

Microcontrollers workshop

ROBOTICS CLUB

IIT GUWAHATI

Microcontrollers:

- ✘ It is just like a computer but with size as small as a 50 paise coin!!
- ✓ The microcontroller includes a CPU, RAM, ROM, I/O ports, and timers like a standard computer.
- ✘ Modern day microcontrollers have tremendous processing power. Some of them even run an operating system eg: Tiny OS and database systems like Tiny DB
- ✘ Here also, we'll use ports to communicate with hardware

Microcontroller

Uses:

- ✓ Microcontrollers have become common in many areas, and can be found in home appliances, computer equipment, and instrumentation.
- ✓ They are often used in automobiles, and have many industrial uses as well, and have become a central part of industrial robotics.
- ✓ Because they are usually used to control a single process and execute simple instructions, microcontrollers do not require significant processing power.

Uses cont...

- ✓ Microcontrollers are hidden inside a surprising number of products these days.
- ✓ If your microwave oven has an LED or LCD screen and a keypad, it contains a microcontroller.
- ✓ All modern automobiles contain at least one microcontroller. The engine is controlled by a microcontroller, as are the anti-lock brakes, the cruise control and so on..

Components of a microcontroller

- ✘ ALU
- ✘ Flash memory – non-volatile – typically contains the program
- ✘ EPROM-non volatile – typically contains some critical static info like serial id, password
- ✘ SRAM- typically used to store temporary data.. Like variables of a C program
- ✘ Timers
- ✘ ADC/DAC
- ✘ Comparators
- ✘ PWM channels
- ✘ UART

- Microcontroller- here we will be using atmel AVR microcontrollers
 - Atmega16
 - Atmega32
 - Atmega8
- Programmer-device to load the program into the microcontroller
for those who have a desktop ,the programmer can be made with inexpensive parts.

This programmer is called the parallel port programmer. To make this programmer u would need a db25 male connector pin.

The connections to the microcontroller are as follows

db25 connector	microcontroller pins
Pin 7	AVR /RESET
Pin 8	AVR SCK (clock input)
Pin 9	AVR MOSI (instruction in)
Pin 10	AVR MISO (data out)
Pin 18	Signal Ground

- Just connect the pins the above order and your programmer is ready
- For those who use laptops with a usb port, there are available usb programmers for sale . you could buy them from the following sites

www.robokits.org

www.nex-robotics.com

www.triindia.com

microcontrollers are also available here

- Then you would need stepper motors or dc motors and the controller ic

- Power supply

batteries of 12 V are usually used.

these are also available at those sites

A voltage regulator ic has to be used.

the ic is 7805 and is available in the local market.this ic supplies all the electronic components with 5 V .this ic works only when the input voltage to it is greater than 7V. it gets hot when the voltage is much higher.

use batteries with higher mAh rating.

- Sensors

the sensors usually used are infrared sensors.

infrared emitters and receivers are available in the market and could be used to make a range sensor, converting the distance from the robot to the obstacle into analog voltage.

the sensors that we make are not accurate in their measurement and are affected by lighting conditions. sensors that are sold in those robot shops are immune to lighting conditions ,as they have a special mechanism of sending a particular frequency and receiving the same.

- Software

winavr is the software used to load the program into the microcontroller. When you install the program of winavr ,2 icons would appear on your screen.

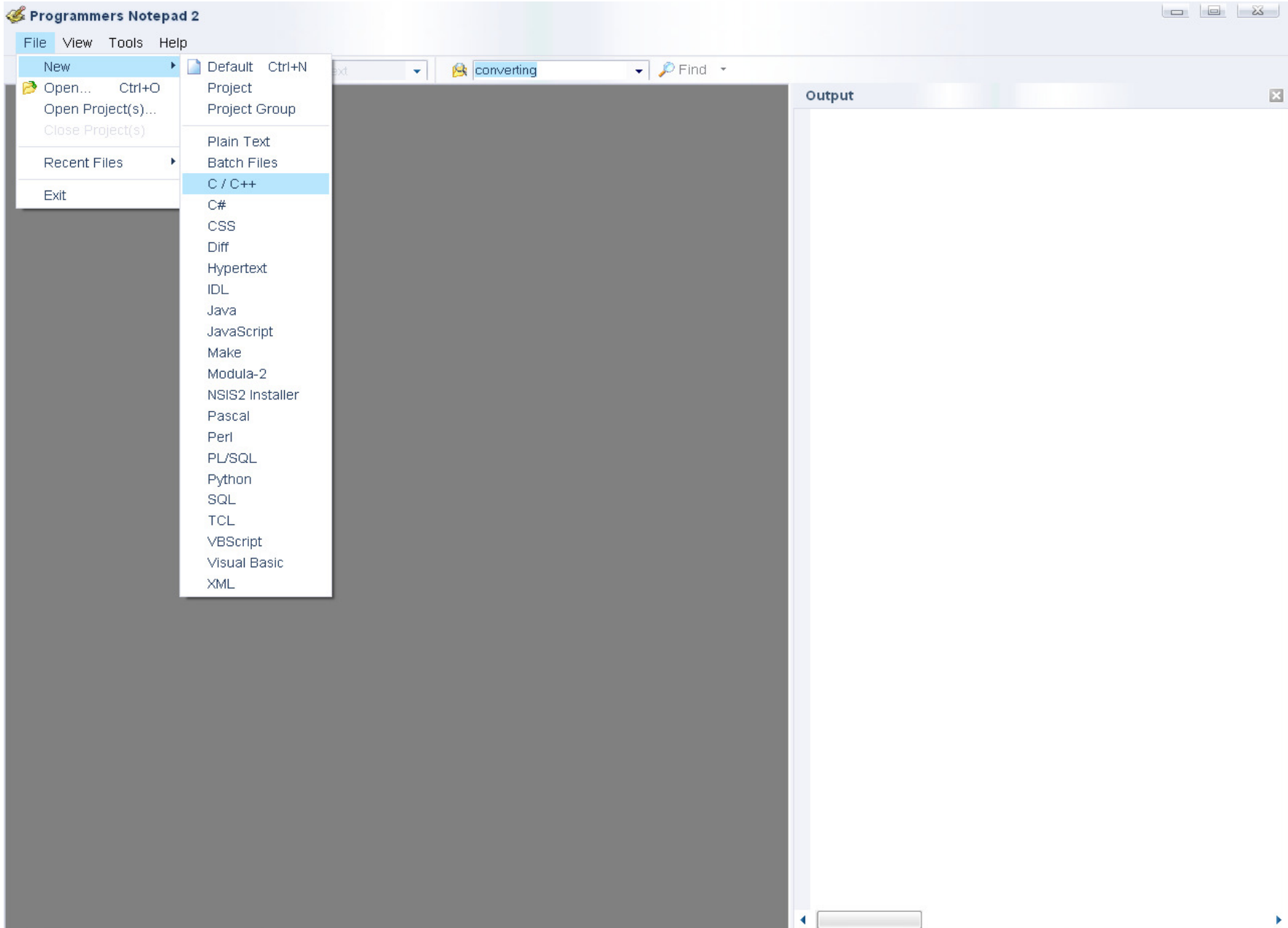
Programmer's note pad –used to write the program.

MFILE – used to convert the written c program into the language of the microcontroller.

Desktop users have to install the giveio driver ,as windows does not give access to parallel port without this driver.

Programming

- Connect the microcontroller to the computer with the programmer.
- Provide the 5V and ground supply to the microcontroller.
- Create a folder where u would like to save the program .
- Write the below program and save the file with the extension of .c or .cpp



Programmers Notepad 2

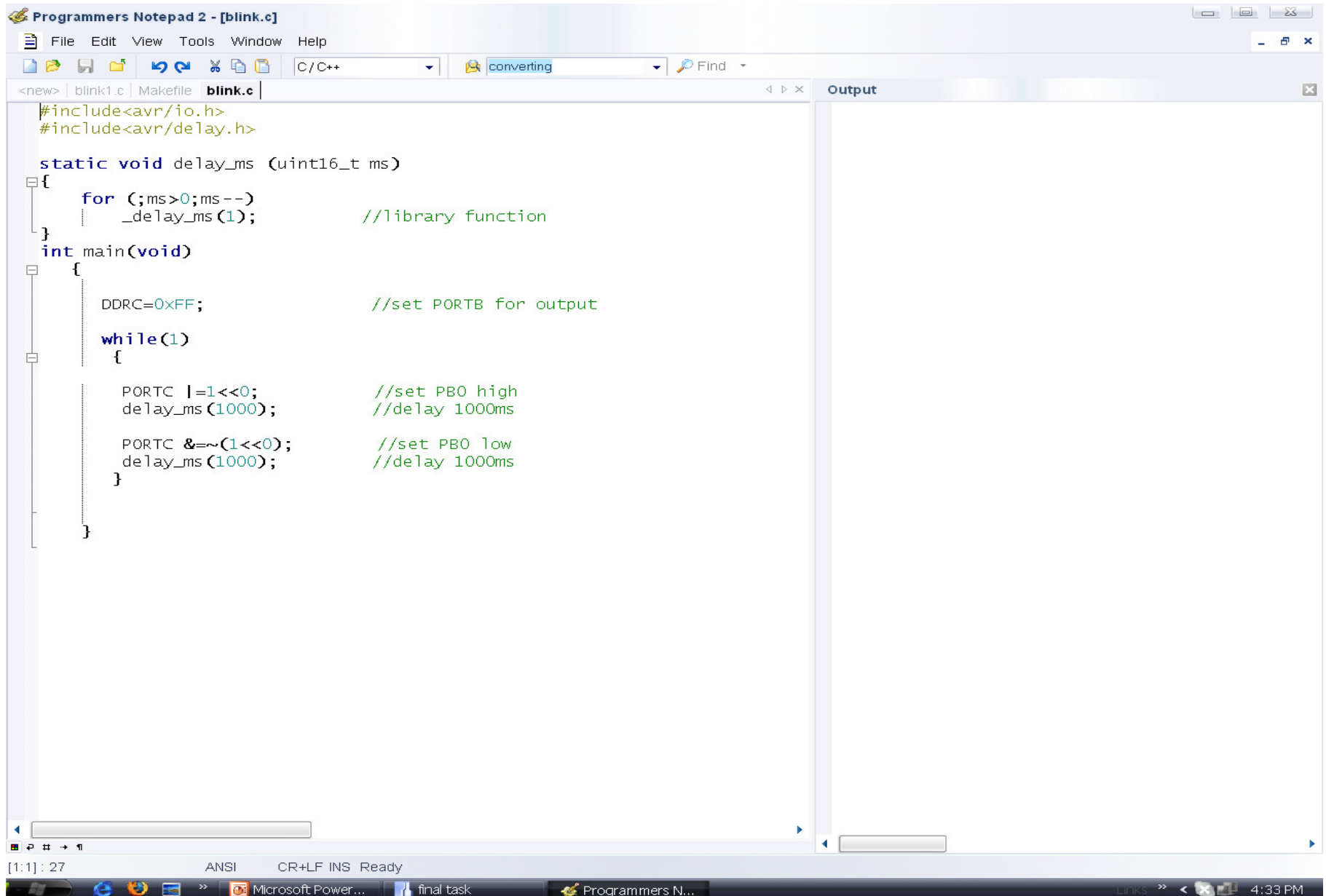
File View Tools Help

- New
 - Open... Ctrl+O
 - Open Project(s)...
 - Close Project(s)
 - Recent Files
 - Exit
- Default Ctrl+N
 - Project
 - Project Group
 - Plain Text
 - Batch Files
 - C / C++
 - C#
 - CSS
 - Diff
 - Hypertext
 - IDL
 - Java
 - JavaScript
 - Make
 - Modula-2
 - NSIS2 Installer
 - Pascal
 - Perl
 - PL/SQL
 - Python
 - SQL
 - TCL
 - VBScript
 - Visual Basic
 - XML

converting Find

Output

Program for blinking LED on PB0



```
Programmers Notepad 2 - [blink.c]
File Edit View Tools Window Help
C/C++ converting Find
<new> blink1.c Makefile blink.c
#include<avr/io.h>
#include<avr/delay.h>

static void delay_ms (uint16_t ms)
{
    for (;ms>0;ms--)
        _delay_ms(1);        //library function
}

int main(void)
{
    DDRC=0xFF;                //set PORTB for output

    while(1)
    {
        PORTC |=1<<0;        //set PB0 high
        delay_ms(1000);      //delay 1000ms

        PORTC &=~(1<<0);     //set PB0 low
        delay_ms(1000);      //delay 1000ms
    }
}

[1:1] : 27 ANSI CR+LF INS Ready
Microsoft Power... final task Programmers N... 4:33 PM
```

```
// contains the definitions of PORT,DDR etc
```

```
static void delay_ms (uint16_t ms)  function to generate a delay
{
    for (;ms>0;ms--)
        _delay_ms(1);                //library function to generate 1 ms delay
}
```

```
int main(void)
{
```

```
    DDRD=0xFF;                        //set PORTD for output
    while(1)
    {
        PORTD |=1<<2;                 //set PD2 high
        delay_ms(1000);               //delay 1000ms

        PORTD &=~(1<<2);              //set PD2 low
        delay_ms(1000);               //delay 1000ms
    }
```

```
}
```

#include<avr/io.h> is a library that contains the definitions of PORT, DDR
PORTA is a register that can be used to read and write to a port.

A port is a set of 8 pins. there are 4 ports for
atmega16. PORTA, PORTB, PORTC, PORTD.

eg

PORTA=0x01;

Implies that in PORTA pin 0 (PA0) has been made high.

Pin no.	7	6	5	4	3	2	1	0
PORTA =	0	0	0	0	0	0	0	1

Writing a 1 to the pin is making the corresponding pin high i.e. 5V.

Writing a 0 to the pin will make the pin low i.e. 0V.

0x01 is hexadecimal number

You could make any no. pins high or low simultaneously.

PORTA=0x32;

Pin no.	7	6	5	4	3	2	1	0
PORTA =	0	0	1	1	0	0	1	0

Pin 5,4 and 1 are made 5V simultaneously (PA5,PA4,PA1) rest all are at 0V.

In some random no . OxPQ

in this the first no. P corresponds to pins 7 6 5 4

the second no Q corresponds to pins 3 2 1 0

eg 0x32

The value of 3 is 0011 that means 7 and 6 are given 0 0 ,5 and 4 are given 1 1.

The value of 2 is 0010 that means 3 and 2 are given 0 0,1 is given 1 (high) and 0 is given 0(low).

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

- DDRA, DDRB, DDRC, DDRD are all data direction registers .they tell the microcontroller whether the port/pin is used as an input or output.

To configure a port as output

DDRA=0xFF;

To configure a port as input

DDRA=0x00;

Writing a 1 makes the pin an output , writing a 0 will make it an input.

Individual pins of each port can be configured separately.

DDRA=0xA9;

Pin no.	7	6	5	4	3	2	1	0
DDRA =	1	0	1	0	1	0	0	1

pinA 7,5,3,0 are made outputs,

6,4,2,1 are inputs.

- The OR and AND operations

or operator is |

and operator is &

eg

```
PORTD=0x00;
```

```
PORTD=PORTD | 0x02 ;
```

Makes pd2 high.

This is equivalent to bit by bit or operation. let P and Q be 2 -8 bit registers

$R = P | Q$ implies— 7th bit of P and 7th bit of Q are operated with or and the result is stored in the 7th bit of R. and so on with the other bits.

Now $PORTD = 0x02$;.....ie the pin PD2 is made high

```
PORTD = PORTD | 0xA4;
```

```
PORTD=0xA6;
```

$PORTD=PORTD \& 0x45$; works similar to or but does the and operation.

```
PORTD= 0x04;
```

- To read a specific pin value u have to use 'PIN', PINA, PINB etc

Eg to read the value(whether high or low) on PC4.

```
If(PINC & 0x10 == 0x10)
```

```
{
```

```
Write your code here
```

```
}
```

This is how it works

If PINC is 0xED, then $0xED \& 0x10 = 0x00$ ---which is not equal to 0x10. the code would not be executed

If PINC=0xB9, then $0xB9 \& 0x10 = 0x10$ ---the code would be executed.

We could check the value on more than one pin of a port simultaneously

```
If(PINC & 0x12 == 0x12)
```

```
{
```

```
}
```

Would check if there are 1 s on pins 4 and 2 (PC4 and PC2)

$(PINC \& 0x12 \neq 0x12)$ could be used to check for zeros.

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
0xED	1	1	1	0	1	1	0	1
0x10	0	0	0	1	0	0	0	0
'AND'	0	0	0	0	0	0	0	0
PC4 is low here....								
	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
0xB9	1	0	1	1	1	0	0	1
0x10	0	0	0	1	0	0	0	0
'AND'	0	0	0	1	0	0	0	0
PC4 is high here....								

Using the adc (analog to digital converter) of the microcontroller

Adc is used to read the analog voltage of the sensor.

```
int main(){
```

```
int x;
```

```
ADCSRA = _BV(ADEN) | _BV(ADPS2);
```

```
·
```

```
·
```

```
    ADMUX=0;
```

```
    ADMUX=(ADMUX | 0x22);           //selects PA2 as the input channel
```

```
    ADCSRA |= _BV(ADSC);           // Start conversion
```

```
    while (ADCSRA & _BV(ADSC) )    // wait until conversion completed
```

```
    {
```

```
    }
```

```
    x = ADCH;
```

```
    // use this x in your code.this conversion is a 8 bit conversion.the result will lie between 0  
    and 255.it is proportional to the voltage
```

```
    x = (255 * V)/5. where v is the input voltage to the pin that is read.
```

```
·
```

```
·
```

```
}
```

- To select other channels

Use

ADMUX=(ADMUX | 0x23);.....PA3

ADMUX=(ADMUX | 0x20);.....PA0

ADMUX=(ADMUX | 0x21);.....PA1

ADMUX=(ADMUX | 0x24);.....PA4

ADMUX=(ADMUX | 0x25);.....PA5

ADMUX=(ADMUX | 0x27);.....PA7

Pin Avcc and AREF are to be given 5 V

Pin no.31 is also to be grounded

- Save the file with the extension .c into a folder
- Open MFILE and do the following
 - click on makefile->mcu type->atmega->atmega16.
 - Debug format->AVR-ext -coff
 - Programmer->bsd (for parallel port programmer)
 - Port->lpt1
- For usb programmer ,a ready made make file is given with the programmer
 - Save this makefile in the folder where u have saved your file.
 - Open this makefile in the folder with word pad.
 - Change cpu frequency to 1000000 .the value that would previously be there is 8000000,change this to 1000000
 - Change the target name to ur file name with out the extension.
 - Save the make file.

- To program the microcontroller

In programmer's note pad ,open the program,then click tools->make all

Then tools->program.

The microcontroller gets programmed.

Microcontroller

Pinout ATmega16

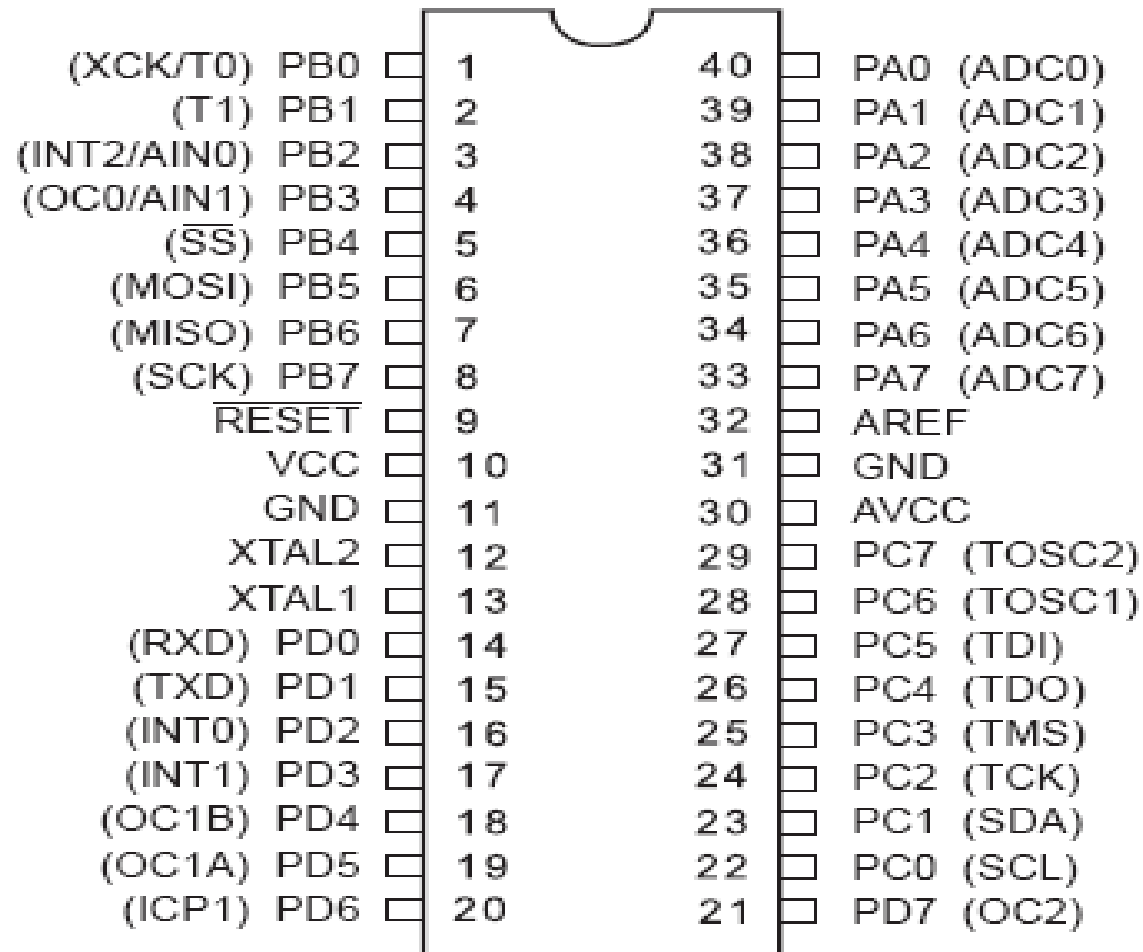


PDIP

(XCK/T0) PB0	1	40	PA0 (ADC0)
(T1) PB1	2	39	PA1 (ADC1)
(INT2/AIN0) PB2	3	38	PA2 (ADC2)
(OC0/AIN1) PB3	4	37	PA3 (ADC3)
(SS) PB4	5	36	PA4 (ADC4)
(MOSI) PB5	6	35	PA5 (ADC5)
(MISO) PB6	7	34	PA6 (ADC6)
(SCK) PB7	8	33	PA7 (ADC7)
RESET	9	32	AREF
VCC	10	31	GND
GND	11	30	AVCC
XTAL2	12	29	PC7 (TOSC2)
XTAL1	13	28	PC6 (TOSC1)
(RXD) PD0	14	27	PC5 (TDI)
(TXD) PD1	15	26	PC4 (TDO)
(INT0) PD2	16	25	PC3 (TMS)
(INT1) PD3	17	24	PC2 (TCK)
(OC1B) PD4	18	23	PC1 (SDA)
(OC1A) PD5	19	22	PC0 (SCL)
(ICP1) PD6	20	21	PD7 (OC2)

Pinout ATmega16

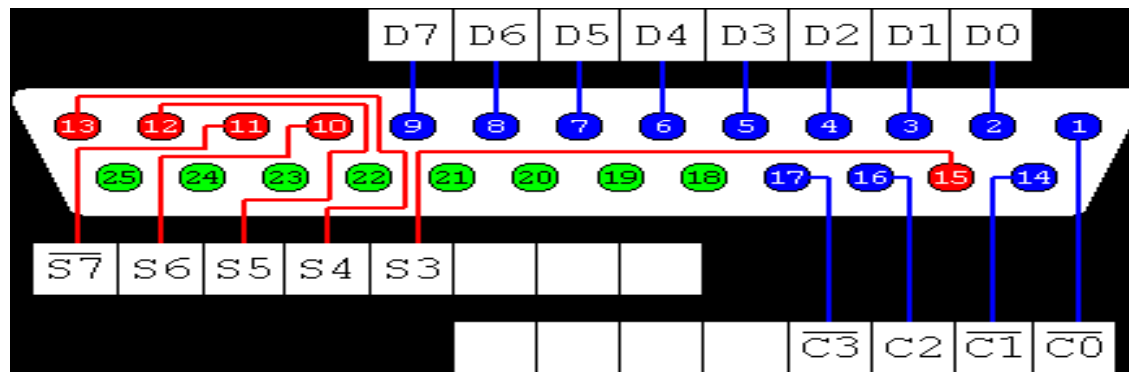
PDIP



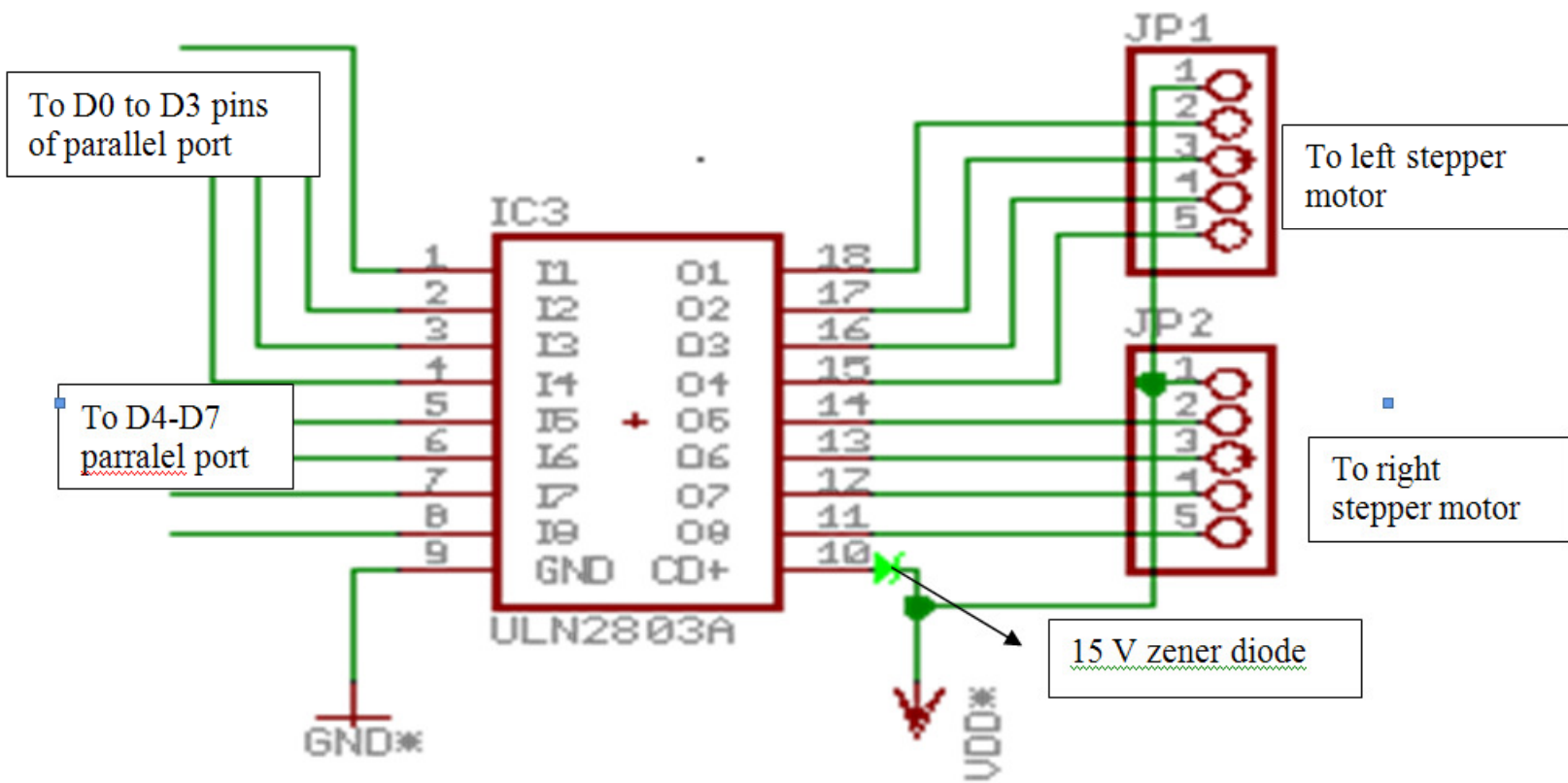
Autonomous robotics

- ✘ In the context of college techfests, it refers to the robots which are controlled by microcontrollers (extremely tiny onboard computers)
- ✘ Unlike the earlier one (where the intelligence was in the PC), no external intelligence required here.
- ✘ The small robot is self-sufficient in all aspects
- ✘ Very useful in situations where there is a serious size and power constraint but the computational requirement is not very huge. Rapid advancement is being made in both these fields

The PC's parallel port



Interfacing circuit



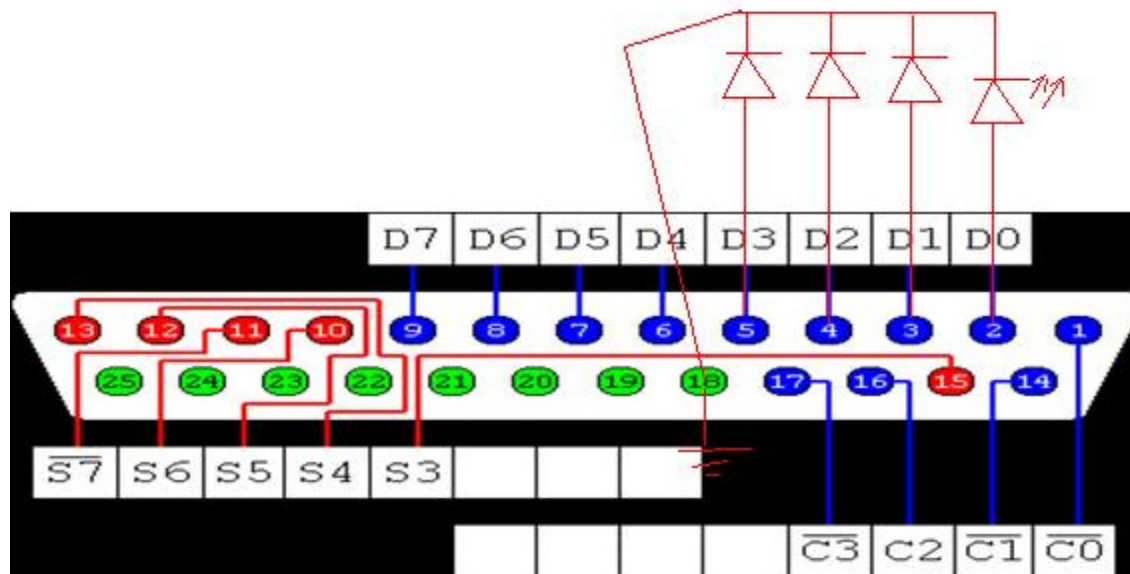
Programming the stepper motor in C

```
//single coil mode
//this program rotates the axle of the stepper motor by 180 degrees

#include <stdio.h>
#include <dos.h>

main()
{ char a[]={1,2,4,8};
  for (int i=0;i<=100;i++)
  { outportb(888,a[i%4]);
    delay(10);
  }
  outportb(888,0);
}
```

Ckt for blinking leds by prev prog



ATmega16

- ✘ 16K bytes of in-system programmable stable flash memory.
- ✘ Endurance 10,000 write/erase cycles.
- ✘ 512 bytes of EEPROM.
- ✘ Endurance:100,000 write/erase cycle.
- ✘ 1k bytes of SRAM.

ATmega16 cont....

- ✘ Two 8 bit and one 16 bit timer/counter with separate prescalers and compare mode.
- ✘ Four PWM channels.
- ✘ 8 channel 10 bit ADC.
- ✘ 32 I/O lines.
- ✘ Six sleep mode : idle, power save , stand by, ADC noise reduction , ext standby, power down.

Ports

- ✘ Just like the ports of the PC, they can be used to communicate with external hardware
- ✘ 4 parallel ports & 1 serial port available in Atmega 16
- ✘ A parallel port is just an array of 8 pins. The on/off pattern of these pins reflects the binary pattern of the number stored in the IO control registers
- ✘ These pins can be used both for input and output depending on the value stored in direction control register(explain).

Start programming

Step1: Install winavr on your computer

this will generate three shortcuts on desktop, TkInfo, programmers Notepad and Mfile.

Step2: Read the user manual (if possible)

Step3: Write a program in c, c++, or assembly language ,use Programmers Notepad

Step4: write a makefile, use Mfile

Programming cont...

Step5: Compile your program

Step6: Simulate program using AVR studio

Step7: Connect programmer hardware with
computer and microcontroller.

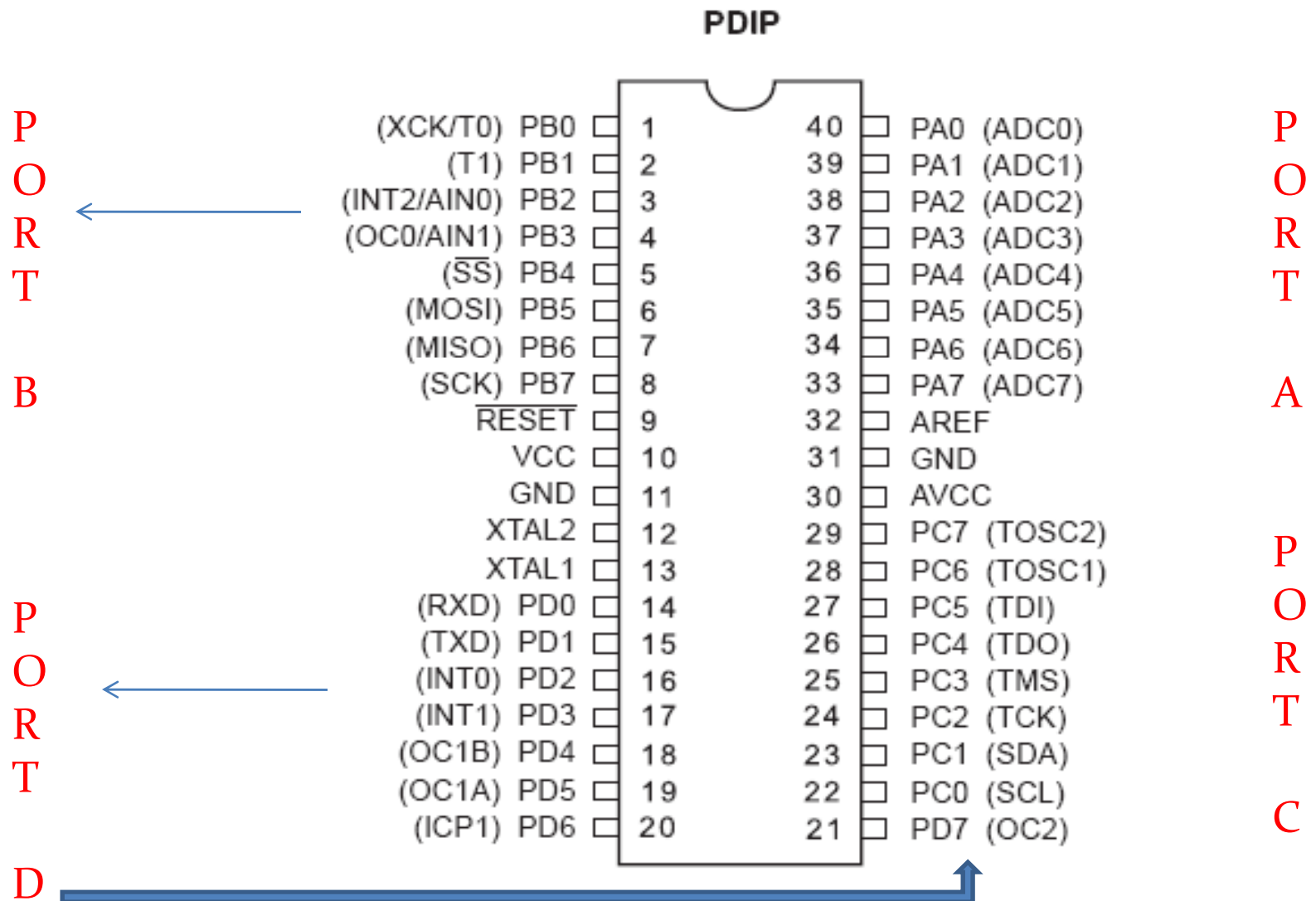
Step 8:Check all the connections, VCC supply

Step9: Burn your program into microcontroller

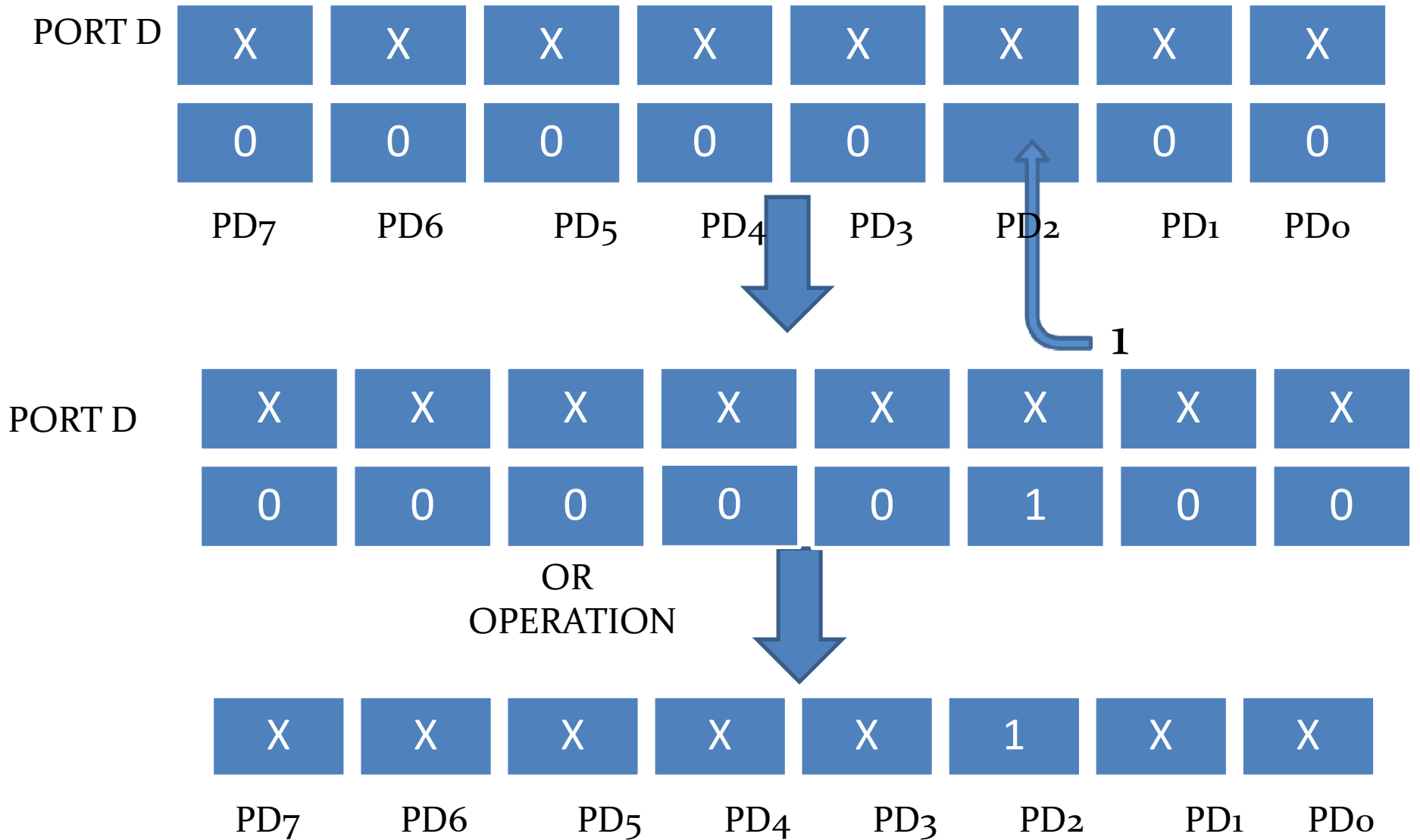
Step10:Remove Programmer

Microcontroller

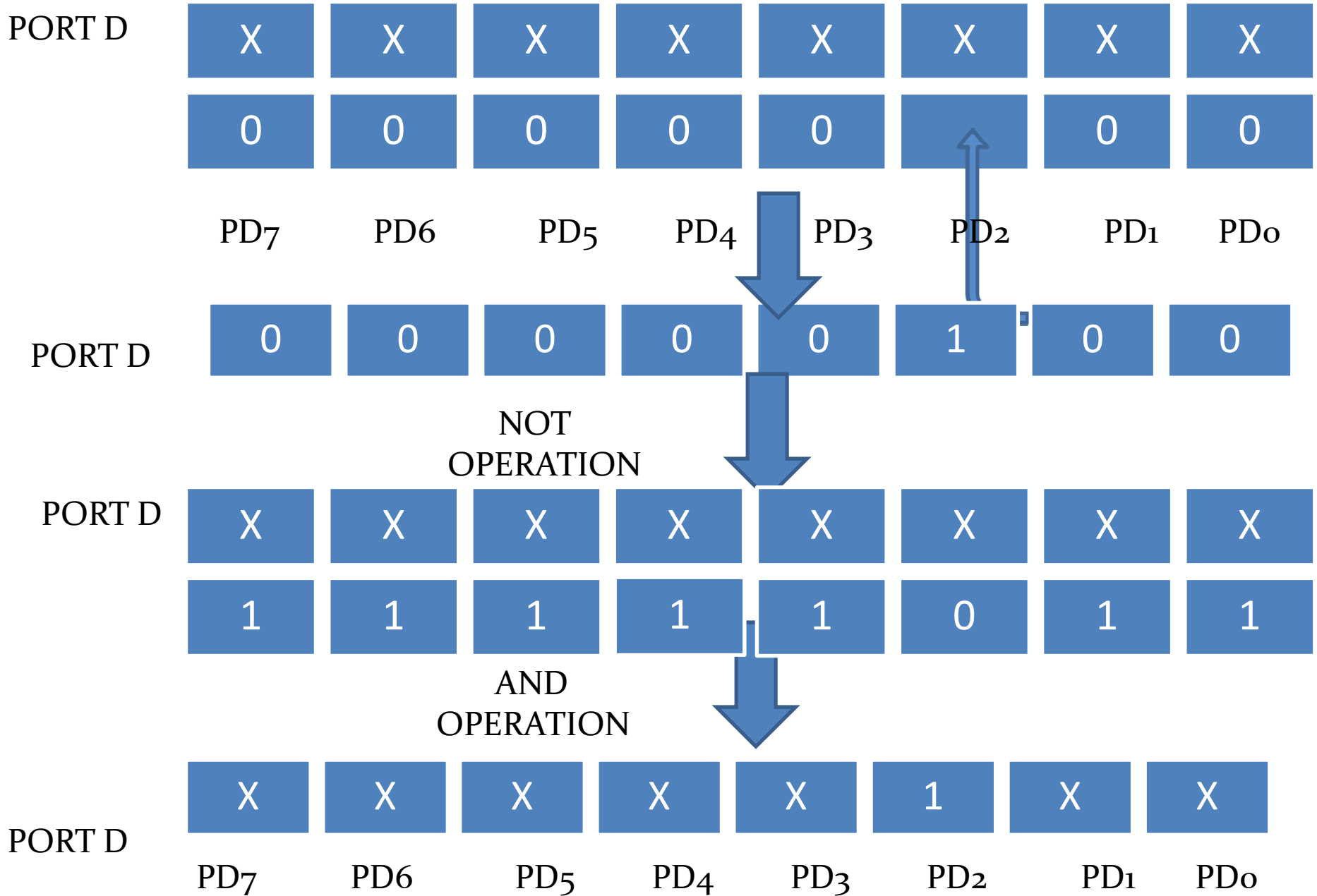
Pinout ATmega16



`PORTD |= 1<<2; //` a command which makes pd2 high, replace 2 with 3 ,then pd3 will be high.



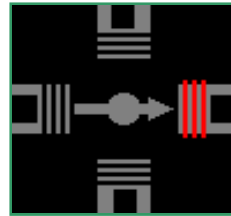
PORTD &= ~(1<<2);



GOOD PROGRAMMING TIPS

- ✓ We define a MACRO for making the pins high or low.
- ✓ MACROS can simplify the program and make it easier to be written.
- ✓ sbi – (set bit) – to make the pin low.
`#define sbi(ADDRESS,BIT) (ADDRESS |= (1<<BIT))`
- ✓ Cbi - (clear bit) – to make the pin high.
`#define cbi(ADDRESS,BIT) (ADDRESS &= ~(1<<BIT))`

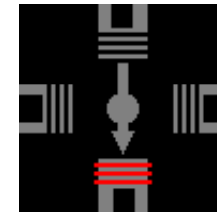
0010



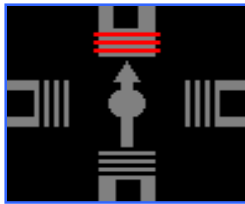
4



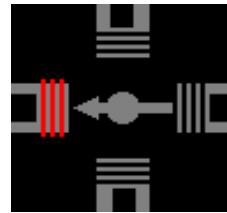
2



0100



0001

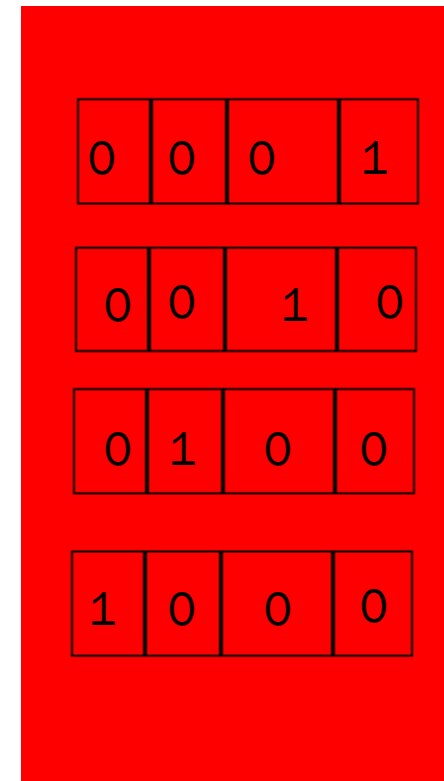


1000

DRIVING A STEPPER

```
Char a[]={1,2,4,8};  
int i=0;
```

```
int main()  
{  
  
PORTA=0xFF;  
  
While ()  
    {  
  
        PORTA=a[i%4];  
        i++;  
        delay_ms(100);  
  
    }  
}
```



TAKE INPUT THROUGH THREE SENSOR AND RUN ACCORDINGLY

- × If three has been used then there will be eight combination
- × 111=0x7,110=0x6,101=0x5, 100=4, 011=3, 010=0x2,001=0x1, 000=0x0
- × So do according to sensors output
- × Read pin as
PINA assuming sensors are connected on PORTA

e.g.

```
if(PINA==0x6)
{
-----
do whatever you want
-----
}
```

- TRY TO DISCUSS CERTAIN PROBLEMS WHICH OUR TEAM HAD FACED WHILE MAKING AN AUTONOMOUS ROBOT.

PROBLEMS

- 1) PROGRAMMING A MICROCONTROLLER
- 2) UNDERSTANDING ITS DATASHEET
- 3) LACK OF RESOURCES
- 4) POWER PROBLEMS
- 5) CALLIBERATION AND TESTING
- 6) OTHER SMALL PROBLEMS -DESIGNING

PROGRAMMING A MICROCONTROLLER

1. Flash Programmable memory
2. In system Programmable
3. Done with C code as told

4. How to put the program in a microcontroller???

CIRCUIT I USED



Problems while programming

1. You connected reset pin to external Vcc or gnd (disconnect it)
2. External power supply may be off
3. Parallel port cable may be not fitted properly.
4. See whether ur parallel port is working or not by using LPT.exe
5. See for short-ckt between pins
6. See for any loose connection in wires using **Continuity tester** of multimeter
- 7.If you are using soldered wires then see for any short circuit or loose connections in solderings
- 8.Problems with the breadboard
- 9.Execute install_giveio.bat for acessing ports

POWER PROBLEMS

- The source may not be able to supply enough current for the whole robot to work.
- SOLUTION :- proper choices of voltage regulator
- Proper placement of sensors and motor
- Read Datasheet

Datasheet

- Try to give time to the data sheet
- Try to screen out useless materials
- Note down the specific registers which needs to be set
- Try to look at some examples ,could be found on net
- Also be under the maximum limits specified

Testing and calliberation

- Best friend – Datasheet
- Problems like – JTAG problem on PORTC on ATEMEGA
- ADC testing problem !
- Try to look at alternating solutions like using potentiometer, multimeter , using LED's to test
- Never exceed voltage above 5V to MCU

Good solution -testing

- Using JTAG – gives all required information regarding internal details like adc, pins status while communicating with the serial port.



GOOD TIPS

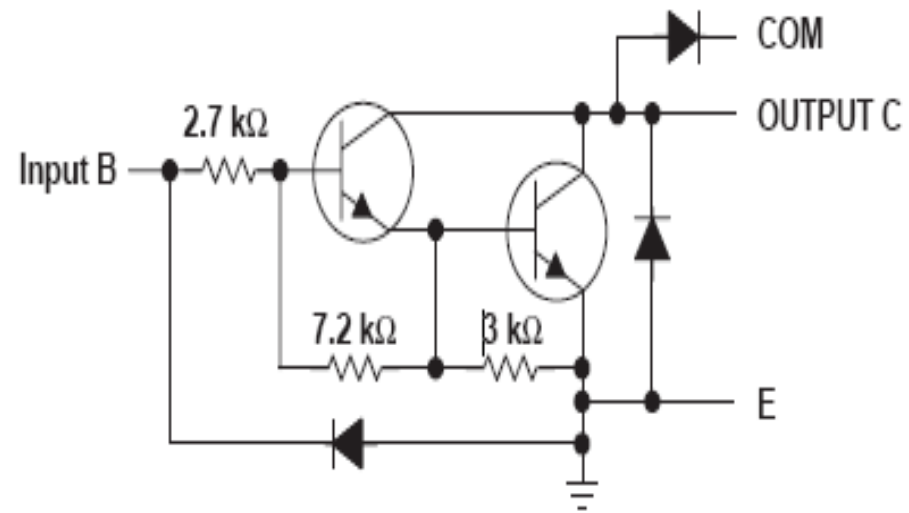
- ✘ Try to use different set of batteries and power circuits for both sensors and motor because both draw a large current and can damage components
- ✘ Before designing the pcb do refer to datasheet otherwise you may later face a problem e.g PORTC
- ✘ Try to check each and every component of the breadboard first.

links

- Roboticsindia.net
- Avrfreaks.com
- Triindia.co.in
- www.8052.com

- Try to program the MCU on the pcb itself.
- Try to keep all components symmetrical
- Try to be as simple as possible
- Try to use all the important componets specified in a datasheet

Single unit of uln2803



Interfacing circuit

