Power Quality And Status Measurement System Using IOT

Barigorla Lalitha[1], S.Ashok Reddy[2], S.Mahaboob Basha[3]
[1]P.G. Scholar,[2]Assistant Professor, [3]Head of the Department
[1,2,3] BRANCH : EMBEDDED SYSTEMS,ECE(Mtech)
[1,2,3]Geethanjali College Of Engineering And Technology
Email: [1]blalitha108@gmail.com, [2]singasaniashokreddy@gmail.com

Abstract

In order to raise awareness about energy waste, an energy monitoring system able to be used either in residential or commercial applications designed in this project. The acquisition system is based on Arduino, Wi-Fi and free software. The system has been designed considering low-power consumption and low-component count while meeting international standards.

Electricity is nowadays one of the most important raw material that is used in industry, institutions or households. Electricity is through the transmission and distribution grids supplied to the end user. During transmission of this energy there are many external factors that can impair the quality of energy supplied. It is very important to identify the source of problems with the quality of electricity and in the case of an internal source of low-quality supply, it is necessary to take corrective action. On the other hand, the electricity should be seen as an energy commodity trading with contractual parameters laid down in the standard EN 50160.

This system measures the electrical parameters like current, voltage and power factor and continuously uploads these values to the web page. The system uses ESP8266 as the Wi-Fi module to upload the parameters on to the web page. The current, voltage and power factor measuring circuits are designed individually and the parameter values are read by Arduino continuously. These values are also displayed on the LCD.

The power status (system on or off status), the current value (current consumed by the loads), the voltage value (the input voltage coming to the loads) and the power factor value will be read and continuously uploaded on the web page.

Keywords:- Building Energy Management system; Internet of Things; Power Quality; Smart Grid

INTRODUCTION

Electricity is an important component for any organization, the electricity quality should be always monitored. University of Lampung (Unila) very concern to perform evaluation of electricity services, especially at Department of Mechanical Engineering Unila. Various policy have been made including the addition of power supply, installation system repairs. However, these efforts did not provided maximum results without being followed by maintaining the quality of the electricity itself, because in reality the amount of electricity tends to change as a result of changes in the load or the power supply itself. These changes should be identified immediately in order to anticipate the consequences that may occur to the system.

As a matter of fact, In general, electrical energy sources use a three-phase system, whereas the loads for household adopt one-phase system. In term of this conditions that not monitored properly as well, then it will certainly result on unbalance of electrical quantities for each phase. Manual measurement conducted was certainly not very effective, because there are so many panels that should be monitored. Besides, the value of electricity will also change every time. For that reason there should be a system that can monitor this situation, measure the changes in the value of electrical quantities, and monitored data online by utilizing the internet network facilities that are already available in University. The proposed of this research is to implementing the Internet of Things (IoT) technology in field area of electrical quantities monitoring to provide useful information to operators, and updating the status.

LITERATURE REVIEW

Related works

Monitoring of electrical energy consumption, room temperature, and internet network bandwidth has been done on previous work but unfortunately Gigih et all in this research only provides information on the magnitude of electrical energy consumption by capturing voltage and current values, while the value of other electrical quantities did not proposed.

In the other work, Despa et all has succeeded created a prototype system that can monitor the quantity of electricity in a 3-phase electric power panel using BCM2835 Single Board Computer. In this research, the electrical quantities that can be monitored were not only electrical energy consumption, it also included voltage, current, power, power factor. The measurement process uses current sensors and voltage sensors. Current sensors used a type of sensor that mounts in series circuit with electrical systems that are at risk of disrupting the electrical system performance in case of damage to the sensor. The researcher on works then continued their research and focus to refine the prototype described in.

The use of current sensors was no longer installed serially and high risk impact to the system, but uses a safer current transformer type. However, in this paper, the real-time report was not provided, in the other hand the display function and data processing was still offline, another disadvantages, the prototype described in paper was still using two embedded system (single board computer BCM2835 and Arduino) which is not efficient economically.

Despa et all on work has created a simple prototype of a system of electrical monitoring in 3-phase electrical power panels using only Arduino. The electrical quantities that can be monitored was focused on voltage, current, power and power factor. The measurement data was stored in to MySQL database and displayed on the web. This monitoring system is the main reference in this study which has some drawbacks, such as the absence of some important parameters of electrical power like frequency of the system and energy consumption, whereas those values are very important to be considered in term of electrical system reliability and estimation. Some previous research related to implementation of Internet of Things technology can be found on works.

INTRODUCTION TO EMBEDDED SYSTEMS

Application Areas

Nearly 99 per cent of the processors manufactured end up in embedded systems. The embedded system market is one of the highest growth areas as these systems are used in very market segment- consumer electronics, office automation, industrial automation, biomedical engineering, wireless communication, data communication, telecommunications, transportation, military and so on.

Consumer appliances At home we use a number of embedded systems which include digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and air-conditioner, VCO player, video game consoles, video recorders etc. Today's high-tech car has about 20 embedded systems for transmission control, engine spark control, air-conditioning, navigation etc. Even wristwatches are now becoming embedded systems. The palmtops are powerful embedded systems using which we can carry out many general-purpose tasks such as playing games and word processing.

Office automation: The office automation products using em embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.

Industrial automation: Today a lot of industries use embedded systems for process control. These include pharmaceutical, cement, sugar, oil exploration, nuclear energy, electricity generation and transmission. The embedded systems for industrial use are designed to carry out specific tasks such as monitoring the temperature, pressure, humidity, voltage, current etc., and then take appropriate action based on the monitored levels to control other devices or to send information to a centralized monitoring station. In hazardous industrial environment, where human presence has to be avoided, robots are used, which are programmed to do specific jobs. The robots are now becoming very powerful and carry out many interesting and complicated tasks such as hardware assembly.

Medical electronics: Almost every medical equipment in the hospital is an embedded system. These equipments include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation, colonscopy, endoscopy etc. Developments in medical electronics have paved way for more accurate diagnosis of diseases.

Computer networking: Computer networking products such as bridges, routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM), X.25 and frame relay switches are embedded systems which implement the necessary data communication protocols. For example, a router interconnects two networks. The two networks may be running different protocol stacks. The router's function is to obtain the data packets from incoming pores, analyze the packets and send them towards the destination after doing necessary protocol conversion. Most networking equipments, other than the end systems (desktop computers) we use to access the networks, are embedded systems

Telecommunications: In the field of telecommunications, the embedded systems can be categorized as subscriber terminals and network equipment. The subscriber terminals such as key telephones, ISDN phones, terminal adapters, web cameras are embedded systems. The network equipment includes multiplexers, multiple access systems, Packet Assemblers Dissemblers (PADs), satellite modems etc. IP phone, IP gateway, IP gatekeeper etc. are the latest embedded systems that provide very low-cost voice communication over the Internet.

Wireless technologies: Advances in mobile communications are paving way for many interesting applications using embedded systems. The mobile phone is one of the marvels of the last decade of the 20th century. It is a very powerful embedded system that provides voice communication while we are on the move. The Personal Digital Assistants and the palmtops can now be used to access multimedia services over the Internet. Mobile communication infrastructure such as base station controllers, mobile switching centers are also powerful embedded systems.

Insemination: Testing and measurement are the fundamental requirements in all scientific and engineering activities. The measuring equipment we use in laboratories to measure parameters such as weight, temperature, pressure, humidity, voltage, current etc. are all embedded systems. Test equipment such as oscilloscope, spectrum analyzer, logic analyzer, protocol analyzer, radio communication test set etc. are embedded systems built around powerful processors. Thank to miniaturization, the test and measuring equipment are now becoming portable facilitating easy testing and measurement in the field by field-personnel.

Security: Security of persons and information has always been a major issue. We need to protect our homes and offices; and also the information we transmit and store. Developing embedded systems for security applications is one of the most lucrative businesses nowadays. Security devices at homes, offices, airports etc. for authentication and verification are embedded systems. Encryption devices are nearly 99 per cent of the processors that are manufactured end up in~ embedded systems. Embedded systems find applications in.every industrial segment- consumer electronics, transportation, avionics, biomedical engineering, manufacturing, process control and industrial automation, data communication, telecommunication, defense, security etc. Used to encrypt the data/voice being transmitted on communication links such as telephone lines. Biometric systems using fingerprint and face recognition are now being extensively used for user authentication in banking applications as well as for access control in high security buildings.

Finance: Financial dealing through cash and cheques are now slowly paving way for transactions using smart cards and ATM (Automatic Teller Machine, also expanded as Any Time Money) machines. Smart card, of the size of a credit card, has a small micro-controller and memory; and it interacts with the smart card reader! ATM machine and acts as an electronic wallet. Smart card technology has the capability of ushering in a cashless society. Well, the list goes on. It is no exaggeration to say that eyes wherever we go, we can see, or at least feel, the work of an embedded system.

HARDWARE IMPLEMENTATION OF THE PROJECT

This chapter briefly explains about the Hardware Implementation of the project. It discusses the design and working of the design with the help of block diagram and circuit diagram and explanation of circuit diagram in detail. It explains the features, timer programming, serial communication, interrupts of atmega328 microcontroller. It also explains the various modules used in this project.

Project Design

The implementation of the project design can be divided in two parts.

- Hardware implementation
- Firmware implementation

Hardware implementation deals in drawing the schematic on the plane paper according to the application, testing the schematic design over the breadboard using the various IC's to find if the design meets the objective, carrying out the PCB layout of the schematic tested on breadboard, finally preparing the board and testing the designed hardware.

The project design and principle are explained in this chapter using the block diagram and circuit diagram. The block diagram discusses about the required components of the design and working condition is explained using circuit diagram and system wiring diagram.

INTRODUCTION TO MICROCONTROLLER

Based on the Processor side Embedded Systems is mainly divided into 3 types

1. Micro Processor : - are for general purpose eg: our personal computer

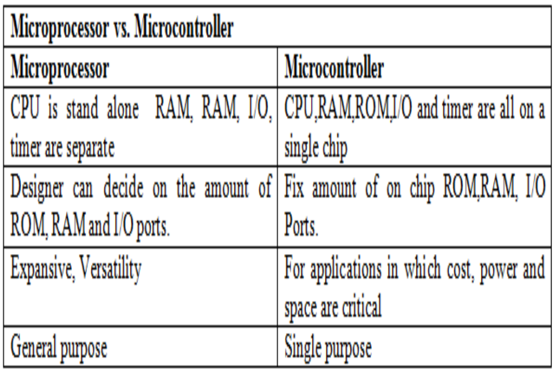
2. Micro Controller:- are for specific applications, because of cheaper cost we will go for these

3. DSP (Digital Signal Processor):- are for high and sensitive application purpose

MICROCONTROLLER VERSUS MICROPROCESSOR

A system designer using a general-purpose microprocessor such as the Pentium or the 68040 must add RAM, ROM, I/O ports, and timers externally to make them functional. Although the addition of external RAM, ROM, and I/O ports makes these systems bulkier and much more expensive, they have the advantage of versatility such that the designer can decide on the amount of RAM, ROM and I/O ports needed to fit the task at hand.

A Microcontroller has a CPU (a microprocessor) in addition to a fixed amount of RAM, ROM, I/O ports, and a timer all on a single chip. In other words, the processor, the RAM, ROM, I/O ports and the timer are all embedded together on one chip; therefore, the designer cannot add any external memory, I/O ports, or timer to it. The fixed amount of on-chip ROM, RAM, and number of I/O ports in Microcontrollers makes them ideal for many applications in which cost and space are critical.



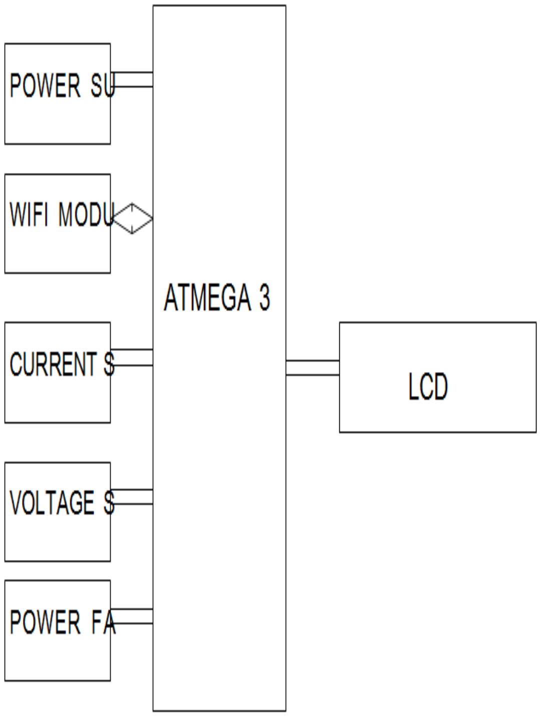
Microprocessor vs. Microcontroller	
Microprocessor	Microcontroller
CPU is stand alone RAM, RAM, I/O, timer are separate	CPU,RAM,ROM,I/O and timer are all on a single chip
Designer can decide on the amount of ROM, RAM and I/O ports.	Fix amount of on chip ROM,RAM, I/O Ports.
Expansive, Versatility	For applications in which cost, power and space are critical
General purpose	Single purpose

Table: Microprocessor versus Microcontroller

Block Diagram of the Project

The block diagram of the design is as shown in the figure below. The brief description about block diagram is given below.

Block diagram of proposed system



Power Supply:

The input to the circuit is applied from the regulated power supply. The a.c. input i.e., 230V from the mains supply is step down by the transformer to 12V and is fed to a rectifier. The output obtained from the rectifier is a pulsating d.c voltage. So in order to get a pure d.c voltage, the output voltage from the rectifier is fed to a filter to remove any a.c components present even after rectification. Now, this voltage is given to a voltage regulator to obtain a pure constant dc voltage.

Voltage regulator:

As the name itself implies, it regulates the input applied to it. A voltage regulator is an electrical regulator designed to automatically maintain a constant voltage level. In this project, power supply of 5V and 12V are required. In order to obtain these voltage levels, 7805 and 7812 voltage regulators are to be used. The first number 78 represents positive supply and the numbers 05, 12 represent the required output voltage levels.

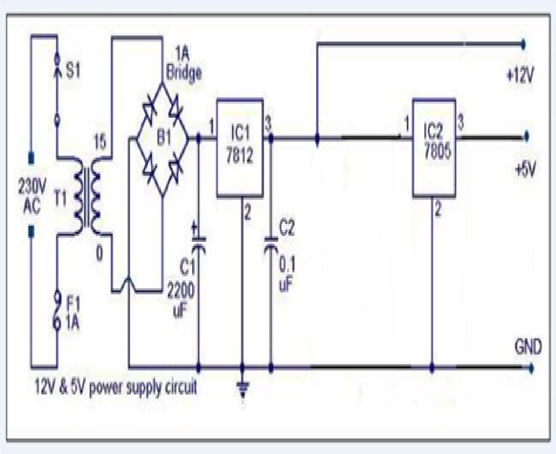


Fig: Power supply circuit diagram

ARDUINO UNO

Arduino is used for building different types of electronic circuits easily using of both a physical programmable circuit board usually microcontroller and piece of code running on computer with USB connection between the computer and Arduino. Programming language used in Arduino is just a simplified version of C++ that can easily replace thousands of wires with words.

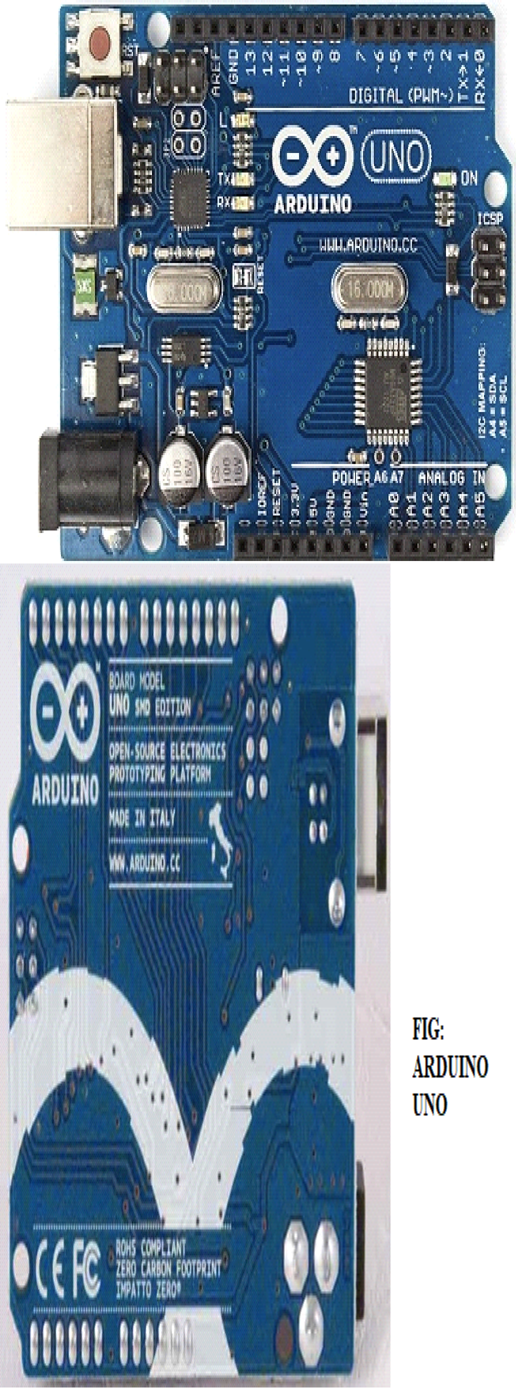


FIG: ARDUINO UNO

ARDUINO UNO-R3 PHYSICAL COMPONENTS

ATMEGA328P-PU microcontroller

The most important element in Arduino Uno R3 is ATMEGA328P-PU is an 8-bit Microcontroller with flash memory reach to 32k bytes.

Features

1. High Performance, Low Power Atmel®AVR® 8-Bit Microcontroller Family

- Advanced RISC Architecture
  - 131 Powerful Instructions
  - Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 32KBytes of In-System Self-Programmable Flash program

2. Memory

  - 1KBytes EEPROM
  - 2KBytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data Retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
- In-System Programming by On-chip Boot Program
- True Read-While-Write Operation
  - Programming Lock for Software Security
- Atmel® QTouch® Library Support
  - Capacitive Touch Buttons, Sliders and Wheels
  - QTouch and QMatrix® Acquisition
  - Up to 64 sense channels

3. Atmel-42735B-ATmega328/P_Datasheet_Complete-11/2016

- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
- Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
- Temperature Measurement
  - Two Master/Slave SPI Serial Interface
  - One Programmable Serial USART
  - One Byte-oriented 2-wire Serial Interface (Philips I2C compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - One On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and

4. Extended Standby

- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
  - 1.8 - 5.5V
- Temperature Range:
  - -40°C to 105°C
- Speed Grade:
  - 0 - 4MHz @ 1.8 - 5.5V
  - 0 - 10MHz @ 2.7 - 5.5V
  - 0 - 20MHz @ 4.5 - 5.5V

    Power Consumption at 1MHz, 1.8V, 25°C

    Active Mode: 0.2mA

    Power-down Mode: 0.1µA
  - Power-save Mode: 0.75µA (Including 32kHz RTC)

OTHER ARDUINO UNO R3 PARTS

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode (), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 k Ohms.

In addition, some pins have specialized functions:

- **Serial: 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: 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: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI**: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off. The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e.1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

  Additionally, some pins have specialized functionality:
- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wire library. There are a couple of other pins on the board:
- **AREF:** Reference voltage for the analog inputs. Used with analogReference().
- **Reset:** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, 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. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino 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 for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a
Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the reference and tutorials.

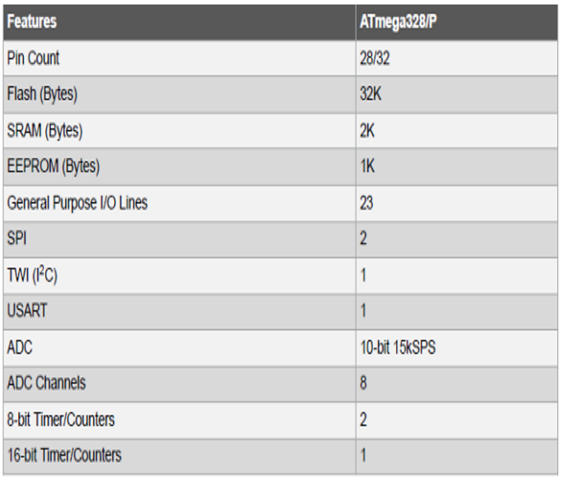
The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available. The
ATmega16 U2/8U2 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. 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).

Description
The Atmel AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in a single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.
Atmel offers the QTouch® library for embedding capacitive touch buttons, sliders and wheels functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS™) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.
The device is manufactured using Atmel’s high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core.
The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega328/P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications. The ATmega328/P is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

Configuration Summary



Features	ATmega328/P
Pin Count	28/32
Flash (Bytes)	32K
SRAM (Bytes)	2K
EEPROM (Bytes)	1K
General Purpose I/O Lines	23
SPI	2
TWI (I²C)	1
USART	1
ADC	10-bit 15kSPS
ADC Channels	8
8-bit Timer/Counters	2
16-bit Timer/Counters	1

Table: Arduino Configurations

Block Diagram

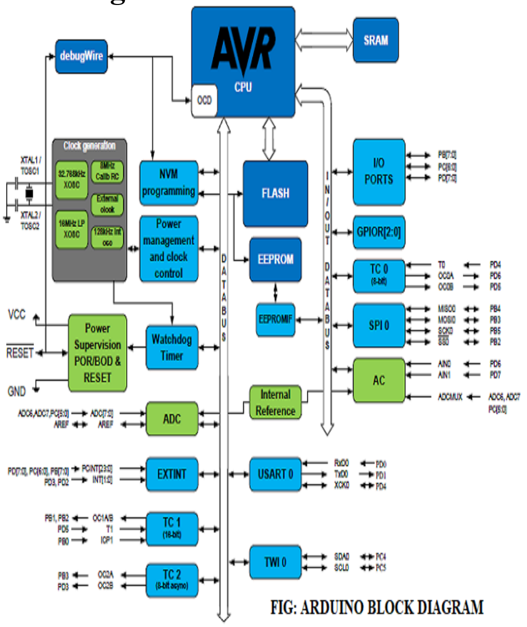


FIG: ARDUINO BLOCK DIAGRAM

Pin Configurations

Pin-out

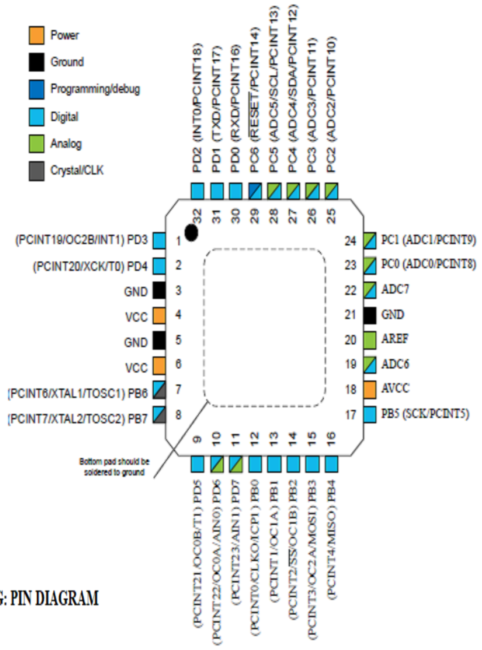


FIG: PIN DIAGRAM

Pin Descriptions

VCC - Digital supply voltage

GND – Ground

Port B (PB[7:0]) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier. If the Internal Calibrated RC Oscillator is used as chip clock source, PB[7:6] is used as TOSC[2:1] input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

Port C (PC[5:0])

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC[5:0] output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

PC6/RESET

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C. If the RSTDISBL Fuse is un programmed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a Reset. The various special features of Port C are elaborated in the *Alternate Functions of Port C* section.

Port D (PD[7:0])

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

AVCC

AVCC is the supply voltage pin for the A/D Converter, PC[3:0], and PE[3:2]. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC[6:4] use digital supply voltage, VCC.

AREF

AREF is the analog reference pin for the A/D Converter.

ADC[7:6] (TQFP and VFQFN Package Only)

In the TQFP and VFQFN package, ADC[7:6] serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

I/O Multiplexing

Each pin is by default controlled by the PORT as a general purpose I/O and alternatively it can be assigned to one of the peripheral functions. The following table describes the peripheral signals multiplexed to the PORT I/O pins.

TYPES OF ARDUINO BOARDS:-

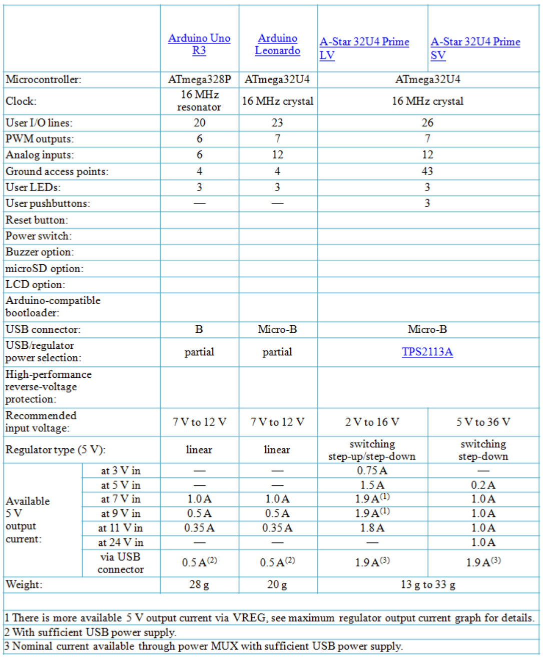
- Arduino UNO.
- Arduino MEGA.
- Arduino MINI.
- Arduino NANO
- Arduino DUE.
- Arduino YUN.
- Arduino Lily pad.
- Arduino Duemilanova.

Apart from this there are many more boards that can be used. As it's open source instead of Arduino you can also find, Freeduino, Arkduino etc. available in the market. Selection of Board should be done according to the application.

Choosing the right controller

The table below compares the Arduino Uno, Leonardo, and our A-Star 32U4 Prime controllers. The A-Star Primes are based on the same ATmega32U4 AVR microcontroller as the Leonardo and ship

with Arduino-compatible bootloaders. The Primes also offer many advantages, including superior power management that enables efficient operation from 2.7 V to 11.8 V (LVversion) or 5 V to 36 V (SV version).



		Arduino Uno R3	Arduino Leonardo	A-Star 32U4 Prime LV	A-Star 32U4 Prime SV
Microcontroller:		ATmega328P	ATmega32U4	ATmega32U4	
Clock:		16 MHz resonator	16 MHz crystal	16 MHz crystal	
User I/O lines:		20	23	26	
PWM outputs:		6	7	7	
Analog inputs:		6	12	12	
Ground access points:		4	4	43	
User LEDs:		3	3	3	
User pushbuttons:		—	—	3	
Reset button:					
Power switch:					
Buzzer option:					
microSD option:					
LCD option:					
Arduino-compatible bootloader:					
USB connector:		B	Micro-B	Micro-B	
USB/regulator power selection:		partial	partial	TPS2113A	
High-performance reverse-voltage protection:					
Recommended input voltage:		7 V to 12 V	7 V to 12 V	2 V to 16 V	5 V to 36 V
Regulator type (5 V):		linear	linear	switching step-up/step-down	switching step-down
Available 5 V output current:	at 3 V in	—	—	0.75 A	—
	at 5 V in	—	—	1.5 A	0.2 A
	at 7 V in	1.0 A	1.0 A	1.9 A(1)	1.0 A
	at 9 V in	0.5 A	0.5 A	1.9 A(1)	1.0 A
	at 11 V in	0.35 A	0.35 A	1.8 A	1.0 A
	at 24 V in	—	—	—	1.0 A
	via USB connector	0.5 A(2)	0.5 A(2)	1.9 A(3)	1.9 A(3)
Weight:		28 g	20 g	13 g to 33 g	

1 There is more available 5 V output current via VREG, see maximum regulator output current graph for details.
2 With sufficient USB power supply.
3 Nominal current available through power MUX with sufficient USB power supply.

LIQUID CRYSTAL DISPLAY:

LCD stands for Liquid Crystal Display. LCD is finding wide spread use replacing LEDs (seven segment LEDs or other multi segment LEDs) because of the following reasons:

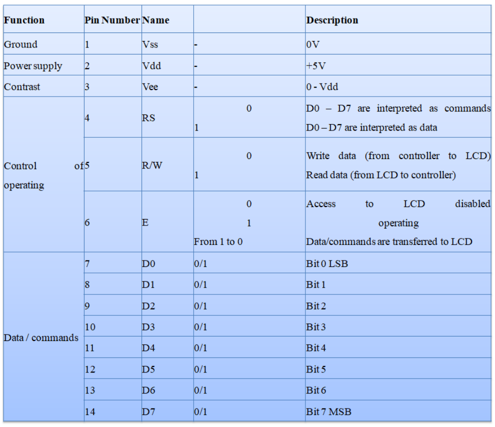
1. The declining prices of LCDs.
2. The ability to display numbers, characters and graphics. This is in contrast to LEDs, which are limited to numbers and a few characters.
3. Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD. In contrast, the LED must be refreshed by the CPU to keep displaying the data.
4. Ease of programming for characters and graphics.

These components are "specialized" for being used with the microcontrollers, which means that they cannot be activated by standard IC circuits. They are used for writing different messages on a miniature LCD.

A model described here is for its low price and great possibilities most frequently used in practice. It is based on the HD44780 microcontroller (*Hitachi*) and can display messages in two lines with 16 characters each. It displays all the alphabets, Greek letters, punctuation marks, mathematical symbols etc. In addition, it is possible to display symbols that user makes up on its own. Automatic shifting message on display (shift left and right), appearance of the pointer, backlight etc. are considered as useful characteristics.

Pins Functions

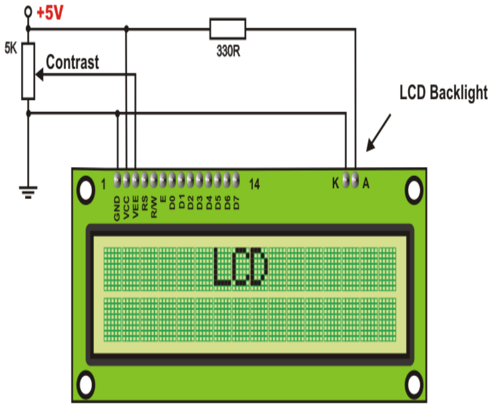
There are pins along one side of the small printed board used for connection to the microcontroller. There are total of 14 pins marked with numbers (16 in case the background light is built in). Their function is described in the table below:



Function	Pin Number	Name		Description
Ground	1	Vss	-	0V
Power supply	2	Vdd	-	+5V
Contrast	3	Vee	-	0 - Vdd
Control of operating	4	RS	0 1	D0 – D7 are interpreted as commands D0 – D7 are interpreted as data
	5	R/W	0 1	Write data (from controller to LCD) Read data (from LCD to controller)
	6	E	0 1 From 1 to 0	Access to LCD disabled operating Data/commands are transferred to LCD
Data / commands	7	D0	0/1	Bit 0 LSB
	8	D1	0/1	Bit 1
	9	D2	0/1	Bit 2
	10	D3	0/1	Bit 3
	11	D4	0/1	Bit 4
	12	D5	0/1	Bit 5
	13	D6	0/1	Bit 6
	14	D7	0/1	Bit 7 MSB

LCD screen:

LCD screen consists of two lines with 16 characters each. Each character consists of 5x7 dot matrix. Contrast on display depends on the power supply voltage and whether messages are displayed in one or two lines. For that reason, variable voltage 0-Vdd is applied on pin marked as Vee. Trimmer potentiometer is usually used for that purpose. Some versions of displays have built in backlight (blue or green diodes). When used during operating, a resistor for current limitation should be used (like with any LE diode).

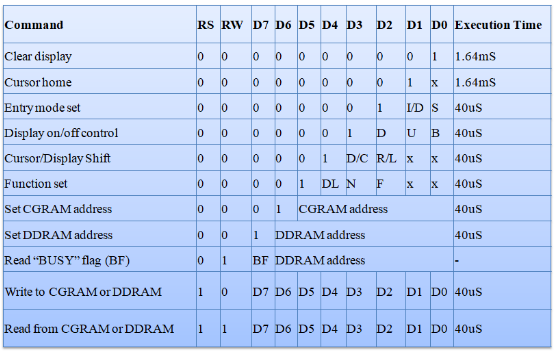


LCD Basic Commands

All data transferred to LCD through outputs D0-D7 will be interpreted as commands or as data, which depends on logic state on pin RS:

RS = 1 - Bits D0 - D7 are addresses of characters that should be displayed. Built in processor addresses built in “map of characters” and displays corresponding symbols. Displaying position is determined by DDRAM address. This address is either previously defined or the address of previously transferred character is automatically incremented.

RS = 0 - Bits D0 - D7 are commands which determine display mode. List of commands which LCD recognizes are given in the table below:



Command	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	Execution Time
Clear display	0	0	0	0	0	0	0	0	0	1	1.64mS
Cursor home	0	0	0	0	0	0	0	0	1	x	1.64mS
Entry mode set	0	0	0	0	0	0	0	1	I/D	S	40uS
Display on/off control	0	0	0	0	0	0	1	D	U	B	40uS
Cursor/Display Shift	0	0	0	0	0	1	D/C	R/L	x	x	40uS
Function set	0	0	0	0	1	DL	N	F	x	x	40uS
Set CGRAM address	0	0	0	1	CGRAM address						40uS
Set DDRAM address	0	0	1	DDRAM address							40uS
Read “BUSY” flag (BF)	0	1	BF	DDRAM address							-
Write to CGRAM or DDRAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	40uS
Read from CGRAM or DDRAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	40uS

I/D 1 = Increment (by 1)
R/L 1 = Shift right
0 = Decrement (by 1)
0 = Shift left
S 1 = Display shift on
DL 1 = 8-bit interface
0 = Display shift off
0 = 4-bit interface
D 1 = Display on
N 1 = Display in two lines
0 = Display off
0 = Display in one line
U 1 = Cursor on
F 1 = Character format 5x10 dots
0 = Cursor off
0 = Character format 5x7 dots
B 1 = Cursor blink on
D/C 1 = Display shift
0 = Cursor blink off
0 = Cursor shift

LCD Initialization:

Once the power supply is turned on, LCD is automatically cleared. This process lasts for approximately 15mS. After that, display is ready to operate. The mode of operating is set by default. This means that:

1. Display is cleared

2. Mode

DL = 1 Communication through 8-bit interface

N = 0 Messages are displayed in one line

F = 0 Character font 5 x 8 dots

3. Display/Cursor on/off

D = 0 Display off

U = 0 Cursor off

B = 0 Cursor blink off

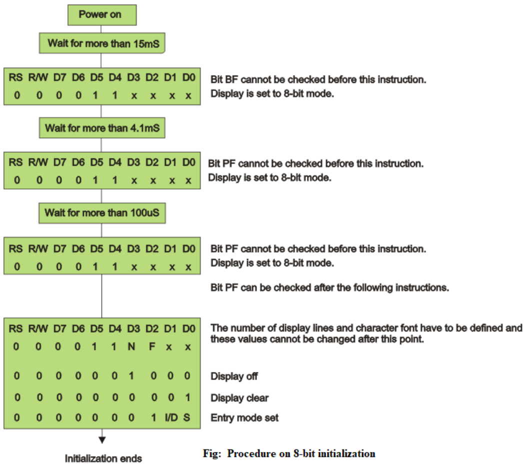
4. Character entry

ID = 1 Addresses on display are automatically incremented by 1

S = 0 Display shift off

Automatic reset is mainly performed without any problems. Mainly but not always! If for any reason power supply voltage does not reach full value in the course of 10mS, display will start perform completely unpredictably. If voltage supply unit can not meet this condition or if it is needed to provide completely safe operating, the process of initialization by which a new reset enabling display to operate normally must be applied.

Algorithm according to the initialization is being performed depends on whether connection to the microcontroller is through 4- or 8-bit interface. All left over to be done after that is to give basic commands and of course- to display messages.



RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	1	x	x	x	x

Bit BF cannot be checked before this instruction.
Display is set to 8-bit mode.

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	1	x	x	x	x

Bit PF cannot be checked before this instruction.
Display is set to 8-bit mode.

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	1	x	x	x	x

Bit PF cannot be checked before this instruction.
Display is set to 8-bit mode.

Bit PF can be checked after the following instructions.

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	0	1	1	N	F	x	x	The number of display lines and character font have to be defined and these values cannot be changed after this point.
0	0	0	0	0	0	1	0	0	0	Display off
0	0	0	0	0	0	0	0	0	1	Display clear
0	0	0	0	0	0	0	1	I/D	S	Entry mode set

Fig: Procedure on 8-bit initialization

ESP8266 WiFi:

Description: The ESP8266 WiFi Module is a self contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your WiFi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266 module comes pre-programmed with an AT command set firmware, meaning, you can simply hook this up to your Controller and get about as much WiFi-ability as a WiFi Shield.

This module has a powerful enough on-board processing and storage capability that allows it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, including the front-end module, is designed to occupy minimal PCB area. The ESP8266 supports APSD for VoIP applications and Bluetooth co-existance interfaces, it contains a self-calibrated RF allowing it to work under all operating conditions, and requires no external RF parts.

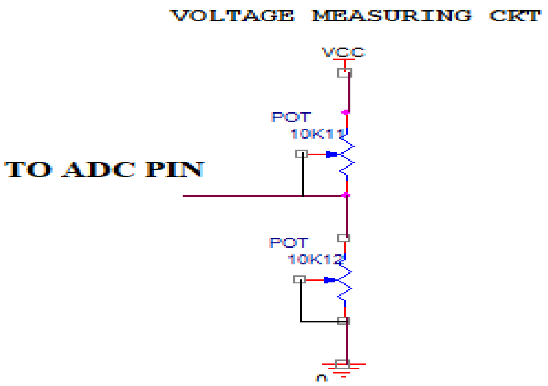
The ESP8266 Module is not capable of 5-3V logic shifting and will require an external Logic Level Converter. Please do not power it directly from your 5V dev board.
This new version of the ESP8266 WiFi Module has increased the flash disk size from 512k to 1MB.

Features:

- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLLs, regulators, DCXO and power management units
- +19.5dBm output power in 802.11b mode
- Power down leakage current of <10uA
- 1MB Flash Memory
- Integrated low power 32-bit CPU could be used as application processor
- SDIO 1.1 / 2.0, SPI, UART
- STBC, 1×1 MIMO, 2×1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4ms guard interval
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)

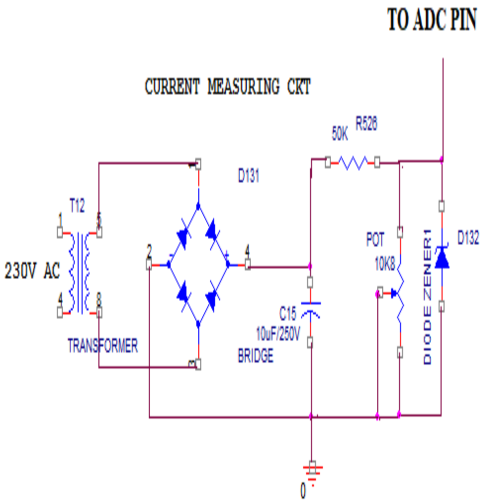
Commands	Description	Type	Set/Execute	Inquiry	test	Parameters and Examples
AT+RST	restart the module	basic	-	-	-	-
AT+GMR	check firmware version	basic	-	-	-	-
AT+CWQAP	quit the AP	wifi	AT+CWQAP	-	AT+CWQAP=?	
AT+CWMODE	wifi mode	wifi	AT+CWMODE=<mode>	AT+CWMODE?	AT+CWMODE=?	1= Sta, 2= AP, 3=both, Sta is the default mode of router, AP is a normal mode for devices
AT+CWLIF	check join devices' IP	wifi	AT+CWLIF	-	-	
AT+CWLAP	list the AP	wifi	AT+CWLAP			
AT+CWJAP	join the AP	wifi	AT+ CWJAP =<ssid>,<pwd >	AT+ CWJAP?	-	ssid = ssid, pwd = wifi password
AT+CIPSTART	set up TCP or UDP connection	TCP/IP	1)single connection (+CIPMUX=0) AT+CIPSTART=<type>,<addr>,<port>; 2) multiple connection (+CIPMUX=1) AT+CIPSTART=<id><type>,<addr>, <port>	-	AT+CIPSTART=?	id = 0-4, type = TCP/UDP, addr = IP address, port= port; eg. Connect to another TCP server, set multiple connection first: AT+CIPMUX=1; connect: AT+CIPSTART=4,"TCP"," X1.X2.X3.X4",9999
AT+CIPSEND	send data	TCP/IP	1)single connection(+CIPMUX=0) AT+CIPSEND=<length>; 2) multiple connection (+CIPMUX=1) AT+CIPSEND=<id>,<length>		AT+CIPSEND=?	eg. send data: AT+CIPSEND=4,15 and then enter the data.
AT+CIPMODE	set data transmission mode	TCP/IP	AT+CIPMODE=<mode>	AT+CIPSEND?		0 not data mode, 1 data mode; return "Link is builded"
AT+CIPCLOSE	close TCP or UDP connection	TCP/IP	AT+CIPCLOSE =<id> or AT+CIPCLOSE		AT+CIPCLOSE=?	
AT+CIFSR	Get IP address	TCP/IP	AT+CIFSR		AT+ CIFSR=?	
AT+ CWSAP	set the parameters of AP	wifi	AT+ CWSAP=<ssid>,<pwd>,<chl>, <ecn>	AT+ CWSAP?		ssid, pwd, chl = channel, ecn = encryption; eg. Connect to your router: AT+CWJAP="www.electrodragon.com","helloworld"; and check if connected: AT+CWJAP?
AT+ CIPSTO	Set the server timeout	AT+CIPSTO=<time>	AT+CIPSTO?		<time>0~28800 in second	
AT+ CIPSTATUS	get the connection status	TCP/IP	AT+ CIPSTATUS			<id>,<type>,<addr>,<port>,<tetype>= client or server mode
AT+ CIPSERVER	set as server	TCP/IP	AT+ CIPSERVER=<mode>[,<port>]			mode 0 to close server mode, mode 1 to open; port = port; eg. turn on as a TCP server: AT+CIPSERVER=1,8888, check the self server IP address: AT+CIFSR=?
AT+ CIPMUX	set mutiple connection	TCP/IP	AT+ CIPMUX=<mode>	AT+ CIPMUX?		0 for single connection 1 for multiple connection
AT	general test	basic	-	-	-	-
+IPD	received data					For Single Connection mode(CIPMUX=0): + IPD, <len>: For Multi Connection mode(CIPMUX=1): + IPD, <id>, <len>: <data>

Voltage measuring circuit:



The supply to the voltage measuring circuit is the Adaptor. Thus, the voltage coming from the adaptor is applied to two potentiometers and the middle point of the two potentiometers is given to ADC pin of Arduino. Thus the voltage value will be displayed on LCD.

Current measuring circuit:

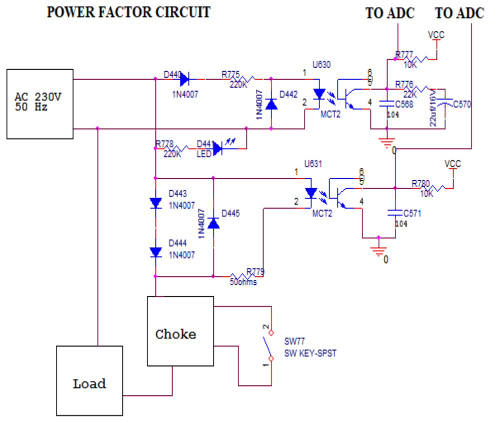


At the secondary winding of the transformer, the bridge rectifier is used that performs the rectification of ac to dc. The capacitor acts as high pass filter which blocks dc and allows ac. Thus, the ac signal is passed through the capacitor leaving the dc. Thus, the pure dc is collected from the output of the capacitor filter. That dc value is applied to the pot in parallel with zener diode and this is given to the ADC pin of Arduino. Thus, the analog value of the current is converted to dc and displayed on LCD.

POWER FACTOR MEASURE:

The power factor is the cosine of the angle between the voltage and current. The power factor is the ratio of the real power used in the circuit to the apparent power in the circuit. When a load is completely resistive, the real power will be equal to the apparent power and the power factor will be 1. When reactance is present in the circuit, some of the energy will be stored in the circuit and then transferred back to the power source. This inefficiency increases the currents in the system which means that the apparent power in the circuit must be greater in order to consume the same amount of real power.

A completely reactive circuit will consume no real power and the subsequent power factor will be 0. A low power factor results in greater distribution losses in the power system and should be avoided. For linear circuits, the power factor is also known as the displacement power factor.



Power Factor Values

For the purely resistive circuit, the power factor is 1 (perfect), because the reactive power equals zero. Here, the power triangle would look like a horizontal line, because the opposite (reactive power) side would have zero length.

For the purely inductive circuit, the power factor is zero, because true power equals zero. Here, the power triangle would look like a vertical line, because the adjacent (true power) side would have zero length.

The same could be said for a purely capacitive circuit. If there are no dissipative (resistive) components in the circuit, then the true power must be equal to zero, making any power in the circuit purely reactive. The power triangle for a purely capacitive circuit would again be a vertical line (pointing down instead of up as it was for the purely inductive circuit).

1.1.1. Importance of Power Factor

Power factor can be an important aspect to consider in an AC circuit because of any power factor less than 1 means that the circuit's wiring has to carry more current than what would be necessary with zero reactance in the circuit to deliver the same amount of (true) power to the resistive load. The poor power factor makes for an inefficient power delivery system.

1.1.2. Poor Power Factor

Poor power factor can be corrected, paradoxically, by adding another load to the circuit drawing an equal and opposite amount of reactive power, to cancel out the effects of the load's inductive reactance. Inductive reactance can only be cancelled by capacitive reactance, so we have to add a capacitor in parallel to our example circuit as the additional load. The effect of these two opposing reactance in parallel is to bring the circuit's total impedance equal to its total resistance (to make the impedance phase angle equal, or at least closer, to zero).

Since we know that the (uncorrected) reactive power is 119.998 VAR (inductive), we need to calculate the correct capacitor size to produce the same quantity of (capacitive) reactive power. Since this capacitor will be directly in parallel with the source (of known voltage), we'll use the power formula which starts from voltage and reactance:

$$Q = \frac{E^2}{X} \qquad X_C = \frac{1}{2\pi fC}$$

FIRMWARE IMPLEMENTATION OF THE PROJECT DESIGN

FIRMWARE IMPLEMENTATION

This chapter briefly explains about the firmware implementation of the project. The required software tools are discussed in the following sections.

Software Tool Required

Arduino 1.0.6 software tools used to program microcontroller. The working of software tool is explained below in detail.

PROGRAMMING MICROCONTROLLER

A compiler for a high level language helps to reduce production time. To program the Arduino UNO microcontroller the Arduino is used. The programming is done strictly in the embedded C language. Arduino is a suite of executable, open source software development tools for the microcontrollers hosted on the Windows platform.

Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

One of the difficulties of programming microcontrollers is the limited amount of resources the programmer has to deal with. In personal computers resources such as RAM and processing speed are basically limitless when compared to microcontrollers. In contrast, the code on microcontrollers should be as low on resources as possible

ABOUT ARDUINO COMPILER

GET AN ARDUINO BOARD AND USB CABLE

You also need a standard USB cable (A plug to B plug): the kind you would connect to a USB printer, for example. (For the Arduino Nano, you'll need an A to Mini-B cable instead.)

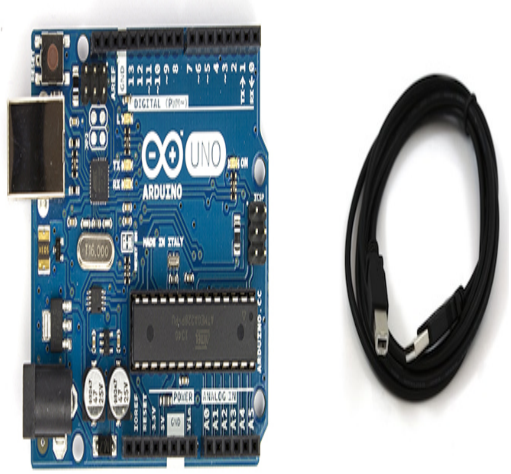


FIG: ARDUINO BOARD AND USB CABLE

CONNECT THE BOARD

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either the USB connection to the computer or an external power supply. If you're using an Arduino Diecimila, you'll need to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it's on the two pins closest to the USB port. Connect the Arduino board to your computer using the USB cable. The green power LED (labelled PWR) should go on.

ARDUINO
AN OPEN PROJECT WRITTEN, DEBUGGED AND SUPPORTED BY MASSIMO BANZI, DAVID CUARTIELLES, TOM IGOE, GIANLUCA MARTINO AND DAVID MELLIS
BASED ON PROCESSING BY CASEY REAS AND BEN FRY

FIG: OPENING THE ARDUINO WINDOW

Open the blink example

Open the LED blink example sketch: File > Examples > 1.Basics > Blink.

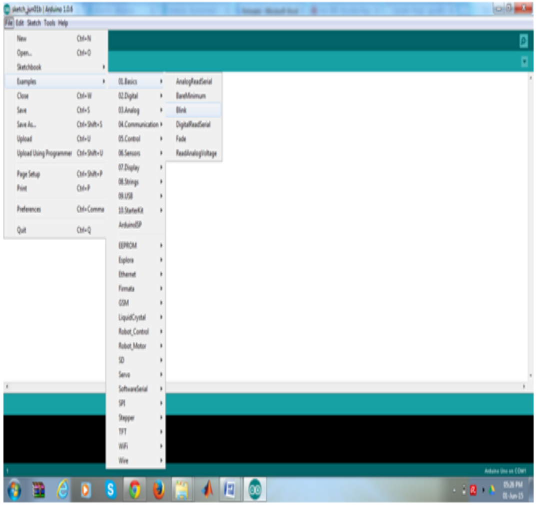
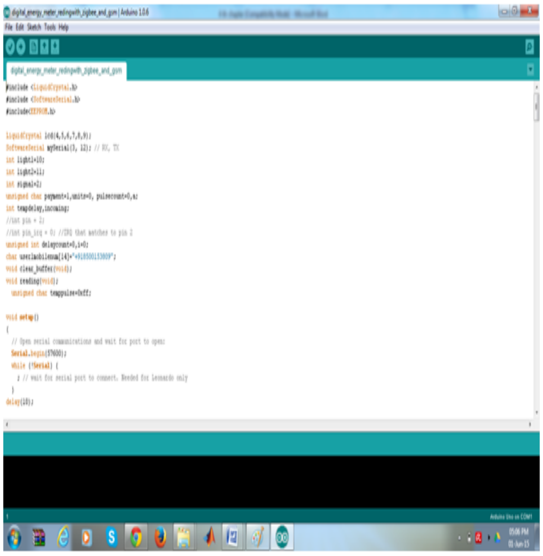


FIG: OPENING BLINK EXAMPLE



digital_energy_meter_redingwith_zigbee_and_gsm | Arduino 1.0.6

```
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
#include<EEPROM.h>

LiquidCrystal lcd(4,5,6,7,8,9);
SoftwareSerial mySerial(3, 12); // RX, TX
int light1=10;
int light2=11;
int signal=2;
unsigned char payment=1,units=0, pulsecount=0,a;
int tempdelay,incoming;
//int pin = 2;
//int pin_irq = 0; //IRQ that matches to pin 2
unsigned int delaycount=0,i=0;
char userlmobilenum[14]="+918500153809";
void clear_buffer(void);
void reading(void);
  unsigned char temppulse=0xff;

void setup()
{
  // Open serial communications and wait for port to open:
  Serial.begin(57600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }
delay(10);
```

FIG: SOURCE CODE WRITTEN IN ARDUINO COMPILER

Select your board

You'll need to select the entry in the Tools > Board menu that corresponds to your Arduino.

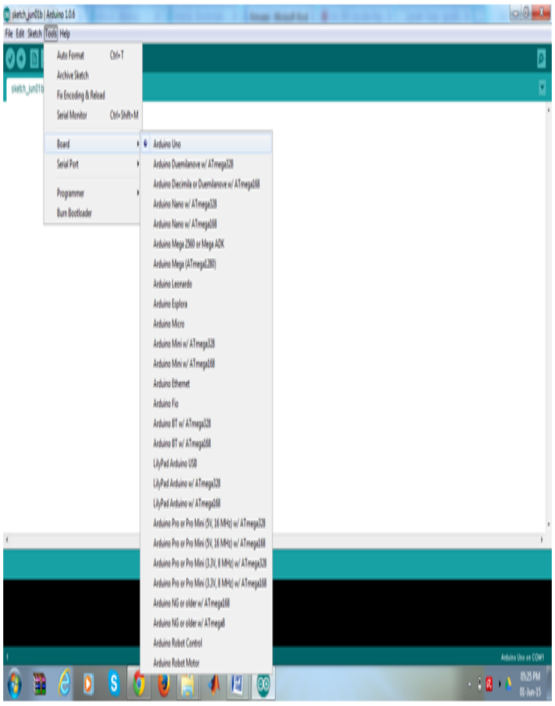
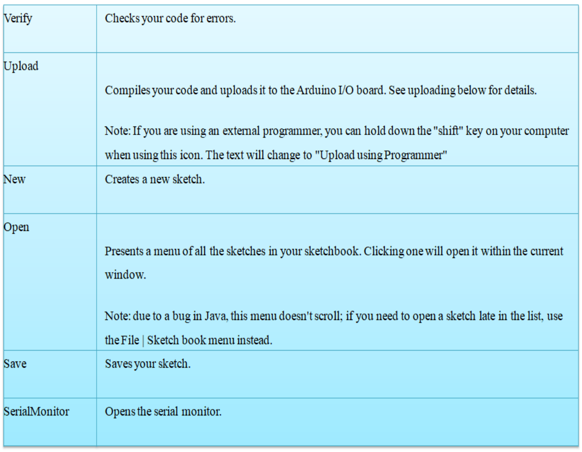

sketch_jun01b | Arduino 1.0.6
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Board
Serial Port
Programmer
Burn Bootloader
Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168
Arduino Nano w/ ATmega328
Arduino Nano w/ ATmega168
Arduino Mega 2560 or Mega ADK
Arduino Mega (ATmega1280)
Arduino Leonardo
Arduino Esplora
Arduino Micro
Arduino Mini w/ ATmega328
Arduino Mini w/ ATmega168
Arduino Ethernet
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino USB
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8
Arduino Robot Control
Arduino Robot Motor
sketch_jun01b

FIG: SELECTING AN ARDUINO UNO

WRITING SKETCHES

Software written using Arduino are called sketches. These sketches are written in the text editor. Sketches are saved with the file extension .ino. It has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino environment including complete error messages and other information. The bottom righthand corner of the window displays the current board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches and open the serial monitor.

NB: Versions of the IDE prior to 1.0 saved sketches with the extension .pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the .ino extension on save.



Verify	Checks your code for errors.
Upload	Compiles your code and uploads it to the Arduino I/O board. See uploading below for details. Note: If you are using an external programmer, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"
New	Creates a new sketch.
Open	Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window. Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketch book menu instead.
Save	Saves your sketch.
SerialMonitor	Opens the serial monitor.

Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help. The menus are context sensitive which means only those items relevant to the work currently being carried out are available.

1.1.3. SELECT YOUR SERIAL PORT

1.1.4. Select the serial device of the Arduino board from the Tools | Serial Port menu. This is likely to be COM3 or higher (COM1and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu; the entry that disappears should be the Arduino board. Reconnect the board and select that serial port.

UPLOAD THE PROGRAM

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Serial Port menus. The boards are described below. On the Mac, the serial port is probably something like /dev/tty.usbmodem241 On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be /dev/ttyUSB0,/dev/ttyUSB1 or similar.

Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the File menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino environment will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

Now, simply click the "Upload" button in the environment. Wait a few seconds - you should see the RX and TX leds on the board flashing. If the upload is successful, the message "Done uploading." will appear in the status bar. (*Note:* If you have an Arduino Mini, NG, or other board, you'll need to physically present the reset button on the board immediately before pressing the upload button.)

Compiling sketch...

FIG: COMPILATION UNDER PROCESS

A few seconds after the upload finishes, you should see the pin 13 (L) LED on the board start to blink (in orange). If it does, congratulations! You've gotten Arduino up-and-running.

RESULTS AND DISCUSSIONS

WORKING PROCEDURE

Power quality and status measurement system demonstrates the power quality of the system. This system measures the electrical parameters like current, voltage and power factor and continuously uploads these values to the web page.

The system uses ESP8266 as the Wi-Fi module to upload the parameters on to the web page. The user can view these values in his mobile phone from any corner of the world. The current, voltage and power factor measuring circuits are designed individually and the parameter values are read by Arduino continuously. These values are also displayed on the LCD.

The power status (system on or off status), the current value (current consumed by the loads), the voltage value (the input voltage coming to the loads) and the power factor value will be read and continuously uploaded on the web page.

REFERENCES:

1. S. M. Amin and B. F. Wollenberg, “Toward a smart grid: power delivery for the 21st century,” IEEE Power and Energy Magazine, vol. 3, no. 5, pp. 34-41, 2005.
2. F. Benzi, N. Anglani, E. Bassi, and L. Frosini, “Electricity Smart Meters Interfacing the Households,” IEEE Transactions on Industrial Electronics, vol. 58, no. 10, pp. 4487-4494, 2011.
3. Zaballos, A. Vallejo, M. Majoral, and J. M. Selga, “Survey and Performance Comparison of AMR Over PLC

Standards," IEEE Transactions on Power Delivery, vol. 24, no. 2, pp. 604-613, 2009.

4.R. Rashed Mohassel, A. Fung, F. Mohammadi, and K. Raahemifar, "A survey on Advanced Metering Infrastructure," International Journal of Electrical Power & Energy Systems, vol. 63, pp. 473-484, 2014 A. M. Vega, F. Santamaria, and E. Rivas, "Modeling for home electric energy management: A review," Renewable and Sustainable Energy Reviews, vol. 52, pp. 948-959, 2015.

5.E. Taktak and I. B. Rodriguez, "Energy Consumption Adaptation Approach for Smart Buildings," in 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), 2017, pp. 1370-1377.

6.M. Beaudin and H. Zareipour, "Home energy management systems: A review of modelling and complexity," Renewable and Sustainable Energy Reviews, vol. 45, pp. 318-335, 2015.

7.K. Athira, A. R. Devidas, M. V. Ramesh, and V. P. Rangan, "User centered energy management scheme for smart buildings," in 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017, pp. 1296-1302.

8.H. G. R. Tan, C. H. Lee, and V. H. Mok, "Automatic power meter reading system using GSM network," in 2007 International Power Engineering Conference (IPEC 2007), 2007, pp. 465-469.

9.M. Wasi-ur-Rahman, M. T. Rahman, T. H. Khan, and S. M. L. Kabir, "Design of an intelligent SMS based remote metering system," in 2009 International Conference on Information and Automation, 2009, pp. 1040-1043.

10.L. Labib, M. Billah, G. M. S. M. Rana, M. N. Sadat, M. G. Kibria, and M. R. Islam, "Design and implementation of low-cost universal smart energy meter with demand side load management," IET Generation, Transmission & Distribution, vol. 11, no. 16, pp. 3938-3945, 2017.

11.H. Primicanta, N. Mohd Yunus, and M. Awan, "ZigBee-GSM based Automatic Meter Reading system," in 2010 International Conference on Intelligent and Advanced Systems, 2010, pp. 1-5.

12.M. A. Prada, P. Reguera, S. Alonso, A. Morán, J. J. Fuertes, and M. Domínguez, "Communication with resource-constrained devices through MQTT for control education," IFAC-PapersOnLine, vol. 49, no. 6, pp. 150-155, 2016