the 24-bit total in this column. PWM features are not documented in this table. AVR microcontrollers Atmel AVR instruction set In-system programming ^ a b c "Arduino UNO for beginners - Projects, Programming and Parts". makerspaces.com. 7 February 2017. Retrieved 4 February 2018. ^ "Arduino FAQ". 5 April 2013. Archived from the original on 27 November 2020. Retrieved 21 February 2018. ^ a b "What is Arduino?". learn.sparkfun.com. Retrieved 4 February 2018. ^ a b "Introduction to Arduino" (PDF). princeton.edu. Archived from the original (PDF) on 3 April 2018. Retrieved 4 February 2018. ^ "Arduino Nano". Arduino Official Store. Retrieved 2022-12-07. ^ "Arduino Leonardo with Headers". Archived from the original on 2021-05-15. ^ "Previous IDE Releases". Retrieved 2023-02-08. ^ "Arduino Older Boards". Retrieved 2023-02-08. ^ a b c d e f g h "Board; Uno R3; Store". Arduino. ^ a b Hernando Barragán (2016-01-01). "The Untold History of Arduino". arduinohistory.github.io. Retrieved 2016-03-06. ^ "Introducing the Arduino UNO R4! - News - SparkFun Electronics". www.sparkfun.com. Retrieved 2023-08-07. ^ a b "MCU; ATmega328P; Docs". Microchip. Archived from the original on March 27, 2023. ^ "What is Arduino UNO? A Getting Started Guide". www.rs-online.com. Retrieved 2021-08-04. ^ "Using Vin pin on Arduino with a shield". Electrical Engineering Stack Exchange. Retrieved 2024-01-20. ^ a b c "Board; Uno R4 Minima; Docs". Arduino. Archived from the original on June 27, 2023. ^ a b c "Board; Uno R4 WiFi; Docs". Arduino. Archived from the original on June 27, 2023. ^ a b c "MCU; R7FA4M1AB; Docs". Renesas. Archived from the original on May 8, 2023. ^ "Qwiic I2C Connect System". SparkFun. Archived from the original on September 7, 2023. ^ "Seeeduino v4.3 (UNO ATmega328P)". Seeed Studio. Archived from the original on November 22, 2023. ^ "Board; Uno R3; Docs". Arduino. Archived from the original on May 17, 2023. ^ "Board; Uno R3 SMD; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Uno R3 SMD; Store". Arduino. ^ "Board; UNO WiFi R2; Docs". Arduino. Archived from the original on March 28, 2023. ^ "Board; Uno WiFi R2; Store". Arduino. ^ "MCU; ATmega4809; Docs". Microchip. Archived from the original on December 6, 2022. ^ "Board; Leonardo; Docs". Arduino. Archived from the original on April 5, 2023. ^ "Board; Leonardo; Store". Arduino. ^ "MCU; ATmega32U4; Docs". Microchip. Archived from the original on April 5, 2023. ^ "Board; Mega 2560 R3; Docs". Arduino. Archived from the original on April 21, 2023. ^ "Board; Mega 2560 R3; Store". Arduino. ^ "MCU; ATmeg2560; Docs". Microchip. Archived from the original on March 1, 2023. ^ "Board; Uno R4 Minima; Store". Arduino. ^ "Board; Uno R4 WiFi; Store". Arduino. ^ "Board; Zero; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Zero; Store". Arduino. ^ "MCU; ATSAMD21G18; Docs". Microchip. Archived from the original on February 1, 2023. ^ "Board; Due; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Due; Store". Arduino. ^ "MCU; ATSAM3X8E; Docs". Microchip. Archived from the original on October 26, 2022. ^ "Board; GIGA R1 WiFi; Docs". Arduino. Archived from the original on May 14, 2023. ^ "Board; GIGA R1 WiFi; Store". Arduino. ^ "MCU; STM32H747XI; Docs". ST. Archived from the original on May 11, 2023. Main article: List of books about Arduino Wikimedia Commons has media related to Arduino UNO. Arduino Uno official webpage What's the difference between UNO R3 and UNO R4 boards? Comparison of Various Arduino Boards Programming Cheat Sheets Sheet1, Sheet2 Pinout Diagrams Arduino Uno Board, ATmega328 DIP IC, ATmega328 SMD IC Electronic Schematics Uno "DIP" R3, Uno SMD R3, Uno R4 Minima, Uno R4 WiFi Differences Between Uno Board Revisions (R1/R2/R3) Mechanical Drawings Dimensions and Hole Patterns Dimensions, Hole Patterns, Header Locations and PCB Templates Retrieved from " Microcontroller board Arduino UnoArduino Uno R3 SMD board with ATmega328P MCU in SMD packageDeveloperarduino.ccManufacturerManyTypeSingle-board microcontroller[1]AvailabilityUno R4 webpageOperating systemNone, with bootloader (default), FreeRTOSCPU Atmel AVR (8-bit) ARM Cortex-M0+ (32-bit) ARM Cortex-M3 (32-bit) Intel Quark (x86) (32-bit) MemorySRAMStorageFlash, EEPROMWebsitearduino.cc The Arduino Uno is a series of open-source microcontroller board based on a diverse range of microcontrollers (MCU). It was initially developed and released by Arduino company in 2010.[2][3] The microcontroller board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[1] The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.[4] It can be powered by a USB cable or a barrel connector that accepts voltages between 7 and 20 volts, such as a rectangular 9-volt battery. It has the same microcontroller as the Arduino Nano board, and the same headers as the Leonardo board.[5][6] The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available. The word "uno" means "one" in Italian and was chosen to mark a major redesign of the Arduino hardware and software.[7] The Uno board was the successor of the Duemilanove release and was the 9th version in a series of USB-based Arduino boards.[8] Version 1.0 of the Arduino IDE for the Arduino Uno board has now evolved to newer releases.[4] The ATmega328 on the board comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.[3] While the Uno communicates using the original STK500 protocol,[1] it differs from all preceding boards in that it does not use a FTDI USB-to-UART serial chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.[9] Arduino RS232 Serial board - a predecessor with ATmega8 MCU The Arduino project started at the Interaction Design Institute Ivrea (IDII) in Ivrea, Italy. At that time, the students used a BASIC Stamp microcontroller, at a cost that was a considerable expense for many students. In 2003, Hernando Barragán created the development platform Wiring as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas, who are known for work on the Processing language. The project goal was to create simple, low-cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a printed circuit board (PCB) with an ATmega168 microcontroller, an IDE based on Processing, and library functions to easily program the microcontroller.[10] In 2003, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, added support for the cheaper ATmega8 microcontroller to Wiring. But instead of continuing the work on Wiring, they forked the project and renamed it Arduino. Early Arduino boards used the FTDI USB-to-UART serial chip and an ATmega168.[10] The Uno differed from all preceding boards by featuring the ATmega328P microcontroller and an ATmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. In June 2023, Arduino released two new flavors of the Uno; R4 Minima and R4 Wifi. These mark a departure from previous boards as they use Renesas RA4M1 ARM Cortex M4 microcontroller, and the R4 Wifi a Espressif ESP32-S3-MINI co-processor. These versions are form factor, pin and power compatible with version R1 to R3, so should be largely be able to be drop in replacements.[11] Arduino Uno R3 board with AVR-based ATmega328P MCU in DIP-28 package Microcontroller (MCU):[12] IC: Microchip ATmega328P (8-bit AVR core) Clock Speed: 16 MHz on Uno board, though IC is capable of 20 MHz maximum at 5 Volts Flash memory: 32 KB, of which 0.5 KB used by the bootloader SRAM: 2 KB EEPROM: 1 KB USART peripherals: 1 (Arduino software default configures USART as a 8N1 UART) SPI peripherals: 1 I²C peripherals: 1 Operating Voltage: 5 Volts Digital I/O Pins: 14 PWM Pins: 6 (Pin # 3, 5, 6, 9, 10 and 11)[13] Analog Input Pins: 6 DC Current per I/O Pin: 20 mA DC Current for 3.3V Pin: 50 mA Size: 68.6 mm x 53.4 mm Weight: 25 g ICSP Header: Yes Power Sources: USB connector. USB bus specification has a voltage range of 4.75 to 5.25 volts. The official Uno boards have a USB-B connector, but 3rd party boards may have a miniUSB / microUSB / USB-C connector. 5.5mm/2.1mm barrel jack connector. Official Uno boards support 6 to 20 volts, though 7 to 12 volts is recommended. The maximum voltage for 3rd party Uno boards varies between board manufactures because various voltage regulators are used, each having a different maximum input rating. Power into this connector is routed through a series diode before connecting to VIN to protect against accidental reverse voltage situations. VIN pin on shield header. It has a similar voltage range of the barrel jack. Since this pin doesn't have reverse voltage protection, power can be injected or pulled from this pin. When supplying power into VIN pin, an external series diode is required in case barrel jack is used. When board is powered by barrel jack, power can be pulled out of this pin.[14] Arduino Uno R4 WiFi with ARM-based R7FA4M1AB MCU in 64pin SMD package Two Uno R4 boards are available: Uno R4 Minima and Uno R4 WiFi. The latter has a WiFi coprocessor and LED matrix, but the Minima doesn't. Common features on both Uno R4 Minima[15] and Uno R4 WiFi[16] boards: Microcontroller (MCU):[17] IC: Renesas R7FA4M1AB (32-bit ARM Cortex-M4F core with single-precision FPU) Clock Speed: 48 MHz Flash memory: 256 KB + bootrom SRAM: 32 KB (16 KB ECC) (16 KB parity) EEPROM: 8 KB (data flash) USART peripherals: 4 SPI peripherals: 2 I²C peripherals: 2 Operating Voltage: 5 Volts USB-C connector. Barrel jack connector and VIN pin on shield header supports up to a maximum of 24 volts DC. Additional features only available on the Uno R4 Minima board:[15] SWD programming connector. This is a 10-pin 5x2 1.27mm header for connecting the microcontroller (R7FA4M1AB) to an external SWD (serial wire debug) programming / debugging device. Additional features only available on the Uno R4 WiFi board:[16] WiFi coprocessor - 240 MHz Espressif ESP32-S3-MINI (IEEE802.11 b/g/n WiFi and Bluetooth 5 LE) and a 6-pin 3x2 2.54mm header for external programming. 12x8 LED matrix - it is driven by 11 GPIO pins using a charlieplexing scheme. Qwiic I²C connector. This 4-pin 1.00mm JST SH connector provides external connection to a 3.3 volt I²C bus. Don't attach 5 volt I²C devices directly to this connector.[18] RTC battery header pin (VRTC). This pin connects an external battery to the RTC (real-time clock) inside the microcontroller (R7FA4M1AB) to keep clock running when board is powered down. Connect this pin to positive side of 1.6 to 3.6 volt battery and negative side of battery to ground header pin (GND), such as a 3 volt lithium coin battery.[17] Remote-Off header pin (OFF). This pin disables the 5 volt buck switching voltage regulator (SL854102) when powered by the barrel jack or VIN header pin. Connect this pin to ground header pin (GND) to disable this voltage regulator. Header pinout of the Arduino Uno board LED: There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it is off. VIN: The input voltage to the Arduino/Genuino board when it is using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. 5V: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board. 3V3: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA. GND: Ground pins. IOREF: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source, or enable voltage translators on the outputs to work with the 5V or 3.3V. Reset: Typically used to add a reset button to shields that block the one on the board.[9] Each of the 14 digital pins and 6 analog pins on the Uno can be used as an input or output, under software control (using pinMode(), digitalWrite(), and digitalRead() functions). They operate at 5 volts. Each pin can provide or receive 20 mA as the recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50K ohm. A maximum of 40mA must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labeled A0 through A5; each provides 10 bits of resolution (i.e. 1024 different values). By default, they measure from ground to 5 volts, though it is possible to change the upper end of the range using the AREF pin and the analogReference() function.[9] In addition, some pins have specialized functions: Serial / UART: pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL serial chip. External interrupts: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. PWM (pulse-width modulation): pins 3, 5, 6, 9, 10, and 11. Can provide 8-bit PWM output with the analogWrite() function. SPI (Serial Peripheral Interface): pins 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK). These pins support SPI communication using the SPI library. TWI (two-wire interface) / I²C: pin SDA (A4) and pin SCL (A5). Support TWI communication using the Wire library. AREF (analog reference): Reference voltage for the analog inputs.[9] The Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows serial communication on any of the Uno's digital pins.[9] Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by the software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.[9] This setup has other implications. When the Uno is connected to a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened.[9] The following table compares official Arduino boards, and has a similar layout as a table in the Arduino Nano article. The table is split with a dark bar into two high-level microcontroller groups: 8-bit AVR cores (upper group), and 32-bit ARM Cortex-M cores (lower group). Though 3rd-party boards have similar board names it doesn't automatically mean they are 100% identical to official Arduino boards. 3rd-party boards often have a different voltage regulator / different USB-to-UART chip / different color solder mask, and some have a different USB connector or additional features, too. [19] BoardName& Part# BoardSizeGroup BoardCommun-ication MCUPart#& Pins MCUI/OVoltage MCUCore MCUClock MCUFlash MCUSRAM MCUEEPROM MCUUSART& UART MCUSPI MCUI²C MCUOther BusPeripherals MCU Timers32/24/16/8/WD/RT/RC MCUADC& DAC MCUEngines Uno R3,[20]A000066,[9]Uno R3 SMD,[21]A000073[22] Uno USB-B ATmega328P,[12]28 pin DIP,32 pin SMD 5V(1.8-5.5V) 8bit AVR 16 MHz* 32 KB 2 KB 1 KB 1, 0 1 1 None 0, 0, 1, 2,WD 10bit,None None Uno WiFi R2,[23]ABX00021[24] Uno USB-B,WiFi,Bluetooth ATmega4809,[25]48 pin 5V(1.8-5.5V) 8bit AVR 16 MHz* 48 KB 6 KB 0.25 KB 4, 0 1 1 None 0, 0, 5, 0,WD, RT 10bit,None None Leonardo,[26]A000057[27] Uno USB-Micro-B ATmega32U4,[28]44 pin 5V(2.7-5.5V) 8bit AVR 16 MHz 32 KB 2.5 KB 1 KB 1, 0 1 1 USB-FS 0, 0, 2, 1,WD, 10bit 10bit,None None Mega 2560 R3,[29]A000067[30] Mega USB-B ATmega2560,[31]100 pin 5V(4.5-5.5V) 8bit AVR 16 MHz 256 KB 8 KB 4 KB 4, 0 1 1 None 0, 0, 4, 2,WD 10bit,None None Uno R4 Minima,[15]ABX00080,[32]Uno R4 WiFi,[16]ABX00087,[33] Uno USB-C,WiFi* R7FA4M1AB,[17]64 pin 5V(1.6-5.5V) 32bit ARMCortex-M4F(FPU) 48 MHz 256 KB+ bootrom 32 KB(ECC)(parity) None+ 8 KBdata flash 4, 0 2 2 USB-FS,CAN-A/B 2, 0, 8, 0,WD, RC,24bit SysTick 14bit,12bit DMA x4,CRC, RNG,Crypto, Touch,LCD Zero,[34]ABX00003[35] Uno USB-Micro-Bx2 ATSAMD21G18,[36]48 pin 3.3V(1.62-3.63V) 32bit ARMCortex-M0+ 48 MHz 256 KB 32 KB None 6, 0 None None USB-FS,I²S 0, 4, 5, 0,WD, RC,24bit SysTick 12bit,10bit DMA x12,CRC32, Touch Due,[37]A000062[38] Mega USB-Micro-Bx2 ATSAM3X8E,[39]144 pin 3.3V(1.62-3.6V) 32bit ARMCortex-M3 84 MHz 512 KB+ bootrom 96 KB None 4, 1 1 2 USB-HS,CAN-A/B x2,I²S, SD 3, 0, 8, 0,WD, RT, RC,24bit SysTick 12bit,12bit x2 DMA x8,RNG GIGA R1 WiFi,[40]ABX00063[41] Mega USB-C,USB-A,WiFi,Bluetooth STM32H747XI,[42]240 pin 3.3V(1.62-3.6V) 32bit ARMCortex-M7FCortex-M4F(dual core)(FPU) 480 MHz(M7F),240 MHz(M4F) 2048 KB+ bootrom 1056 KB(ECC) None 4, 5 6 4 USB-HS & FS,CAN-A/B/FD x2,I²S x4, SD x2,S/PDIF x4, CEC,SWP, QSPI 2, 0, 18, 0,WD, RC,24bit SysTick 16bit x3,12bit x2 DMA x4,CRC, RNG,Graphics Arduino Leonardo board with ATmega32U4 MCU Arduino Due board with ATSAM3X8E MCU Table notes Board Size Group column - Simplified board dimension size grouping: Uno means similar size as Arduino Uno R3 and Duemilanove (predecessor) boards, Mega means similar size as the longer Arduino Mega 2560 R3 and Mega (predecessor) boards. This table has a similar layout as a table in the Arduino Nano article. MCU Part# / Pins column - MCU means microcontroller. All MCU information in this table was sourced from official datasheets in this column. The pin count is useful to determine the quantity of internal MCU features that are available. All MCU hardware features may not be available at the shield header pins because the MCU IC package has more pins than the shield header pins on the Arduino board (*). MCU I/O Voltage column - Microcontrollers on official Arduino boards are powered at a fixed voltage of either 3.3 or 5 volts, though some 3rd party boards have a voltage selection switch. The voltage rating of the microcontroller is stated inside parenthesis, though Arduino boards don't support this full range. MCU Clock column - MHz means 106 Hertz. The ATmega328P MPU and ATmega4809 MCU are rated for a maximum of 20 MHz, but the Uno R3 and Uno WiFi R2 boards both operate at 16 MHz. The following Arduino boards have a 32.768 kHz crystal too: Uno WiFi R2, Zero, Due, GIGA R1 WiFi. The Uno R4 Minima has SMD footprints for a 32.768KHz crystal and two capacitors, but aren't installed. MCU memory columns - KB means 1024 bytes, MB means 10242 bytes. The R7FA4M1AB MCU (Uno R4 boards) contains data flash memory instead of EEPROM memory. MCU SRAM column - SRAM size doesn't include caches or peripheral buffers. ECC means SRAM has error correction code checking, Par means SRAM has parity checking. MCU USART/UART column - USARTs are software configurable to be a: UART / SPI / other peripherals (varies across MCUs). MCU Other Bus Peripherals column - For USB bus, "FS" means Full Speed (12 Mbps max), "HS" means High Speed (480 Mbps max). For CAN bus, "A" means CAN 2.0A, "B" means CAN 2.0B, "FD" means CAN-FD. Some buses require additional external circuitry to operate. MCU Timers column - The numbers in this column are the total number of each timer bit width, for example, the ATmega328P has one 16-bit timer and two 8-bit timers. "WD" means Watchdog timer, "RT" means Real Time Counter/Timer, "RC" means Real Time Clock (sec/min/hr). The 24-bit SysTick timer(s) inside the ARM cores aren't included in the 24-bit total in this column. PWM features are not documented in this table. AVR microcontrollers Atmel AVR instruction set In-system programming ^ a b c "Arduino UNO for beginners - Projects, Programming and Parts". makerspaces.com. 7 February 2017. Retrieved 4 February 2018. ^ "Arduino FAQ". 5 April 2013. Archived from the original on 27 November 2020. Retrieved 21 February 2018. ^ a b "What is Arduino?". learn.sparkfun.com. Retrieved 4 February 2018. ^ a b "Introduction to Arduino" (PDF). princeton.edu. Archived from the original (PDF) on 3 April 2018. Retrieved 4 February 2018. ^ "Arduino Nano". Arduino Official Store. Retrieved 2022-12-07. ^ "Arduino Leonardo with Headers". Archived from the original on 2021-05-15. ^ "Previous IDE Releases". Retrieved 2023-02-08. ^ "Arduino Older Boards". Retrieved 2023-02-08. ^ a b c d e f g h "Board; Uno R3; Store". Arduino. ^ a b Hernando Barragán (2016-01-01). "The Untold History of Arduino". arduinohistory.github.io. Retrieved 2016-03-06. ^ "Introducing the Arduino UNO R4! - News - SparkFun Electronics". www.sparkfun.com. Retrieved 2023-08-07. ^ a b "MCU; ATmega328P; Docs". Microchip. Archived from the original on March 27, 2023. ^ "What is Arduino UNO? A Getting Started Guide". www.rs-online.com. Retrieved 2021-08-04. ^ "Using Vin pin on Arduino with a shield". Electrical Engineering Stack Exchange. Retrieved 2024-01-20. ^ a b c "Board; Uno R4 Minima; Docs". Arduino. Archived from the original on June 27, 2023. ^ a b c "Board; Uno R4 WiFi; Docs". Arduino. Archived from the original on June 27, 2023. ^ a b c "MCU; R7FA4M1AB; Docs". Renesas. Archived from the original on May 8, 2023. ^ "Qwiic I2C Connect System". SparkFun. Archived from the original on September 7, 2023. ^ "Seeeduino v4.3 (UNO ATmega328P)". Seeed Studio. Archived from the original on November 22, 2023. ^ "Board; Uno R3; Docs". Arduino. Archived from the original on May 17, 2023. ^ "Board; Uno R3 SMD; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Uno R3 SMD; Store". Arduino. ^ "Board; UNO WiFi R2; Docs". Arduino. Archived from the original on March 28, 2023. ^ "Board; Uno WiFi R2; Store". Arduino. ^ "MCU; ATmega4809; Docs". Microchip. Archived from the original on December 6, 2022. ^ "Board; Leonardo; Docs". Arduino. Archived from the original on April 5, 2023. ^ "Board; Leonardo; Store". Arduino. ^ "MCU; ATmega32U4; Docs". Microchip. Archived from the original on April 5, 2023. ^ "Board; Mega 2560 R3; Docs". Arduino. Archived from the original on April 21, 2023. ^ "Board; Mega 2560 R3; Store". Arduino. ^ "MCU; ATmeg2560; Docs". Microchip. Archived from the original on March 1, 2023. ^ "Board; Uno R4 Minima; Store". Arduino. ^ "Board; Uno R4 WiFi; Store". Arduino. ^ "Board; Zero; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Zero; Store". Arduino. ^ "MCU; ATSAMD21G18; Docs". Microchip. Archived from the original on February 1, 2023. ^ "Board; Due; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Due; Store". Arduino. ^ "MCU; ATSAM3X8E; Docs". Microchip. Archived from the original on October 26, 2022. ^ "Board; GIGA R1 WiFi; Docs". Arduino. Archived from the original on May 14, 2023. ^ "Board; GIGA R1 WiFi; Store". Arduino. ^ "MCU; STM32H747XI; Docs". ST. Archived from the original on May 11, 2023. Main article: List of books about Arduino Wikimedia Commons has media related to Arduino UNO. Arduino Uno official webpage What's the difference between UNO R3 and UNO R4 boards? Comparison of Various Arduino Boards Programming Cheat Sheets Sheet1, Sheet2 Pinout Diagrams Arduino Uno Board, ATmega328 DIP IC, ATmega328 SMD IC Electronic Schematics Uno "DIP" R3, Uno SMD R3, Uno R4 Minima, Uno R4 WiFi Differences Between Uno Board Revisions (R1/R2/R3) Mechanical Drawings Dimensions and Hole Patterns Dimensions, Hole Patterns, Header Locations and PCB Templates Retrieved from " This chapter describes the general physical and electrical characteristics of specific Arduino boards, from the Diecimila through recent types like the Leonardo, Esplora, and Micro. Topics covered include pinout descriptions and the physical dimensions of most current Arduino models, from the so-called baseline types like the Uno, to the large form-factor Mega boards and the unique Esplora, to the small-outline boards such as the Mini, Micro, and Nano models. Table 4-1 is a comparison of the most common Arduino board types. If you compare this table with the tables in Chapter 1 it is obvious that the basic capabilities of an Arduino board are the capabilities supplied by its microcontroller. However, because the Arduino designs allocate certain pins on the AVR processors to specific functions, or don't bring out all of the processor's pins, not all of the capabilities of the microcontrollers are available at the terminals of an Arduino. The term "pin" is used in this and other sections when referring to the pin sockets on an Arduino. This is mainly to maintain consistency with the terminology encountered elsewhere, but it's not completely technically correct. The connection points on an Arduino board are sockets, and the jumpers and shields that plug into these sockets are the actual pins. You can think of a "pin" as a connection point of some sort, be it a lead on an IC package, a position on a 0.1 inch (2.54 mm) socket header, or the pins extending from the bottom of a shield PCB. Table 4-1. Arduino hardware features Board name Processor VCC (V) Clock (MHz) AIN pinsa DIO pinsb PWM pins USBc ArduinoBT ATmega328 5 16 6 14 6 None Duemilanove ATmega168 5 16 6 14 6 Regular Duemilanove ATmega328 5 16 6 14 6 Regular Diecimila ATmega168 5 16 6 14 6 Regular Esplora ATmega32U4 5 16 - - - Micro Ethernet ATmega328 5 16 6 14 4 Regular Fio ATmega328P 3.3 8 8 14 6 Mini Leonardo ATmega32U4 5 16 12 20 7 Micro LilyPad ATmega168V 2.7–5.5 8 6 14 6 None LilyPad ATmega328V 2.7–5.5 8 6 14 6 None Mega ATmega1280 5 16 16 54 15 Regular Mega ADK ATmega2560 5 16 16 54 15 Regular Mega 2560 ATmega2560 5 16 16 54 15 Regular Micro ATmega32U4 5 16 12 20 7 Micro Mini ATmega328 5 16 8 14 6 None Mini Pro ATmega168 3.3 8 6 14 6 None Mini Pro ATmega168 5 16 6 14 6 None Nano ATmega168 5 16 8 14 6 Mini-B Nano ATmega328 5 16 8 14 6 Mini-B Pro (168) ATmega168 3.3 8 6 14 6 None Pro (328) ATmega328 5 16 6 14 6 None Uno ATmega328 5 16 6 14 6 Regular Yún ATmega32U4 5 16 12 20 7 Host (A) Starting with the Leonardo board (2012), the ATmega32U4 XMEGA microcontroller has been used as the primary processor. This part has a built-in USB interface, which eliminates the need for the additional chip seen on earlier Arduino models with a USB interface. The Leonardo (2012), Esplora (2012), Micro (2012), and Yún (2013) all use the ATmega32U4 processor. The older Arduino models with USB used an FTDI interface chip (the FT232RL), an ATmega8 (Uno), or an ATmega16U2 (Mega2560 and Uno R3). The FT232RL converts between standard serial (such as RS-232) and USB. In the Uno, Uno R3, and Mega2560 the additional small ATmega processors are preprogrammed to serve as a USB interface. The operation of these parts is transparent when using the Arduino IDE to create and load program sketches. Those boards that do not have a USB interface must be programmed using an external adapter. Arduino types that use the FTDI FT232RL serial-to-USB interface chip are essentially identical internally, and consist of a DC voltage regulation circuit and two ICs. Figure 4-1 shows a block diagram of the Diecimila and Duemilanove models with an FTDI interface chip. Since around 2010, the Uno R2 and Uno SMD boards have employed the ATmega16U2 part instead of the FTDI FT232RL for the USB interface. The Uno R3 also has the ATmega16U2 serving as the USB interface. The ATmega16U2 incorporates a built-in USB 2.0 interface and is basically the same as the ATmega32U4, just with less memory. Figure 4-2 shows a block diagram of the Uno R2 with an ATmega16U2 providing the USB interface. The Uno, with an ATmega8, has the same internal functional arrangement as the Uno R2, just with a different MCU serving as the USB interface. Arduino boards come in a variety of shapes and sizes, but generally they can be organized into four groups: full-size or baseline boards, mega-size boards, small form-factor boards, and special-purpose boards. The board dimensions given in this section, while generally close, are approximate, as there may be some slight variations between boards from different sources. Refer to the PCB layout from Arduino.cc, which is available for each board, if you need accurate dimensions. Or better yet, just take the measurements from an actual board. To give some idea of scale, Figure 4-3 shows a lineup of several common Arduino boards. Shown here in clockwise order from the lower left are a Duemilanove, a Leonardo, a clone Mega2560 with an extended I/O pin layout from SainSmart, and an official Arduino Mega2560, with an Arduino Nano sitting in the center. Figure 4-4 shows the physical layouts for six different baseline Arduino boards, from the Diecimila to the Leonardo. In between there are the Duemilanove and Uno variants. Baseline, in this context, refers to the "classic" Arduino PCB layout that determines the physical design of most shields and other add-on components. The functions of the I/O and other pins on each PCB are described in "Arduino Pinout Configurations". With the Diecimila, the Duemilanove, the Uno R2 (revision 2), and the Uno SMD the arrangement of the I/O socket headers along the edges of the PCBs is unchanged. This book refers to this as the baseline Arduino form factor. Also, starting with the Uno R2 a new block of six pins appeared on the PCB, in addition to the block that already existed on earlier boards. This is the ICSP (In-Circuit Serial Programming) interface for the ATmega16U2 processor that is used for the USB interface. The Uno SMD also has this new ICSP interface. The Uno R3 introduced the new extended I/O pin configuration. This is a backward-compatible extension, meaning that a shield intended for an older model like a Duemilanove will still work with the newer boards. The extension only adds new signal pin sockets, but no new signals, and it doesn't alter any of the pin functions found in the baseline layout. The Leonardo PCB uses the ATmega32U4 processor, which has built-in USB support, so there is only one microcontroller IC on a Leonardo PCB and only one ICSP port. It has the same I/O pin layout as earlier boards, although the actual microcontroller ports used are different. All full-size baseline Arduino boards have the same physical dimensions, as shown in Figure 4-5. The locations of the mounting holes for the PCB vary slightly between models depending on the version of the board, but the overall exterior dimensions are consistent. The Mega form-factor boards incorporate the baseline pinout along with additional pins to accommodate the extended capabilities of the ATmega1280 and ATmega2560 microcontrollers (these devices are described in "ATmega1280/ATmega2560" in Chapter 3). The Mega and Mega2560 are essentially the same layout, with the primary difference being the type of AVR device on the boards. The Mega2560 replaces the Mega, which is no longer produced by Arduino.cc, although some second-source clone boards are still available. Considering the enhanced memory of the Mega2560, there really isn't any reason to purchase a Mega. Figure 4-6 shows the overall dimensions of a Mega or Mega2560 board. Note that a baseline (Uno, Leonardo, etc.) shield will work with a Mega board. The I/O pins on the Mega are arranged such that the basic digital I/O and the A/D inputs 0 through 5 are compatible with the baseline pin layout. The Mega ADK is based on the Mega2560, but features a USB host interface that allows it to connect to Android phones and similar devices. Other than an additional USB connector located between the B type USB connector and the DC power jack, it is identical to the Mega2560 in terms of dimensions. Like with the Mega2560, a standard baseline-type shield can be used with the Mega ADK. The full-size Arduino boards were the first to make an appearance around 2005, and by 2007 the layout had settled into the baseline and extended forms seen today. But the Arduino team and their partners realized that the full-size board just wouldn't work for some applications, so they came up with the miniature formats. The miniature boards include the Mini, Micro, Nano, and Fio layouts. These PCBs are smaller in both width and length, but still have the same AVR processors as the full-size types. The Mini is intended for use on breadboards or in other applications where space is limited. It does not have a USB connector, and an external programmer interface must be used to transfer executable code to the microcontroller. Its dimensions are shown in Figure 4-7. The Pro Mini is similar to the Mini with regard to pin layout and form factor, but unlike the Mini it is intended for permanent or semipermanent installation. The Pro Mini was designed and manufactured by SparkFun Electronics. Its dimensions are shown in Figure 4-8. Similar to the Mini, the Nano is a small form-factor board suitable for use with solderless breadboards and as a plug-in module for a larger PCB. It was designed and produced by Gravitech. Its dimensions are given in Figure 4-9. The Fio is intended for wireless applications, primarily XBee, and as such it lacks some of the direct connection programmability of other Arduino types. A Fio can be programmed using a serial-to-USB adapter or wirelessly using a USB-to-XBee adapter. It was designed and manufactured by SparkFun Electronics; its dimensions are shown in Figure 4-10. The Micro employs a DIP (dual in-line pin) form factor and uses an ATmega32U4 processor, which is identical to the Leonardo board. Like the Nano, the Micro is suitable for use with a solderless breadboard and as a plug-in module using a conventional IC socket. It was developed in conjunction with Adafruit. The Micro's dimensions are shown in Figure 4-11. Arduino boards aren't limited to simple shapes like rectangles. The LilyPad is a small disk with connection points arranged around the edge. It can be integrated into clothing to build wearable creations. The Esplora is physically configured like a conventional game controller, although as it is an Arduino it can be programmed to do much more than just play games. The LilyPad and its variations are intended for wearable applications. The board itself measures about 2 inches (50 mm) in diameter, as shown in Figure 4-12. The Esplora is supplied with four pushbuttons, a switch-type joystick, and a micro USB connector. Four mounting holes are available to affix the board to a chassis or panel. The Esplora PCB dimensions are shown in Figure 4-13. When creating a shield board for the Arduino, the convention is to follow the common baseline pin layout pattern described here. This configuration is found on the "standard" baseline Arduino boards built between 2007 and 2012. Boards using the newer "extended" pin layout (the Uno R3 and Leonardo), as well as the "Mega" boards, also support the baseline connections, but add new capabilities by extending the rows of terminals along the sides of the PCBs. The baseline Arduino pin layout as it exists today appeared with the Diecimila model. Over the years it has become a de facto standard upon which numerous shield boards have been based. The Arduino boards that utilize the baseline pin layout are listed in Table 4-2. Table 4-2. Baseline layout Arduino boards Board name Year Microcontroller Diecimila 2007 ATmega168 Duemilanove 2008 ATmega168/ATmega328 Uno (R2 and SMD) 2010 ATmega328P Figure 4-14 shows the pinout of a full-size baseline Arduino board. This includes the Diecimila, Duemilanove, Uno R2, and Uno SMD models. The gray boxes in Figure 4-14 give the chip pin number and port designations for the ATmega168/328 parts. The common baseline I/O and power pin layout for the Arduino consists of 14 discrete digital I/O pins, an analog reference, 3 ground pins, 6 analog input pins, pins for 3.3V and 5V, and a reset line. As shown in Figure 4-14, these pins are arranged as two eight-position connectors and two six-position connectors along the sides of the PCB. From a programming perspective, each interface pin on a Diecimila, Duemilanove, Uno R2, or Uno SMD PCB has a predefined name used to identify it in software. These names are reflected by the labels screened onto the Arduino PCB. Table 4-3 lists the pin assignments for a baseline or R2 Arduino with an ATmega168 or ATmega328 MCU. See the pin assignments for the Arduino Ethernet (Table 4-3) for the Uno SMD board. Table 4-3. Arduino ATmega168/328 pin assignments Digital pin (Dn) Analog pin (An) AVR pin AVR port AVR function(s) AVR PWM 0 2 PD0 RxD 1 3 PD1 TxD 2 4 PD2 INT0 3 5 PD3 INT1, OC2B Yes 4 6 PD4 T0, XCK 5 11 PD5 T1 Yes 6 12 PD6 AIN0 Yes 7 13 PD7 AIN1 8 14 PB0 CLK0, ICP1 9 15 PB1 OC1A Yes 10 16 PB2 OC1B, SS Yes 11 17 PB3 OC2A, MOSI Yes 12 18 PB4 MISO 13 19 PB5 SCK 14 0 23 PC0 15 1 24 PC1 16 2 25 PC2 17 3 26 PC3 18 4 27 PC4 SDA 19 5 28 PC5 SCL Starting with the R3 version of the Uno, four additional pins appeared on the Arduino PCB. Two of these are near the relocated reset button and provide additional connections for I2C (the SCL and SDA lines). The other two appeared next to the reset connection on the opposite side of the board. One is designated as IOREF (the nominal I/O voltage, may be either 3.3V or 5V depending on board type) and the other is not presently connected. Table 4-4 lists the extended baseline layout boards. Table 4-4. Extended layout Arduino boards Board name Year Microcontroller Uno R3 2010 ATmega328 Ethernet 2011 ATmega328 Leonardo 2012 ATmega32U4 Like the Uno R2 and Uno SMD, the Uno R3 utilizes a second microcontroller to handle USB communications. The Arduino Ethernet does not have built-in USB. Figure 4-15 shows the block diagram for the Uno R3 and Uno SMD boards. The pin functions for the Uno R3 are shown in Figure 4-16. The extended baseline (R3) Arduino boards with the ATmega328 MCU have the same pin assignments as given in Table 4-3, but with the additional pins for ADC4 and ADC5 (A4 and A5). The Leonardo pin functions are defined in "Leonardo". The Ethernet deviates from the Arduino conventions seen up through the Uno R3 with its inclusion of a 100Mb Ethernet interface and an RJ45 jack. It has no USB interface. The MCU is a surface-mount version of the ATmega328, with different pin functions and numbering from the ATmega328. A WIZnet W5100 chip is used for the Ethernet interface. Figure 4-17 shows a block diagram of the Ethernet board. An FTDI-type interface is used to program the Ethernet with an adapter, like the SparkFun or Adafruit FTDI-type devices. This interface is brought out on a right-angle six-pin header along one edge of the PCB next to the microSD carrier. Figure 4-18 shows the pinouts of the Ethernet board. This product has been retired by Arduino.cc, but it is still available from multiple sources. Ethernet connectivity can be obtained by using an Ethernet shield (see Chapter 8 for more details on shields). Table 4-5 lists the pin assignments for the Arduino Ethernet. Note that pins 10, 11, 12, and 13 are reserved for the Ethernet interface and are not available for general-purpose use. Table 4-5. Arduino Ethernet pin assignments Digital pin (Dn) Analog pin (An) AVR pin AVR port AVR function(s) AVR PWM 0 30 PD0 RxD 1 31 PD1 TxD 2 32 PD2 INT0 3 1 PD3 INT1, OC2B Yes 4 2 PD4 T0, XCK 5 9 PD5 T1, OC0B Yes 6 10 PD6 AIN0, OC0A Yes 7 11 PD7 AIN1 8 12 PB0 CLK0, ICP1 9 13 PB1 OC1A Yes 10 14 PB2 OC1B, SS Yes 11 15 PB3 OC2A, MOSI Yes 12 16 PB4 MISO 13 17 PB5 SCK 14 0 23 PC0 15 1 24 PC1 16 2 25 PC2 17 3 26 PC3 18 4 27 PC4 SDA 19 5 28 PC5 SCL The Leonardo introduced the ATmega32U4 processor, which contains a built-in USB interface and enhanced functionality. This simplified the PCB layout, as can be seen in Figure 4-19. Also, note that the Leonardo uses a mini-USB connector instead of the full-size type B connector found on older Arduino boards. This was a much-needed change, and it allows the Leonardo to work with shields that would have interfered with the B type USB connector on the older models. The Uno R3 and the Leonardo both use the same PCB pin layout, but some of the microcontroller functions are different. In the Arduino IDE this is handled by using a set of definitions specific to each board type to map functions to specific pins. Table 4-6 lists the pin assignments for an extended or R3 Arduino with an ATmega32U4 MCU. Table 4-6. Arduino ATmega32U4 pin assignments Digital pin (Dn) Analog pin (An) AVR pin AVR port AVR function(s) AVR PWM 0 20 PD2 INT3, RxD1 1 21 PD3 INT2, TxD1 2 19 PD1 INT1, SDA 3 18 PD0 INT0, OC0B, SCL Yes 4 25 PD4 ICP1, ADC8 5 31 PC6 OC3A, OC4A Yes 6 27 PD7 OC4D, ADC10, T0 Yes 7 1 PE6 INT6, AIN0 8 28 PB4 ADC11 9 29 PB5 OC1A, ADC12, *OC4B Yes 10 30 PB6 OC1B, ADC13, OC4B Yes 11 12 PB7 OC0A Yes 12 26 PD6 *OC4D, ADC9, T1 Yes 13 32 PC7 OC4A, ICP3 Yes 14 0 36 PF7 TDI 15 1 37 PF6 TDO 16 2 38 PF5 TMS 17 3 39 PF4 TCK 18 4 40 PF1 19 5 41 PF0 The Mega series (which use the ATmega1280 and ATmega2560 processors) also incorporate the standard pinout pattern, but include additional pins to accommodate the extended I/O capabilities of the larger processors. The Mega pin layout is shown in Figure 4-20. The boards that utilize this layout are listed in Table 4-7. Most common shields will work with the Mega boards. In Figure 4-20, the PCINT pin functions are not shown for the sake of clarity. Also note that the R3 version of the Mega2560 contains pins not found on earlier versions, but these do not interfere with baseline layout–style shields. Table 4-7. Mega pin layout boards Board name Year Microcontroller Mega 2009 ATmega1280 Mega2560 2010 ATmega2560 Mega ADK 2011 ATmega2560 In terms of nonstandard pinout configurations (nonstandard in the sense of being physically incompatible with conventional Arduino shields), the most radical is the LilyPad, with its circular form factor and use of solder pads for connections. The small form-factor Nano, Mini, Mini Pro, and Micro have pins soldered to the underside of the board, and are suitable for use with a solderless breadboard block or as a component on a large PCB. The Fio uses solder pads with spacing compatible with standard header pin strips, and the Esplora has a game controller–type form factor. None of the nonstandard layout boards can be used directly with a standard shield. The boards that fall into this category are listed in Table 4-8, and the pin functions for these boards are shown in Figures 4-21 through 4-27 in the following sections. Table 4-8. Nonstandard pin layout boards Board name Year Microcontroller LilyPad 2007 ATmega168V/ATmega328V Nano 2008 ATmega328/ATmega168 Mini 2008 ATmega168 Pro Mini 2008 ATmega328 Fio 2010 ATmega328P Esplora 2012 ATmega32U4 Micro 2012 ATmega32U4 The Atmel website offers a selection of datasheets, example software, and other resources for working with the AVR microcontrollers. Note that these are only for the AVR, not the Arduino. Additional information about the various Arduino boards can be found on the Arduino website. Get Arduino: A Technical Reference now with the O'Reilly learning platform. O'Reilly members experience books, live events, courses curated by job role, and more from O'Reilly and nearly 200 top publishers. The Arduino Uno is one kind of microcontroller board based on ATmega328, and Uno is an Italian term which means one. Arduino Uno is named for marking the upcoming release of microcontroller board namely Arduino Uno Board 1.0. This board includes digital I/O pins-14, a power jack, analog i/ps-6, ceramic resonator-A16 MHz, a USB connection, an RST button, and an ICSP header. All these can support the microcontroller for further operation by connecting this board to the computer. The power supply of this board can be done with the help of an AC to DC adapter, a USB cable, otherwise a battery. This article discusses what is an Arduino Uno microcontroller, pin configuration, Arduino Uno specifications or features, and applications. What is Arduino Uno ATmega328? The ATmega328 is one kind of single-chip microcontroller formed with Atmel within the megaAVR family. The architecture of this Arduino Uno is a customized Harvard architecture with 8 bit RISC processor core. Other boards of Arduino Uno include Arduino Pro Mini, Arduino Nano, Arduino Due, Arduino Mega, and Arduino Leonardo. Arduino Uno ATmega328 Features of Arduino Uno Board The features of Arduino Uno ATmega328 includes the following. The operating voltage is 5V The recommended input voltage will range from 7v to 12V The input voltage ranges from 6v to 20V Digital input/output pins are 14 Analog i/p pins are 6 DC Current for each input/output pin is 40 mA DC Current for 3.3V Pin is 50 mA Flash Memory is 32 KB SRAM is 2 KB EEPROM is 1 KB CLK Speed is 16 MHz Arduino Uno Pin Diagram The Arduino Uno board can be built with power pins, analog pins, ATmegs328, ICSP header, Reset button, power LED, digital pins, test led 13, TX/RX pins, USB interface, an external power supply. The Arduino UNO board description is discussed below. Arduino Uno Board Pin Configuration Power Supply The Arduino Uno power supply can be done with the help of a USB cable or an external power supply. The external power supplies mainly include AC to DC adapter otherwise a battery. The adapter can be connected to the Arduino Uno by plugging into the power jack of the Arduino board. Similarly, the battery leads can be connected to the Vin pin and the GND pin of the POWER connector. The suggested voltage range will be 7 volts to 12 volts. Input & Output The 14 digital pins on the Arduino Uno can be used as input & output with the help of the functions like pinMode(), digitalWrite(), & Digital Read(). Pin1 (TX) & Pin0 (RX) (Serial): This pin is used to transmit & receive TTL serial data, and these are connected to the ATmega8U2 USB to TTL Serial chip equivalent pins. Pin 2 & Pin 3 (External Interrupts): External pins can be connected to activate an interrupt over a low value, change in value. Pins 3, 5, 6, 9, 10, & 11 (PWM): This pin gives 8-bit PWM o/p by the function of analogWrite(). SPI Pins (Pin-10 (SS), Pin-11 (MOSI), Pin-12 (MISO), Pin-13 (SCK): These pins maintain SPI-communication, even though offered by the fundamental hardware, is not presently included within the Arduino language. Pin-13(LED): The inbuilt LED can be connected to pin-13 (digital pin). As the HIGH-value pin, the light emitting diode is activated, whenever the pin is LOW. Pin-4 (SDA) & Pin-5 (SCL) (I2C): It supports TWI-communication with the help of the Wire library. AREF (Reference Voltage): The reference voltage is for the analog i/ps with analogReference(). Reset Pin: This pin is used for reset (RST) the microcontroller. Memory The memory of this Atmega328 Arduino microcontroller includes flash memory-32 KB for storing code, SRAM-2 KB EEPROM-1 KB. Communication The Arduino Uno ATmega328 offers UART TTL-serial communication, and it is accessible on digital pins like TX (1) and RX (0). The software of an Arduino has a serial monitor that permits easy data. There are two LEDs on the board like RX & TX which will blink whenever data is being broadcasted through the USB. A SoftwareSerial library permits for serial communication on Arduino Uno digital pins and the ATmega328P supports TWI (I2C) as well as SPI-communication. The Arduino software contains a wired library for simplifying the utilization of the I2C bus. How to Use an Arduino Uno? Arduino Uno can detect the surroundings from the input. Here the input is a variety of sensors and these can affect its surroundings through controlling motors, lights, other actuators, etc. The ATmega328 microcontroller on the Arduino board can be programmed with the help of an Arduino programming language and the IDE (Integrated Development Environment). Arduino projects can communicate by software while running on a PC. Arduino Programming Once the Arduino IDE tool is installed in the PC, attach the Arduino board to the computer with the help of USB cable. Open the Arduino IDE & select the right board by choosing Tools–>Board..>Arduino Uno, and select the right Port by choosing Tools–>Port. This board can be programmed with the help of an Arduino programming language depends on Wiring. To activate the Arduino board & flash the LED on the board, dump the program code with the selection of Files–> Examples..>Basics..>Flash. When the programming codes are dumped into the IDE, and then click the button 'upload' on the top bar. Once this process is completed, check the LED flash on the board. High Voltage Protection of USB The Arduino Uno board has a rearrangeable poly fuse that defends the USB port of the PC from the over-voltage. Though most of the PCs have their own inner protection, the fuse gives an additional coating of safety. If above 500mA is given to the USB port, then the fuse will routinely crack the connection until the over-voltage is removed. Physical Characteristics The physical characteristics of an Arduino board mainly include length and width. The printed circuit board of the Arduino Uno length and width are 2.7 X 2.1 inches, but the power jack and the USB connector will extend beyond the previous measurement. The board can be attached on the surface otherwise case with the screw holes. Applications of Arduino Uno ATmega328 The applications of Arduino Uno include the following. Arduino Uno is used in Do-it-Yourself projects prototyping. In developing projects based on code-based control Development of Automation System Designing of basic circuit designs. Thus, this is all about Arduino Uno datasheet. From the above information finally, we can conclude that this is an 8-bit ATmega328P microcontroller. It has different components like serial communication, crystal oscillator, the voltage regulator for supporting the microcontroller. This board includes a USB connection, digital I/O pins-14, analog i/p pins-6, a power-barrel jack, a reset button, and an ICSP header. Here is a question for you, what is the Arduino Uno price in India? Family of microcontrollers This article is about the series of AVR microcontrollers. For the AVR instruction set, see Atmel AVR instruction set. Not to be confused with automatic voltage regulator. AVR logo Various older AVR microcontrollers: ATmega8 in 28-pin narrow dual in-line package (DIP-28N), ATxmega128A1 in 100-pin thin quad flat pack (TQFP-100) package, ATtiny45 in 8-pin small outline (SO-8) package ATmega328P in 28-pin narrow dual in-line package (DIP-28N). It is commonly found on Arduino boards. AVR is a family of microcontrollers developed since 1996 by Atmel, acquired by Microchip Technology in 2016. They are 8-bit RISC single-chip microcontrollers based on a modified Harvard architecture. AVR was one of the first microcontroller families to use on-chip flash memory for program storage, as opposed to one-time programmable ROM, EPROM, or EEPROM used by other microcontrollers at the time. AVR microcontrollers are used numerously as embedded systems. They are especially common in hobbyist and educational embedded applications, popularized by their inclusion in many of the Arduino line of open hardware development boards. The AVR 8-bit microcontroller architecture was introduced in 1997. By 2003, Atmel had shipped 500 million AVR flash microcontrollers.[1] The AVR architecture was conceived by two students at the Norwegian Institute of Technology (NTH),[2] Alf-Egil Bogen[3] and Vegard Wollan.[4] Atmel says that the name AVR is not an acronym and does not stand for anything in particular. The creators of the AVR give no definitive answer as to what the term "AVR" stands for.[4] However, it is commonly accepted that AVR stands for Alf and Vegard's RISC processor.[5] Note that the use of "AVR" in this article generally refers to the 8-bit RISC line of Atmel AVR microcontrollers. The original AVR MCU was developed at a local ASIC house[clarification needed] in Trondheim, Norway, called Nordic VLSI at the time, now Nordic Semiconductor, where Bogen and Wollan were working as students.[citation needed] It was known as a μRISC (Micro RISC)[6] and was available as silicon IP/building block from Nordic VLSI.[7] When the technology was sold to Atmel from Nordic VLSI, the internal architecture was further developed by Bogen and Wollan at Atmel Norway, a subsidiary of Atmel. The designers worked closely with compiler writers at IAR Systems to ensure that the AVR instruction set provided efficient compilation of high-level languages.[8] Among the first of the AVR line was the AT90S8515, which in a 40-pin DIP package has the same pinout as an 8051 microcontroller, including the external multiplexed address and data bus. The polarity of the RESET line was opposite (8051's having an active-high RESET, while the AVR has an active-low RESET), but other than that the pinout was identical. The Arduino platform, developed for simple electronics projects, was released in 2005 and featured ATmega8 AVR microcontrollers. The AVR is a modified Harvard architecture machine, where program and data are stored in separate physical memory systems that appear in different address spaces, but having the ability to read data items from program memory using special instructions. AVRs are generally classified into following: tinyAVR – the ATtiny series Main article: ATtiny microcontroller comparison chart Flash size Frequency[MHz] Package SRAM EEPROM 0.5–32 KB 1.6–20 6–32-pin package 32–3072 bytes 64–512 bytes The ATtiny series features small package microcontrollers with a limited peripheral set available. However, the improved tinyAVR 0/1/2-series (released in 2016) include: Peripherals equal to or exceed megaAVR 0-series Event System Improved AVRxt instruction set (improved timing of calls), hardware multiply megaAVR – the ATmega series Flash size Frequency[MHz] Package SRAM EEPROM 4–256 KB 1.6–20 28–100-pin package 256–16384 bytes 256–4096 bytes The ATmega series features microcontrollers that provide an extended instruction set (multiply instructions and instructions for handling larger program memories), an extensive peripheral set, a solid amount of program memory, as well as a wide range of pins available. The megaAVR 0-series (released in 2016) also has functionality such as: Event system New peripherals with enhanced functionality Improved AVRxt instruction set (improved timing of calls) AVR Dx – The AVR Dx family features multiple microcontroller series, focused on HCI, analog signal conditioning and functional safety. Flash size Frequency[MHz] Package SRAM EEPROM Release year 16–128 KB 20–24 at 1.8–5.5 V 14–64-pin package 4–16 KB 512 bytes 2020 The parts numbers is formatted as AVRffDxpp, where ff is flash size, x is family, and pp is number of pins. Example: AVR128DA64 – 64-pin DA-series with 128k flash. All devices in the AVR Dx family include: an Async Type D timer that can run faster than the CPU 12-bit ADC 10-bit DAC AVR DA-series (early 2020) – The high memory density makes these MCUs well suited for both wired and wireless communication-stack-intensive functions. integrated sensors for capacitative touch measurement (HCI) updated core independent peripherals (CIPs) and analog peripherals no external high frequency crystal AVR DB-series (mid-late 2020) – inherits many features from the DA-family, while adding its own: 2 or 3 on-chip opamps MultiVoltage IO (MVIO) on PORTC Supports external HF crystal AVR DD-series 16–64 KiB Flash 2–8 KiB SRAM 14–32-pin package internal 24 MHz oscillator 7–23-channel 130 kS/s 12-bit differential Analog-to-Digital Converter (ADC) no amplifiers 1 analog comparator Two USARTs, one SPI, one dual-mode TWI Multi-Voltage Input/Output (MVIO) support on 3 or 4 pins on Port C 4 Configurable Custom Logic (CCL) cells, 6 Event System channels AVR EA-series 8–64 KiB Flash 28–48-pin package internal 20 MHz oscillator 24–32-channel 130 kS/s 12-bit differential Analog-to-Digital Converter (ADC) Programmable Gain Amplifier (PGA) with up to 16x gain 2 analog comparators Three USARTs, one SPI, one dual-mode TWI no Multi-Voltage Input/Output (MVIO) 4 Configurable Custom Logic (CCL) cells, 6 Event System channels XMEGA Flash size Frequency[MHz] Package SRAM EEPROM Release year 16–256 KB 32 44–100-pin package 1–32 KB 512–2048 bytes — the ATxmega series offers a wide variety of peripherals and functionality such as: Extended performance features, such as DMA, "Event System", and cryptography support Extensive peripheral set with ADCs Application-specific AVR megaAVRs with special features not found on the other members of the AVR family, such as LCD controller, USB controller, advanced PWM, CAN, etc. FPSLIC (AVR with FPGA) FPGA 5k to 40k gates SRAM for the AVR program code, unlike all other AVRs AVR core can run at up to 50 MHz[9] 32-bit AVRsMain article: AVR32 In 2006, Atmel released microcontrollers based on the 32-bit AVR32 architecture. This was a completely different architecture unrelated to the 8-bit AVR, intended to compete with the ARM-based processors. It had a 32-bit data path, SIMD and DSP instructions, along with other audio- and video-processing features. The instruction set was similar to other RISC cores, but it was not compatible with the original AVR (nor any of the various ARM cores). Since then support for AVR32 has been dropped from Linux as of kernel 4.12; compiler support for the architecture in GCC was never mainlined into the compiler's central source-code repository and was available primarily in a vendor-supported fork. At the time that AVR32 was introduced, Atmel had already been a licensee of the ARM architecture, with both ARM7 and ARM9 microcontrollers having been released prior to and concurrently with the AVR32; later Atmel focused most development effort on 32-bit chips with ARM Cortex-M and Cortex-A cores. Atmel ATxmega128A1 in 100-pin TQFP package ATMEL MEGA32U4 die shot The AVRs have 32 single-byte registers and are classified as 8-bit RISC devices. Flash, EEPROM, and SRAM are all integrated onto a single chip, removing the need for external memory in most applications. Some devices have a parallel external bus option to allow adding additional data memory or memory-mapped devices. Almost all devices (except the smallest TinyAVR chips) have serial interfaces, which can be used to connect larger serial EEPROMs or flash chips. Program instructions are stored in non-volatile flash memory. Although the MCUs are 8-bit, each instruction takes one or two 16-bit words. The size of the program memory is usually indicated in the naming of the device itself (e.g., the ATmega64x line has 64 KB of flash, while the ATmega32x line has 32 KB). There is no provision for off-chip program memory; all code executed by the AVR core must reside in the on-chip flash. However, this limitation does not apply to the AT94 FPSLIC AVR/FPGA chips. The data address space consists of the register file, I/O registers, and SRAM. Some small models also map the program ROM into the data address space, but larger models do not. In the tinyAVR and megaAVR variants of the AVR architecture, the working registers are mapped in as the first 32 data memory addresses (000016–001F16), followed by 64 I/O registers (002016–005F16). In devices with many peripherals, these registers are followed by 160 "extended I/O" registers, only accessible as memory-mapped I/O (006016–00FF16). Actual SRAM starts after these register sections, at address 006016 or, in devices with "extended I/O", at 010016. Even though there are separate addressing schemes and optimized opcodes for accessing the register file and the first 64 I/O registers, all can also be addressed and manipulated as if they were in SRAM. The very smallest of the tinyAVR variants use a reduced architecture with only 16 registers (r0 through r15 are omitted) which are not addressable as memory locations. I/O memory begins at address 000016, followed by SRAM. In addition, these devices have slight deviations from the standard AVR instruction set. Most notably, the direct load/store instructions (LDS/STS) have been reduced from 2 words (32 bits) to 1 word (16 bits), limiting the total direct addressable memory (the sum of both I/O and SRAM) to 128 bytes. Conversely, the indirect load instruction's (LD) 16-bit address space is expanded to also include non-volatile memory such as Flash and configuration bits; therefore, the Load Program Memory (LPM) instruction is unnecessary and omitted. (For detailed info, see Atmel AVR instruction set.) In the XMEGA variant, the working register file is not mapped into the data address space; as such, it is not possible to treat any of the XMEGA's working registers as though they were SRAM. Instead, the I/O registers are mapped into the data address space starting at the very beginning of the address space. Additionally, the amount of data address space dedicated to I/O registers has grown substantially to 4096 bytes (000016–0FFF16). As with previous generations, however, the fast I/O manipulation instructions can only reach the first 64 I/O register locations (the first 32 locations for bitwise instructions). Following the I/O registers, the XMEGA series sets aside a 4096 byte range of the data address space, which can be used optionally for mapping the internal EEPROM to the data address space (100016–1FFF16). The actual SRAM is located after these ranges, starting at 200016. Each GPIO port on a tiny or mega AVR drives up to eight pins and is controlled by three 8-bit registers: DDRx, PORTx and PINx, where x is the port identifier. DDRx: Data Direction Register, configures the pins as either inputs or outputs. PORTx: Output port register. Sets the output value on pins configured as outputs. Enables or disables the pull-up resistor on pins configured as inputs. PINx: Input register, used to read an input signal. On some devices, this register can be used for pin toggling: writing a logic one to a PINx bit toggles the corresponding bit in PORTx, irrespective of the setting of the DDRx bit.[10] Newer ATtiny AVRs, like ATtiny817 and its siblings, have their port control registers somewhat differently defined. xmegaAVR have additional registers for push/pull, totem-pole and pullup configurations. Almost all AVR microcontrollers have internal EEPROM for semi-permanent data storage. Like flash memory, EEPROM can maintain its contents when electrical power is removed. In most variants of the AVR architecture, this internal EEPROM memory is not mapped into the MCU's addressable memory space. It can only be accessed the same way an external peripheral device is, using special pointer registers and read/write instructions, which makes EEPROM access much slower than other internal RAM. However, some devices in the SecureAVR (AT90SC) family[11] use a special EEPROM mapping to the data or program memory, depending on the configuration. The XMEGA family also allows the EEPROM to be mapped into the data address space. Since the number of writes to EEPROM is limited – Atmel specifies 100,000 write cycles in their datasheets – a well designed EEPROM write routine should compare the contents of an EEPROM address with desired contents and only perform an actual write if the contents need to be changed. Atmel's AVRs have a two-stage, single-level pipeline design, meaning that the next machine instruction is fetched as the current one is executing. Most instructions take just one or two clock cycles, making AVRs relatively fast among eight-bit microcontrollers. The AVR processors were designed with the efficient execution of compiled C code in mind and have several built-in pointers for the task. Main article: Atmel AVR instruction set The AVR instruction set is more orthogonal than those of most eight-bit microcontrollers, in particular the 8051 clones and PIC microcontrollers with which AVR has competed. However, it is not completely regular: Pointer registers X, Y, and Z have addressing capabilities that are different from each other. Register locations R0 to R15 have more limited addressing capabilities than register locations R16 to R31. I/O ports 0 to 31 can be bit addressed, unlike I/O ports 32 to 63. CLR (clear all bits to zero) affects flags, while SER (set all bits to one) does not, even though they are complementary instructions. (CLR is pseudo-op for EOR R, R; while SER is short for LDI R,$FF. Arithmetic operations such as EOR modify flags, while moves/loads/stores/branches such as LDI do not.) Accessing read-only data stored in the program memory (flash) requires special LPM instructions; the flash bus is otherwise reserved for instruction memory. Some chip-specific differences affect code generation. Code pointers (including return addresses on the stack) are two bytes long on chips with up to 128 KB of flash memory, but three bytes long on larger chips; not all chips have hardware multipliers; chips with over 8 KB of flash have branch and call instructions with longer ranges; and so forth. The mostly regular instruction set makes C (and even Ada) compilers fairly straightforward and efficient. GCC has included AVR support for quite some time, and that support is widely used. LLVM also has rudimentary AVR support. In fact, Atmel solicited input from major developers of compilers for small microcontrollers, to determine the instruction set features that were most useful in a compiler for high-level languages.[8] The AVR line can normally support clock speeds from 0 to 20 MHz, with some devices reaching 32 MHz. Lower-powered operation usually requires a reduced clock speed. All recent (Tiny, Mega, and Xmega, but not 90S) AVRs feature an on-chip oscillator, removing the need for external clocks or resonator circuitry. Some AVRs also have a system clock prescaler that can divide down the system clock by up to 1024. This prescaler can be reconfigured by software during run-time, allowing the clock speed to be optimized. Since all operations (excluding multiplication and 16-bit add/subtract) on registers R0–R31 are single-cycle, the AVR can achieve up to 1 MIPS per MHz, i.e. an 8 MHz processor can achieve up to 8 MIPS. Loads and stores to/from memory take two cycles, branching takes two cycles. Branches in the latest "3-byte PC" parts such as ATmega2560 are one cycle slower than on previous devices. AVRs have a large following due to the free and inexpensive development tools available, including reasonably priced development boards and free development software. The AVRs are sold under various names that share the same basic core, but with different peripheral and memory combinations. Compatibility between chips in each family is fairly good, although I/O controller features may vary. See external links for sites relating to AVR development. AVRs offer a wide range of features: Multifunction, bi-directional general-purpose I/O ports with configurable, built-in pull-up resistors Multiple internal oscillators, including RC oscillator without external parts Internal, self-programmable instruction flash memory up to 256 KB (384 KB on XMega) In-system programmable using serial/parallel low-voltage proprietary interfaces or JTAG Optional boot code section with independent lock bits for protection On-chip debugging (OCD) support through JTAG or debugWIRE on most devices The JTAG signals (TMS, TDI, TDO, and TCK) are multiplexed on GPIOs. These pins can be configured to function as JTAG or GPIO depending on the setting of a fuse bit, which can be programmed via in-system programming (ISP) or HVSP. By default, AVRs with JTAG come with the JTAG interface enabled. debugWIRE uses the /RESET pin as a bi-directional communication channel to access on-chip debug circuitry. It is present on devices with lower pin counts, as it only requires one pin. Internal data EEPROM up to 4 KB Internal SRAM up to 16 KB (32 KB on XMega) External 64 KB little endian data space on certain models, including the Mega8515 and Mega162. The external data space is overlaid with the internal data space, such that the full 64 KB address space does not appear on the external bus and accesses to e.g. address 010016 will access internal RAM, not the external bus. In certain members of the XMega series, the external data space has been enhanced to support both SRAM and SDRAM. As well, the data addressing modes have been expanded to allow up to 16 MB of data memory to be directly addressed. 8-bit and 16-bit timers PWM output (some devices have an enhanced PWM peripheral which includes a dead-time generator) Input capture that record a time stamp triggered by a signal edge analog comparator 10 or 12-bit A/D converters, with multiplex of up to 16 channels 12-bit D/A converters A variety of serial interfaces, including I²C compatible Two-Wire Interface (TWI) Synchronous/asynchronous serial peripherals (UART/USART) (used with RS-232, RS-485, and more) Serial Peripheral Interface Bus (SPI) Universal Serial Interface (USI): a multi-purpose hardware communication module that can be used to implement an SPI,[12] I2C[13][14] or UART[15] interface. Brownout detection Watchdog timer (WDT) Multiple power-saving sleep modes Lighting and motor control (PWM-specific) controller models CAN controller support USB controller support Proper full-speed (12 Mbit/s) hardware & Hub controller with embedded AVR. Also freely available low-speed (1.5 Mbit/s) (HID) bitbanging software emulations Ethernet controller support LCD controller support Low-voltage devices operating down to 1.8 V (to 0.7 V for parts with built-in DC–DC upconverter) picoPower devices DMA controllers and "event system" peripheral communication. Fast cryptography support for AES and DES There are many means to load program code into an AVR chip. The methods to program AVR chips varies from AVR family to family. Most of the methods described below use the RESET line to enter programming mode. In order to avoid the chip accidentally entering such mode, it is advised to connect a pull-up resistor between the RESET pin and the positive power supply.[16] 6- and 10-pin ISP header diagrams The in-system programming (ISP) programming method is functionally performed through SPI, plus some twiddling of the Reset line. As long as the SPI pins of the AVR are not connected to anything disruptive, the AVR chip can stay soldered on a PCB while reprogramming. All that is needed is a 6-pin connector and programming adapter. This is the most common way to develop with an AVR. The Atmel-ICE device or AVRISP mkII (Legacy device) connects to a computer's USB port and performs in-system programming using Atmel's software. AVRDUDE (AVR Downloader/UploaDEr) runs on Linux, FreeBSD, Windows, and Mac OS X, and supports a variety of in-system programming hardware, including Atmel AVRISP mkII, Atmel JTAG ICE, older Atmel serial-port based programmers, and various third-party and "do-it-yourself" programmers.[17] The Program and Debug Interface (PDI) is an Atmel proprietary interface for external programming and on-chip debugging of XMEGA devices. The PDI supports high-speed programming of all non-volatile memory (NVM) spaces; flash, EEPROM, fuses, lock-bits and the User Signature Row. This is done by accessing the XMEGA NVM controller through the PDI interface, and executing NVM controller commands. The PDI is a 2-pin interface using the Reset pin for clock input (PDI_CLK) and a dedicated data pin (PDI_DATA) for input and output.[18] The Unified Program and Debug Interface (UPDI) is a one-wire interface for external programming and on-chip debugging of newer ATtiny and ATmega devices. UPDI chips can be programmed by an Atmel-ICE, a PICkit 4, an Arduino (flashed with jtag2updi),[19] or though a UART (with a 1 kΩ resistor between the TX and RX pins) controlled by Microchip's Python utility pymcuprog.[20] High-voltage serial programming (HVSP)[21] is mostly the backup mode on smaller AVRs. An 8-pin AVR package does not leave many unique signal combinations to place the AVR into a programming mode. A 12-volt signal, however, is something the AVR should only see during programming and never during normal operation. The high voltage mode can also be used in some devices where the reset pin was disabled by fuses. High-voltage parallel programming (HVPP) is considered the "final resort" and may be the only way to correct bad fuse settings on an AVR chip. Most AVR models can reserve a bootloader region, 256 bytes to 4 KB, where re-programming code can reside. At reset, the bootloader runs first and does some user-programmed determination whether to re-program or to jump to the main application. The code can re-program through any interface available, or it could read an encrypted binary through an Ethernet adapter like PXE. Atmel has application notes and code pertaining to many bus interfaces.[22][23][24][25] The AT90SC series of AVRs are available with a factory mask-ROM for program memory, instead of flash.[26] Because of the large up-front cost and minimum order quantity, a mask-ROM is only cost-effective for high-production runs. aWire is a new one-wire debug interface available on the new UC3L AVR32 devices. The AVR offers several options for debugging, mostly involving on-chip debugging while the chip is in the target system. Main article: debugWIRE debugWIRE is Atmel's solution for providing on-chip debug capabilities via a single microcontroller pin. It is useful for lower pin-count parts which cannot provide the four "spare" pins needed for JTAG. The JTAGICE mkII, mkIII and the AVR Dragon support debugWIRE. debugWIRE was developed after the original JTAGICE release, and now clones support it. The Joint Test Action Group (JTAG) feature provides access to on-chip debugging functionality while the chip is running in the target system.[27] JTAG allows accessing internal memory and registers, setting breakpoints on code, and single-stepping execution to observe system behaviour. Atmel provides a series of JTAG adapters for the AVR: The Atmel-ICE[28] is the latest adapter. It supports JTAG, debugWire, aWire, SPI, TPI, and PDI interfaces. The JTAGICE 3[29] is a midrange debugger in the JTAGICE family (JTAGICE mkIII). It supports JTAG, aWire, SPI, and PDI interfaces. The JTAGICE mkII[30] replaces the JTAGICE and is similarly priced. The JTAGICE mkII interfaces to the PC via USB, and supports both JTAG and the newer debugWIRE interface. Numerous third-party clones of the Atmel JTAGICE mkII device started shipping after Atmel released the communication protocol.[31] The AVR Dragon[32] is a low-cost (approximately $50) substitute for the JTAGICE mkII for certain target parts. The AVR Dragon provides in-system serial programming, high-voltage serial programming and parallel programming, as well as JTAG or debugWIRE emulation for parts with 32 KB of program memory or less. ATMEL changed the debugging feature of AVR Dragon with the latest firmware of AVR Studio 4 – AVR Studio 5 and now it supports devices over 32 KB of program memory. The JTAGICE adapter interfaces to the PC via a standard serial port.[33] Although the JTAGICE adapter has been declared "end-of-life" by Atmel, it is still supported in AVR Studio and other tools. JTAG can also be used to perform a boundary scan test,[34] which tests the electrical connections between AVRs and other boundary scan capable chips in a system. Boundary scan is well-suited for a production line, while the hobbyist is probably better off testing with a multimeter or oscilloscope. Atmel STK500 development board Official Atmel AVR development tools and evaluation kits contain a number of starter kits and debugging tools with support for most AVR devices: The STK600 starter kit and development system is an update to the STK500.[35] The STK600 uses a base board, a signal routing board, and a target board. The base board is similar to the STK500, in that it provides a power supply, clock, in-system programming, an RS-232 port and a CAN (Controller Area Network, an automotive standard) port via DE9 connectors, and stake pins for all of the GPIO signals from the target device. The target boards have ZIF sockets for DIP, SOIC, QFN, or QFP packages, depending on the board. The signal routing board sits between the base board and the target board, and routes the signals to the proper pin on the device board. There are many different signal routing boards that could be used with a single target board, depending on what device is in the ZIF socket. The STK600 allows in-system programming from the PC via USB, leaving the RS-232 port available for the target microcontroller. A 4 pin header on the STK600 labeled 'RS-232 spare' can connect any TTL level USART port on the chip to an onboard MAX232 chip to translate the signals to RS-232 levels. The RS-232 signals are connected to the RX, TX, CTS, and RTS pins on the DB-9 connector. The STK500 starter kit and development system features ISP and high voltage programming (HVP) for all AVR devices, either directly or through extension boards. The board is fitted with DIP sockets for all AVRs available in DIP packages. STK500 Expansion Modules: Several expansion modules are available for the STK500 board: STK501 – Adds support for microcontrollers in 64-pin TQFP packages. STK502 – Adds support for LCD AVRs in 64-pin TQFP packages. STK503 – Adds support for microcontrollers in 100-pin TQFP packages. STK504 – Adds support for LCD AVRs in 100-pin TQFP packages. STK505 – Adds support for 14 and 20-pin AVRs. STK520 – Adds support for 14 and 20, and 32-pin microcontrollers from the AT90PWM and ATmega family. STK524 – Adds support for the ATmega32M1/C1 32-pin CAN/LIN/Motor Control family. STK525 – Adds support for the AT90USB microcontrollers in 64-pin TQFP packages. STK526 – Adds support for the AT90USB microcontrollers in 32-pin TQFP packages. The STK200 starter kit and development system has a DIP socket that can host an AVR chip in a 40, 20, or 8-pin package. The board has a 4 MHz clock source, 8 light-emitting diode (LED)s, 8 input buttons, an RS-232 port, a socket for a 32 KB SRAM and numerous general I/O. The chip can be programmed with a dongle connected to the parallel port. Supported microcontrollers (according to the manual) Chip Flash size EEPROM SRAM Frequency[MHz] Package AT90S1200 1 KB 64 B 0 B 12 PDIP-20 AT90S2313 2 KB 128 B 128 B 10 PDIP-20 AT90S/LS2323 2 KB 128 B 128 B 10 PDIP-8 AT90S/LS2343 2 KB 128 B 128 B 10 PDIP-8 AT90S4414 4 KB 256 B 256 B 8 PDIP-40 AT90S/LS4434 4 KB 256 B 256 B 8 PDIP-40 AT90S8515 8 KB 512 B 512 B 8 PDIP-40 AT90S/LS8535 8 KB 512 B 512 B 8 PDIP-40 The Atmel ICE is the currently supported inexpensive tool to program and debug all AVR devices (unlike the AVRISP/AVRISP mkII, Dragon, etc. discussed below). It connects to and receives power from a PC via USB, and supports JTAG, PDI, aWire, debugWIRE, SPI, SWD, TPI, and UPDI (the Microchip Unified Program and Debug Interface) interfaces. The ICE can program and debug all AVRs via the JTAG interface, and program with additional interfaces as supported on each device: 8-bit AVR XMEGA devices via the PDI 2-wire interface 8-bit megaAVR and tinyAVR devices via SPI for all with OCD (on-chip debugger) support 8-bit tinyAVR microcontrollers with TPI support 32-bit SAM Arm Cortex-M based microcontrollers via SWD Target operating voltage ranges of 1.62V to 5.5V are supported as well as the following clock ranges: Supports JTAG & PDI clock frequencies from 32 kHz to 7.5 MHz Supports aWire baud rates from 7.5 kbit/s to 7 Mbit/s Supports debugWIRE baud rates from 4 kbit/s to 0.5 Mbit/s Supports SPI clock frequencies from 8 kHz to 5 MHz Supports SWD clock frequencies from 32 kHz to 2 MHz The ICE is supported by the Microchip Studio IDE, as well as a command line interface (atprogram). The Atmel-ICE supports a limited implementation of the Data Gateway Interface (DGI) when debugging and programming features are not in use. The Data Gateway Interface is an interface for streaming data from a target device to the connected computer. This is meant as a useful adjunct to the unit to allow for demonstration of application features and as an aid in application level debugging. AVRISP mkII The AVRISP and AVRISP mkII are inexpensive tools allowing all AVRs to be programmed via ICSP. The AVRISP connects to a PC via a serial port and draws power from the target system. The AVRISP allows using either of the "standard" ICSP pinouts, either the 10-pin or 6-pin connector. The AVRISP mkII connects to a PC via USB and draws power from USB. LEDs visible through the translucent case indicate the state of target power. As the AVRISP mkII lacks driver/buffer ICs,[36] it can have trouble programming target boards with multiple loads on its SPI lines. In such occurrences, a programmer capable of sourcing greater current is required. Alternatively, the AVRISP mkII can still be used if low-value (~150 ohm) load-limiting resistors can be placed on the SPI lines before each peripheral device. Both the AVRISP and the AVRISP mkII are now discontinued, with product pages removed from the Microchip website. As of July 2019 the AVRISP mkII is still in stock at a number of distributors. There are also a number of 3rd party clones available. AVR Dragon with ISP programming cable and attached, blue/greenish ZIF Socket The Atmel Dragon is an inexpensive tool which connects to a PC via USB. The Dragon can program all AVRs via JTAG, HVP, PDI,[37] or ICSP. The Dragon also allows debugging of all AVRs via JTAG, PDI, or debugWire; a previous limitation to devices with 32 KB or less program memory has been removed in AVR Studio 4.18.[38] The Dragon has a small prototype area which can accommodate an 8, 28, or 40-pin AVR, including connections to power and programming pins. There is no area for any additional circuitry, although this can be provided by a third-party product called the "Dragon Rider".[39] The JTAG In Circuit Emulator (JTAGICE) debugging tool supports on-chip debugging (OCD) of AVRs with a JTAG interface. The original JTAGICE (sometimes retroactively referred to as JTAGICE mkI) uses an RS-232 interface to a PC and can only program AVRs with a JTAG interface. The JTAGICE mkI is no longer in production, however it has been replaced by the JTAGICE mkII. The JTAGICE mkII debugging tool supports on-chip debugging (OCD) of AVRs with SPI, JTAG, PDI, and debugWIRE interfaces. The debugWire interface enables debugging using only one pin (the Reset pin), allowing debugging of applications running on low pin-count microcontrollers. The JTAGICE mkII connects using USB, but there is an alternate connection via a serial port, which requires using a separate power supply. In addition to JTAG, the mkII supports ISP programming (using 6-pin or 10-pin adapters). Both the USB and serial links use a variant of the STK500 protocol. The JTAGICE3 updates the mkII with more advanced debugging capabilities and faster programming. It connects via USB and supports the JTAG, aWire, SPI, and PDI interfaces.[40] The kit includes several adapters for use with most interface pinouts. The AVR ONE! is a professional development tool for all Atmel 8-bit and 32-bit AVR devices with On-Chip Debug capability. It supports SPI, JTAG, PDI, and aWire programming modes and debugging using debugWIRE, JTAG, PDI, and aWire interfaces.[41] Atmel ATmega169 in 64-pad MLF package on the back of an Atmel AVR Butterfly board Main article: AVR Butterfly The very popular AVR Butterfly demonstration board is a self-contained, battery-powered computer running the Atmel AVR ATmega169V microcontroller. It was built to show off the AVR family, especially a then new built-in LCD interface. The board includes the LCD screen, joystick, speaker, serial port, real time clock (RTC), flash memory chip, and both temperature and voltage sensors. Earlier versions of the AVR Butterfly also contained a CdS photoresistor; it is not present on Butterfly boards produced after June 2006 to allow RoHS compliance.[42] The small board has a shirt pin on its back so it can be worn as a name badge. The AVR Butterfly comes preloaded with software to demonstrate the capabilities of the microcontroller. Factory firmware can scroll your name, display the sensor readings, and show the time. The AVR Butterfly also has a piezoelectric transducer that can be used to reproduce sounds and music. The AVR Butterfly demonstrates LCD driving by running a 14-segment, six alpha-numeric character display. However, the LCD interface consumes many of the I/O pins. The Butterfly's ATmega169 CPU is capable of speeds up to 8 MHz, but it is factory set by software to 2 MHz to preserve the button battery life. A pre-installed bootloader program allows the board to be re-programmed via a standard RS-232 serial plug with new programs that users can write with the free Atmel IDE tools. This small board, about half the size of a business card, is priced at slightly more than an AVR Butterfly. It includes an AT90USB1287 with USB On-The-Go (OTG) support, 16 MB of DataFlash, LEDs, a small joystick, and a temperature sensor. The board includes software, which lets it act as a USB mass storage device (its documentation is shipped on the DataFlash), a USB joystick, and more. To support the USB host capability, it must be operated from a battery, but when running as a USB peripheral, it only needs the power provided over USB. Only the JTAG port uses conventional 2.54 mm pinout. All the other AVR I/O ports require more compact 1.27 mm headers. The AVR Dragon can both program and debug since the 32 KB limitation was removed in AVR Studio 4.18, and the JTAGICE mkII is capable of both programming and debugging the processor. The processor can also be programmed through USB from a Windows or Linux host, using the USB "Device Firmware Update" protocols. Atmel ships proprietary (source code included but distribution restricted) example programs and a USB protocol stack with the device. LUFA[43] is a third-party free software (MIT license) USB protocol stack for the USBKey and other 8-bit USB AVRs. The RAVEN kit supports wireless development using Atmel's IEEE 802.15.4 chipsets, for Zigbee and other wireless stacks. It resembles a pair of wireless more-powerful Butterfly cards, plus a wireless USBKey; and costing about that much (under $US100). All these boards support JTAG-based development. The kit includes two AVR Raven boards, each with a 2.4 GHz transceiver supporting IEEE 802.15.4 (and a freely licensed Zigbee stack). The radios are driven with ATmega1284p processors, which are supported by a custom segmented LCD driven by an ATmega3290p processor. Raven peripherals resemble the Butterfly: piezo speaker, DataFlash (bigger), external EEPROM, sensors, 32 kHz crystal for RTC, and so on. These are intended for use in developing remote sensor nodes, to control relays, or whatever is needed. The USB stick uses an AT90USB1287 for connections to a USB host and to the 2.4 GHz wireless links. These are intended to monitor and control the remote nodes, relying on host power rather than local batteries. A wide variety of third-party programming and debugging tools are available for the AVR. These devices use various interfaces, including RS-232, PC parallel port, and USB.[44] Atmel AVR ATmega328 28-pin DIP on an Arduino Duemilanove board Atmel AVR ATmega8 28-pin DIP on a custom development board AVRs have been used in various automotive applications such as security, safety, powertrain and entertainment systems. Atmel has recently launched a new publication "Atmel Automotive Compilation" to help developers with automotive applications. Some current usages are in BMW, Daimler-Chrysler and TRW. The Arduino physical computing platform is based on an ATmega328 microcontroller (ATmega168 or ATmega8 in board versions older than the Diecimila). The ATmega1280 and ATmega2560, with more pinout and memory capabilities, have also been employed to develop the Arduino Mega platform. Arduino boards can be used with its language and IDE, or with more conventional programming environments (C, assembler, etc.) as just standardized and widely available AVR platforms. USB-based AVRs have been used in the Microsoft Xbox hand controllers. The link between the controllers and Xbox is USB. Numerous companies produce AVR-based microcontroller boards intended for use by hobbyists, robot builders, experimenters and small system developers including: Cubloc,[45] gnusb,[46] BasicX,[47] Oak Micros,[48] ZX Microcontrollers,[49] and myAVR.[50] There is also a large community of Arduino-compatible boards supporting similar users. Schneider Electric used to produce the M3000 Motor and Motion Control Chip, incorporating an Atmel AVR Core and an advanced motion controller for use in a variety of motion applications but this has been discontinued.[51] With the growing popularity of FPGAs among the open source community, people have started developing open source processors compatible with the AVR instruction set. The OpenCores website lists the following major AVR clone projects: pAVR,[52] written in VHDL, is aimed at creating the fastest and maximally featured AVR processor, by implementing techniques not found in the original AVR processor such as deeper pipelining. avr_core,[53] written in VHDL, is a clone aimed at being as close as possible to the ATmega103. Navré,[54] written in Verilog, implements all Classic Core instructions and is aimed at high performance and low resource usage. It does not support interrupts. softavrcore,[55] written in Verilog, implements the AVR instruction set up to AVR5, supports interrupts along with optional automatic interrupt acknowledgement, power saving via sleep mode plus some peripheral interfaces and hardware accelerators (such as UART, SPI, cyclic redundancy check calculation unit and system timers). These peripherals demonstrate how could these be attached to and configured for this core. Within the package, a full-featured FreeRTOS port is also available as an example for the core + peripheral utilization. The opencores project CPU lecture[56] written in VHDL by Dr. Jürgen Sauermann explains in detail how to design a complete AVR-based system on a chip (SoC). In addition to the chips manufactured by Atmel, clones are available from LogicGreen Technologies.[57] These parts are not exact clones – they have a few features not found in the chips they are "clones" of, and higher maximum clock speeds, but use SWD (Serial Wire Debug, a variant of JTAG from ARM) instead of ISP for programming, so different programming tools must be used. Microcontrollers using the ATmega architecture are being manufactured by NIIET in Voronezh, Russia, as part of the 1887 series of integrated circuits. This includes an ATmega128 under the designation 1887VE7T (Russian: 1887ВЕ7Т).[58]

^ Atmel press release. "Atmel's AVR Microcontroller Ships 500 Million Units". ^ Since 1996, NTH has become part of the Norwegian University of Science and Technology (NTNU) ^ alfbogen.com blog ^ a b Archived at Ghostarchive and the Wayback Machine: "The Story of AVR". youtube.com. ^ "UNSW School of Computer Science and Engineering - General AVR Info". Cse.unsw.edu.au. Archived from the original on 2012-06-23. Retrieved 2012-09-19. ^ An introduction to Atmel and the AVR microcontroller[permanent dead link] ^ "Embedded Systems and Microcontrollers" (PDF). Archived from the original (PDF) on 2004-12-24. Retrieved 2018-10-01. ^ a b Myklebust, Gaute. The AVR Microcontroller and C Compiler Co-Design (PDF). Atmel Norway. CiteSeerX 10.1.1.63.1447. Retrieved 2012-09-19. ^ Field Programmable System Level Integrated Circuit. Archived 2012-11-27 at the Wayback Machine. ^ atmel.com ^ Atmel Smart Card ICs ^ "AVR319: Using the USI module for SPI communication" (PDF). Atmel. 2004. Archived (PDF) from the original on 2012-06-17. Retrieved 10 June 2014. ^ "Atmel AVR310: Using the USI Module as a I2C Master" (PDF). Atmel. 2013. Archived (PDF) from the original on 2014-07-14. Retrieved 10 June 2014. ^ "AVR312: Using the USI module as a I2C slave" (PDF). Atmel. 2005. Archived (PDF) from the original on 2014-07-14. Retrieved 10 June 2014. ^ "AVR307: Half Duplex UART Using the USI Module" (PDF). Atmel. 2003. Archived (PDF) from the original on 2014-07-14. Retrieved 10 June 2014. ^ "AVR Hardware Design Considerations" (PDF) (application note). Atmel Corporation. Jun 2015. p. 5. Archived (PDF) from the original on 2014-12-22. Retrieved 14 Jun 2015. The reset line has an internal pull-up resistor, but if the environment is noisy it can be insufficient and reset can therefore occur sporadically. ^ "AVRDUDE programmer". Savannah.nongnu.org. Retrieved 2012-09-19. ^ "PDI programming driver" (PDF). Archived (PDF) from the original on 2020-03-25. Retrieved 2012-09-19. ^ "GitHub - ElTangas/Jtag2updi: UPDI programmer software for Arduino (Targets Tiny AVR-0/1/2, Mega AVR-0 and AVR-DA/DB MCUs)". GitHub. 17 December 2021. ^ "pymcuprog - Python MCU programmer". Github. Microchip PIC&AVR Tools. 13 November 2022. Retrieved 18 November 2022. ^ "HVSP_Description". Support.atmel.no. Archived from the original on 2009-10-12. Retrieved 2012-09-19. ^ "DES-encrypted AVR Bootloader" (PDF). Archived (PDF) from the original on 2005-05-16. Retrieved 2012-09-19. ^ "AES-encrypted AVR Bootloader" (PDF). Retrieved 2012-09-19. ^ "XMEGA Bootloader" (PDF). Retrieved 2012-09-19. ^ "AVR USB Bootloader" (PDF). Archived (PDF) from the original on 2006-06-28. Retrieved 2012-09-19. ^ "Atmel's Self-Programming Flash Microcontrollers" (PDF). Retrieved 12 March 2020. ^ "Guide to understanding JTAG and security fuses on the AVR". Retrieved 2012-09-19.[permanent dead link] ^ "Atmel-ICE - Atmel Corporation". Atmel.com. Retrieved 2015-09-11. ^ "JTAGICE 3- Atmel Corporation". Atmel.com. Retrieved 2012-09-19. ^ "AVR JTAGICE mkII". Atmel. Archived from the original on 15 February 2013. Retrieved 13 January 2013. ^ "JTAGICE mkII Communication Protocol" (PDF). Archived (PDF) from the original on 2005-05-16. Retrieved 2012-09-19. ^ "AVR Dragon". Atmel. Retrieved 13 January 2013. ^ "AVR JTAGICE mkII User's Guide" (PDF). microchip.com. Archived (PDF) from the original on 2017-07-02. Retrieved 25 March 2020. ^ JTAGICE Press Release, 2004. Archived 2011-07-07 at the Wayback Machine ^ "STK600". Atmel. Archived from the original on 15 February 2013. Retrieved 13 January 2013. ^ "AVRISP mkII Disassembled". Archived from the original on 2014-11-08. Retrieved 2014-11-08. ^ "AVR1005: Getting started with XMEGA, page 7" (PDF). Atmel. Archived (PDF) from the original on 2009-10-07. Retrieved 7 November 2011. ^ "AVR Studio v4.18 Release Notes". Retrieved 2012-09-19. ^ "ECROS Technology - Dragon Rider". Ecrostech.com. 2008-03-02. Retrieved 2012-09-19. ^ JTAGICE3 Product Page ^ AVR ONE! Product Page ^ AVR Butterfly ^ "LUFA (Formerly MyUSB)". Four Walled Cubicle. Retrieved 2012-09-19. ^ See avrffreaks.net for a comprehensive list. ^ "Comfile Technology". Comfile Technology, Inc. Archived from the original on 17 January 2013. Retrieved 13 January 2013. ^ "gnusb: Open Source USB Sensor Box". Retrieved 13 January 2013. ^ "BasicX". NetMedia, Inc. Archived from the original on 23 May 2013. Retrieved 13 January 2013. ^ "Welcome to Oak Micros". Oak Micros. Oak Micros. Archived from the original on 2012-10-25. Retrieved 13 January 2013. ^ "ZBasic". Retrieved 13 January 2013. ^ "myAVR". Laser & Co. Solutions GmbH. Retrieved 13 January 2013. ^ "M3000 Motion controller on a chip". imshome.com. Schneider Electric Motion USA. Archived from the original on 2009-12-02. Retrieved 2011-08-02. ^ "pAVR :: Overview". OpenCores. Retrieved 2012-09-19. ^ "AVR Core :: Overview". OpenCores. Retrieved 2012-09-19. ^ "Navré AVR clone (8-bit RISC) Overview". OpenCores. Retrieved 2012-09-19. ^ "Soft AVR Core + Interfaces Overview". OpenCores. Retrieved 2020-06-16. ^ "CPU lecture". OpenCores. Retrieved 2015-02-16. ^ "LGT8F88A FLASH Microcontroller". LogicGreen Technologies. Archived from the original on 2017-08-29. Retrieved 2019-01-18, a clone of the ATmega88. ^ "Микроконтроллеры" [Microcontrollers] (in Russian). Voronezh: OAO "NIIET". Archived from the original on 22 August 2017. Retrieved 22 August 2017. See also: List of books about Arduino Williams, Elliot (2014). AVR Programming: Learning to Write Software for Hardware. Maker Media. ISBN 978-1449355784. Schmidt, Maik (2011). Arduino: A Quick Start Guide. Pragmatic Bookshelf. ISBN 978-1-934356-66-1. Margush, Timothy S. (2011). Some Assembly Required: Assembly Language Programming with the AVR Microcontroller. CRC Press. ISBN 978-1439820643. Mazidi, Muhammad Ali; Naimi, Sarmad; Naimi, Sepehr (2010). AVR Microcontroller and Embedded Systems: Using Assembly and C. Pearson. ISBN 978-0138003319. Wikimedia Commons has media related to Atmel AVR. The Wikibook Embedded Systems has a page on the topic of: Atmel AVR Electronics portal Official Website Atmel AVR Official Community AVR Freaks community Microchip Forum Pinout Diagrams AVR DIP-Packages: ATtiny44/45/84/85, ATmega328P, ATmega644P, ATmega1284P AVR SMD-Packages: ATmega328, ATmega2560, ATmega32U4 Simulators AVR8js (8-bit in-browser simulator) Retrieved from " 2Microcontroller family 78K Family microcontrollersGeneral informationLaunched1986; 39 years ago (1986)DiscontinuedcurrentCommon manufacturerRenesas Electronics(formerly NEC)PerformanceMax. CPU clock rate32 kHz to 24 MHzData width16/8Address width20(24)/16Architecture and classificationApplicationEmbeddedInstruction set78K FamilyPhysical specificationsCores1Products, models, variantsVariant78K0R, 78K0S, 78K0,78K4, 78K6, 78K3,78K7,78K1, 78K2HistoryPredecessors87AD Family,17K FamilySuccessorRL78 Family 78K0/KX1+ board with in-circuit emulator; MINICUBE 78K0S/KA1+ Do It board 78K0R/KG3 Cool It board with in-circuit emulator; IECUBE (formerly, MINICUBE2) 78K is the trademark name of 16- and 8-bit microcontroller family[1]:23-4–23-5[2]:78 manufactured by Renesas Electronics, originally developed by NEC[3][4]:229 started in 1986.[5]:7,line 2 The basis of 78K Family is an accumulator-based register-bank CISC architecture. 78K is a single-chip microcontroller, which usually integrates; program ROM, data RAM, serial interfaces, timers, I/O ports, an A/D converter, an interrupt controller, and a CPU core, on one die.[6][7]:412 Its application area is mainly simple mechanical system controls and man-machine interfaces.[8][9][10] Regarding software development tools, C compilers and macro-assemblers are available.[11]:99 As for development tool hardware, full probing-pod type and debug port type in-circuit emulators,[12][13] and flash ROM programmers[14]:22–24 are available. Historically, the family has 11 series with 9 instruction set architectures. As of 2018, 3 instruction set architectures, those are 8-bit 78K0, 8-bit 78K0S, and 16-/8-bit 78K0R, are still promoted for customers' new designs.[14] But in most of cases, migration to RL78 Family,[15] which is a successor of 78K0R and almost binary level compatible with 78K0R,[16]:20 is recommended.[17] 78K0 Series [de; jp] (also known as 78K/0) is a long-running 8-bit single chip microcontroller,[18] which is the basis of 78K0S [jp] and 78K0R Series. It contains 8× 8-bit registers ×4 banks. For 16-bit calculating instructions, it performs ALU operation twice. Each instructions are performed serially without instruction pipelining. It has 16-bit 64K Byte address space.[19] Some variants of 78K0 have affordable and compact type 8-bit R-2R D/A converter, which does not have monotonicity because it is not trimmed for adjustment nor followed by operational amplifier. In its earlier stage, the Program Memory was one-time PROM (OTP), UV-EPROM, or mask ROM.[20] But with the times, it became flash memory.[21][22] 78K0S Series (also known as 78K/0S) is a low-end version of 78K0.[23][24][25] It has 8× 8-bit registers, but without any banks. In addition, some instructions, such as multiplication and division, are removed from 78K0 instruction set architecture.[26] 78K0R Series is a 16-bit single-chip microcontroller with 3-stage instruction pipelining.[27] Its instruction set is similar to 78K0 and covers 16- and 8-bit operations. It has 20-bit 1M Byte address space.[28] 75 instructions out of 80 are identical with that of RL78 Family; its successor.[16]:20[15][29] 178K0 Series (also known as 178K/0) is a successor of NEC's 17K Family 4-bit microcontroller for DTS (Digital Tuning Systems) and remote controls.[30] It integrates 17K family's peripheral functions with the 78K0 8-bit CPU core on a chip.[31] 178K0S Series (also known as 178K/0S) is also a successor of 17K Family with the 78K0S CPU core.[32] 78K4 Series (also known as 78K/4) is a 16-bit single-chip microcontroller with 16 and 8-bit operations.[33][34][35][36] It has 16× 8-bit registers ×4 banks, which can be also used for 8× 16-bit registers ×4 banks. Some of these registers can be also used as 24-bit extension for addressing modes.[37] It has 24-bit 16M Byte address space. It has microcode-based operations named Macro Service with interrupt functions.[38]:§23.8,560–593 78K7 Series (also known as 78K/7) is a 32-bit single-chip microcontroller with 32, 16 and 8 bit operations. It has 8× 32-bit registers ×16 banks, which can be also used for 16× 16-bit registers ×16 banks and 16× 8-bit registers ×16 banks. It has microcode-based operations named Macro Service with interrupt functions. It has 24-bit 16M Byte linear address space. It is used for some Quantum Fireball products,[39]:Photo 2 but shortly replaced with V850 Family 32-bit RISC microcontrollers. 78K6 Series (also known as 78K/6) is a 16-bit single-chip microcontroller. Its life-time was short, and less variants. 78K1 Series (also known as 78K/1) is an 8-bit single-chip microcontroller. It has 8× 8-bit registers ×4 banks. 78K1 series is targeted for servo controls of videocassette recorders. μPD78148 sub-series integrates 2 operational amplifiers.[40] 78K3 Series (also known as 78K/3) is a 16-bit single-chip microcontroller with 16 and 8 bit operations. It has 16× 8-bit ×8 banks, which can be also used for 8× 16-bit registers ×8 banks. Its address space is 16-bit 64K Byte. It is developed as high-end series of 78K Family. It has microcode-based operations named Macro Service with interrupt functions.[41]:§13.4,261–280 This series is used for hard disk drives, especially Quantum Fireball Series.[42] μPD78364 sub-series is used for inverter compressor controls.[43] It is also used for traction control systems of some cars. 78K2 Series (also known as 78K/2) is an 8-bit single-chip microcontroller. It has 8× 8-bit registers ×4 banks. It is developed as general purpose series of 78K Family.[44] 87AD Family[4]:229 is an 8-bit single-chip microcontroller. It has 8× 8-bit registers ×4 banks. Its instruction set architecture became the basis of 78K.[45] 17K Family[4]:229 is a 4-bit single-chip microcontroller, especially dedicated for DTS (Digital Tuning Systems) and remote controls. It has 2 plane of 128× 4-bit register files, and sophisticated fully orthogonal instruction set. This instruction set is completely different from that of 78K Family.[30] Series ALU Registers Instructions Pipeline Remark Documents RL78-S3 16-bit 8× 8-bit ×4 banks 81 (75+6) 3-stage Successor of 78K0R [29]:8 RL78-S2 16-bit 8× 8-bit ×4 banks 75 3-stage RL78-S1 8-bit 8× 8-bit (no bank) 74 (75-1) 3-stage 78K0R 16-bit 8× 8-bit ×4 banks 80 (75+5) 3-stage Extended 78K/0 [28]:18 78K0S 8-bit 8× 8-bit (no bank) 47 none Simplified 78K/0 [25] 78K0 8-bit 8× 8-bit ×4 banks 48 none Basic 78K/0 core [19] 178K0S 8-bit 8× 8-bit 47 none 78K/0S for DTS;Digital Tuning System [25][32] 178K0 8-bit 8× 8-bit ×4 banks 48 none 78K/0 for DTS [19][31] 78K4 16-bit 16× 8-bit ×4 banks 113 none Macro service available [37]:24,128 78K7 32-bit 16× 16-bit ×16 banks none Macro service available [46] 78K6 16-bit Macro service available 78K1 8-bit 8× 8-bit ×4 banks 64 none For VCR servo controls [40]:3,39 78K3 16-bit 16× 8-bit ×8 banks 113–115 none Macro service available [47]:3–28,45 78K2 8-bit 8× 8-bit ×4 banks 65 none General purpose [44]:16,50 87AD 8-bit 8× 8-bit ×2 banks CMOS: 159NMOS: 158 none Predecessor of 78K [45]:21,39 (17K) 4-bit 128× 4-bit ×2 banks 47 none Predecessor of 178K [30] RL78 NEC V20 V850 Renesas 740 IEBus ^ Oklobdzija, Vojin G. (2001). The Computer Engineering Handbook. CRC Press. ISBN 9780849308857. ^ Edwards, Lewin A. R. W. (2006). So, You Wanna be an Embedded Engineer: The Guide to Embedded Engineering, from Consultancy to the Corporate Ladder. Newnes. p. 78. ISBN 9780750679534. NEC 78K. ^ "78k | The CPU Shack Museum". www.cpushack.com. ^ a b c Parai, Manas Kumar; Das, Banasree; Das, Gautam (January 2013). "An Overview of Microcontroller Unit: From Proper Selection to Specific Application". International Journal of Soft Computing and Engineering. 2 (6): 228–231. ISSN 2231-2307. S2CID 11529467. ^ NECエレクトロニクス 8ビットマイコンのあゆみ [History of 8-bit microcontrollers of NEC Electronics] (PDF) (in Japanese). Sunhayato Corp. ^ "microcontroller". The Free Dictionary. ^ Oklobdzija, Vojin G. (2017). Digital Systems and Applications. CRC Press. ISBN 9781351838108. ^ "NEC Electronics Introduces 12 New 16-bit All Flash Microcontrollers with LCD Controller/Driver Circuit". Business Wire. 2009-01-16. ^ Garcia, Pedro Castillo; Lozano, Rogelio; Dzul, Alejandro Enrique (2006). Modelling and Control of Mini-Flying Machines. Springer Science & Business Media. ISBN 9781846281792. ^ Archived: "MPU & MCU | Renesas Electronics". 7 October 2012. Archived from the original on 2012-10-07. ^ Emilio, Maurizio Di Paolo (2014). Embedded Systems Design for High-Speed Data Acquisition and Control. Springer. ISBN 9783319068657. ^ Electronic Specifier (2009-04-19). "NEC Electronics' new 78K Primer Kit". www.electronicspecifier.com. ^ Bender, Klaus; Jack, Peter; Koç, Ali; Péter, Istvan; Megyeri, Gergely (2001). Qualitätssicherung eingebetteter Software : Methoden und Best-Practices : [FUSIM] (in German). München: Herbert Utz Verlag. ISBN 9783831600243. ^ a b Renesas official: Renesas MPUs & MCUs 78K MCU Selection Guide. Renesas Electronics. ^ a b Dean, Alexander G.; Conrad, James M. (2012). Creating Fast, Responsive and Energy-Efficient Embedded Systems using the Renesas RL78 Microcontroller (PDF). Weston, FL: Micrium Press. ISBN 9781935772989. ^ a b Renesas official: Porting guide from 78K0R/FC3 to RL78/F14. Renesas Electronics. ^ "Renesas Electronics Introduces the New RL78 Microcontroller Family to Deliver Solutions for Next-Generation 8-/16-bit Embedded Applications". Renesas Electronics. ^ Hausmann, G.; Gebing, E. (1997). "The realisation of specific automotive applications with "Full" CAN functionality at "Basic" CAN cost on highly integrated 8-Bit microcontroller of NEC's 78K/0 family" (PDF). 4th International CAN Conference, ICC. 97: 4-02–11. ^ a b c Renesas official: 78K/0 Series for Instructions. Renesas Electronics. ^ Renesas official: UPD78054,78054Y Subseries User's Manual. Renesas Electronics. ^ "NEC launches 14 new 8-bit MCUs for automotive dashboard applications | EE Times". EETimes. ^ Renesas official: 78K0/Dx2 User's Manual: Hardware. Renesas Electronics. ^ Suzuki, Tetsuya (2007-06-22). Google Translate - Introduction site of: Cコンパイラで遊ぶ78K0Sマイコン [Play with the C compiler 78K0S microcomputer] (in Japanese). Tokyo, Japan: Socym Co,.Ltd. ISBN 9784883375394. ^ Skorobogatov, Sergei (17 August 2010). "Flash Memory 'Bumping' Attacks". Cryptographic Hardware and Embedded Systems, CHES 2010 (PDF). Lecture Notes in Computer Science. Vol. 6225. Springer, Berlin, Heidelberg. pp. 158–172. doi:10.1007/978-3-642-15031-9_11. ISBN 9783642150319. ^ a b c Renesas official: 78K/0S Series for Instructions. Renesas Electronics. ^ Renesas official: "Difference on 78K0 and 78K0S in 8-bit All Flash microcontrollers". Renesas Electronics - Knowledgebase. 28 June 2016. ^ Kim, Dahoo; Hida, Itaru; Fukuda, Eric S.; Asai, Tetsuya; Motomura, Masato (November 2014). A Study of Transparent On-chip Instruction Cache for NV Microcontrollers. The Seventh International Conference on Advances in Circuits, Electronics and Micro-electronics. pp. 26–29. CiteSeerX 10.1.1.676.6935. ISBN 978-1-61208-379-7. ISSN 2308-426X. ^ a b Renesas official: 78K0R Microcontrollers User's Manual: Instructions. Renesas Electronics. ^ a b Renesas official: RL78 family User's Manual: Software. Renesas Electronics. ^ a b c 17K 4-bit Microcontroller Data Book (1992). NEC. ^ a b Renesas official: UPD178024 Subseries User's Manual. Renesas Electronics. ^ Renesas official: UPD179327 Subseries User's Manual. Renesas Electronics. ^ JPRS Report: Science & technology. Japan. Foreign Broadcast Information Service. 1994. p. 25. The 78K/IV has upward compatibility regarding the instruction sets of the existing 78K/0, 78K/II and 78K/III. Major features of the 78K/IV are: 1) linear addressing of 16 M bytes, 2) wide operative voltage = 2.7-6.0 V, 3) efficient power management, 4) instruction sets for C compiler. NEC has developed the 1st product "puPD784026 subseries" that has upward-compatible peripheral functions of the 78K/II series. ^ Ohuchi, Mitsuro; Kawata, Kazuhide; Akiyama, Shin-ichiro; Imamura, Hirohisa; Fukushima, Kiyoshi; Ishizaki, Norihiko; Imamizu, Jun-ichi; Mori, Takehiko; Ono, Hirohihiko; Nakata, Shigeru (1994). "16ビットシングルチップマイクロコンピュ-タ78K/4シリ-ズ (半導体デバイス)" [16-Bit Single Chip Microcomputer 78K/IV Series.]. NEC Technical Journal. 47 (3): 122–127. ^ "NEC: News Release 96/10/30-01". www.nec.co.jp. ^ "Micro Controller (Data Part)". www.cpe.ku.ac.th. ^ a b Renesas official: 78K/IV Series Instructions. Renesas Electronics. ^ Renesas official: UPD784908 Subseries Hardware (Preliminary). Renesas Electronics. ^ Lui, Dr. Gough (16 August 2013). "Salvage: Quantum Fireball 1280Mb AT Hard Drive". Gough's Tech Zone. Phto 2. ^ a b Renesas official: UPD78148 User's Manual. Renesas Electronics. ^ Renesas official:

UPD78334 User's Manual. Renesas Electronics. ^ "Software repair of hard disks HDD (Google Translate)". www.phantom.sannata.ru (in Russian). 4X_Pro. ^ Renesas official: UPD78366A Hardware. Renesas Electronics. ^ a b Renesas official: UPD78234 Sub-Series Hardware. Renesas Electronics. ^ a b Renesas official: 87AD Series UPD78C18 User's Manual. Renesas Electronics. ^ 787012 User's Manual Hardware Edition (1994). NEC. ^ Renesas official: UPD78356 Instructions. Renesas Electronics. Technical documents Application note: 78K/0 Series Basic (I) | Renesas Electronics Application note: 78K/0 Series Basic(II) | Renesas Electronics Application note: 78K/0 Series Basic (III) | Renesas Electronics Web sites 78K Family | Renesas Electronics 78K Family Software & Tools | Renesas Electronics CPU of the Day: NEC 78C11 Sample and the 78K family | The CPU Shack MUseum TESSERA TECHNOLOGY, Inc. Retrieved from " 3 The following pages link to 78K External tools (link count transclusion count sorted list) · See help page for transcluding these entries Showing 50 items. View (previous 50 | next 50) (20 | 50 | 100 | 250 | 500)Intel 80186 (links | edit) Motorola 68HC11 (links | edit) Microcontroller (links | edit) PowerPC (links | edit) Playdia (links | edit) Zilog Z80 (links | edit) Intel MCS-51 (links | edit) Intel MCS-48 (links | edit) ARM architecture family (links | edit) AVR microcontrollers (links | edit) SuperH (links | edit) H8 Family (links | edit) PIC microcontrollers (links | edit) TI MSP430 (links | edit) In-circuit emulation (links | edit) Packard Bell (links | edit) Renesas Electronics (links | edit) NEC (links | edit) NEC V20 (links | edit) JTAG (links | edit) COP8 (links | edit) CompactRISC (links | edit) Zilog eZ80 (links | edit) AMD Am29000 (links | edit) Walter Tenney Carleton (links | edit) Green Rockets Tokatsu (links | edit) MPC5xx (links | edit) Zilog Z8 (links | edit) Sharp NEC Display Solutions (links | edit) ARM7 (links | edit) Kunihiko Iwadare (links | edit) Zenith Data Systems (links | edit) Infineon TriCore (links | edit) In-system programming (links | edit) NEC µPD7720 (links | edit) M32R (links | edit) V850 (links | edit) FR-V (microprocessor) (links | edit) NEC V25 (links | edit) R8C (links | edit) NEC Software Solutions (links | edit) NEC Supertower (links | edit) Comparison of operating system kernels (links | edit) NEC Cup (China) (links | edit) NEC Shun-Ei (links | edit) Atmel ARM-based processors (links | edit) ARM9 (links | edit) Cypress PSoC (links | edit) Tadahiro Sekimoto (links | edit) View (previous 50 | next 50) (20 | 50 | 100 | 250 | 500) Retrieved from " WhatLinksHere/78K" Microcontroller board Arduino UnoArduino Uno R3 SMD board with ATmega328P MCU in SMD packageDeveloperarduino.ccManufacturerManyTypeSingle-board microcontroller[1]AvailabilityUno R4 webpageOperating systemNone, with bootloader (default), FreeRTOSCPU Atmel AVR (8-bit) ARM Cortex-M0+ (32-bit) ARM Cortex-M3 (32-bit) Intel Quark (x86) (32-bit) MemorySRAMStorageFlash, EEPROMWebsitearduino.cc The Arduino Uno is a series of open-source microcontroller board based on a diverse range of microcontrollers (MCU). It was initially developed and released by Arduino company in 2010.[2][3] The microcontroller board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[1] The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.[4] It can be powered by a USB cable or a barrel connector that accepts voltages between 7 and 20 volts, such as a rectangular 9-volt battery. It has the same microcontroller as the Arduino Nano board, and the same headers as the Leonardo board.[5][6] The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available. The word "uno" means "one" in Italian and was chosen to mark a major redesign of the Arduino hardware and software.[7] The Uno board was the successor of the Duemilanove release and was the 9th version in a series of USB-based Arduino boards.[8] Version 1.0 of the Arduino IDE for the Arduino Uno board has now evolved to newer releases.[4] The ATmega328 on the board comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.[3] While the Uno communicates using the original STK500 protocol,[1] it differs from all preceding boards in that it does not use a FTDI USB-to-UART serial chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.[9] Arduino RS232 Serial board - a predecessor with ATmega8 MCU The Arduino project started at the Interaction Design Institute Ivrea (IDII) in Ivrea, Italy. At that time, the students used a BASIC Stamp microcontroller, at a cost that was a considerable expense for many students. In 2003, Hernando Barragán created the development platform Wiring as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas, who are known for work on the Processing language. The project goal was to create simple, low-cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a printed circuit board (PCB) with an ATmega168 microcontroller, an IDE based on Processing, and library functions to easily program the microcontroller.[10] In 2003, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, added support for the cheaper ATmega8 microcontroller to Wiring. But instead of continuing the work on Wiring, they forked the project and renamed it Arduino. Early Arduino boards used the FTDI USB-to-UART serial chip and an ATmega168.[10] The Uno differed from all preceding boards by featuring the ATmega328P microcontroller and an ATmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. In June 2023, Arduino released two new flavors of the Uno; the R4 Minima and R4 Wifi. These mark a departure from previous boards as they use Renesas RA4M1 ARM Cortex M4 microcontroller with the R4 Wifi a Espressif ESP32-S3-MINI co-processor. These versions are form factor, pin and power compatible with version R1 to R3, so should be largely be able to be drop in replacements.[11] Arduino Uno R3 board with AVR-based ATmega328P MCU in DIP-28 package Microcontroller (MCU):[12] IC: Microchip ATmega328P (8-bit AVR core) Clock Speed: 16 MHz on Uno board, though IC is capable of 20 MHz maximum at 5 Volts Flash memory: 32 KB, of which 0.5 KB used by the bootloader SRAM: 2 KB EEPROM: 1 KB USART peripherals: 1 (Arduino software default configures USART as a 8N1 UART) SPI peripherals: 1 I²C peripherals: 1 Operating Voltage: 5 Volts Digital I/O Pins: 14 PWM Pins: 6 (Pin # 3, 5, 6, 9, 10 and 11)[13] Analog Input Pins: 6 DC Current per I/O Pin: 20 mA DC Current for 3.3V Pin: 50 mA Size: 68.6 mm x 53.4 mm Weight: 25 g ICSP Header: Yes Power Sources: USB connector. USB bus specification has a voltage range of 4.75 to 5.25 volts. The official Uno boards have a USB-B connector, but 3rd party boards may have a miniUSB / microUSB / USB-C connector. 5.5mm/2.1mm barrel jack connector. Official Uno boards support 6 to 20 volts, though 7 to 12 volts is recommended. The maximum voltage for 3rd party Uno boards varies between board manufacturers because various voltage regulators are used, each having a different maximum input rating. Power into this connector is routed through a series diode before connecting to VIN to protect against accidental reverse voltage situations. VIN pin on shield header. It has a similar voltage range of the barrel jack. Since this pin doesn't have reverse voltage protection, power can be injected or pulled from this pin. When supplying power into VIN pin, an external series diode is required in case barrel jack is used. When board is powered by barrel jack, power can be pulled out of this pin.[14] Arduino Uno R4 WiFi with ARM-based R7FA4M1AB MCU in 64pin SMD package Two Uno R4 boards are available: Uno R4 Minima and Uno R4 WiFi. The latter has a WiFi coprocessor and LED matrix, but the Minima doesn't. Common features on both Uno R4 Minima[15] and Uno R4 WiFi[16] boards: Microcontroller (MCU):[17] IC: Renesas R7FA4M1AB (32-bit ARM Cortex-M4F core with single-precision FPU) Clock Speed: 48 MHz Flash memory: 256 KB + bootrom SRAM: 32 KB (16 KB ECC) (16 KB parity) EEPROM: 8 KB (data flash) USART peripherals: 4 SPI peripherals: 2 I²C peripherals: 2 Operating Voltage: 5 Volts USB-C connector. Barrel jack connector and VIN pin on shield header supports up to a maximum of 24 volts DC. Additional features only available on the Uno R4 Minima board:[15] SWD programming connector. This is a 10-pin 5x2 1.27mm header for connecting the microcontroller (R7FA4M1AB) to an external SWD (serial wire debug) programming / debugging device. Additional features only available on the Uno R4 WiFi board:[16] WiFi coprocessor - 240 MHz Espressif ESP32-S3-MINI (IEEE802.11 b/g/n WiFi and Bluetooth 5 LE) and a 6-pin 3x2 2.54mm header for external programming. 12x8 LED matrix - it is driven by 11 GPIO pins using a charlieplexing scheme. Qwiic I²C connector. This 4-pin 1.00mm JST SH connector provides external connection to a 3.3 volt I²C bus. Don't attach 5 volt I²C devices directly to this connector.[18] RTC battery header pin (VRTC). This pin connects an external battery to the RTC (real-time clock) inside the microcontroller (R7FA4M1AB) to keep clock running when board is powered down. Connect this pin to positive side of 1.6 to 3.6 volt battery and negative side of battery to ground header pin (GND), such as a 3 volt lithium coin battery.[17] Remote-Off header pin (OFF). This pin disables the 5 volt buck switching voltage regulator (SL854102) when powered by the barrel jack or VIN header pin. Connect this pin to ground header pin (GND) to disable this voltage regulator. Header pinout of the Arduino Uno board LED: There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it is off. VIN: The input voltage to the Arduino/Genuino board when it is using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. 5V: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board. 3V3: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA. GND: Ground pins. IOREF: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source, or enable voltage translators on the outputs to work with the 5V or 3.3V. Reset: Typically used to add a reset button to shields that block the one on the board.[9] Each of the 14 digital pins and 6 analog pins on the Uno can be used as an input or output, under software control (using pinMode(), digitalWrite(), and digitalRead() functions). They operate at 5 volts. Each pin can provide or receive 20 mA as the recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50K ohm. A maximum of 40mA must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labeled A0 through A5; each provides 10 bits of resolution (i.e. 1024 different values). By default, they measure from ground to 5 volts, though it is possible to change the upper end of the range using the AREF pin and the analogReference() function.[9] In addition, some pins have specialized functions: Serial / UART: pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL serial chip. External interrupts: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. PWM (pulse-width modulation): pins 3, 5, 6, 9, 10, and 11. Can provide 8-bit PWM output with the analogWrite() function. SPI (Serial Peripheral Interface): pins 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK). These pins support SPI communication using the SPI library. TWI (two-wire interface) / I²C: pin SDA (A4) and pin SCL (A5). Support TWI communication using the Wire library. AREF (analog reference): Reference voltage for the analog inputs.[9] The Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows serial communication on any of the Uno's digital pins.[9] Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by the software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.[9] This setup has other implications. When the Uno is connected to a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened.[9] The following table compares official Arduino boards, and has a similar layout as a table in the Arduino Nano article. The table is split with a dark bar into two high-level microcontroller groups: 8-bit AVR cores (upper group), and 32-bit ARM Cortex-M cores (lower group). 3rd-party boards board names it doesn't automatically mean they are 100% identical to official Arduino boards. 3rd-party boards often have a different voltage regulator / different USB-to-UART chip / different color solder mask, and some have a different USB connector or additional features, too. [19] BoardName& Part# BoardSizeGroup BoardCommun-ication MCUPart# & Pins MCUI/OVoltage MCUCore MCUClock MCUFlash MCUSRAM MCUEEPROM MCUUSART& UART MCUSPI MCUI²C MCUOther BusPeripherals MCU Timers32/24/16/8/WD/RT/RC MCUADC& DAC MCUEngines Uno R3,[20]A000066,[9]Uno R3 SMD,[21]A000073[22] Uno USB-B ATmega328P,[12]28 pin DIP,32 pin SMD 5V(1.8-5.5V) 8bit AVR 16 MHz* 32 KB 2 KB 1 KB 1, 0 1 1 None 0, 0, 1, 2,WD 10bit,None None Uno WiFi R2,[23]ABX00021[24] Uno USB-B,WiFi,Bluetooth ATmega4809,[25]48 pin 5V(1.8-5.5V) 8bit AVR 16 MHz* 48 KB 6 KB 0.25 KB 4, 0 1 1 None 0, 0, 5,0,WD, RT 10bit,None None Leonardo,[26]A000057[27] Uno USB-Micro-B ATmega32U4,[28]44 pin 5V(2.7-5.5V) 8bit AVR 16 MHz 32 KB 2.5 KB 1 KB 1, 0 1 1 USB-FS 0, 0, 2, 1,WD, 10bit 10bit,None None Mega 2560 R3,[29]A000067[30] Mega USB-B ATmega2560,[31]100 pin 5V(4.5-5.5V) 8bit AVR 16 MHz 256 KB 8 KB 4 KB 4, 0 1 1 None 0, 0, 4, 2,WD 10bit,None None Uno R4 Minima,[15]ABX00080,[32]Uno R4 WiFi,[16]ABX00087,[33] Uno USB-C,WiFi* R7FA4M1AB,[17]64 pin 5V(1.6-5.5V) 32bit ARMCortex-M4F(FPU) 48 MHz 256 KB+ bootrom 32 KB(ECC)(parity) None+ 8 KBdata flash 4, 0 2 2 USB-FS,CAN-A/B 2, 0, 8, 0,WD, RC,24bit SysTick 14bit,12bit DMA x4,CRC, RNG,Crypto, Touch,LCD Zero,[34]ABX00003[35] Uno USB-Micro-Bx2 ATSAMD21G18,[36]48 pin 3.3V(1.62-3.63V) 32bit ARMCortex-M0+ 48 MHz 256 KB 32 KB None 6, 0 None None USB-FS,I²S 0, 4, 5, 0,WD, RC,24bit SysTick 12bit,10bit DMA x12,CRC32, Touch Due,[37]A000062[38] Mega USB-Micro-Bx2 ATSAM3X8E,[39]144 pin 3.3V(1.62-3.6V) 32bit ARMCortex-M3 84 MHz 512 KB+ bootrom 96 KB None 4, 1 1 2 USB-HS,CAN-A/B x2,I²S, SD 3, 0, 8, 0,WD, RT, RC,24bit SysTick 12bit,12bit x2 DMA x8,RNG GIGA R1 WiFi,[40]ABX00063[41] Mega USB-C,USB-A,WiFi,Bluetooth STM32H747XI,[42]240 pin 3.3V(1.62-3.6V) 32bit ARMCortex-M7F(M7)+ARMCortex-M4F(dual core)(FPU) 480 MHz(M7F),240 MHz(M4F) 2048 KB+ bootrom 1056 KB(ECC) None 4, 5 6 4 USB-HS & FS,CAN-A/B/FD x2,I²S x4, SD x2,S/PDIF x4, CEC,SWP, QSPI 2, 0, 18, 0,WD, RC,24bit SysTick 16bit x3,12bit x2 DMA x4,CRC, RNG,Graphics Arduino Leonardo board with ATmega32U4 MCU Arduino Due board with ATSAM3X8E MCU Table notes Board Size Group column - Simplified board dimension size grouping: Uno means similar size as Arduino Uno R3 and Duemilanove (predecessor) boards, Mega means similar size as the longer Arduino Mega 2560 R3 and Mega (predecessor) boards. This table has a similar layout as a table in the Arduino Nano article. MCU Part# / Pins column - MCU means microcontroller. All MCU information in this table was sourced from official datasheets in this column. The pin count is useful to determine the quantity of internal MCU features that are available. All MCU hardware features may not be available at the shield header pins because the MCU IC package has more pins than the shield header pins on the Arduino board (*). MCU I/O Voltage column - Microcontrollers on official Arduino boards are powered at a fixed voltage of either 3.3 or 5 volts, though some 3rd party boards have a voltage selection switch. The voltage rating of the microcontroller is stated inside parenthesis, though Arduino boards don't support this full range. MCU Clock column - MHz means 106 Hertz. The ATmega328P MPU and ATmega4809 MCU are rated for a maximum of 20 MHz, but the Uno R3 and Uno WiFi R2 boards both operate at 16 MHz. The following Arduino boards have a 32.768KHz crystal too: Uno WiFi R2, Zero, Due, GIGA R1 WiFi. The Uno R4 Minima has SMD footprints for a 32.768KHz crystal and two capacitors, but aren't installed. MCU memory columns - KB means 1024 bytes, MB means 10242 bytes. The R7FA4M1AB MCU (Uno R4 boards) contains data flash memory instead of EEPROM memory. MCU SRAM column - SRAM size doesn't include caches or peripheral buffers. ECC means SRAM has error correction code checking, Par means SRAM has parity checking. MCU USART/UART column - USARTs are software configurable to be a: UART / SPI / other peripherals (varies across MCUs). MCU Other Bus Peripherals column - For USB bus, "FS" means Full Speed (12 Mbps max), "HS" means High Speed (480 Mbps max). For CAN bus, "A" means CAN 2.0A, "B" means CAN 2.0B, "FD" means CAN-FD. Some buses require additional external circuitry to operate. MCU Timers column - The numbers in this column are the total number of each timer bit width, for example, the ATmega328P has one 16-bit timer and two 8-bit timers. "WD" means Watchdog timer, "RT" means Real Time Counter/Timer, "RC" means Real Time Clock (sec/min/hr). The 24-bit SysTick timer(s) inside the ARM cores aren't included in the 24-bit total in this column. PWM features are not documented in this table. AVR microcontrollers Atmel AVR instruction set In-system programming ^ a b c "Arduino UNO for beginners - Projects, Programming and Parts". makerspaces.com. 7 February 2017. Retrieved 4 February 2018. ^ "Arduino FAQ". 5 April 2013. Archived from the original on 27 November 2020. Retrieved 21 February 2018. ^ a b "What is Arduino?". learn.sparkfun.com. Retrieved 4 February 2018. ^ a b "Introduction to Arduino" (PDF). princeton.edu. Archived from the original (PDF) on 3 April 2018. Retrieved 4 February 2018. ^ "Arduino Nano". Arduino Official Store. Retrieved 2022-12-07. ^ "Arduino Leonardo with Headers". Archived from the original on 2021-05-15. ^ "Previous IDE Releases". Retrieved 2023-02-08. ^ "Arduino Older Boards". Retrieved 2023-02-08. ^ a b c d e f g h "Board; Uno R3; Store". Arduino. ^ a b Hernando Barragán (2016-01-01). "The Untold History of Arduino". arduinohistory.github.io. Retrieved 2016-03-06. ^ "Introducing the Arduino UNO R4! - News - SparkFun Electronics". www.sparkfun.com. Retrieved 2023-08-07. ^ a b "MCU; ATmega328P; Docs". Microchip. Archived from the original on March 27, 2023. ^ "What is Arduino? A Getting Started Guide". www.rs-online.com. Retrieved 2021-08-04. ^ "Using Vin pin on Arduino with a shield". Electrical Engineering Stack Exchange. Retrieved 2024-01-20. ^ a b c "Board; Uno R4 Minima; Docs". Arduino. Archived from the original on June 27, 2023. ^ a b c "Board; Uno R4 WiFi; Docs". Arduino. Archived from the original on June 27, 2023. ^ a b c "MCU; R7FA4M1AB; Docs". Renesas. Archived from the original on May 8, 2023. ^ "Qwiic I2C Connect System". SparkFun. Archived from the original on September 7, 2023. ^ "Seeeduino v4.3 (UNO ATmega328P)". Seeed Studio. Archived from the original on November 22, 2023. ^ "Board; Uno R3; Docs". Arduino. Archived from the original on May 17, 2023. ^ "Board; Uno R3 SMD; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Uno R3 SMD; Store". Arduino. ^ "Board; Uno WiFi R2; Docs". Arduino. Archived from the original on March 28, 2023. ^ "Board; Uno WiFi R2; Store". Arduino. ^ "MCU; ATmega4809; Docs". Microchip. Archived from the original on December 6, 2022. ^ "Board; Leonardo; Docs". Arduino. Archived from the original on April 5, 2023. ^ "Board; Leonardo; Store". Arduino. ^ "MCU; ATmega32U4; Docs". Microchip. Archived from the original on April 5, 2023. ^ "Board; Mega 2560 R3; Docs". Arduino. Archived from the original on April 21, 2023. ^ "Board; Mega 2560 R3; Store". Arduino. ^ "MCU; ATmega2560; Docs". Microchip. Archived from the original on March 1, 2023. ^ "Board; Uno R4 Minima; Store". Arduino. ^ "Board; Uno R4 WiFi; Store". Arduino. ^ "Board; Zero; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Zero; Store". Arduino. ^ "MCU; ATSAMD21G18; Docs". Microchip. Archived from the original on February 1, 2023. ^ "Board; Due; Docs". Arduino. Archived from the original on May 8, 2023. ^ "Board; Due; Store". Arduino. ^ "MCU; ATSAM3X8E; Docs". Microchip. Archived from the original on October 26, 2022. ^ "Board; GIGA R1 WiFi; Docs". Arduino. Archived from the original on May 14, 2023. ^ "Board; GIGA R1 WiFi; Store". Arduino. ^ "MCU; STM32H747XI; Docs". ST. Archived from the original on May 11, 2023. Attribution: This article incorporates text available under the CC BY-SA 3.0 license. Main article: List of books about Arduino Wikimedia Commons has media related to Arduino UNO. Arduino UNO official webpage What's the difference between UNO R3 and UNO R4 boards? Comparison of Various Arduino Boards Programming Cheat Sheets Sheet1, Sheet2 Pinout Diagrams Arduino Uno Board, ATmega328 DIP IC, ATmega328 SMD IC Electronic Schematics Uno "DIP" R3, Uno SMD R3, Uno R4 Minima, Uno R4 WiFi Differences Between Uno Board Revisions (R1/R2/R3) Mechanical Drawings Dimensions and Hole Patterns Dimensions, Hole Patterns, Header Locations and PCB Templates Retrieved from " Arduino Uno is a popular microcontroller development board based on 8-bit ATmega328P microcontroller. Along with ATmega328P MCU IC, it consists of other components such as crystal oscillator, serial communication, voltage regulator, etc. to support the microcontroller. This article explores the Arduino UNO pin diagram in detail along with basics on how to use this board and upload your first code. Please note that this article discusses the popular Arduino UNO R3 development board and not the latest Arduino UNO R4 board which was launched recently. If you are new you can also check out our article on Arduino UNO R3 vs Arduino UNO R4 to understand the differences between these two boards. Arduino Uno Pinout Configuration Pin Category Pin Name Pin Description Power Vin, 3.3V, 5V, GND Vin: Input voltage to Arduino when using an external power source. 5V: Regulated power supply used to power microcontroller and other components on the board. 3.3V: 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA. GND: ground pins. Reset Reset Resets the microcontroller. Analog Pins A0 – A5 Used to provide analog input in the range of 0-5V Input/Output Pins Digital Pins 0 - 13 Can be used as input or output pins. Serial 0(Rx), 1(Tx) Used to receive and transmit TTL serial data. External Interrupts 2, 3 To trigger an interrupt. PWM 3, 5, 6, 9, 11 Provides 8-bit PWM output. SPI 10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK) Used for SPI communication. Inbuilt LED 13 To turn on the inbuilt LED. TWI A4 (SDA), A5 (SCA) Used for TWI communication. AREF AREF To provide reference voltage for input voltage. Arduino Uno Technical Specifications Microcontroller ATmega328P – 8 bit AVR family microcontroller Operating Voltage 5V Recommended Input Voltage 7-12V Input Voltage Limits 6-20V Analog Input Pins 6 (A0 – A5) Digital I/O Pins 14 (Out of which 6 provide PWM output) DC Current on I/O Pins 40 mA DC Current on 3.3V Pin 50 mA Flash Memory 32 KB (0.5 KB is used for Bootloader) SRAM 2 KB EEPROM 1 KB Frequency (Clock Speed) 16 MHz Note: Complete technical information can be found in the Arduino UNO Datasheet, linked at the bottom of this page. Other Arduino Boards Arduino Nano, Arduino Pro Mini, Arduino Mega, Arduino Due, Arduino MKR1000 Wi-Fi Board, Arduino Leonardo Overview Arduino Uno is a microcontroller board based on 8-bit ATmega328P microcontroller. Along with ATmega328P, it consists other components such as crystal oscillator, serial communication, voltage regulator, etc. to support the microcontroller. Arduino Uno has 14 digital input/output pins (out of which 6 can be used as PWM outputs), 6 analog input pins, a USB connection, A Power barrel jack, an ICSP header and a reset button. Arduino UNO Pin Layout Overview The Arduino UNO pin layout is organized into distinct categories, including Power Pins, Digital Pins, Analog Pins, and Special Function Pins. Each category plays a specific role in enabling the functionality of the board. The Arduino UNO pinouts available under each category is shown in the image below Now, lets understand the function of each pin under in detail under different category Power Pins Power pins are essential for operating the board and connected devices. The main pins include: VIN: Accepts external power sources (7-12V). 5V and 3.3V: Provide regulated voltage outputs for peripherals. GND (Ground): Completes the circuit. IOREF: Supplies a voltage reference for I/O pins. Tip: Always verify the voltage compatibility of connected components to avoid damage. Digital Pins (0-13) The Arduino UNO has 14 digital pins that can function as inputs or outputs. Pins 0 (RX) and 1 (TX): Reserved for serial communication. Pins 2-13: General-purpose I/O pins. PWM Pins (3, 5, 6, 9, 10, 11): Support Pulse Width Modulation, ideal for applications like controlling motors and dimming LEDs. Use functions like pinMode(), digitalWrite(), and digitalRead() to interact with these pins. Analog Pins (A0-A5) Analog pins allow reading continuous voltage signals, often from sensors. Resolution: 10-bit (0 to 1023 range). Flexibility: Can also function as digital I/O pins when required. Special Function Pins Reset Pin: Resets the board when triggered. AREF: Used to provide an external voltage reference for analog inputs. Serial Pins (RX/TX): Facilitate UART communication for serial data exchange. ICSP Header The ICSP (In-Circuit Serial Programming) header allows direct programming of the microcontroller or connecting additional peripherals. MISO (Master-In-Slave-Out): Receives data from peripherals. MOSI (Master-Out-Slave-In): Sends data to peripherals. SCK (Serial Clock): Synchronizes data transfer. Communication Pins I2C Pins: SCL (Clock line) and SDA (Data line) are located on A5 and A4, respectively. SPI Pins: Share functionality with the ICSP header (MISO, MOSI, and SCK). UART Pins: TX (Pin 1) and RX (Pin 0) handle serial communication. Arduino Uno to ATmega328 Pin Mapping When ATmega328 chip is used in place of Arduino Uno, or vice versa, the image below shows the pin mapping between the two. Software (Arduino IDE) Arduino IDE (Integrated Development Environment) is required to program the Arduino Uno board. Download it from here. Programming Arduino Once arduino IDE is installed on the computer, connect the board with computer using USB cable. Now open the arduino IDE and choose the correct board by selecting Tools>Boards>Arduino/Genuino Uno, and choose the correct Port by selecting Tools>Port. Arduino Uno is programmed using Arduino programming language based on Wiring. To get it started with Arduino Uno board and blink the built-in LED, load the example code by selecting Files>Examples>Basics>Blink. Once the example code (also shown below) is loaded into your IDE, click on the 'upload' button given on the top bar. Once the upload is finished, you should see the Arduino's built-in LED blinking. Below is the example code for blinking: // the setup function runs once when you press reset or power the board void setup() { // initialize digital pin LED_BUILTIN as an output. pinMode(LED_BUILTIN, OUTPUT); } // the loop function runs over and over again forever void loop() { digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) delay(1000); // wait for a second digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW delay(1000); // wait for a second } Applications Prototyping of Electronics Products and Systems Multiple DIY Arduino Projects. Easy to use for beginner level DIYers and makers. Projects requiring Multiple I/O interfaces and communications. Commonly Asked Question when working with Arduino UNO Q1. What is the function of PWM pins on Arduino UNO? PWM pins generate variable output signals for tasks like motor control and dimming LEDs. Q2. Can I use analog pins as digital pins? Yes, analog pins (A0-A5) can be configured as digital I/O pins. Q3. What is the role of the ICSP header? It allows direct programming of the microcontroller or connecting advanced peripherals. Q4. What happens if I supply more than 5V to an I/O pin? Excess voltage can permanently damage the microcontroller. Q5. Are all digital pins PWM-capable? No, only pins 3, 5, 6, 9, 10, and 11 support PWM. Q6. How do I power the Arduino UNO? You can use the USB port, VIN pin, or DC power jack. 2D Model and Dimensions Share — copy and redistribute the material in any medium or format for any purpose, even commercially. Adapt — remix, transform, and build upon the material for any purpose, even commercially. The licensor cannot revoke these freedoms as long as you follow the license terms. Attribution — You must give appropriate credit , provide a link to the license, and indicate if changes were made . You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original. No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits. You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation . No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material. Arduino Uno Board Arduino is a single-board microcontroller meant to make the application more accessible which are interactive objects and its surroundings. The hardware features with an open-source hardware board designed around an 8-bit Atmel AVR microcontroller or a 32-bit Atmel ARM. Current models consists a USB interface, 6 analog input pins and 14 digital I/O pins that allows the user to attach various extension boards. The Arduino Uno board is a microcontroller based on the ATmega328. It has 14 digital input/output pins in which 6 can be used as PWM outputs, a 16 MHz ceramic resonator, an ICSP header, a USB connection, 6 analog inputs, a power jack and a reset button. This contains all the required support needed for microcontroller. In order to get started, they are simply connected to a computer with a USB cable or with a AC-to-DC adapter or battery. Arduino Uno Board varies from all other boards and they will not use the FTDI USB-to-serial driver chip in them. It is featured by the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. Arduino Uno with Digital Input/Output There are various types of Arduino boards in which many of them were third-party compatible versions. The most official versions available are the Arduino Uno R3 and the Arduino Nano V3. Both of these run a 16MHz Atmel ATmega328P 8-bit microcontroller with 32KB of flash RAM 14 digital I/O and six analogue I/O and the 32KB will not sound like as if running Windows. Arduino projects can be stand-alone or they can communicate with software on running on a computer. For e.g. Flash, Processing, Max/MSP). The board is clocked by a 16 MHz ceramic resonator and has a USB connection for power and communication. You can easily add micro SD/SD card storage for bigger tasks. Features of the Arduino Uno Board: It is an easy USB interface. This allows interface with USB as this is like a serial device. The chip on the board plugs straight into your USB port and supports on your computer as a virtual serial port. The benefit of this setup is that serial communication is an extremely easy protocol which is time-tested and USB makes connection with modern computers and makes it comfortable. It is easy-to-find the microcontroller brain which is the ATmega328 chip. It has more number of hardware features like timers, external and internal interrupts, PWM pins and multiple sleep modes. It is an open source design and there is an advantage of being open source is that it has a large community of people using and troubleshooting it. This makes it easy to help in debugging projects. It is a 16 MHz clock which is fast enough for most applications and does not speeds up the microcontroller. It is very convenient to manage power inside it and it had a feature of built-in voltage regulation. This can also be powered directly off a USB port without any external power. You can connect an external power source of upto 12v and this regulates it to both 5v and 3.3v. 13 digital pins and 6 analog pins. This sort of pins allows you to connect hardware to your Arduino Uno board externally. These pins are used as a key for extending the computing capability of the Arduino Uno into the real world. Simply plug your electronic devices and sensors into the sockets that correspond to each of these pins and you are good to go. This has an ICSP connector for bypassing the USB port and interfacing the Arduino directly as a serial device. This port is necessary to re-bootload your chip if it corrupts and can no longer used to your computer. It has a 32 KB of flash memory for storing your code. An on-board LED is attached to digital pin 13 to make fast the debugging of code and to make the debug process easy. Finally, it has a button to reset the program on the chip. Arduino was created in the year 2005 by two Italian engineers David Cuartielles and Massimo Banzi with the goal of keeping in mind about students to make them learn how to program the Arduino uno microcontroller and improve their skills about electronics and use it in the real world. Arduino uno microcontroller can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). ATmega168/328-Arduino Pin Mapping: ATmega168-328Arduino Pin Mapping Programming: The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and is derived from the IDE for the Processing programming language and the Wiring projects The Arduino Uno board can be programmed with the Arduino software. Select "Arduino Uno from the Tools > Board menu (according to the microcontroller on your board). The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol. You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . Pin Diagram of Arduino Uno The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by: On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode. Please refer to this link to know more about Arduino Projects for Engineering Students You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). Arduino Uno Starter Kit Microcontroller ATmega328 Operating Voltage 5V Input Voltage (recommended) 7-12V Input Voltage (limits) 6-20V Digital I/O Pins 14 (of which 6 provide PWM output) Analog Input Pins 6 DC Current per I/O Pin 40 Ma DC Current for 3.3V Pin 50 Ma Flash Memory 32 KB (ATmega328) of which 0.5 KB used by bootloader SRAM 2 KB (ATmega328) EEPROM 1 KB (ATmega328) Clock Speed 16 MHz Home / Hardware / UNO R3 Arduino UNO is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.The ATmega328P can easily be replaced, as it is not soldered to the board.The ATmega328P also features 1kb of EEPROM, a memory which is not erased when powered off.The Arduino UNO features a barrel plug connector, that works great with a standard 9V battery. Home / Hardware / UNO R3 Arduino UNO is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.The ATmega328P can easily be replaced, as it is not soldered to the board.The ATmega328P also features 1kb of EEPROM, a memory which is not erased when powered off.The Arduino UNO features a barrel plug connector, that works great with a standard 9V battery. Arduino is an incredibly important part of modern-day electronics. The ease with which these Arduino boards can be programmed makes them the best choice especially when it comes to integrating them with large-scale projects. In this article, we will get an overview of the basic components that make up an Arduino board. This will include talking about the brain of the Arduino i.e. the microcontroller, the pins, and the power supplies. What is an Arduino Board?Here is an overview of the whole Arduino board. Anybody who has worked on Arduino will know that is a small board consisting of multiple components like ICs, and USB which are interconnected to form a whole connection. Here is a list of all the components Arduino Board Analog Reference pinDigital Ground Digital Pins 2-13 Digital Pins 0-1/Serial In/Out - TX/RXReset Button - S1 In-circuit Serial Programmer ICSP pin Analog In Pins 0-5 Power and Ground PinsExternal Power Supply In (9-12VDC) - X1Toggles External Power and USB Power- SV1 USB (universal serial bus)Crystal Oscillator MicrocontrollersThe microcontroller used on the Arduino board is essentially used for controlling all major operations. The microcontroller is used to coordinate the input taken and execute the code written in a high-level language. This code is then implemented and relevant output is generated. The choice of microcontroller varies on the requirements of the project. The microcontroller used in the Arduino shown above is ATmega328P manufactured by ATMEL Company and it is the most common choice. Here are some features of this microcontroller. ATmega328P has 14 Digital I/O Pins. Out of the 14 pins, 6 provide PWM(pulse width modulation) output.ATmega328P can have 6 (DIP) or 8 (SMD) Analog Input Pins.The DC Current which is supplied to each I/O Pin is around 40 mA.ATmega328P has a flash memory of 32 KB.ATmega328P has a SRAM(Static Random-Access Memory) of 2 KB.ATmega328P has EEPROM(Electrically Erasable Programmable Read-Only Memory) of 1 KB. Communication Interface To function optimally, the Arduino needs to communicate with external devices like computers, sensors, and LEDs. By adding a communication interface, we can ensure that Arduino can receive and transfer data to external devices and therefore generate the required output. Let's understand the components that make up the Communication Interface of Arduino. Serial Communication (UART): The UART is a protocol used by Arduino for serial communication with other devices. UART stands for Universal Asynchronous Receiver/Transmitter and is used for bit data transfer. The built-in hardware in Arduino aids it in this communication with other sensors, actuators, Rasperry pies, and other boards.Inter-Integrated Circuit (I2C): This is another communication protocol that comes into the picture when we want multiple connections but with minimal wiring. The way it allows communication between multiple channels is using two wires known as the SDA - Serial Data Line and SCL - Serial Clock Line. Arduino is designed with pins that help the Arduino to connect with sensors and displays without any inconvenience.Serial Peripheral Interface (SPI): The last serial communication protocol that is used when we need high speed for data transfer. The multiple lines used in this protocol help to connect the microcontroller to other devices. Unlike I2C, it uses different wires to coordinate different tasks like communication, clock controls, etc. This protocol is suitable for connecting Arduino with SD cards, display modules, and digital-to-analog converters (DACs). Digital PinsIn general, digital pins are used for general purposes like taking input or generating output. The commands that are used for setting the modes of the pins are pinMode(), digitalRead(), and digitalWrite() commands. digitalWrite() is used to turn the resistors in each pin ON or OFF which will assign a HIGH or LOW value to the pins. The maximum current that can flow in each pin is 40 mA. Here are some digital pins. Serial: These pins are categorized into two types namely receive (RX) and transmit (TX) serial data. On the Arduino Diecimila, the two pins are usually numbered '0' and '1' when they perform the task of communication. They are also present at pin 12 where TX flashes the LED while data is sent and RX flashes when data is being received. Sometimes, they are used with an external TTL serial module (e.g. the Mini-USB Adapter). External Interrupts: As the name suggests, external interrupts are used to trigger an interrupt when required. This interruption can be due to a rising or falling edge, or a change in value. Once an interrupt is called, the Arduino will come to a halt and begin working only when told. These pins are PIN '2' and '3' which are controlled using the attachInterrupt() function.PWM: PWM stands for pulse width modulation. The pin numbers 3, 5, 6, 9, 10, and 11 are PWM pins. The analogWrite() function is used for generating an 8-bit output. So when a large output is to be received or transmitted, the 8-bit output is generated. On certain boards like ATmega8, these pins are limited and present at 9, 10, and 11.SPI(serial peripheral interface): This is a synchronous serial data protocol generally used by microcontrollers. This is present at pin number 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK) which are used by microcontrollers for communicating with different devices. The relationship can be understood as the output device acting as a slave to the master of the SPI bus.LED: Present at pin number 13 in some Arduino, LED is often used for testing purposes. The LED glows when the pin is HIGH, and turns off when the pin is LOW. Sometimes it is also possible to connect some external LEDS by using breadboard and jumper wires. Analog PinsIn general, the analog pins are used for general purposes like supporting 10-bit analog-to-digital conversion (ADC) which is performed using analog the Read() function. These analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. Analog pins are particularly helpful since they can store 0-255 bits which is not possible using digital pins. This feature is not available on every Arduino board. I2C(Inter-Integrated Circuit): These pins are present at numbers 4 (SDA) and 5 (SCL) and are used to perform I2C (TWI) communication. Note that we need to import the Wire library to use this protocol. Power PinsThe power pins are used to supply the power needed for operating the Arduino. In case some external source like a jack is being used to drive the power then it can be connected to this PIN. The supply of power that each board can take varies from one design to another and it is necessary to know this range for the board that you are using. Some Arduinos don't have the VIN pin since they only accept a regulated input, one such example is lililyPad 5V(Power Supply): This is the voltage that is used for driving components like the microcontroller on the board. This power can only come either from VIN or a source that can provide a regulated voltage of around 5V. Any voltage less than this will not turn the Arduino on.GND: This is known as the Ground pin and is used to set a reference level as the ground. This is automatically considered to be at the potential 0V. Other PinsAREF: The analog reference pin is often used to set an upper limit to the voltage for analog pins. This is set using the analogReference() function.Reset: This pin is used to reset the state of the microcontroller by setting all values to their default values. Once all the actions have been performed or some wrong program is executed then you might want to reset the Arduino so you can use this pin for that. Crystal OscillatorThe crystal oscillator is a device on Arduino that deals with issues involving time. The Arduino calculates time using this oscillator only. If you observe, you will see the number '16.000H9H' printed on top of the Arduino crystal. This indicates that the Arduino operates at a frequency of 16,000,000 Hertz pf. Crystal oscillators are very precise and accurate devices. For example, a crystal oscillator is also present on he Arduino to provide clock pulses to the microcontroller Atmega 328 and help it control all commands and order of execution. Applications of Arduino Arduino finds its applications in various fields due to their ability to perform different things. Let us see some of its applications: Arduinos are used in 3D printing where they perform the task of selecting how the printing will be performed.Arduinos are used for creating basic designs by makers, designers, hackers, and creators across the globe to create some great projects. Some of the projects are Laser Turret Midi Controller, Retro Gaming With an OLED Display, and Traffic Light Controller.Arduinos are used by college students to understand programmable electronics and to explore their interest in programming.Arduinos are used in the field of robotics for programming robots and adding basic features like sensing and responding to environmental conditions.Arduino is used in IoT(Internet of Things) since it can collect information using sensors. The collected data is then processed and transmitted for developing various smart devices.Conclusion We have seen how Arduino is made of various components that control the entire working of Arduino. These components are connected to coordinate all the tasks and ensure the Arduino can work as required. Apart from the microcontroller, which is the Braun of the Arduino, we studied various components like the pins and the oscillators. It is important to know the different components that make the hardware and software parts work together. Readers are advised to refer to the frequently asked questions in case of any doubts.