LOAD SHEDDING TIME MANAGEMENT WITH PROGRAMMABLE INTERFACE

A. HIMA BINDU , S. SHAMEEM , S. VAMSI KRISHNA , A. SAI KUMAR, N.SANTHOSH KUMAR

Department of Electrical & Electronics Engineering

Annamacharya Institute of Technology & Sciences,

Rajampet (Autonomous)

Kadapa, Andhra Pradesh,516216

Abstract:

As demand of power is increasing continuously, with already installed power generation system, we cannot replace the new generating system every time. Therefore Load Shedding is the most preferred solution to meet this problem. Since Load Shedding is monitored manually by allotment of some in charge person to turn On/Off the switch, so it is not very efficient as well as non reliable.

Therefore " Load Shedding Time Management System using microcontroller " can be designed which can automatically turn On/Off the switch as per the command fixed into it with Real time clock system operated using arduino. This project would be providing information of how a Real time clock system interfaced with microcontroller will help in shedding the load of various zones as per the command fed to it.

Introduction

Load-shedding is a strategy through which the electricity division manages the inadequacy of the electricity consumed by the society. Shedding process has been done to reduce the load of electricity usage in a society via different substations that are linked to the main power station. During the low frequency of voltage, the generator deteriorates to fabricate the recommended voltage. Under the circumstances authorization lacks to provide the expected quantity of electricity which impels the authorization to execute a shedding. To symmetry the availability and the demand of electricity the concerned authorization has to implement the load shedding process.

The load-shedding process is more vulnerable to human mistakes as a machinist has physically switched the load ON/OFF.

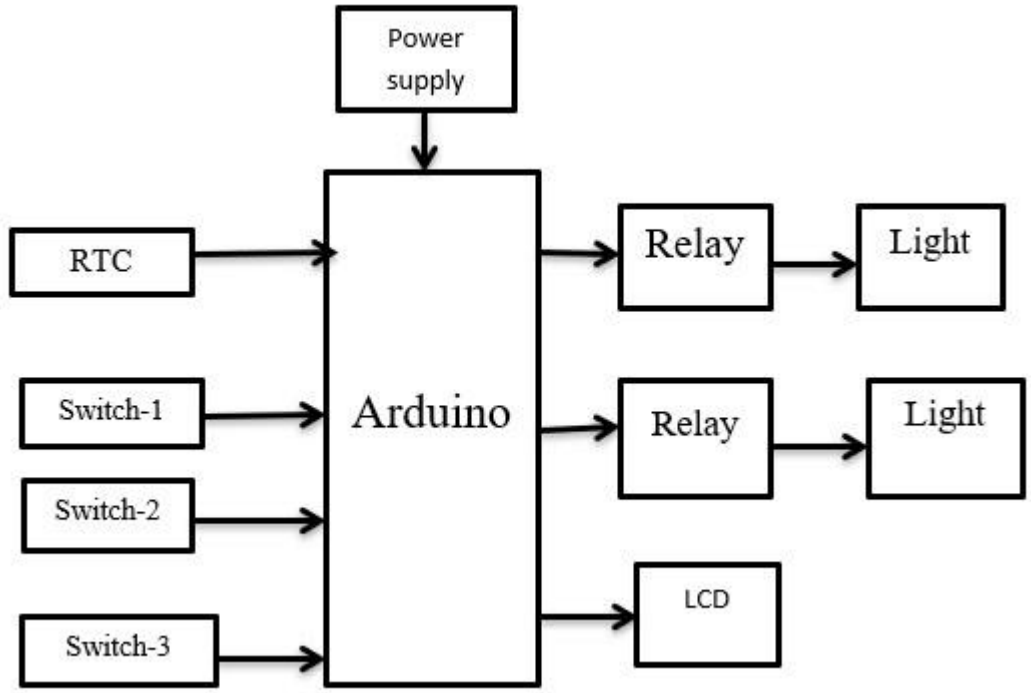
Existing system:

some man power may be employed or by using computer it can be controlled efficiently. Detaching of power is done to minimize the consumer load provided through several substations, Which are connected to the main power station.

Proposed system:

So In this project “The Programmable load shedding time management system” we are connecting two loads operating through microcontroller using relay circuits. Here 230V AC supply is rectified to 12V DC which is then converted into input circuit supply of 5V DC with the help of voltage regulator. As we know that in power system relays are used to trip the circuit at a time of any fault or disturbance. So to shed the particular load, relay receives the command from microcontroller.
Here according to time we set by using RTC the load will turn off/on.

Block diagram:



Hardware requirements

ARDUINO

The Arduino microcontroller is an easy to use yet powerful single board computer that has gained considerable traction in the hobby and professional market. The Arduino is open-source, which means hardware is reasonably priced and development software is free. This guide is for students in ME 2011, or students anywhere who are confronting the Arduino for the first time. For advanced Arduino users, prowl the web; there are lots of resources.

This is what the Arduino board looks like.

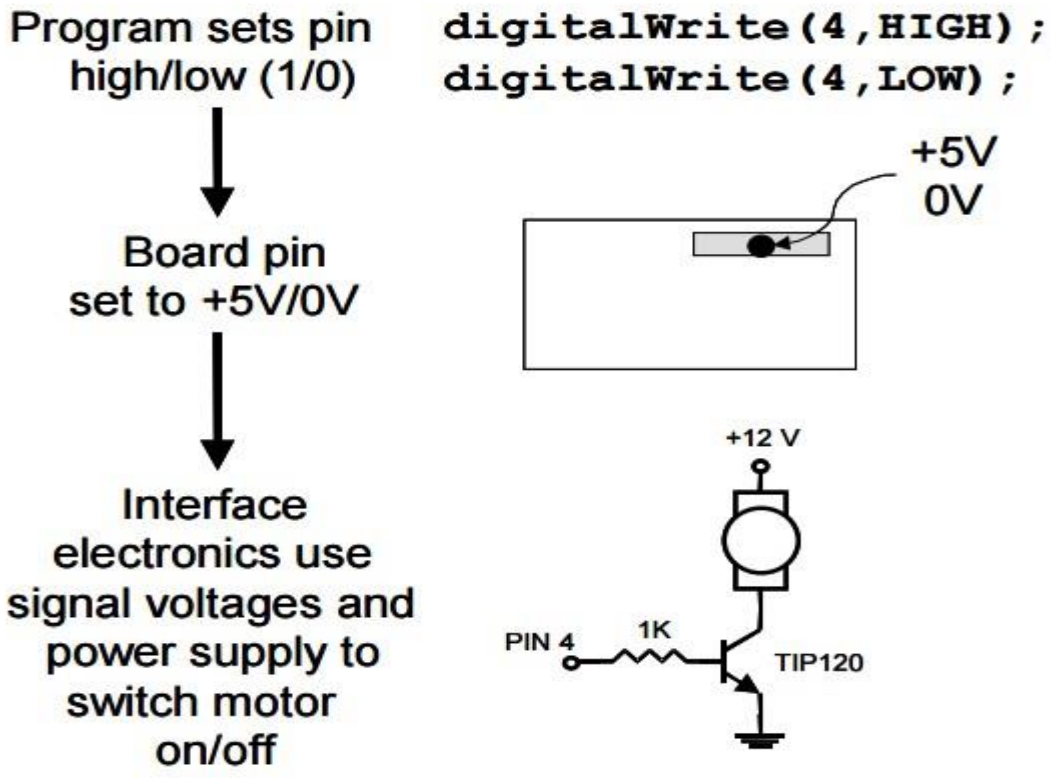
The Arduino programming language is a simplified version of C/C++. If you know C, programming the Arduino will be familiar. If you do not know C, no need to worry as only a few commands are needed to perform useful functions.

Arduino Hardware

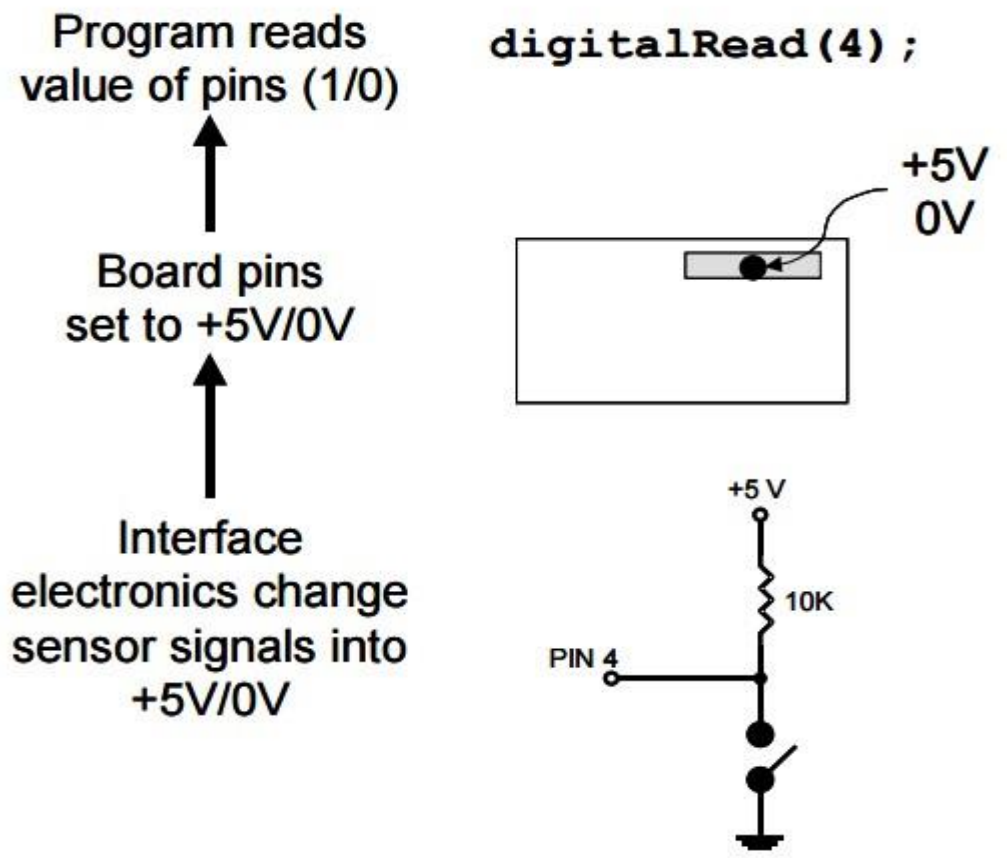
The power of the Arduino is not its ability to crunch code, but rather its ability to interact with the outside world through its input-output (I/O) pins. The Arduino has 14 digital I/O pins labeled 0 to 13 that can be used to turn motors and lights on and off and read the state of switches.

Each digital pin can sink or source about 40 mA of current. This is more than adequate for interfacing to most devices, but does mean that interface circuits are needed to control devices other than simple LED's. In other words, you cannot run a motor directly using the current available from an Arduino pin, but rather must have the pin drive an interface circuit that in turn drives the motor. A later section of this document shows how to interface to a small motor.

To interact with the outside world, the program sets digital pins to a high or low value using C code instructions, which corresponds to +5 V or 0 V at the pin. The pin is connected to external interface electronics and then to the device being switched on and off. The sequence of events is shown in this figure.



To determine the state of switches and other sensors, the Arduino is able to read the voltage value applied to its pins as a binary number. The interface circuitry translates the sensor signal into a 0 or +5 V signal applied to the digital I/O pin. Through a program command, the Ardiomp interrogates the state of the pin. If the pin is at 0 V, the program will read it as a 0 or LOW. If it is at +5 V, the program will read it as a 1 or HIGH. If more than +5 V is applied, you may blow out your board, so be careful. The sequence of events to read a pin is shown in this figure.



Interacting with the world has two sides. First, the designer must create electronic interface circuits that allow motors and other devices to be controlled by a low (1-10 mA) current signal that switches between 0 and 5 V, and other circuits that convert sensor readings into a switched 0 or 5 V signal. Second, the designer must write a program using the set of Arduino commands that set and read the I/O pins. Examples of both can be found in the Arduino resources section of the ME2011 web site.

Atmega328p features:

- High Performance, Low Power AVR® 8-Bit Microcontroller
- 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 20 MHz
- On-chip 2-cycle Multiplier

- High Endurance Non-volatile Memory Segments
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash progam memory (ATmega48PA/88PA/168PA/328P)
  - 256/512/512/1K Bytes EEPROM (ATmega48PA/88PA/168PA/328P)
  - 512/1K/1K/2K Bytes Internal SRAM (ATmega48PA/88PA/168PA/328P)
  - 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 Onchip Boot Program True Read-While-Write Operation
  - Programming Lock for Software Security
- 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
  - Programmable Serial USART

– Master/Slave SPI Serial Interface

Byte-oriented 2-wire Serial Interface (Philips I2 C compatible)

Programmable Watchdog Timer with Separate On-chip Oscillator

– 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 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 for ATmega48PA/88PA/168PA/328P

- Temperature Range:

  – -40°C to 85°C

- Speed Grade:

  – 0 - 20 MHz @ 1.8 - 5.5V

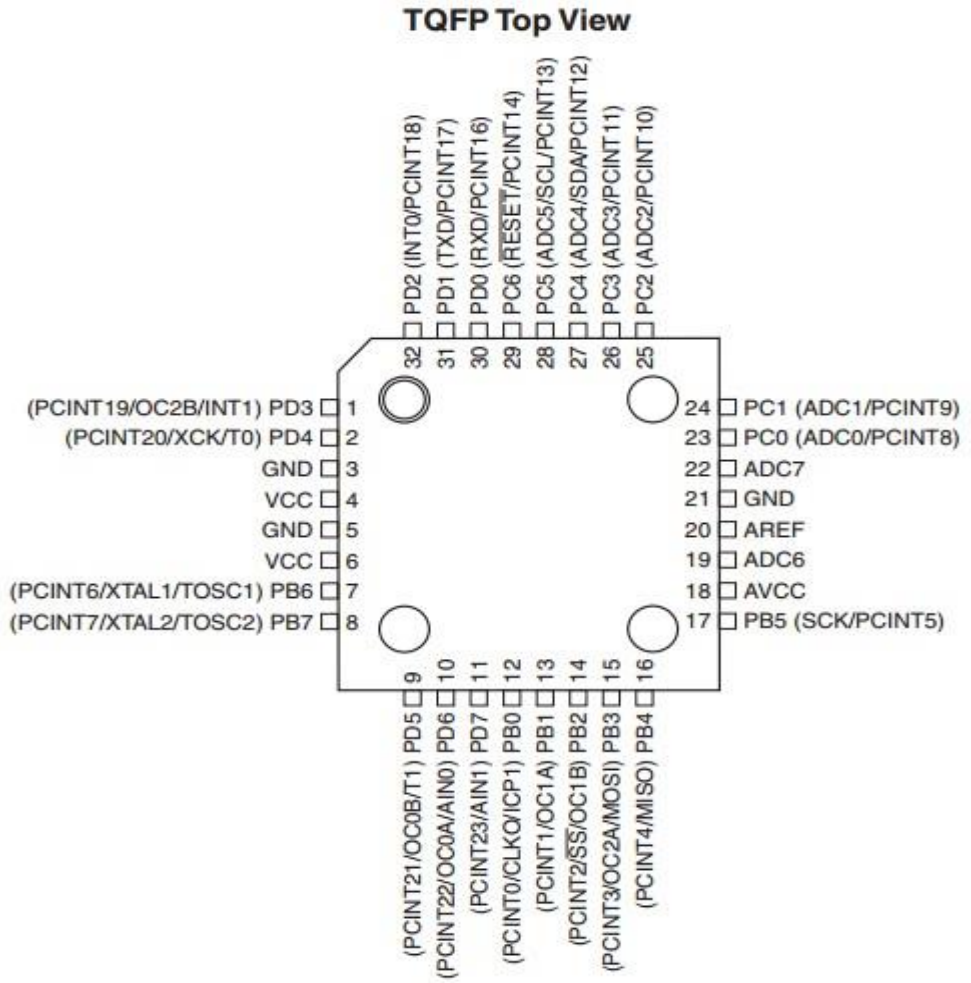
- Low Power Consumption at 1 MHz, 1.8V, 25°C for ATmega48PA/88PA/168PA/328P:

– Active Mode: 0.2 mA

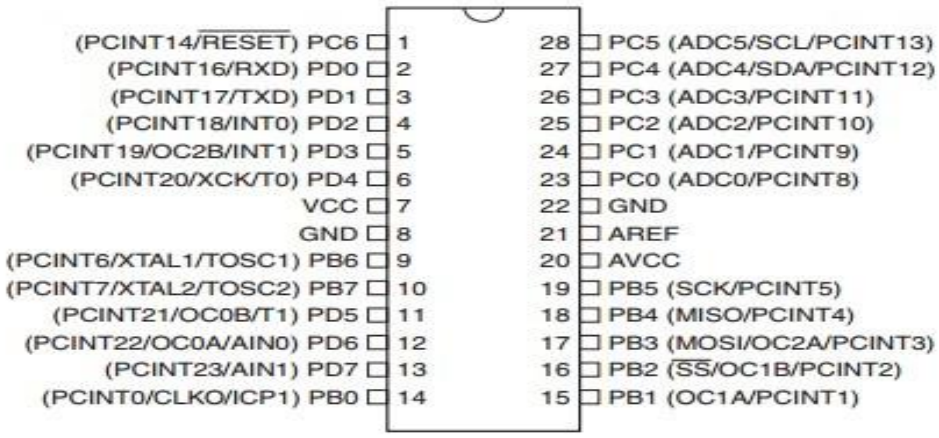
Power-down Mode: 0.1 µA

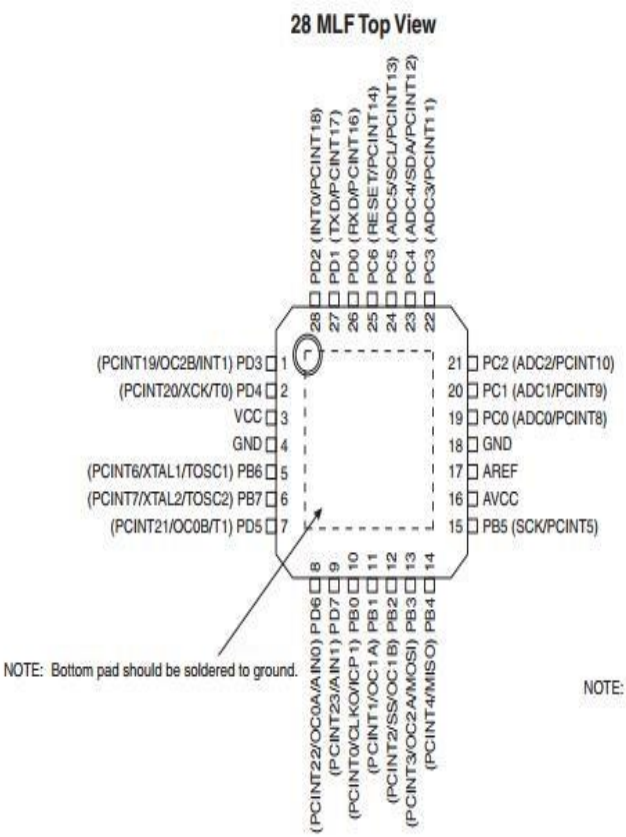
Power-save Mode: 0.75 µA (Including 32 kHz RTC)

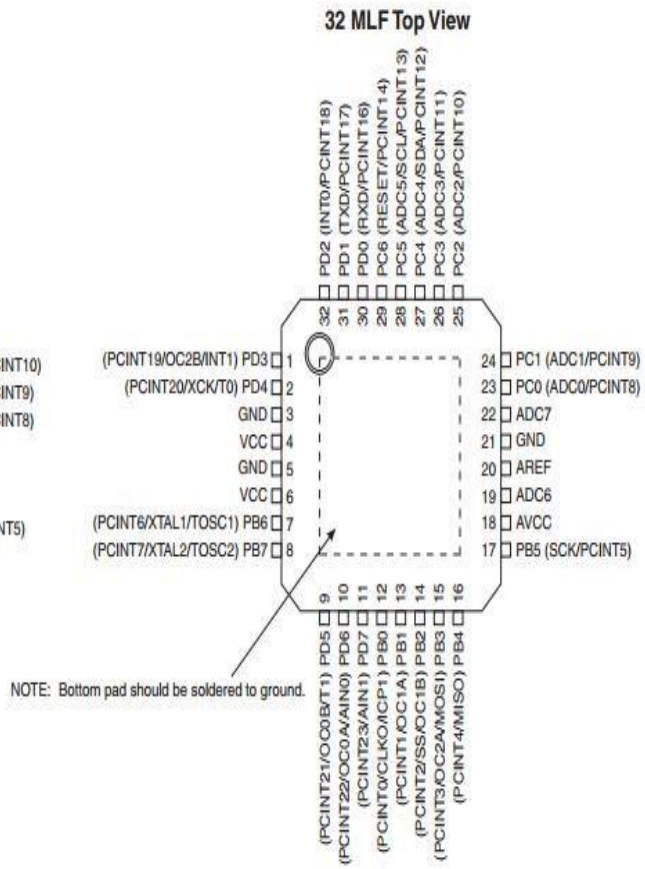
PIN CONFIGURATION



PDIP


(PCINT14/RESET) PC6 1
(PCINT16/RXD) PD0 2
(PCINT17/TXD) PD1 3
(PCINT18/INT0) PD2 4
(PCINT19/OC2B/INT1) PD3 5
(PCINT20/XCK/T0) PD4 6
VCC 7
GND 8
(PCINT6/XTAL1/TOSC1) PB6 9
(PCINT7/XTAL2/TOSC2) PB7 10
(PCINT21/OC0B/T1) PD5 11
(PCINT22/OC0A/AIN0) PD6 12
(PCINT23/AIN1) PD7 13
(PCINT0/CLKO/ICP1) PB0 14
28 PC5 (ADC5/SCL/PCINT13)
27 PC4 (ADC4/SDA/PCINT12)
26 PC3 (ADC3/PCINT11)
25 PC2 (ADC2/PCINT10)
24 PC1 (ADC1/PCINT9)
23 PC0 (ADC0/PCINT8)
22 GND
21 AREF
20 AVCC
19 PB5 (SCK/PCINT5)
18 PB4 (MISO/PCINT4)
17 PB3 (MOSI/OC2A/PCINT3)
16 PB2 (SS/OC1B/PCINT2)
15 PB1 (OC1A/PCINT1)


28 MLF Top View
(PCINT19/OC2B/INT1) PD3 1
(PCINT20/XCK/T0) PD4 2
VCC 3
GND 4
(PCINT6/XTAL1/TOSC1) PB6 5
(PCINT7/XTAL2/TOSC2) PB7 6
(PCINT21/OC0B/T1) PD5 7
21 PC2 (ADC2/PCINT10)
20 PC1 (ADC1/PCINT9)
19 PC0 (ADC0/PCINT8)
18 GND
17 AREF
16 AVCC
15 PB5 (SCK/PCINT5)
NOTE: Bottom pad should be soldered to ground.


32 MLF Top View
(PCINT19/OC2B/INT1) PD3 1
(PCINT20/XCK/T0) PD4 2
GND 3
VCC 4
GND 5
VCC 6
(PCINT6/XTAL1/TOSC1) PB6 7
(PCINT7/XTAL2/TOSC2) PB7 8
24 PC1 (ADC1/PCINT9)
23 PC0 (ADC0/PCINT8)
22 ADC7
21 GND
20 AREF
19 ADC6
18 AVCC
17 PB5 (SCK/PCINT5)
NOTE: Bottom pad should be soldered to ground.

Pin Descriptions

VCC: Digital supply voltage.

GND: Ground.

Port B (PB7:0) 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, PB7..6 is used as TOSC2..1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set. The various special features of Port B are elaborated in ”Alternate Functions of Port B” on page 76 and ”System Clock and Clock Options” on page 26.

Port C (PC5:0): Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5..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 unprogrammed, 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. The minimum pulse length is given in Table 28-3 on page 308. Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated in ”Alternate Functions of Port C” on page 79.

Port D (PD7:0): 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. The various special features of Port D are elaborated in ”Alternate Functions of Port D” on page 82.

AVCC: AVCC is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. 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 PC6..4 use digital supply voltage, VCC.

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

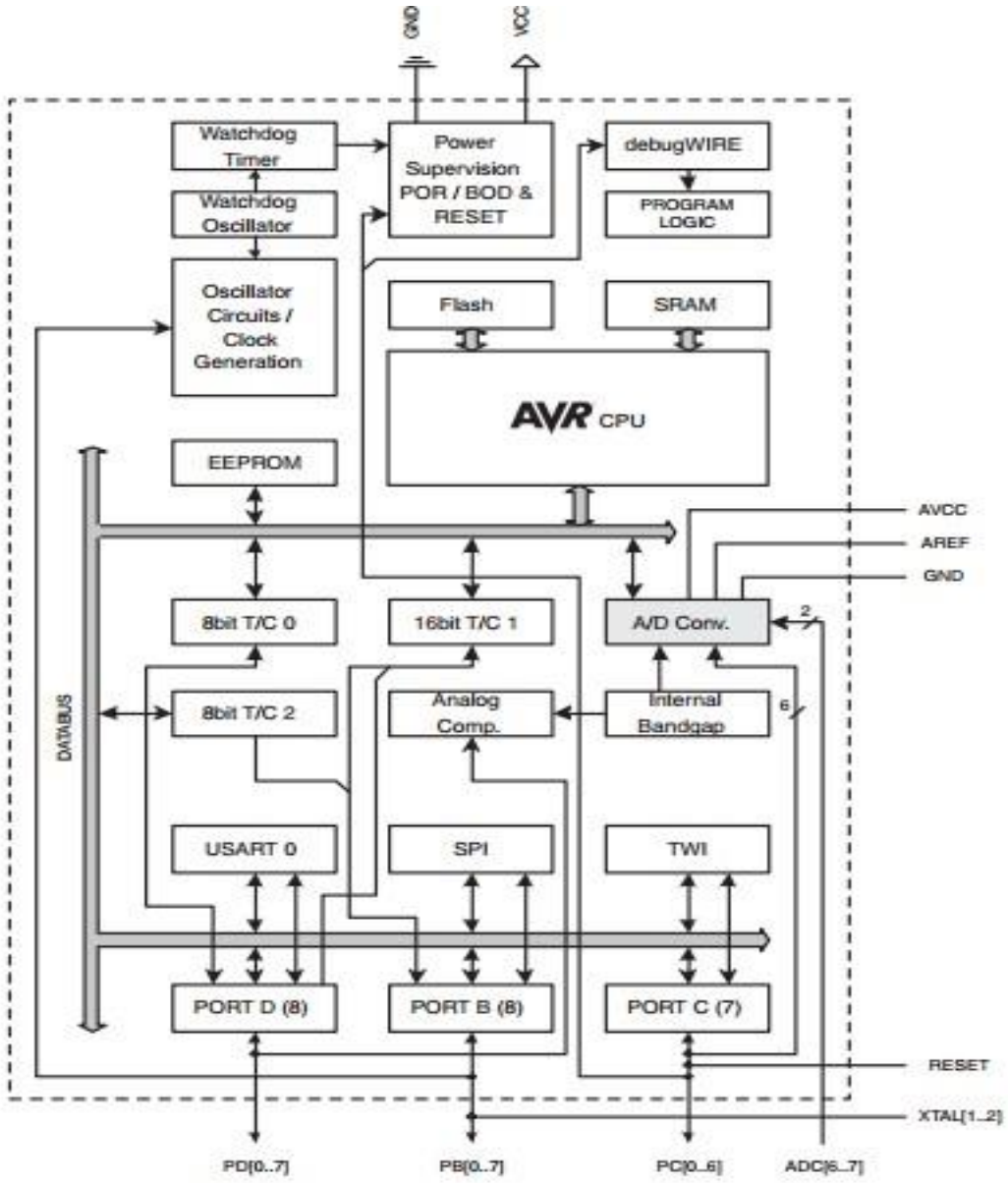
ADC7:6 (TQFP and QFN/MLF Package Only): In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

OVERVIEW

The ATmega48PA/88PA/168PA/328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48PA/88PA/168PA/328P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

The 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 one 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.

BLOCK DIAGRAM



The ATmega48PA/88PA/168PA/328P provides the following features: 4/8/16/32K bytes of InSystem Programmable Flash with Read-While-Write capabilities, 256/512/512/1K bytes EEPROM, 512/1K/1K/2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable Watchdog Timer with internal Oscillator, and five

software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

The device is manufactured using Atmel's high density non-volatile memory technology. The Onchip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile 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 ATmega48PA/88PA/168PA/328P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications. The ATmega48PA/88PA/168PA/328P AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, InCircuit Emulators, and Evaluation kits.

Comparison Between ATmega48PA, ATmega88PA, ATmega168PA and ATmega328P

The ATmega48PA, ATmega88PA, ATmega168PA and ATmega328P differ only in memory sizes, boot loader support, and interrupt vector sizes. Table 2-1 summarizes the different memory and interrupt vector sizes for the three devices.

Table 2-1. Memory Size Summary

Device	Flash	EEPROM	RAM	Interrupt Vector Size
ATmega48PA	4K Bytes	256 Bytes	512 Bytes	1 instruction word/vector
ATmega88PA	8K Bytes	512 Bytes	1K Bytes	1 instruction word/vector
ATmega168PA	16K Bytes	512 Bytes	1K Bytes	2 instruction words/vector
ATmega328P	32K Bytes	1K Bytes	2K Bytes	2 instruction words/vector

ATmega88PA, ATmega168PA and ATmega328P support a real Read-While-Write SelfProgramming mechanism. There is a separate Boot Loader Section, and the SPM instruction can only execute from there. In ATmega48PA, there is no Read-While-Write support and no separate Boot Loader Section. The SPM instruction can execute from the entire Flash.

POWER:

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an ACto-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm centerpositive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. The power pins are as follows:

- **VIN**. The input voltage to the Arduino board when it's 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 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3**. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

- **GND.** Ground pins.

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).

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 kOhms. 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. See the attachInterrupt() function for details.
- **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.

See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

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 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.

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). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by 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. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment.

This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either 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 halfsecond 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.

If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data. The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be

able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

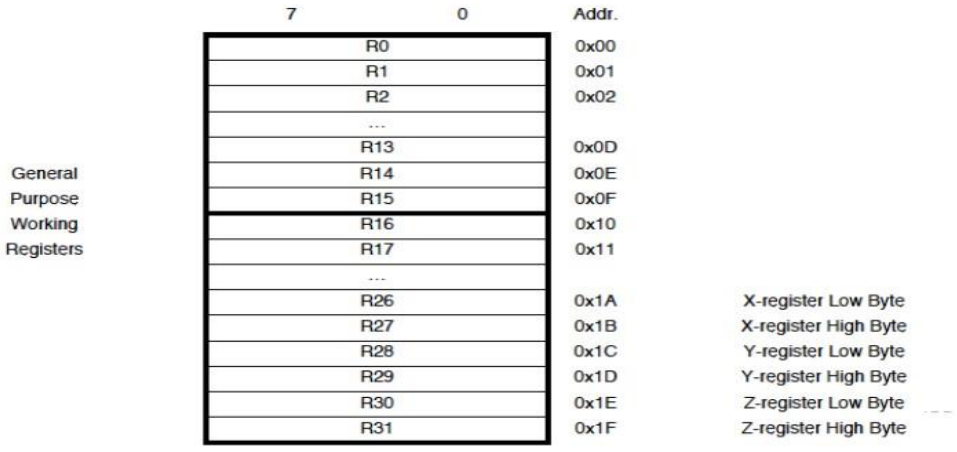
The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

Register File

- 32 8-bit GP registers
- Part of SRAM memory space

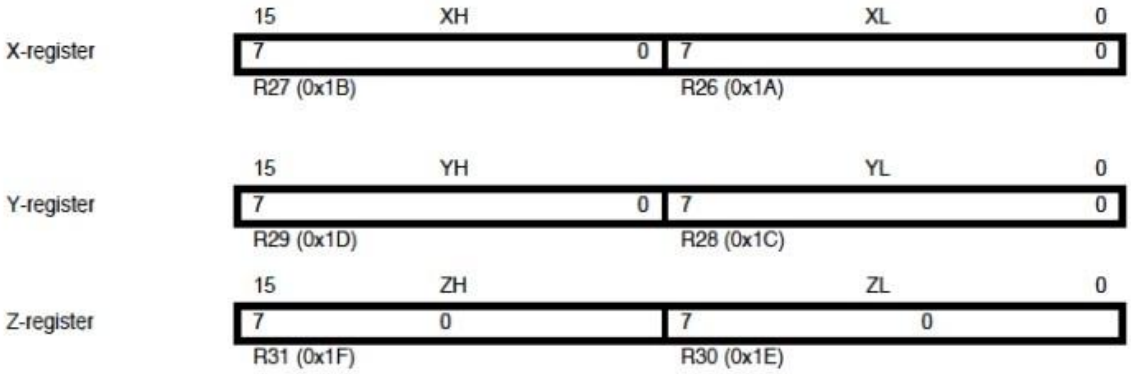


Special Addressing Registers

- X, Y and Z registers

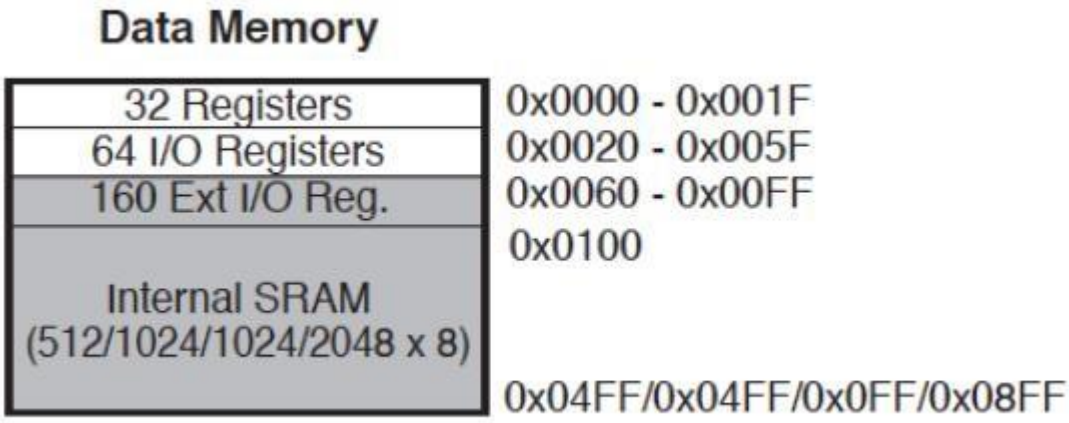
  16-bit registers made using registers 26 – 31

- Support indirect addressing



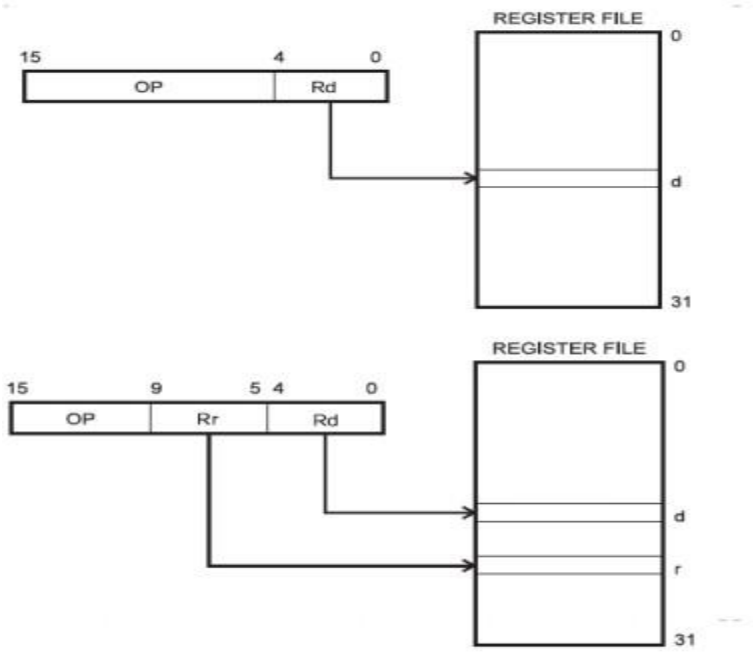
AVR Memory

- Program memory – Flash
- Data memory – SRAM

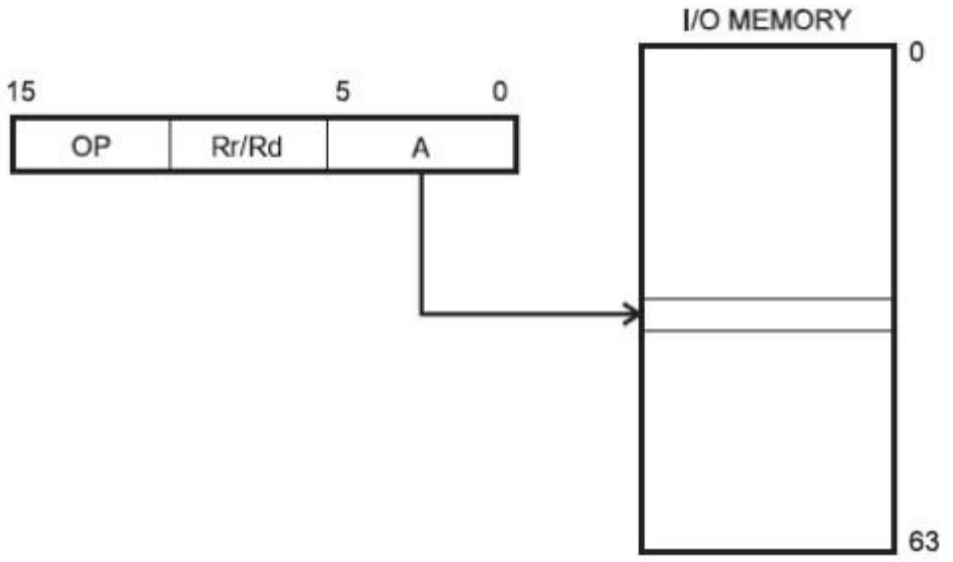


Addressing Modes

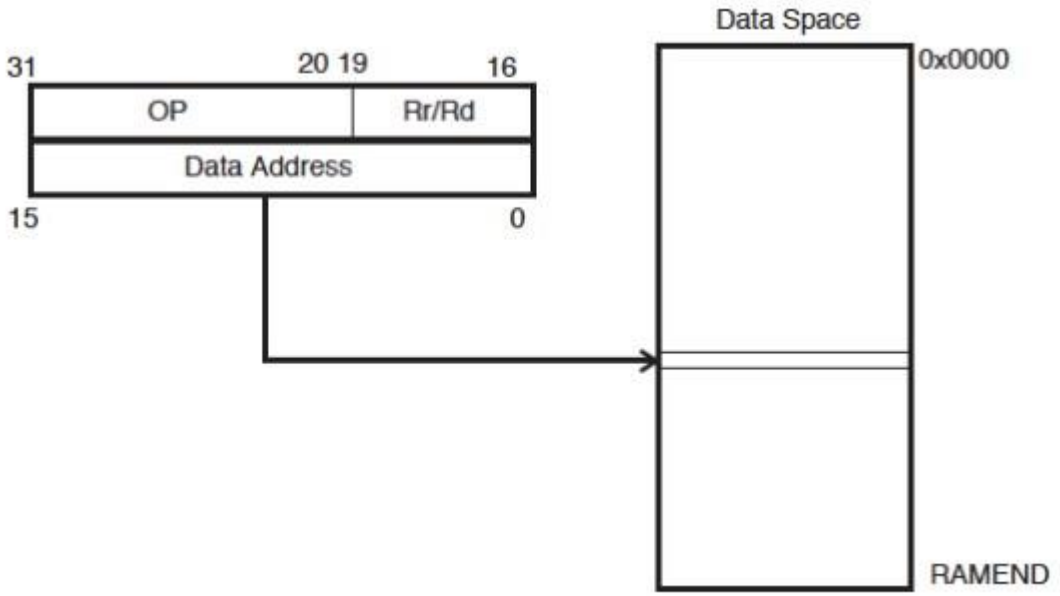
- Direct register addressing



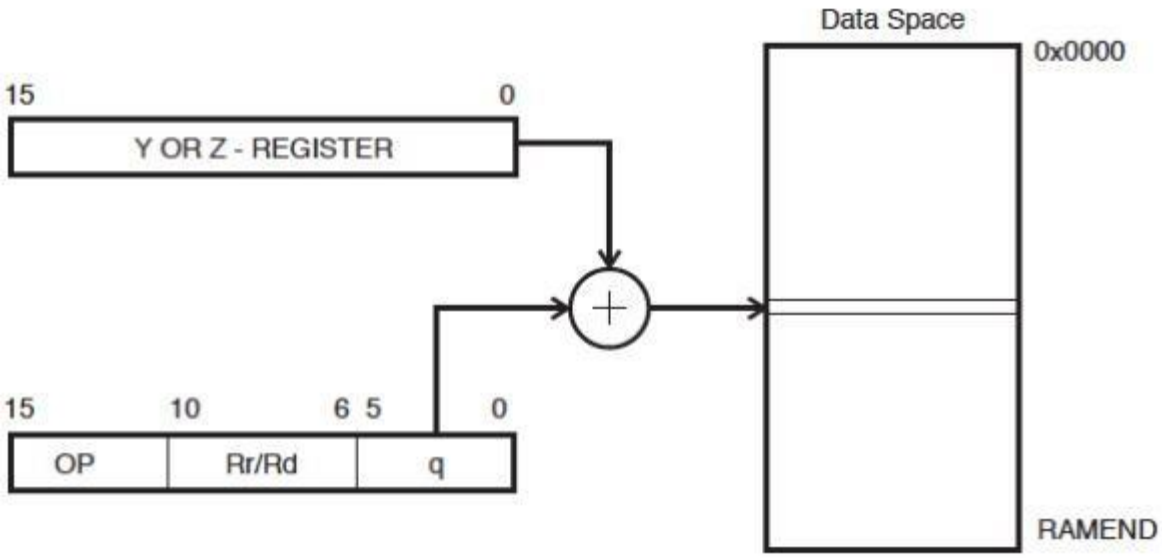
- Direct I/O addressing



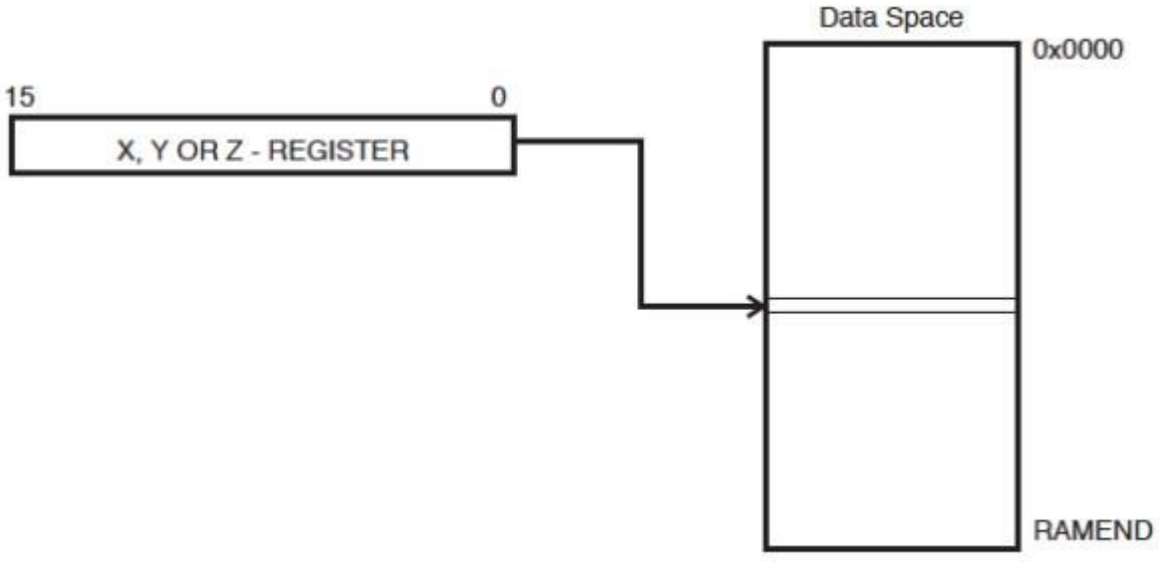
Direct data memory addressing



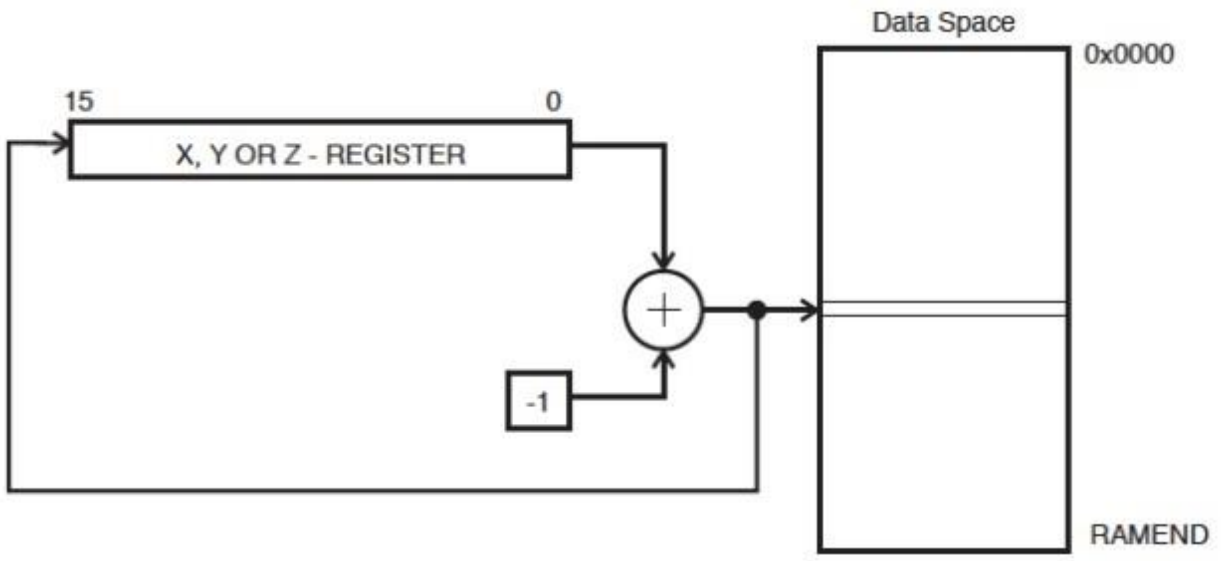
Direct data memory with displacement addressing



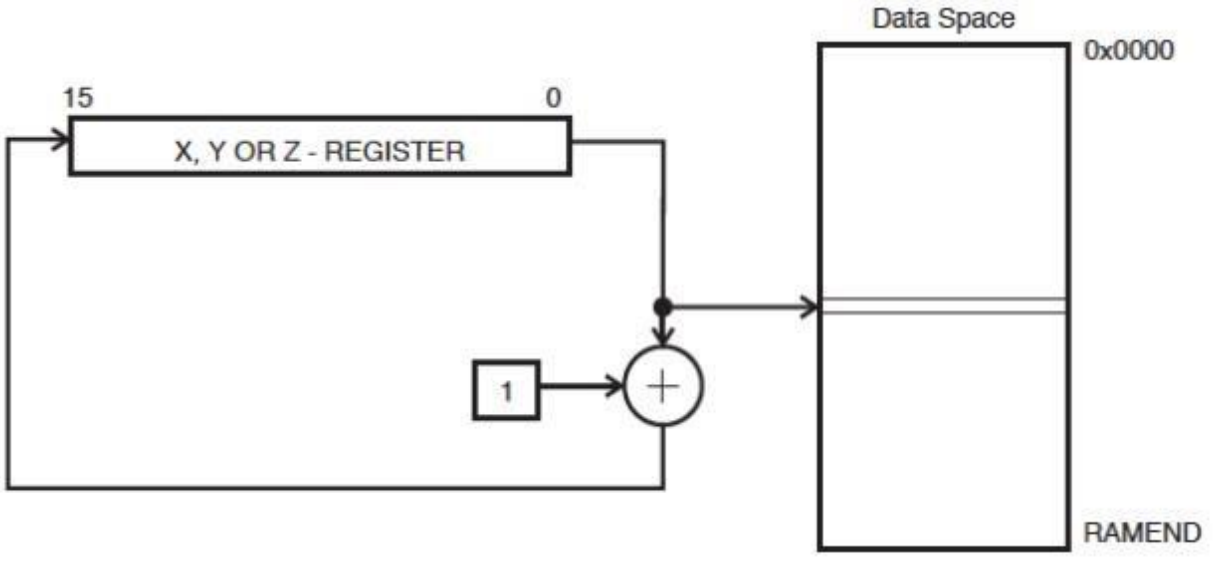
Indirect data memory addressing



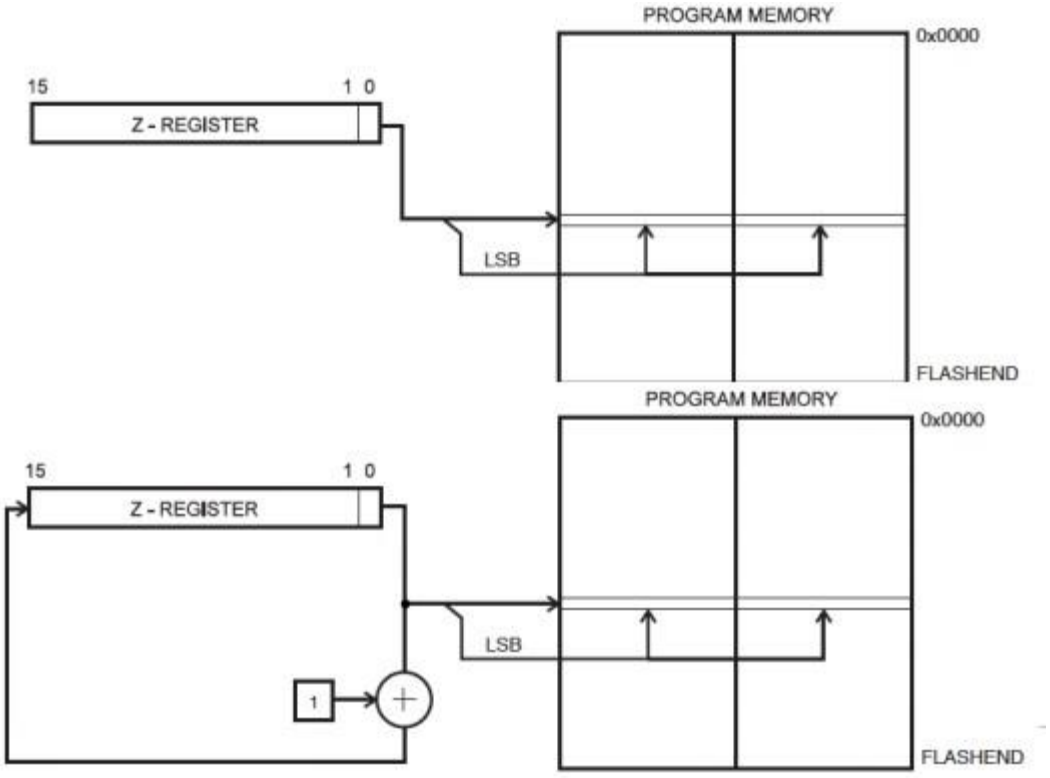
Indirect data memory addressing with pre-decrement



Indirect data memory addressing with post-increment

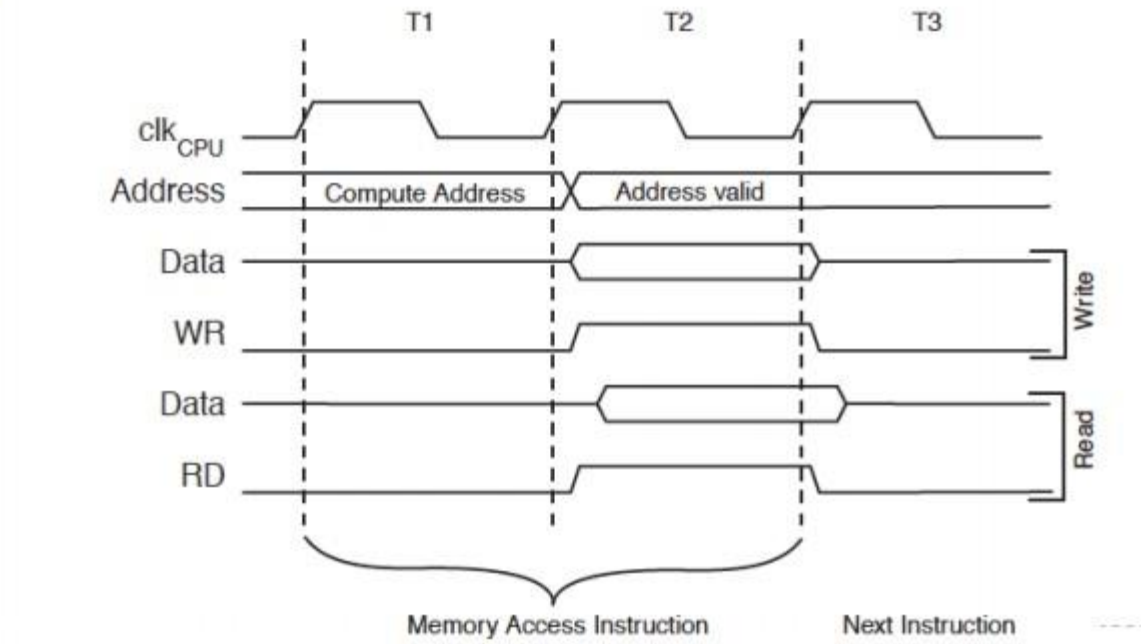


Program memory addressing (constant data)



SRAM Read/Write Timing

Figure 7-4. On-chip Data SRAM Access Cycles



Stack Pointer Register

- Special register in I/O space [3E, 3D]
  - Enough bits to address data space
  - Initialized to RAMEND (address of highest memory address)
- Instructions that use the stack pointer

Instruction	Stack pointer	Description
PUSH	Decremented by 1	Data is pushed onto the stack
CALL ICALL RCALL	Decremented by 2	Return address is pushed onto the stack with a subroutine call or interrupt
POP	Incremented by 1	Data is popped from the stack
RET RETI	Incremented by 2	Return address is popped from the stack with return from subroutine or return from interrupt

Program Status Register (PSR)

Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- Status bits set by instructions/Checked by Branch/Skip instructions
  - I – Global interrupt enable
  - T – Flag bit
  - H – Half carry (BCD arithmetic)
  - S – Sign
  - V – Overflow
  - N – Negative
  - Z – Zero
  - C – Carry

I/O Ports

- 3 8-bit Ports (B, C, D)
- Each port controlled by 3 8-bit registers
  - Each bit controls one I/O pin
  - DDRx – Direction register
    - Defines whether a pin is an input (0) or and output (1)
  - PINx – Pin input value
    - Reading this "register" returns value of pin
  - PORTx – Pin output value
    - Writing this register sets value of pin

LCD

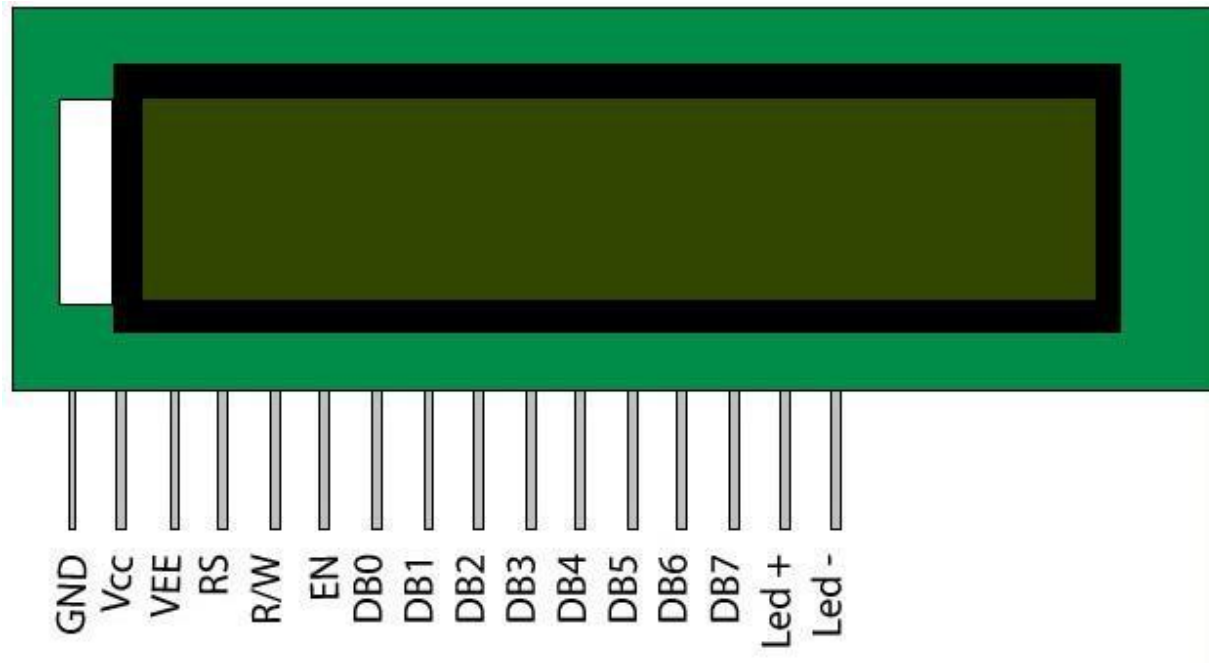
LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs.

The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on.

A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD. Click to learn more about internal structure of a LCD.

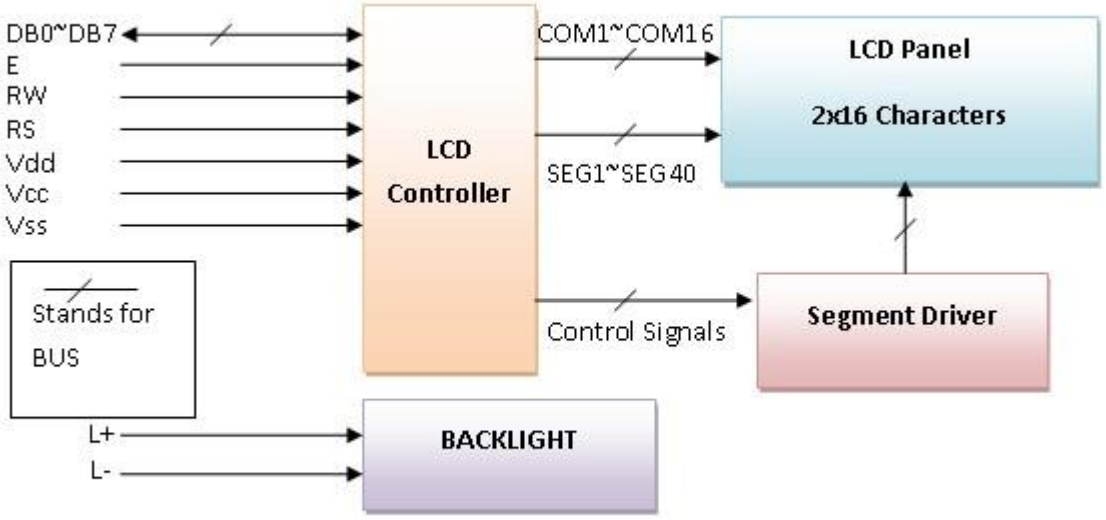
Pin Diagram:



Pin Description:

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7		DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

Block Diagram of LCD Display:-



Control and display commands

	Instruction Code		

Instruction	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	**Instruction Code Description**	**Execution time**
Read Data From RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM	1.53-1.64ms
Write data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM)	1.53-1.64ms
Busy flag & Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Busy flag (BF: 1→ LCD Busy) and contents of address counter in bits AC6AC0.	39 µs

Set DDRAM Address	0	0	1	AC 6	AC 5	AC 4	AC 3	AC 2	AC 1	AC 0	Set DDRAM address in address counter.	39 µs
Set CGRAM Address	0	0	0	1	AC 5	AC 4	AC 3	AC 2	AC 1	AC 0	Set CGRAM Address in address counter.	39 µs
Function Set	0	0	0	0	1	DL	N	F	X	X	Set interface data length (DL: 4bit/8bit), Numbers of display line (N: 1-line/2-line) display font type (F:0→ 5×8 dots, F:1→ 5×11 dots)	39 µs
Cursor or Display Shift	0	0	0	0	0	1	S/ C	R/ L	X	X	Set cursor moving and display shift control bit, and the direction	39 µs
											without changing DDRAM data	
Display & Cursor On/Off	0	0	0	0	0	0	1	D	C	B	Set Display(D),Cursor(C) and cursor blink(b) on/off control	39 µs
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	SH	Assign cursor moving direction and enable shift entire display.	0µs

Return Home	0	0	0	0	0	0	0	0	1	X	Set DDRAM Address to "00H" from AC and return cursor to its original position if shifted.	43µs
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM and set DDRAM Address to "00H" from AC	43µs

AC -Address Counter

Outline

Now the instruction can be divided mainly in four kinds

1) Function set instructions
2) Address set instructions
3) Data transfer instructions with internal RAM
4) Others

Details of the Instructions 1) Read Data from RAM

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	1	D7	D6	D5	D4	D3	D2	D1	D0

Read 8bit binary data from DDRAM/CGRAM

The selection of RAM is set by the previous address set instruction. If the address set instruction of RAM is not performed before this instruction, the data that is read first is invalid, because the direction of AC is not determined. If the RAM data is read several times without RAM address set instruction before read operation, the correct RAM data from the second, but the first data would be incorrect, as there is no time to transfer RAM data. In case of DDRAM read operation, cursor shift instruction plays the same role as DDRAM address set instruction; it also transfers RAM data to the output data registers.

After read operation, the data address counter is automatically increased or decreased by 1 according to the entry mode. After CGRAM read operation, display shift may not be executed properly.

*In case of RAM write operation, AC is increased or decreased by 1 like that of the read operation. In this time AC indicates the next address position, but the previous data can only by the read instruction.

2) Write data to ram

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0	D7	D6	D5	D4	D3	D2	D1	D0

Write binary 8bit data to DDRAM/CGRAM. The selection of CGRAM or DRAM is set by the previous address set instruction; DDRAM address set, CGRAM address set. RAM set instruction can also determine the AC direction to RAM.

After write operation, the address is automatically increased or decreased by 1 according to the entry mode.

3) Read Busy Flag and Address

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0

By making this read out operation, it can be determined if the LCD is performing some internal operation or not. If Busy Flag (BF) is high, some internal operation is going inside the LCD at that particular moment. To perform further operation the data source (e.g. micro controller) must wait for the BF to go low. Here, the address counter value can also be read.

4) Set DDRAM Address

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0

Set DDRAM address to AC, this instruction makes DDRAM data available from MPU. In 1-line display mode, DDRAM address rangers from "00H" to "4FH". In 2-line display mode, DDRAM address in the first line ranges from "00H" to "27H", and DDRAM address in the 2nd line is from "40H" to "67H".

5) Set CGRAM address

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0

Set CGRAM address to AC. This instruction makes CGRAM data available from MPU.

6) Function Set

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	F	X	X

DL: Interface data length control bit

DL='1' means 8bit mode of data transfer.

DL=’0’ means 4bit mode of data transfer

When 4 bit mode is activated, the data needs to be transferred in two parts, first higher 4bits, and then lower 4 bits.

N: display line number control bit

N=’1’ will allows to characters to display in 2-lines

N=’0’ will allows to characters to display in the first line only

F: display font control bit

F=’0’ will use 5×8 dots format display mode

F=’1’ will use 5×11 dots format display mode

7) Cursor or display Shift

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	S/C	R/L	X	X

Without writing or reading the display data, shifting right/left cursor position or display.

This instruction is made to correct or search or display data. During 2-line display mode, cursor moves to the 2nd line after the 40th digit of the 1st line.

When displayed data is shifted repeatedly, each line shifts individually.

When display shift is performed, the contents of the address counter are not changed.

8) Display On/Off Control

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0

0	0	0	0	0	0	1	D	C	B

This instruction controls Display, Cursor and cursor blink.

D: Display On/Off control bit

D='1' means entire display is turned on

D='0' means entire display is turned off. But Display data remains in DDRAM.

C: cursor On/Off control bit

C='1' turns on the cursor

C='0' turns off the cursor. But I/D register retains the data

B: Cursor blink On/Off control bit

B='1' makes cursor blink periodically.

B='0' stops the cursor to blink and cursor looks steady if the Cursor is turned on.

9) Entry Mode Set

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	SH

This instruction sets the moving direction of cursor and display.

When I/D= '1' cursor moves to the right and DDRAM address is increased by 1.

When I/D= '0' cursor moves to the left and DDRAM address is decreased by 1. CGRAM operates in the same way in this setting.

10) Return Home

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	1	X

This instruction sets the address counter to '00H', and returns the cursor to the first column of first line. And if display is shifted previously, this instruction shifts this too. The DDRAM contents don't change in this instruction.

11) Clear display

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

Clear all the display data by writing "20H" (ASCII code of 'space' character) to all DDRAM address, AND set value DDRAM address counter (AC) to "00H". It returns the cursor to the first column of first line and sets the entry mode to increment mode (I/D='1').

8-bit and 4-bit interfacing of LCD

Now the question is how to display data in the LCD or give command to it. There is two modes of data transfer are supported by LCD displays. One is 4bit mode, another is 8 bit mode. To transfer data In 8 bit mode, first put your data in the 8bit bus, then put command in the command bus and then pulse the enable signal.

To send data in 4bit mode; first put upper 4bit in the 4 bit data bus connected to 4MSB pins of LCD display, then put control signals in the control bus, then pulse the E pin once. Next put the lower 4 bit in the data bus and pulse the E pin again. Here is a flowchart simply describing it. **LCD Display Interfacing – Flowchart:-**

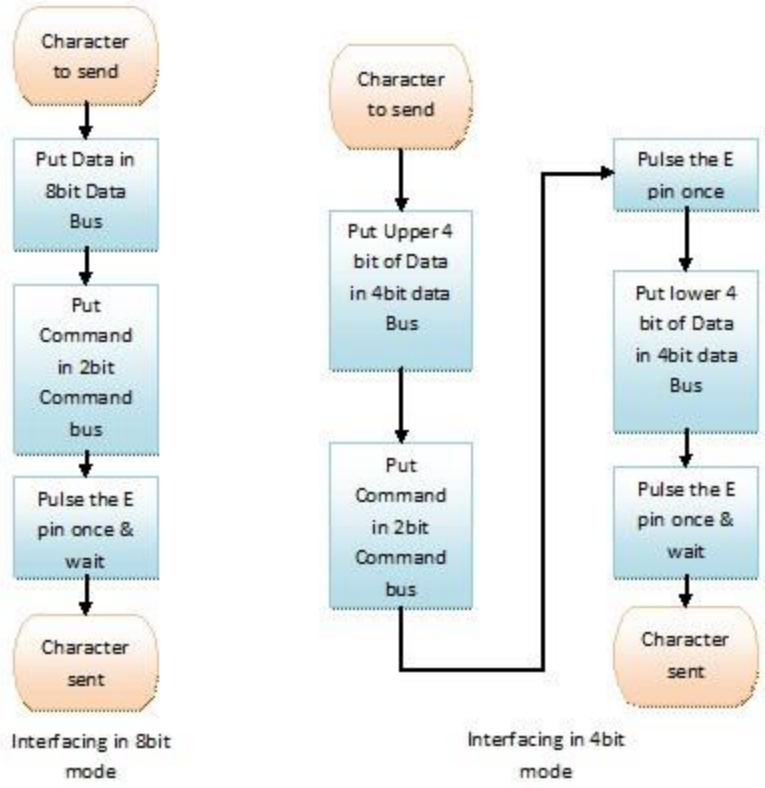
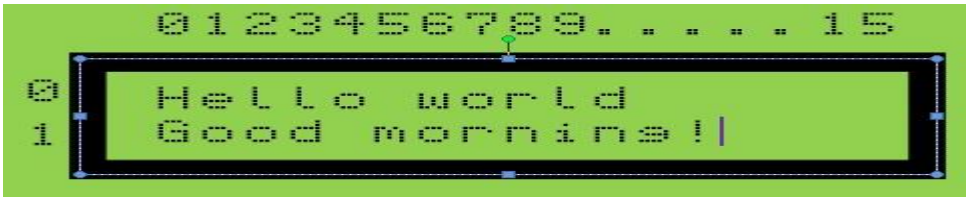


Fig: Flow chart of interfacing LCD display

8 BIT MODE

There is lot of stuff that can be done with the LCDs, to start with we will simple display a couple of strings on the 2 lines of the LCD as shown in the image.



Schematic description

- **Data Lines:** In this mode, all of the 8 datalines DB0 to DB7 are connected from the micrcontroller to a LCD module as shown the schematic.
- **Control Lines:'** The RS, RW and E are control lines, as discussed earlier.
- **Power & contrast**:Apart from that the LCD should be powered with 5V between **PIN 2(VCC)** and**PIN 1(gnd)**. **PIN 3** is the contrast pin and is output of center terminal of potentiometer (voltage divider) which varies voltage between 0 to 5v to vary the contrast.

- **Back-light:** The PIN 15 and 16 are used as backlight. The led backlight can be powered through a simple current limiting resistor as we do with normal leds.

4 BIT MODES:

Schematic

There are following differences in 4 bit mode.

- Only data lines D4 to D7 are used as shown in the schematic below.
- In code, we need to send the command to select 4 bit mode as shown in the instruction set above.

The main program remains exactly as in 8 bit mode, we simply include the lcd_4_bit.c to work in 4 bit mode.

POWER SUPPLY

WHAT IS A POWER SUPPLY?

A power supply is a device which delivers an exact voltage to another device as per its needs.There are many power supplies available today in the market like regulated, unregulated, variable etc, and the decision to pick the correct one depends entirely on what device you are trying to operate with the power supply. Power supplies, often called power adapters, or simply adapters, are available in various voltages, with varying current capacities, which is nothing but the maximum capacity of a power supply to deliver current to a load (Load is the device you are trying to supply power to).

WHY MAKE ONE YOURSELF?

`One would ask himself, “Why do I make it myself, when it available in the market?” Well, the answer is- even if you buy one, it is bound to stop working in a while (and believe me, power supplies stop working without any prior indication, one day they’ll work, the next day they’ll just stop working!). So, if you build one yourself, you will always know how to repair it, as you will know exactly what component/part of the circuit is doing what. And further, knowing how to build one, will allow you to repair the ones you have already bought, without wasting your money on a new one.

WHAT YOU NEED?

1. Copper wires, with at least 1A current carrying capacity for AC mains
2. Step Down Transformer
3. 1N4007 Silica Diodes (×4)
4. 1000µF Capacitor
5. 10µF Capacitor
6. Voltage regulator (78XX) (XX is the output voltage reqd. I'll explain this concept later)
7. Soldering iron
8. Solder
9. General Purpose PCB
10. Adapter jack (to provide the output voltage to a device with a particular socket)
11. 2 Pin plug

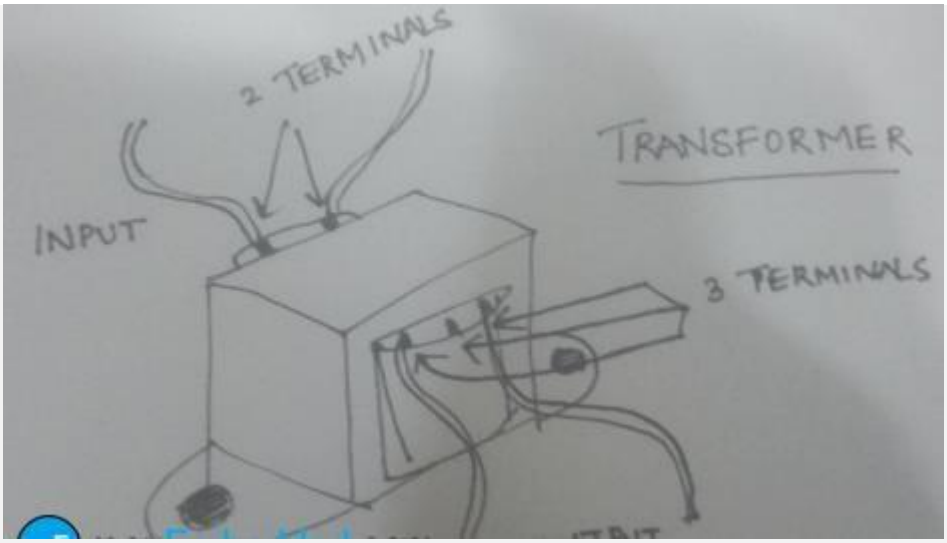
Optional

a. LED (for indication)
b. Resistor (Value explained later)
c. Heat Sink for The Voltage Regulator (For higher current outputs)
d. SPST Switch

SOME BASIC CONCEPTS RELATED TO POWER SUPPLIES

Transformers

Transformers are devices which step down a relatively higher AC input Voltage into a lower AC output voltage. To find the input and output terminals of a transformer is very tricky. Refer to the following illustration or the internet to understand where what is.



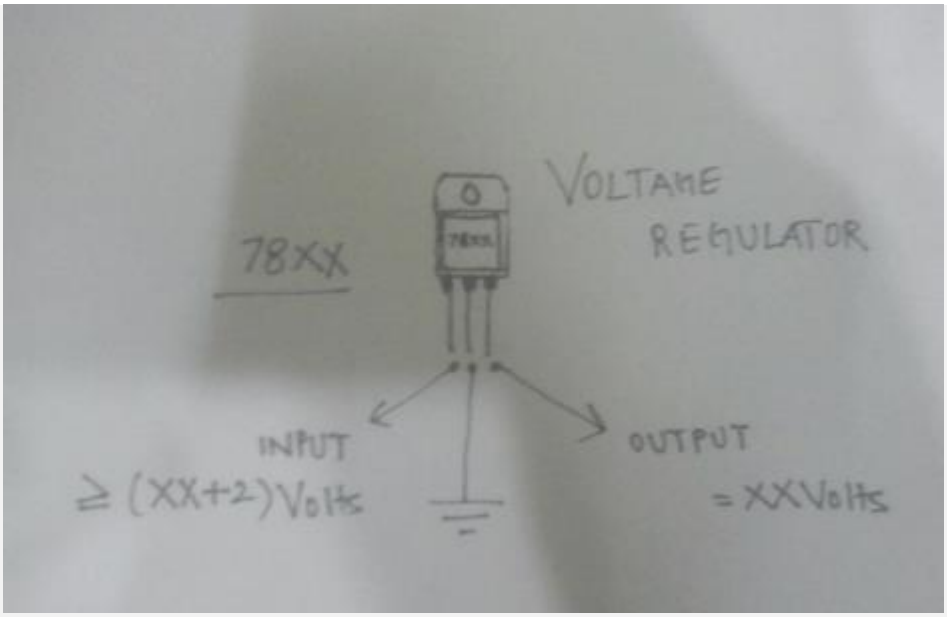
I/O Terminals of a Transformer

Basically, there are two sides in a transformer where the coil winding inside the transformer ends. Both ends have two wires each (unless you are using a center-tapped transformer for full wave rectification). On the transformer, one side will have three terminals and the other will have two. The one with the three terminals is the stepped down output of the transformer, and the one with the two terminals is where the input voltage is to be provided.

Voltage Regulators

The 78XX series of voltage regulators is a widely used range of regulators all over the world. The XX denotes the voltage that the regulator will regulate as output, from the input voltage. For instance, 7805, will regulate the voltage to 5V. Similarly, 7812 will regulate the voltage to 12V. The thing to remember with these voltage regulators is that they need at least 2 volts more than their output voltage as input. For instance, 7805 will need at least 7V, and 7812, at least 14 volts as inputs. This excess voltage which needs to be given to voltage regulators is called **Dropout Voltage**.

NOTE: The input pin is denoted as ‘1’, ground as ‘2’ and output as ‘3’.



Voltage Regulator Schematic

Diode Bridge

A bridge rectifier consists of an assembly of four ordinary diodes, by means of which we can convert AC Voltage into DC Voltage. It is found to be the best model for AC to DC conversion, over Full wave and Half wave rectifiers. You can use any model you want, but I use this for the sake of high efficiency (If you are using the full wave rectifier model, you'll need a center-tapped transformer, and you will only be able to use half of the transformed voltage).

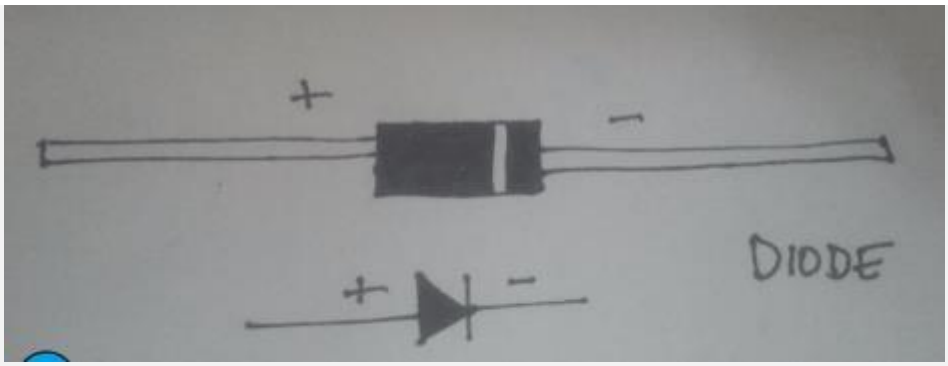
One thing to note about diodes is that they drop about 0.7V each when operated in forward bias. So, in bridge rectification we will drop 1.4V because at one instant two diodes are conducting and each will drop 0.7V. In case of Full wave rectifier, only 0.7V will be dropped.

So how does this drop affect us? Well, this comes in handy while choosing the correct step down voltage for the transformer. See, our voltage regulator needs 2 Volts more than its output voltage. For the sake of explanation, let's assume that we are making a 12V adapter. So the voltage regulator needs at least 14 Volts as input.

So the output of the diodes (which goes into the voltage regulator) will have to be more than or equal to 14 Volts. Now for the diodes' input voltage. They'll drop 1.4 Volts in total, so the input to them

has to be greater than or equal to 14.0 + 1.4 = 15.4Volts. So I would probably use a 220 to 18 Volt step down transformer for that.

So basically, the transformer step down voltage should be at least 3.4V more than the desired Power Supply output.



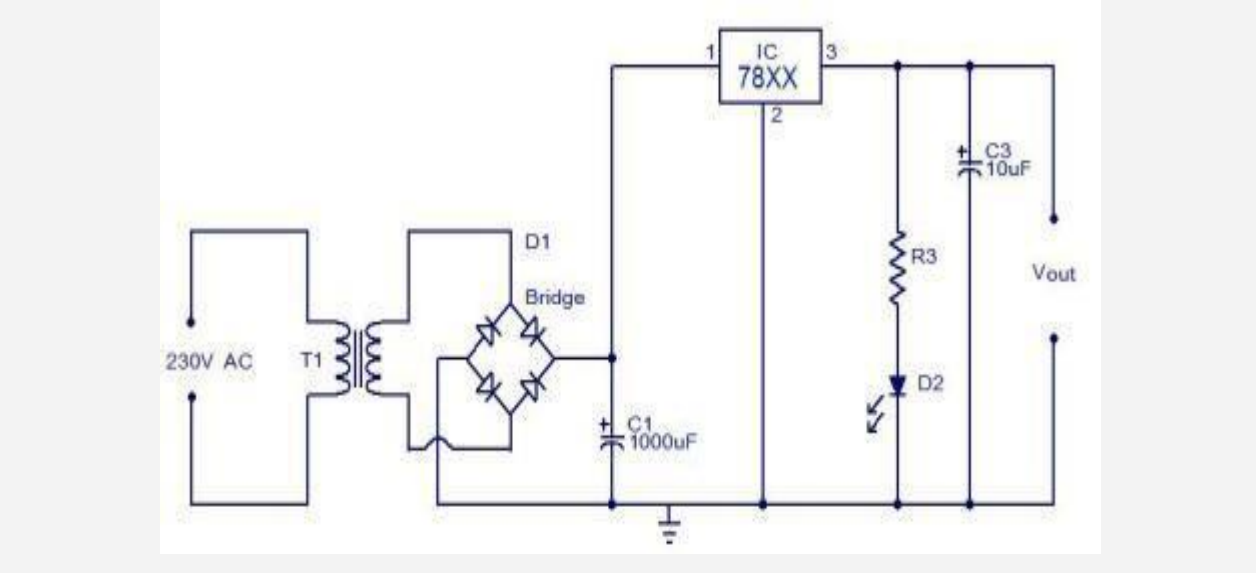
Schematic and Illustration of a Diode

Filter Circuit

We filter, at the output of the voltage rectifier in order to get the smoothest DC Voltage as possible, from our adapter, for which we use capacitors. Capacitors are the simplest current filters available, they let AC current pass through and block DC, so they are used in parallel to the output. Furthermore, if there is a ripple in the input or output, a capacitor rectifies it by discharging the charge stored in it.

HOW TO BUILD IT?

Here is the circuit diagram for the Power Supply:



Circuit Diagram

How it works

The AC mains are fed to the transformer, which steps down the 230 Volts to the desired voltage. The bridge rectifier follows the transformer thus converting AC voltage into a DC output and through a filtering capacitor feeds it directly into the input (Pin 1) of the voltage regulator. The common pin (Pin 2) of the voltage regulator is grounded. The output (Pin 3) of the voltage regulator is first filtered by a capacitor, and then the output is taken.

Make the circuit on a general purpose PCB and use a 2 Pin (5A) plug to connect the transformer input to the AC mains via insulated copper wires.

WHAT IS A RELAY?

We know that most of the high end industrial application devices have relays for their effective working. Relays are simple switches which are operated both electrically and mechanically. Relays consist of an

electromagnet and also a set of contacts. The switching mechanism is carried out with the help of the electromagnet. There are also other operating principles for its working. But they differ according to their applications. Most of the devices have the application of relays.

WHY IS A RELAY USED?

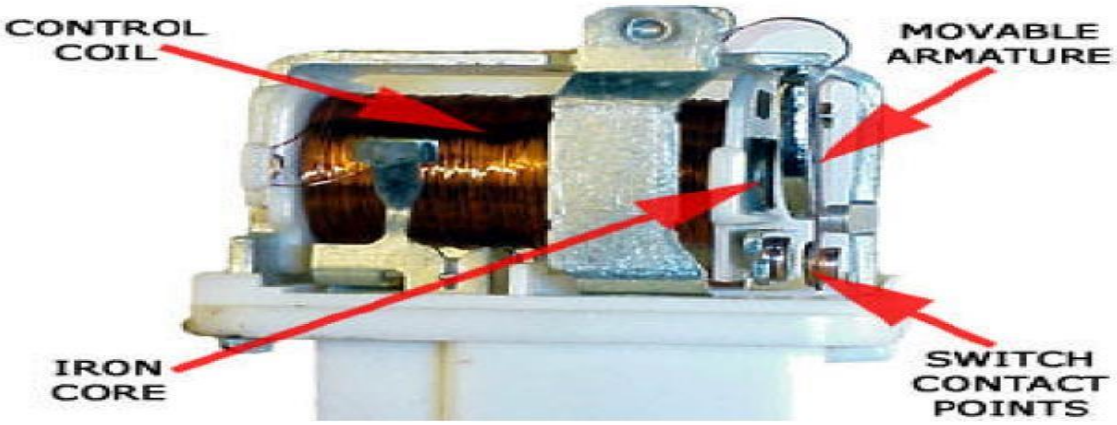
The main operation of a relay comes in places where only a low-power signal can be used to control a circuit. It is also used in places where only one signal can be used to control a lot of circuits. The application of relays started during the invention of telephones. They played an important role in switching calls in telephone exchanges. They were also used in long distance telegraphy. They were used to switch the signal coming from one source to another destination. After the invention of computers they were also used to perform Boolean and other logical operations. The high end applications of relays require high power to be driven by electric motors and so on. Such relays are called contactors.

RELAY DESIGN

There are only four main parts in a relay. They are

- Electromagnet
- Movable Armature
- Switch point contacts
- Spring

The figures given below show the actual design of a simple relay.

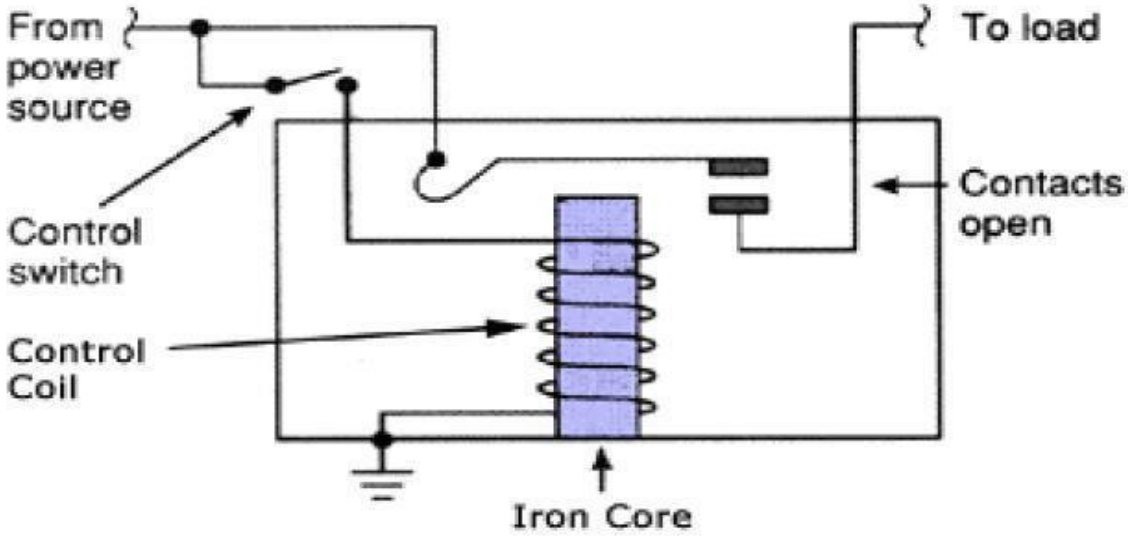


3.3.1 Relay Construction

It is an electro-magnetic relay with a wire coil, surrounded by an iron core. A path of very low reluctance for the magnetic flux is provided for the movable armature and also the switch point contacts. The movable armature is connected to the yoke which is mechanically connected to the switch point contacts. These parts are safely held with the help of a spring. The spring is used so as to produce an air gap in the circuit when the relay becomes de-energized.

HOW RELAY WORKS?

The working of a relay can be better understood by explaining the following diagram given below.



3.4.1 Relay Design

The diagram shows an inner section diagram of a relay. An iron core is surrounded by a control coil. As shown, the power source is given to the electromagnet through a control switch and through contacts to the load. When current starts flowing through the control coil, the electromagnet starts energizing and thus intensifies the magnetic field. Thus the upper contact arm starts to be attracted to the lower fixed arm and thus closes the contacts causing a short circuit for the power to the load. On the other hand, if the relay was already de-energized when the contacts were closed, then the contact move oppositely and make an open circuit.

As soon as the coil current is off, the movable armature will be returned by a force back to its initial position. This force will be almost equal to half the strength of the magnetic force. This force is mainly provided by two factors. They are the spring and also gravity.

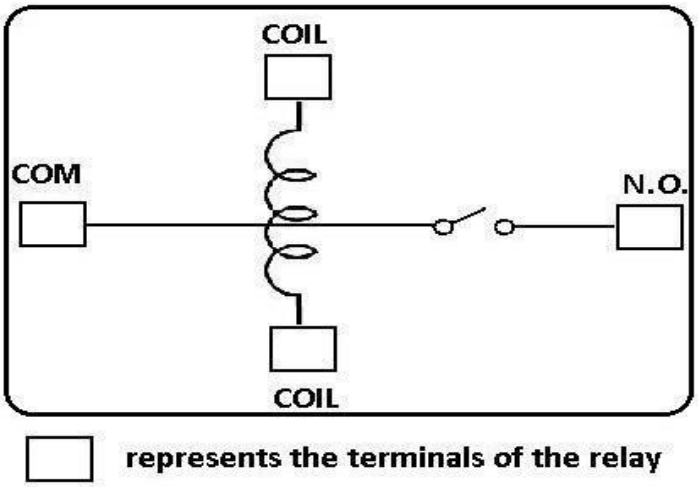
Relays are mainly made for two basic operations. One is low voltage application and the other is high voltage. For low voltage applications, more preference will be given to reduce the noise of the whole circuit. For high voltage applications, they are mainly designed to reduce a phenomenon called arcing.

HOW TO CONNECT A SINGLE POLE SINGLE THROW (SPST) RELAY IN CIRCUIT

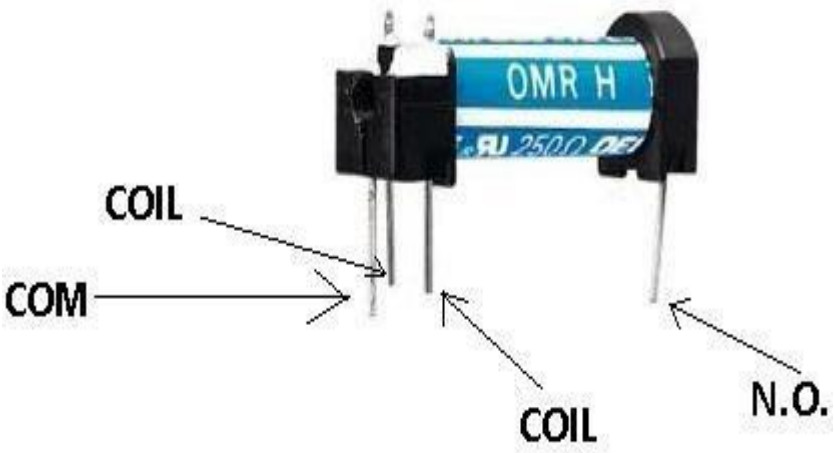
In order to know how to connect a single pole single throw (SPST) relay, you must know what each pin terminal represents and how the relay works.

Terminal Pins

A Single Pole Single Throw Relay comes with four terminal points. The terminals are COIL, COIL , COM, and N.O.



This correlates to the following in the relay:



Terminal Descriptions

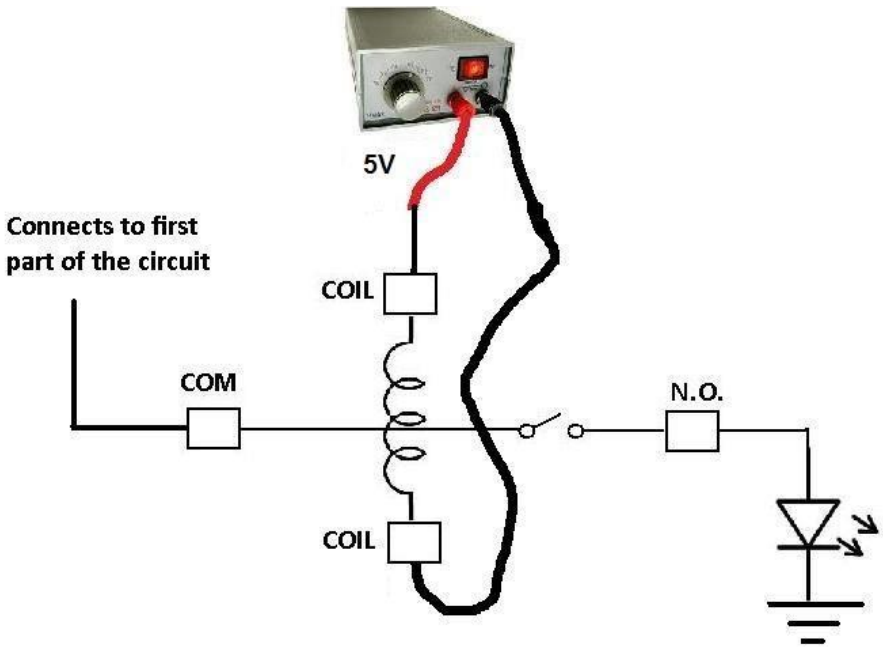
-COIL- This is one end of the coil.

COIL- This is the other end of the coil. These are the terminals where you apply voltage to in order to give power to the coils (which then will close the switch). Polarity does not matter. One side gets positive voltage and the other side gets negative voltage. Polarity only matters if a diode is used.

NO- This is Normally Open switch. This is the terminal where you connect the device that you want the relay to power when the relay is powered, meaning when the COIL receives sufficient voltage. The device connected to NO will be off when the relay has no power and will turn on when the relay receives power.

COM- This is the common of the relay. If the relay is powered and the switch is closed, COM and N.O. have continuity. This is the terminal of the relay where you connect the first part of your circuit to.

Now that we know what each terminal pin represents, we now wire it to a circuit for it to do a realworld function. We're going to connect a single pole single throw relay to a circuit to light up a LED. This is the circuit below to connect a single pole single throw relay to light a LED:



Since the relay is rated for 5V, it should receive 5 volts in order to power on. It may work with less voltage, but 5V is really what it should receive. This goes into either side of the COIL terminals. Even if you switched the positive and negative voltage of the power supply, it should work exactly the same.

The COM terminal of the relay gets connected to the first part of the circuit. If there is no first part of the circuit, this terminal can be left open.

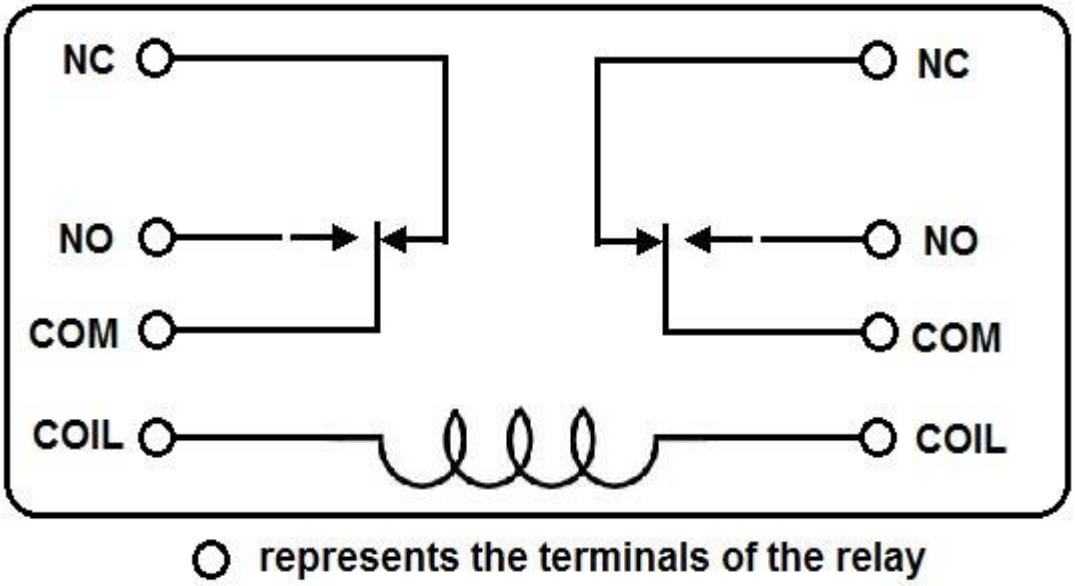
The N.O. terminal of the relay gets connected to the output of the device that it powers on or off. In this case, we are using a LED as output. When the relay receives 5V of power in its coil, the LED will light up. If the 5v are cut off from it, the LED will no longer light.

HOW TO CONNECT A DPDT RELAY IN A CIRCUIT

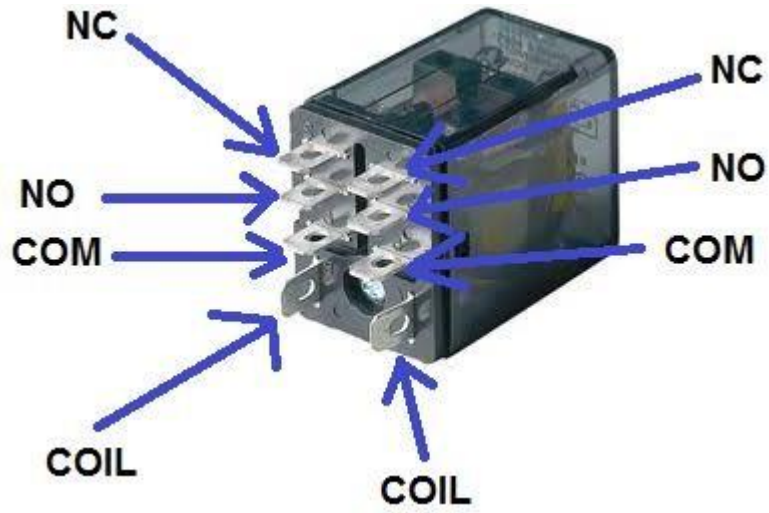
In order to know how to connect a DPDT relay, you must know what each pin terminal represents and how the relay works.

Terminal Pins

A Double Pole Double Throw Relay comes with 8 terminal points. The terminals are COIL, COIL, COM, COM, NO, NO, NC, NC.



This correlates to the following in the relay:



Terminal Descriptions

COIL- This the is the COIL terminal. These are the terminals where you apply voltage to in order to give power to the coils (which then will close the switch). Polarity does not matter. One side gets positive

voltage and the other side gets negative voltage. It doesn't matter which order. Polarity only matters if a diode is used.

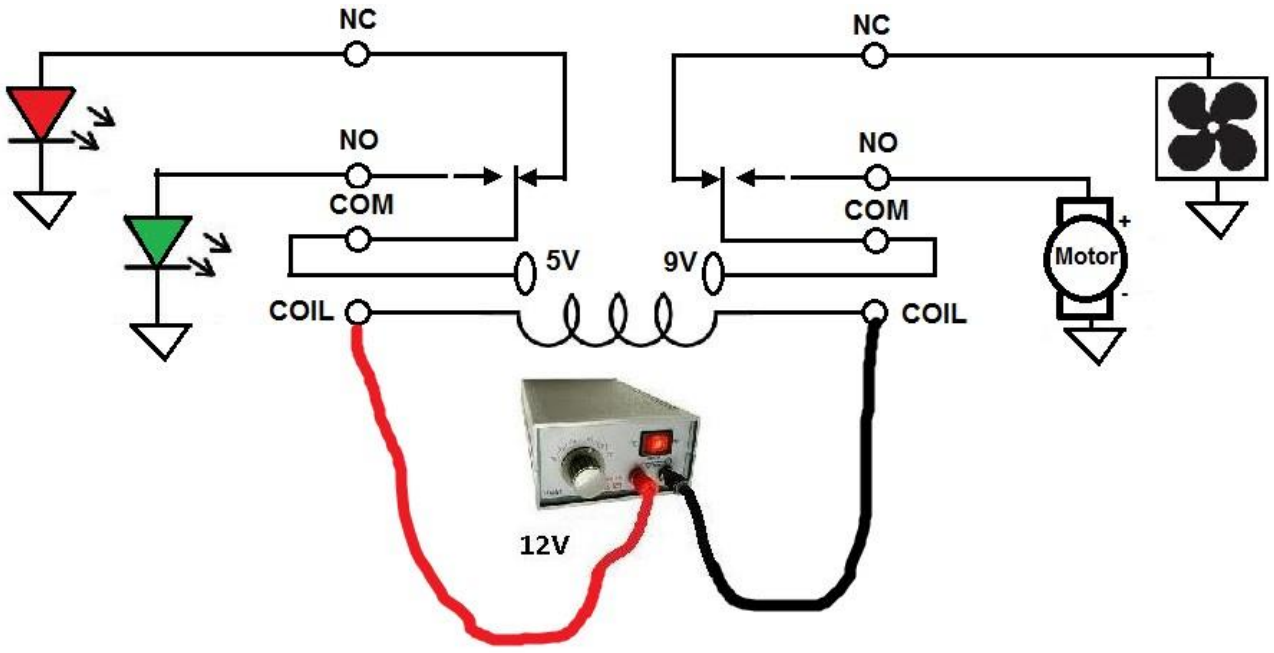
NO- This is Normally Open switch. This is the terminal where you connect the device that you want the relay to power, when the relay is powered, meaning when the COIL receives sufficient voltage. The device connected to NO will be off when the relay has no power and will turn on when the relay receives power.

NC- This is the Normally Closed Switch. This is the terminal where you connect the device that you want powered when the relay receives no power. The device connected to NC will be on when the relay has no power and will turn off when the relay receives power.

COM- This is the common of the relay. If the relay is powered and the switch is closed, COM and NO have continuity. If the relay isn't powered and the switch is open, COM and NC have continuity. This is the terminal of the relay where you connect the first part of your circuit to.

Now that we know what each terminal pin represents, we now wire it to a circuit for it to do a real-world function. We're going to connect a Double pole double throw relay to a circuit to light up LEDs. When the relay isn't powered, both the red LED and the DC fan are on,. When the relay is powered, the red LED and the fan shut off and the green LED and the DC motor turn on.

This is the circuit below:



Since the relay is rated for 12V, it should receive 12 volts in order to power on. It may work with less voltage, but 12V is really what it should receive. This goes into either side of the COIL terminals. Even if you switched the positive and negative voltage of the power supply, it should work exactly the same.

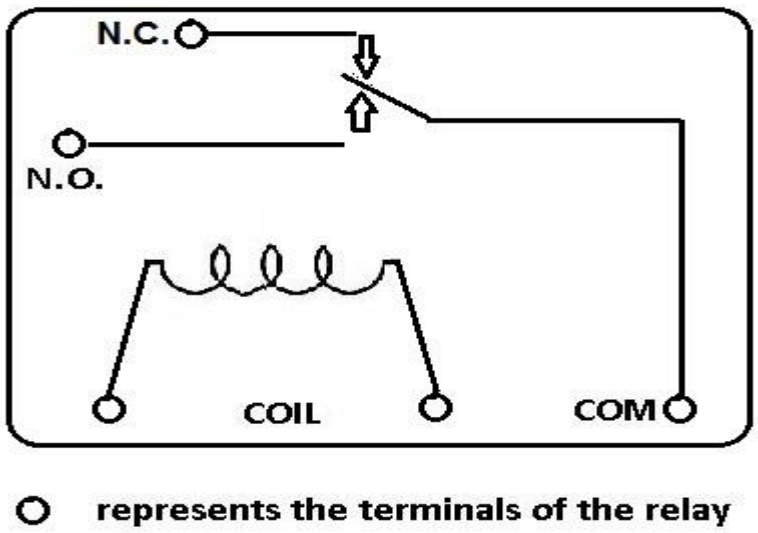
The COM terminals of the relay get connected to the first part of the circuit. If there is no first part of the circuit, this terminal can be left open. In this case, the first part of the circuit is the 5-volt power supply and the 9-volt power supply to light the LEDs and the DC fan and DC motor.

The NC terminals of the relay get power even when the relay isn't powered. This means that as long as the 5-volt power supply is on, the red LED and the DC fan will be on and operating.

The NO terminals of the relay get power only when the relay is powered. When the relay receives 12 volts of power, the relay snaps from the NC position to the NO position. The red LED and the DC fan now shut off and the green LED and the DC motor now turn on and operate.

SPDT RELAY

The Single Pole Double Throw SPDT relay is quite useful in certain applications because of its internal configuration. It has one common terminal and 2 contacts in 2 different configurations: one can be Normally Closed and the other one is opened or it can beNormally Open and the other one closed. So basically you can see the SPDT relay as a way of switching between 2 circuits: when there is no voltage applied to the coil one circuit "receives" current, the other one doesn't and when the coil gets energized the opposite is happening.



How does the Single Pole Double Throw relay works?

No DC voltage is applied to the coil so the terminal T is connected to contact 1 therefore the current can flow through 1 and it cannot flow through 2.

When DC voltage is applied to the coil and terminal T is now connected to contact 2 therefore the current doesn't flow anymore through 1 but now it flows through 2.

RTC:

Real time clocks (RTC), as the name recommends are clock modules. The DS1307 real time clock (RTC) IC is an 8 pin device using an I2C interface. The DS1307 is a low-power clock/calendar with 56 bytes of battery backup SRAM. The clock/calendar provides seconds, minutes, hours, day, date, month and year

qualified data. The end date of each month is automatically adjusted, especially for months with less than 31 days.

They are available as integrated circuits (ICs) and supervise timing like a clock and also operate date like a calendar. The main advantage of RTC is that they have an arrangement of battery backup which keeps the clock/calendar running even if there is power failure. An exceptionally little current is required for keeping the RTC animated. We can find these RTCs in many applications like embedded systems and computer mother boards, etc. In this article we are going to see about one of the real time clock (RTC), i.e. DS1307.

Pin Description of DS1307:

Pin 1, 2: Connections for standard 32.768 kHz quartz crystal. The internal oscillator circuitry is intended for operation with a crystal having a specified load capacitance of 12.5pF. X1 is the input to the oscillator and can alternatively be connected to an external 32.768 kHz oscillator. The output of the internal oscillator, X2 is drifted if an external oscillator is connected to X1.

Pin 3: Battery input for any standard 3V lithium cell or other energy source. Battery voltage should be between 2V and 3.5V for suitable operation. The nominal write protect trip point voltage at which access to the RTC and user RAM is denied is set by the internal circuitry as 1.25 x VBAT nominal. A lithium battery with 48mAhr or greater will backup the DS1307 for more than 10 years in the absence of power at 25ºC. UL recognized to ensure against reverse charging current when utilized as a part of conjunction with a lithium battery.

Pin 4: Ground.

Pin 5: Serial data input/output. The input/output for the I2C serial interface is the SDA, which is open drain and requires a pull up resistor, allowing a pull up voltage up to 5.5V. Regardless of the voltage on VCC.

Pin 6: Serial clock input. It is the I2C interface clock input and is used in data synchronization.

Pin 7: Square wave/output driver. When enabled, the SQWE bit set to 1, the SQW/OUT pin outputs one of four square-wave frequencies (1Hz, 4 kHz, 8 kHz, and 32 kHz). This is also open drain and requires an

external pull-up resistor. It requires application of either Vcc or Vb at to operate SQW/OUT, with an allowable pull up voltage of 5.5V and can be left floating, if not used.

Pin 8: Primary power supply. When voltage is applied within normal limits, the device is fully accessible and data can be written and read. When a backup supply is connected to the device and VCC is below VTP, read and writes are inhibited. However at low voltages, the timekeeping function still functions.

Features:

- Programmable square wave output signal
- Automatic power-fail detect and switch circuitry
- Consumes less than 500nA in battery backup mode with oscillator running
- Available in 8-pin DIP or SOIC
- Underwriters Laboratory (UL) recognized
- Real-time clock (RTC) counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap-year compensation valid up to 2100
- 56-byte non-volatile RAM for data storage
- Two-wire interface (I2C)

Using the DS1307 is primarily written to and read the registers of this chip. The memory contains all 64 DS1307 8-bit registers are addressed from 0 to 63 (from 00H to 3FH the hexadecimal system). The first eight registers are used for the clock register the remaining 56 vacant can be used as RAM contains temporary variable if desired. The first seven registers contain information about the time of the clock including: seconds, minutes, hours, secondary, date, month and year. The DS1307 include several components such as power circuits, oscillator circuits, logic controller and I2C interface circuit and the address pointer register (or RAM).

Working of DS1307:

In the simple circuit the two inputs X1 and X2 are connected to a 32.768 kHz crystal oscillator as the source for the chip. VBAT is connected to positive culture of a 3V battery chip. Vcc power to the I2C interface is

5V and can be given using microcontrollers. If the power supply Vcc is not granted read and writes are inhibited.

START and STOP conditions are required when a device wants to establish communication with a device in the I2C network.

- By providing a device identification code and a register address, we can implement the START condition to access the device.
- The registers can be accessed in serial order until a STOP condition is implemented

The DS1307 has the 2-wire bus connected to two I/O port pins of the DS5000: SCL – P1.0, SDA – P1.1. The V_{DD} voltage is 5V, R_P = 5KΩ and the DS5000 is by means of a 12-MHz crystal. The other secondary device could be any other device that recognizes the 2-wire protocol, such as the DS1621 Digital Thermometer and Thermostat. The interface with the D5000 was skilled using the DS5000T Kit hardware and software. These development kits allow the PC to be used as a dumb terminal using the DS5000's serial ports to substitute a few words with the keyboard and monitor.

Typical 2-wire bus arrangement, the following bus protocol has been defined during data exchange information; the data line must remain stable whenever the clock line is high. Changes in the data line while the clock line is high will be interpreted as control signals.

Accordingly, the following bus conditions have been defined:

Start data transfer: A change in the state of the data line from high to low, while the clock line is high, defines a START condition.

Stop data transfer: A change in the state of the data line from low to high, while the clock line is high, defines the STOP condition.

Data valid: The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the high period of the clock signal. The data on the line must be changed during the low period of the clock signal. There is one clock pulse per bit of data. Each data transfer is initiated with a START condition and terminated with a STOP condition. The number of data bytes transferred

between the START and the STOP conditions is not limited, and is determined by the master device. The information is transferred byte-wise and each receiver acknowledges with a ninth bit.

Software requirements

EMBEDDED C

Implanted C makes use of KEIL IDE programming. The framework program written in implanted C can be placed away in Microcontroller. The accompanying is a portion of the actual motives behind composing applications in C as opposed to get collectively. It is much less disturbing and much less tedious to write down in C then amassing. C is less traumatic to trade and refresh. You can utilize code available in capacity libraries. C code is compact to different microcontrollers with subsequent to 0 alteration. Genuine, installed C programming need nonstandard expansions to the C driver with a view to bolster charming components, as an example, settled point range catching, numerous unmistakable reminiscence banks, and fundamental I/O operations.

In 2008, the C Standards Committee prolonged the C data to deal with these problems via giving a normal wellknown to all executions to purchaser to contains numerous additives not handy in standard C, for example, settled factor wide variety catching, named address spaces, and vital I/O equipment tending to.

Installed C utilize the greater part of the grammar and semantics of wellknown C, e.G., number one() paintings, variable definition, facts type statement, contingent proclamations (if, switch. Case), circles (even as, for), capacities, exhibits and strings, structures and union, piece operations, macros, unions, and so on.

Embedded systems programming

Installed frameworks writing computer programs is not quite the same as creating applications on a desktop PCs. Key attributes of an implanted framework, when contrasted with PCs, are as per the following:

•Embedded gadgets have asset limitations (restricted ROM, constrained RAM, constrained stack space, less handling power)

•Components utilized as a part of installed framework and PCs are distinctive; implanted frameworks ordinarily utilizes littler, less power devouring segments. Inserted frameworks are more fixing to the equipment.

Two remarkable components of Embedded Programming are code speed and code estimate. Code speed is represented by the handling power, timing requirements, while code size is administered by accessible program memory and utilization of programming dialect. Objective of implanted framework writing computer programs is to get greatest elements in least space and least time.

Implanted frameworks are modified utilizing distinctive sort of dialects:

•Machine Code

•Low level dialect, i.e., get together

•High level dialect like C, C++, and Java and so on.

•Application level dialect like Visual Basic, scripts, Access, and so on..,

Arduino IDE:

The Arduino IDE software is a open source software, where we can have the example codes for the beginners. In the Present world there are lot of version in the Arduino IDE in which present usage is Version1.0.5. It is very easy to connect the PC with Arduino Board.

First we have to install the Arduino IDE software according to the below instructions:

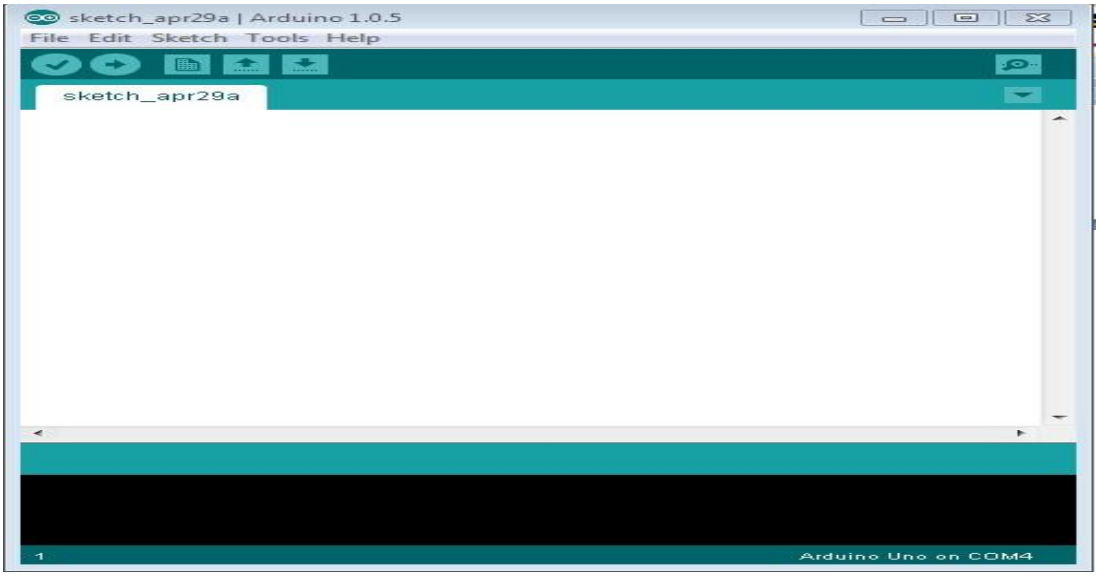
- Insert the CD-ROM or PENDRIVE which Contains the software and then Copy the Setup File to your desired location.
- After Copying, now click on the setup you will see an window shown below
- Click On NO, not this time. Then after NEXT
- Another Window opens –select Install from a list of specific location and NEXT

- Select “include this location in the search” and then click Browse option available in it

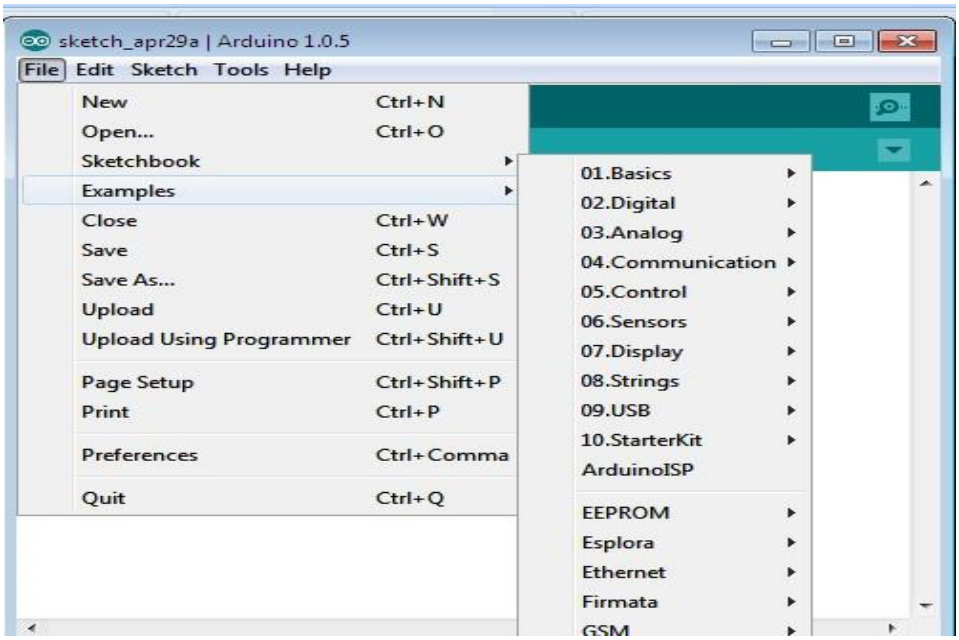
- Now it will Automatically check the USB driver and the software is installed click Finish

.

- Now click Finish, the Software will be downloaded.
- Now click on the Arduino IDE icon present on your Desktop. A window will appear like this.



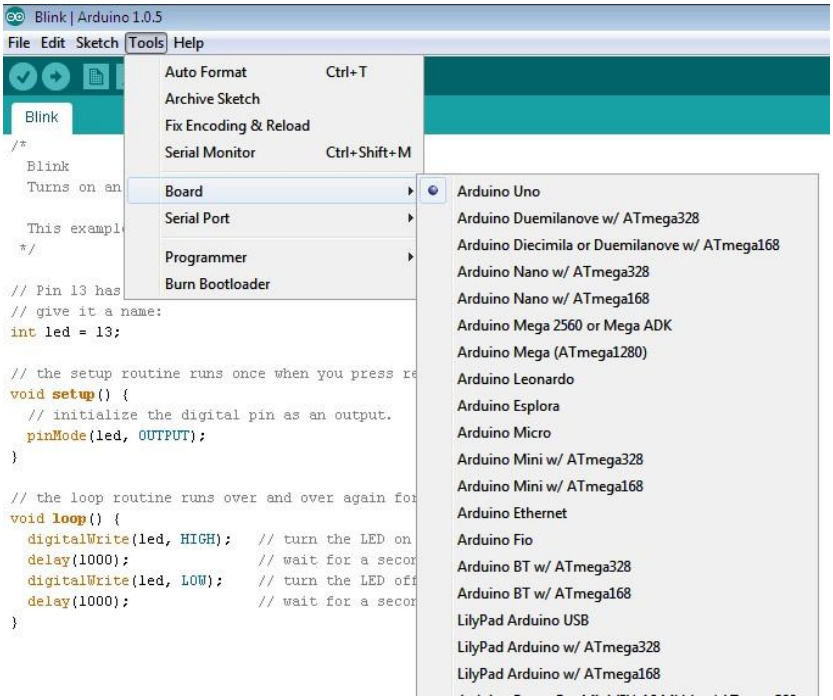
- For any sample programs, select FILE option➋Examples.



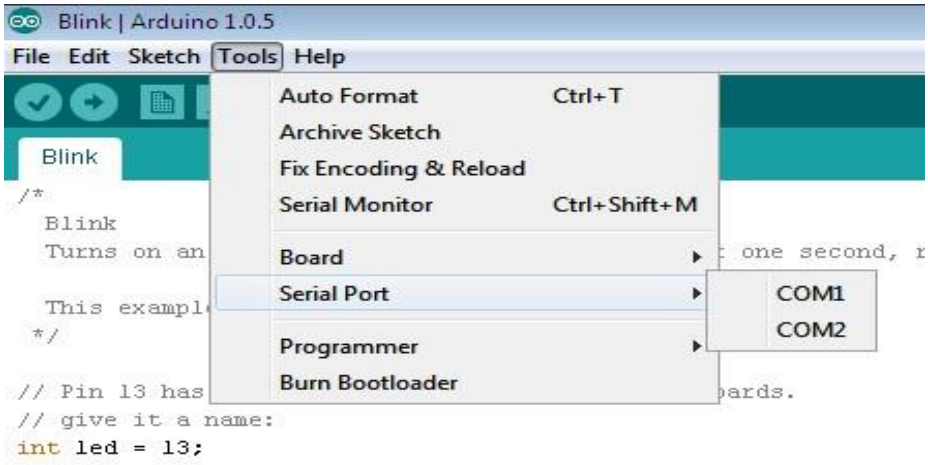
- After Entering the Sample Code in the file, it would look like this

- Before Connecting we have to select which Board is used by the user, Basically UNO.

  By selecting TOOLS➷Board➷ARDUINO UNO

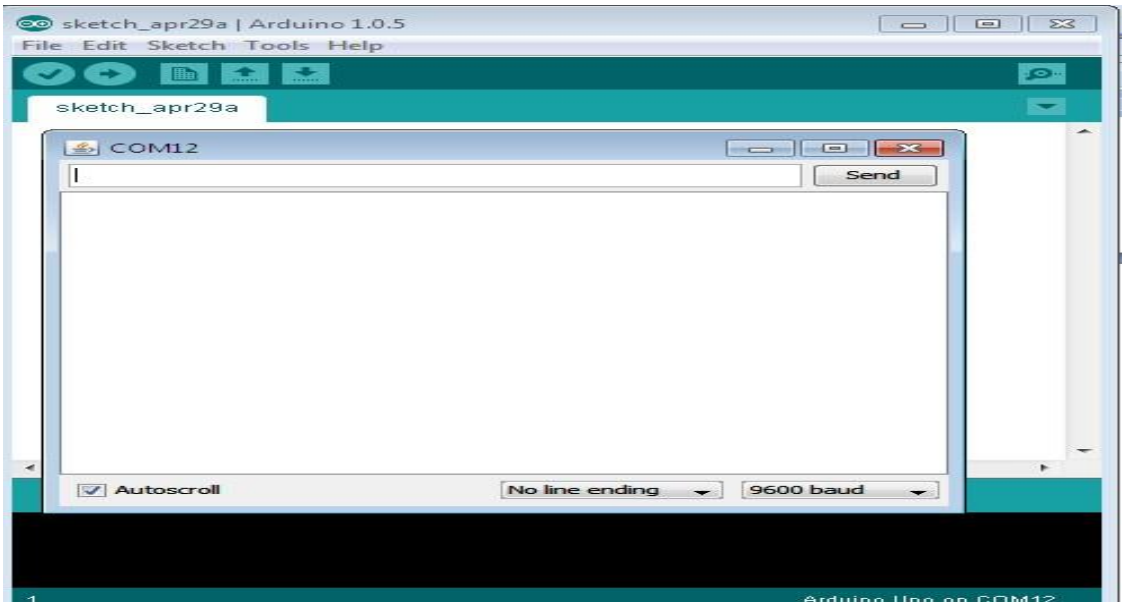


- Now to dump the in the board Connect the Arduino to the PC through the USB port available in it. Like this TOOLS➏SERIAL PORT➏COMM4,COMM8 etc;



- To verify the written Program select COMPILE option available in the software (√).
- Now Connect the Board and select the COMM port and then UPLOAD the file in ARDUINO(➏)

- To OPEN the Previous ARDUINO FILE selects (↑) option.
- To enter new files select NEW option.
- To Save the Existing File, Click on the(↓).
- To Send the Data Through Serial Monitor, Click on the (Q).



- Here we can see the Serial Data.

Advantages:

- Power consumption is reduced
- easy

Advantages:

- In household applications
- Industries

Conclusion

The project presents the programmable based load shedding system helps in improving the quality and stability of power system. The technologies and infrastructure are designed to be in place and will take care of all the challenges and vulnerabilities of automatic load shedding system. The system is technically feasible therefore nothing should prevent the transition into the smart load shedding system.

References

[1] Data Sheet - Texas Instruments of ULN2003 Rekha, A,Dr. Rathina Kumar, M., Ashok Kumar, N., Research Article An Intelligent Load Shedding To Improve Power System Congestion published in 13 April 2013.

[2] Datasheet of AT89S52 microcontroller from Atmel [online] available. www.atmel.com/images/doc0265.pdf

[3] Data Sheet - Texas Instruments of ULN2003 [online] www.ti.com/lit/ds/symlink/uln2003a.pdf .

[4] MVP Rao - A Research in Real Time Scheduling Policy for Embedded system-2009 the paper is available at www.clei.org/cleiej/papers/v12i2p4.pdf .

[5] DS1307 - Part Number Search - Maxim datasheets.maximintegrated.com/en/ds/DS1307.pdf .

[6] Demand for electricity changes through the days www.eia.gov .

[7] Electricity Demand http://www.mpoweruk.com/electricity_demand.htm