Basic I/O Interfacing on Firebird V

e-Yantra Team

Embedded Real-Time Systems (ERTS) Lab
Indian Institute of Technology, Bombay

Agenda for Discussion

1. Input-Output Ports in ATmega 2560
   - Overview of Ports
   - Ports in ATmega 2560
   - Accessing Ports
   - Examples
2. Write Your First Embedded C Program
   - Buzzer Interfacing
   - Programming Tools
   - C code
3. Assignment

What are Ports?

What are Ports?

- Junctions where peripheral devices are connected.

What are Ports?

- Junctions where peripheral devices are connected.
- Peripheral devices can be:

What are Ports?

- Junctions where peripheral devices are connected.
- Peripheral devices can be:
  1. Input Device:

     Example: Switch, Sensors, etc...

What are Ports?

- Junctions where peripheral devices are connected.
- Peripheral devices can be:
  1. Input Device:

     Example: Switch, Sensors, etc...
  2. Output Device:

     Example: Buzzer, LCD, Motors, LED, etc...

Ports in ATmega 2560

Ports in ATmega 2560

- ATmega 2560 is a 100 pin micro-controller.

Ports in ATmega 2560

- ATmega 2560 is a 100 pin micro-controller.
- 86 pins can be used as Input/Output pins.

Ports in ATmega 2560

- ATmega 2560 is a 100 pin micro-controller.
- 86 pins can be used as Input/Output pins.
- Pins are grouped together and are called as Port.

Ports in ATmega 2560

- ATmega 2560 is a 100 pin micro-controller.
- 86 pins can be used as Input/Output pins.
- Pins are grouped together and are called as Port.

1. ATmega 2560 has ten 8-bit Ports

   Port x; x = A to F and H, J, K, L

Ports in ATmega 2560

- ATmega 2560 is a 100 pin micro-controller.
- 86 pins can be used as Input/Output pins.
- Pins are grouped together and are called as Port.
  1. ATmega 2560 has ten 8-bit Ports

     Port x; x = A to F and H, J, K, L
  2. ATmega 2560 has one 6-bit Port

     Port G;

Ports in ATmega 2560

- ATmega 2560 is a 100 pin micro-controller.
- 86 pins can be used as Input/Output pins.
- Pins are grouped together and are called as Port.
  1. ATmega 2560 has ten 8-bit Ports

     Port x; x = A to F and H, J, K, L
  2. ATmega 2560 has one 6-bit Port

     Port G;
- All Port pins can be individually configured as Input/Output.

Accessing Ports

Accessing Ports

Each Port has three associated registers with it:

Accessing Ports

Each Port has three associated registers with it:

1. DDRx x = A to H and J, K, L

Accessing Ports

Each Port has three associated registers with it:

1. DDRx x = A to H and J, K, L
2. PORTx x = A to H and J, K, L

Accessing Ports

Each Port has three associated registers with it:

1. DDRx x = A to H and J, K, L
2. PORTx x = A to H and J, K, L
3. PINx x = A to H and J, K, L

Understanding DDRx Register

Understanding DDRx Register

- Data Direction Register

Understanding DDRx Register

- Data Direction Register
- Purpose: To define Port pins as Input/Output

Understanding DDRx Register

- Data Direction Register
- Purpose: To define Port pins as Input/Output
  a. DDRx bit = 0 → Portx pin is defined as Input.

Understanding DDRx Register

- Data Direction Register
- Purpose: To define Port pins as Input/Output
  a. DDRx bit = 0 → Portx pin is defined as Input.
  b. DDRx bit = 1 → Portx pin is defined as Output.

Understanding DDRx Register

- Data Direction Register
- Purpose: To define Port pins as Input/Output
  a. DDRx bit = 0 → Portx pin is defined as Input.
  b. DDRx bit = 1 → Portx pin is defined as Output.
- Example: For Port B, make lower nibble as Input and upper nibble as Output.

Understanding DDRx Register

- Data Direction Register
- Purpose: To define Port pins as Input/Output
  a. DDRx bit = 0 → Portx pin is defined as Input.
  b. DDRx bit = 1 → Portx pin is defined as Output.
- Example: For Port B, make lower nibble as Input and upper nibble as Output.

Understanding DDRx Register

- Data Direction Register
- Purpose: To define Port pins as Input/Output
  a. DDRx bit = 0 → Portx pin is defined as Input.
  b. DDRx bit = 1 → Portx pin is defined as Output.
- Example: For Port B, make lower nibble as Input and upper nibble as Output.

DDRB =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	0	0	0	0

Understanding DDRx Register

- Data Direction Register
- Purpose: To define Port pins as Input/Output
  a. DDRx bit = 0 → Portx pin is defined as Input.
  b. DDRx bit = 1 → Portx pin is defined as Output.
- Example: For Port B, make lower nibble as Input and upper nibble as Output.

DDRB =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	0	0	0	0

DDRB = 0xF0

Understanding PINx Register

Understanding PINx Register

1. Purpose: To read data present on Port x pins.

Understanding PINx Register

1. Purpose: To read data present on Port x pins.
2. Save the value of register in a variable.

Understanding PINx Register

1. Purpose: To read data present on Port x pins.
2. Save the value of register in a variable.
3. Example:

   Read data from Port C

Understanding PINx Register

1. Purpose: To read data present on Port x pins.
2. Save the value of register in a variable.
3. Example:

   Read data from Port C

Understanding PINx Register

1. Purpose: To read data present on Port x pins.
2. Save the value of register in a variable.
3. Example:

   Read data from Port C

PortC =

P7	P6	P5	P4	P3	P2	P1	P0
1	1	1	1	0	0	0	0

Understanding PINx Register

1. Purpose: To read data present on Port x pins.
2. Save the value of register in a variable.
3. Example:

   Read data from Port C

PortC =

P7	P6	P5	P4	P3	P2	P1	P0
1	1	1	1	0	0	0	0

x = PINC

x = 0xF0

Understanding PORTx Register

Understanding PORTx Register

Case 1: When Port x is defined as Output

Understanding PORTx Register

Case 1: When Port x is defined as Output

1. Purpose: Send data on Port x pins

Understanding PORTx Register

Case 1: When Port x is defined as Output

1. Purpose: Send data on Port x pins
2. Example:

Understanding PORTx Register

Case 1: When Port x is defined as Output

1. Purpose: Send data on Port x pins
2. Example:

Understanding PORTx Register

Case 1: When Port x is defined as Output

1. Purpose: Send data on Port x pins
2. Example:

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

Understanding PORTx Register

Case 1: When Port x is defined as Output

1. Purpose: Send data on Port x pins
2. Example:

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

DDRA = 0xFF

Understanding PORTx Register

Case 1: When Port x is defined as Output

1. Purpose: Send data on Port x pins
2. Example:

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

DDRA = 0xFF

PORTA = 0xFF

Understanding PORTx Register

Understanding PORTx Register

Case 2: When Port x is defined as Input

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor

Understanding PORTx Register

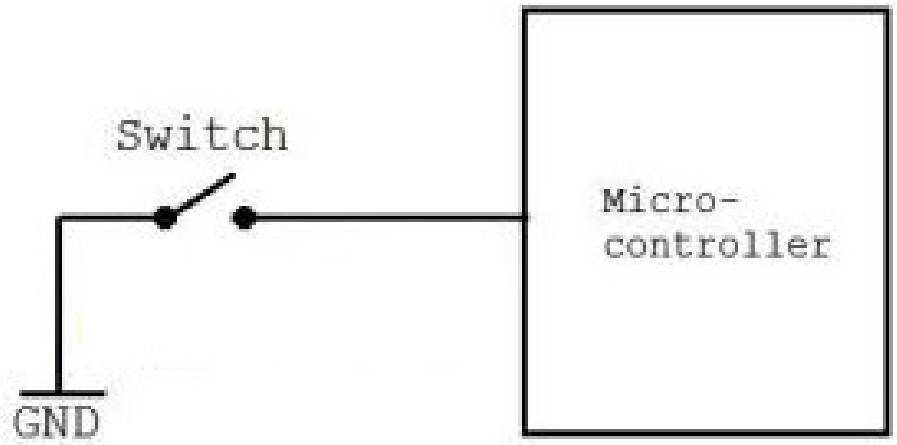
Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor

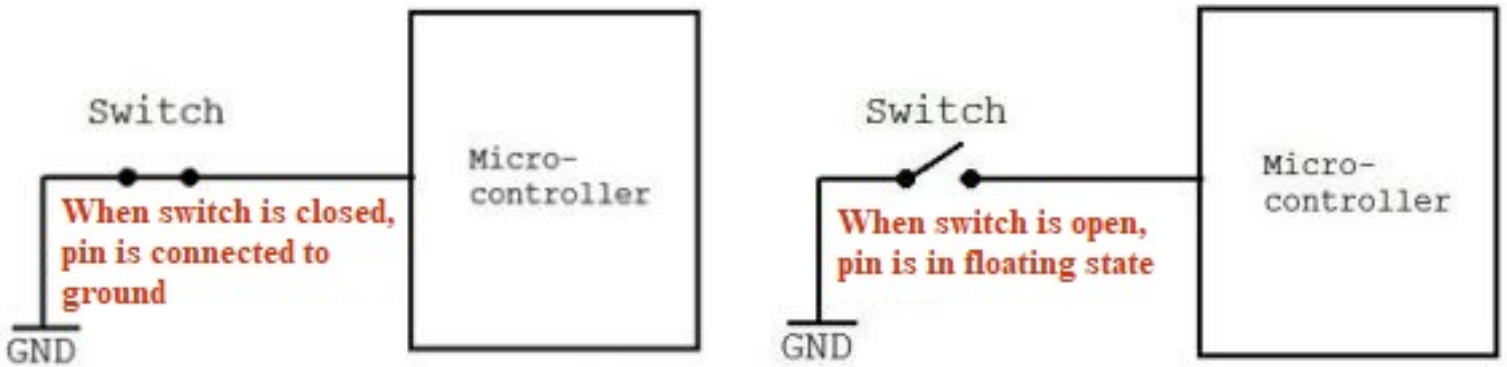


Understanding PORTx Register

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor

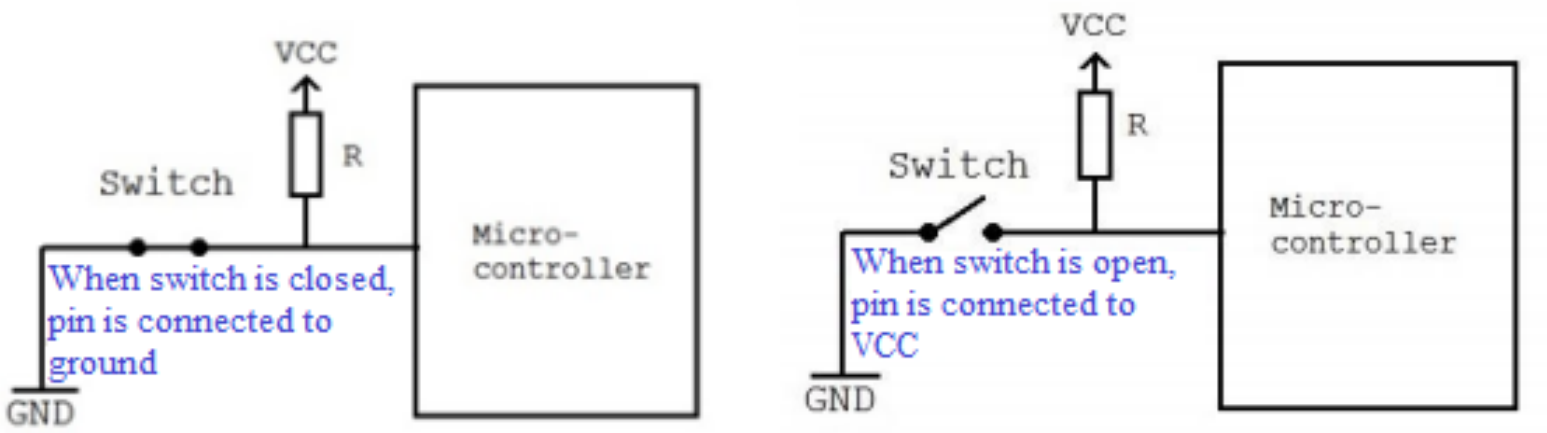


Understanding PORTx Register

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor



Understanding PORTx Register

Understanding PORTx Register

Case 2: When Port x is defined as Input

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor
   a. PORTx bit = 1 → Pull up is activated on Portx pin.

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor
   a. PORTx bit = 1 → Pull up is activated on Portx pin.
   b. PORTx bit = 0 → Pull up is deactivated on Portx pin.

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor
   a. PORTx bit $= 1 \rightarrow$ Pull up is activated on Portx pin.
   b. PORTx bit $= 0 \rightarrow$ Pull up is deactivated on Portx pin.
2. Example:

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor
   a. PORTx bit = 1 → Pull up is activated on Portx pin.
   b. PORTx bit = 0 → Pull up is deactivated on Portx pin.
2. Example:

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor
   a. PORTx bit = 1 → Pull up is activated on Portx pin.
   b. PORTx bit = 0 → Pull up is deactivated on Portx pin.
2. Example:

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor
   a. PORTx bit = 1 → Pull up is activated on Portx pin.
   b. PORTx bit = 0 → Pull up is deactivated on Portx pin.
2. Example:

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor
   a. PORTx bit = 1 → Pull up is activated on Portx pin.
   b. PORTx bit = 0 → Pull up is deactivated on Portx pin.
2. Example:

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

PORTA = 0xFF

Understanding PORTx Register

Case 2: When Port x is defined as Input

1. Purpose: Activate/deactivate Pull-up resistor
   a. PORTx bit = 1 → Pull up is activated on Portx pin.
   b. PORTx bit = 0 → Pull up is deactivated on Portx pin.
2. Example:

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

PORTA = 0xFF

Pull-Up is activated for all Pins of PortA.

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

   DDRD =

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

DDRD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

DDRD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

DDRD = 0xFF

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

DDRD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

DDRD = 0xFF

2. Step 2: Put data on the Port D

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

DDRD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

DDRD = 0xFF

2. Step 2: Put data on the Port D

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

DDRD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

DDRD = 0xFF

2. Step 2: Put data on the Port D

PORTD =

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

DDRD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

DDRD = 0xFF

2. Step 2: Put data on the Port D

PORTD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	0	1	0	1	0	1

Examples

- Example 1: Make PortD as output port and send hex value 'D5'.

1. Step 1: Make Port D as Output port

DDRD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

DDRD = 0xFF

2. Step 2: Put data on the Port D

PORTD =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	0	1	0	1	0	1

PORTD = 0xD5

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

   DDRA =

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

2. Step 2: To activate Pull-up Resistor send data on Port A

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

2. Step 2: To activate Pull-up Resistor send data on Port A

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

2. Step 2: To activate Pull-up Resistor send data on Port A

PORTA =

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

2. Step 2: To activate Pull-up Resistor send data on Port A

PORTA =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

Examples (Cont..)

- Example 2: Make PortA input port with pull-up activated on all pins

1. Step 1: Make Port A as Input port

DDRA =

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

DDRA = 0x00

2. Step 2: To activate Pull-up Resistor send data on Port A

PORTA =

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1

PORTA = 0xFF

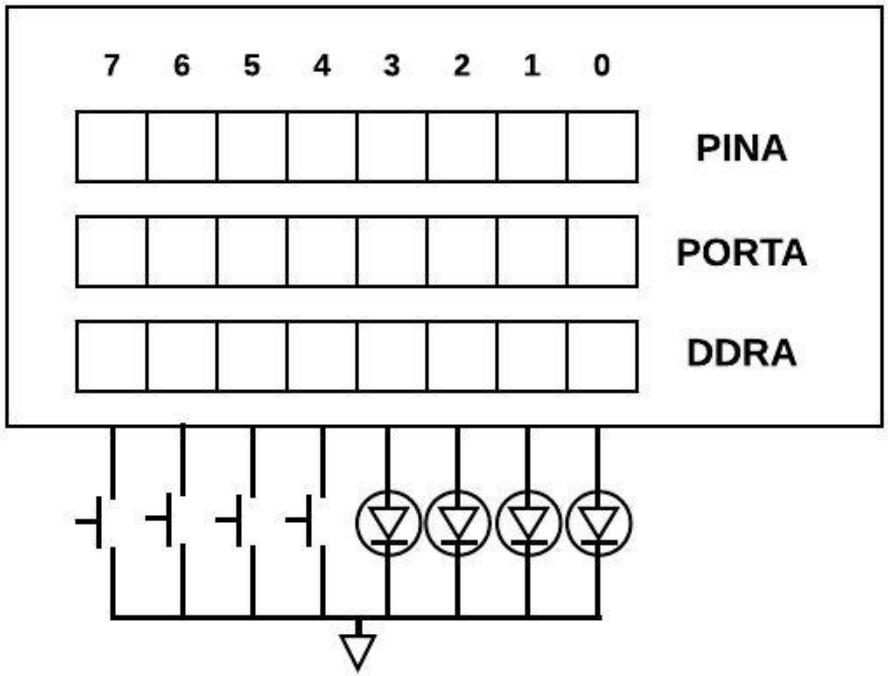
Examples

Examples

- Example: Connect LEDs to lower nibble and Switches to upper nibble of PortA. Turn ON alternate LEDs (0 and 2) and activate pull up for all Switches. Read data using PIN register. What will be the content of PINA register, if only Switch at pin 5 is pressed?

Examples

- Example: Connect LEDs to lower nibble and Switches to upper nibble of PortA. Turn ON alternate LEDs (0 and 2) and activate pull up for all Switches. Read data using PIN register. What will be the content of PINA register, if only Switch at pin 5 is pressed?



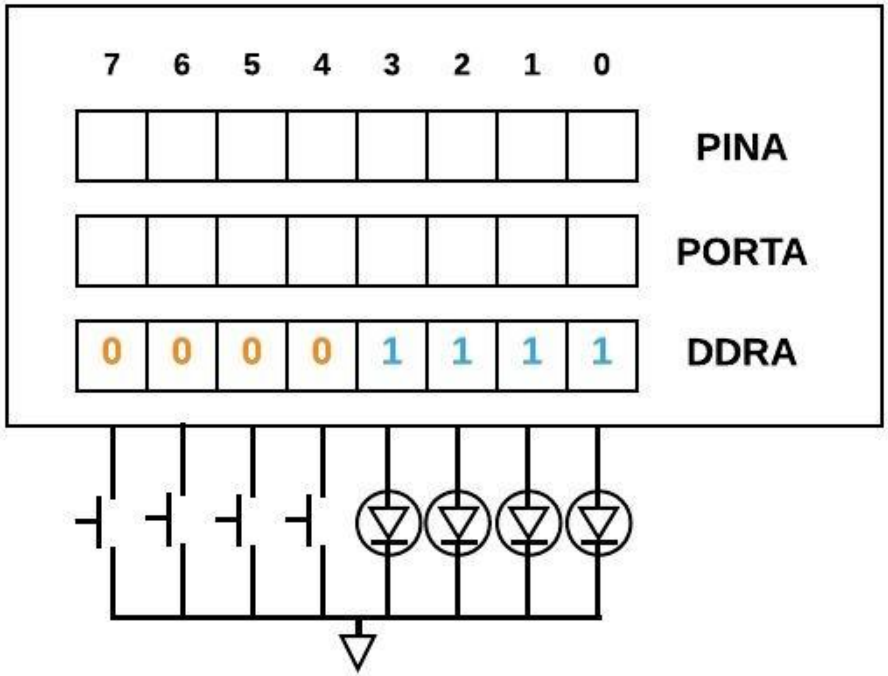
Examples

Examples

- Step 1: Make upper nibble as Input and lower nibble as Output.

Examples

- Step 1: Make upper nibble as Input and lower nibble as Output.



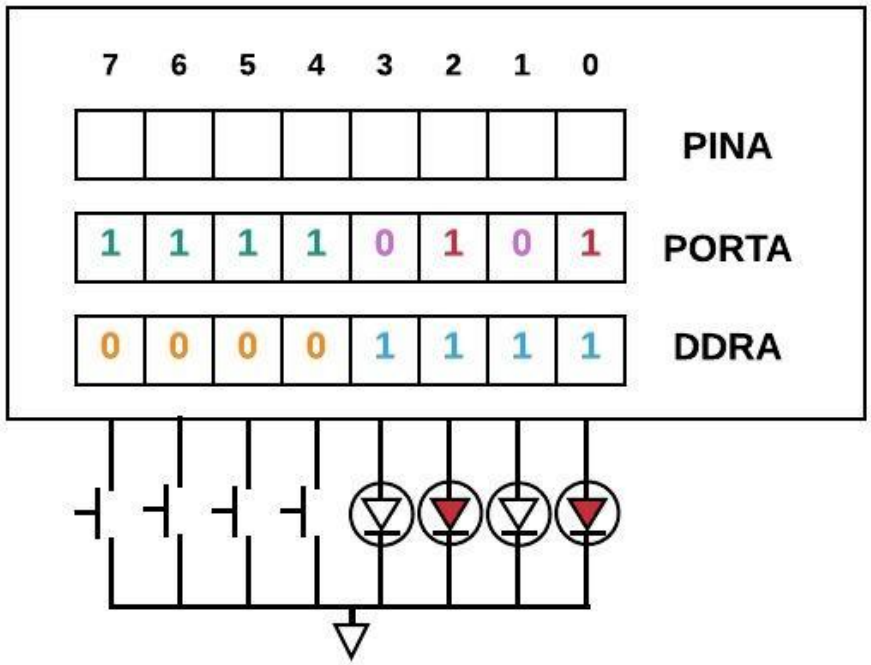
Examples

Examples

- Step 2: Turn ON alternate LEDs (0 and 2) and activate pull up for Switches.

Examples

- Step 2: Turn ON alternate LEDs (0 and 2) and activate pull up for Switches.



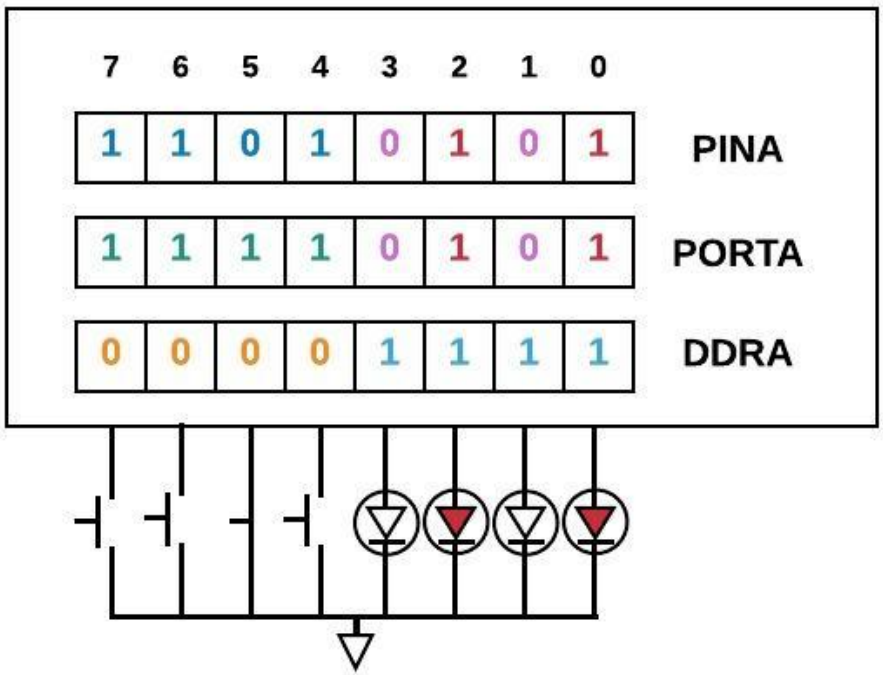
Examples

Examples

- Step 3: Read data from PINA. On lower nibble we will get the same data and on upper nibble depending on Switch position, data will change.

Examples

- Step 3: Read data from PINA. On lower nibble we will get the same data and on upper nibble depending on Switch position, data will change.



Buzzer Interfacing in Firebird V

Buzzer Interfacing in Firebird V

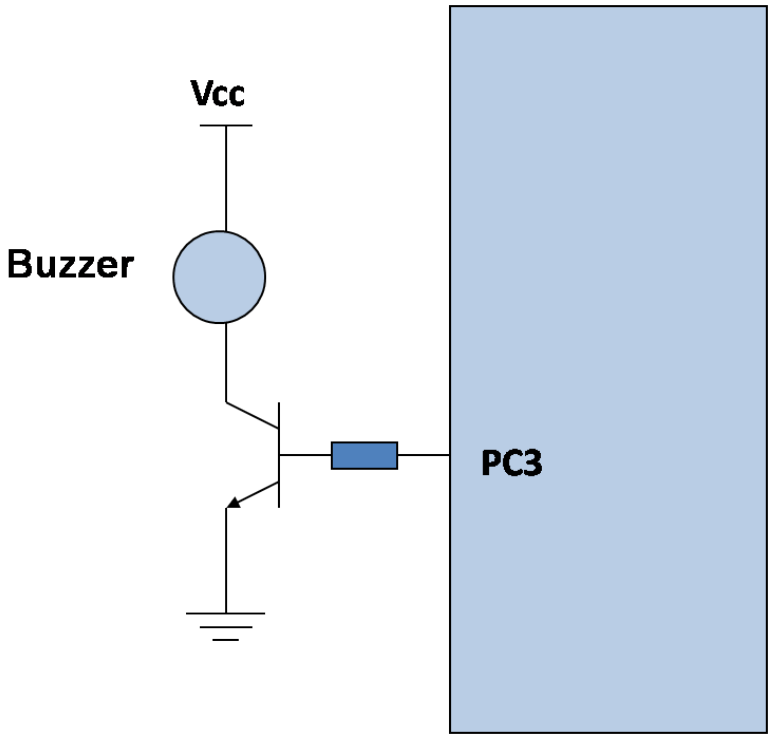
1. Buzzer is connected to Port C pin 3

Buzzer Interfacing in Firebird V

1. Buzzer is connected to Port C pin 3

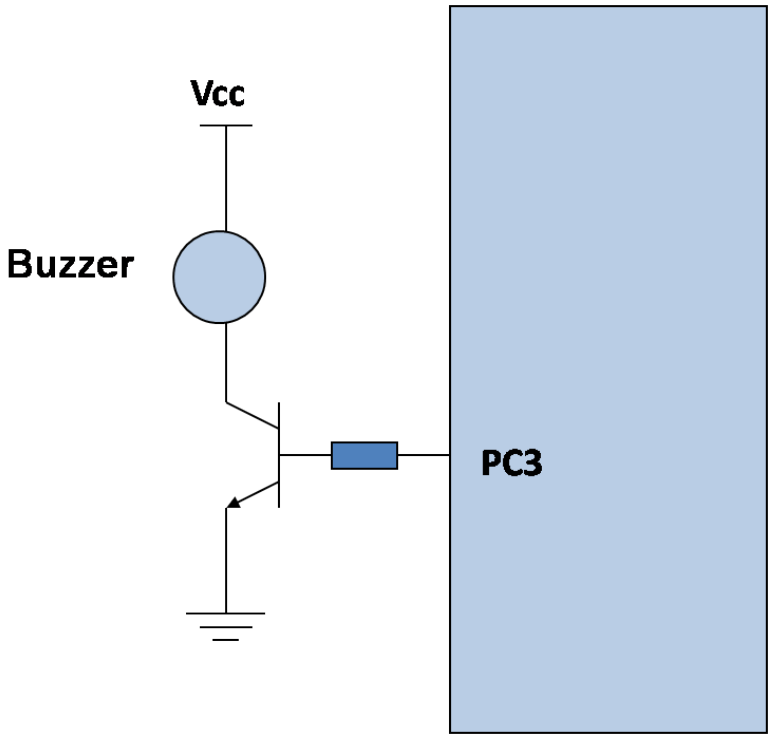
Buzzer Interfacing in Firebird V

1. Buzzer is connected to Port C pin 3



Buzzer Interfacing in Firebird V

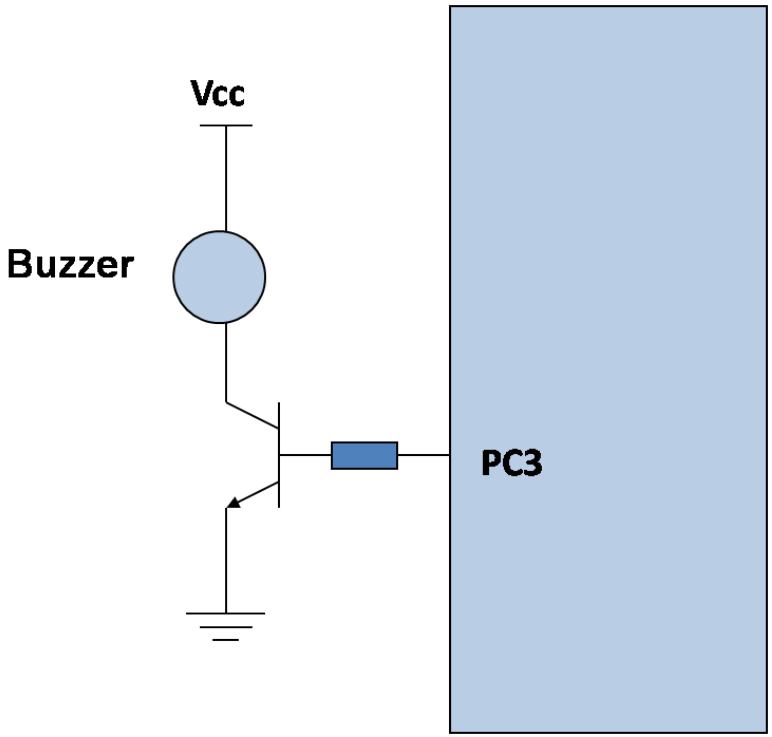
1. Buzzer is connected to Port C pin 3



2. To turn ON buzzer:

Buzzer Interfacing in Firebird V

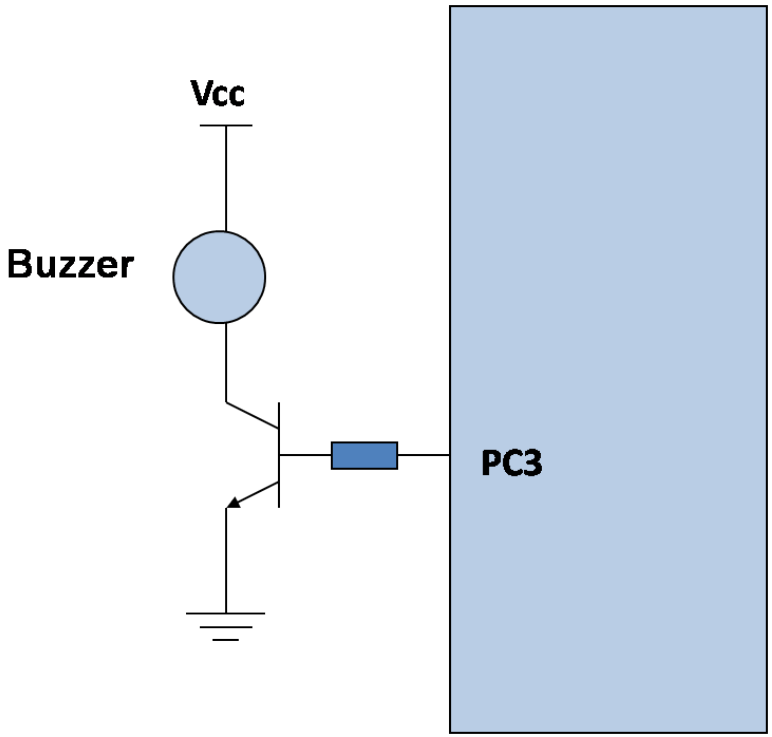
1. Buzzer is connected to Port C pin 3



2. To turn ON buzzer:

Buzzer Interfacing in Firebird V

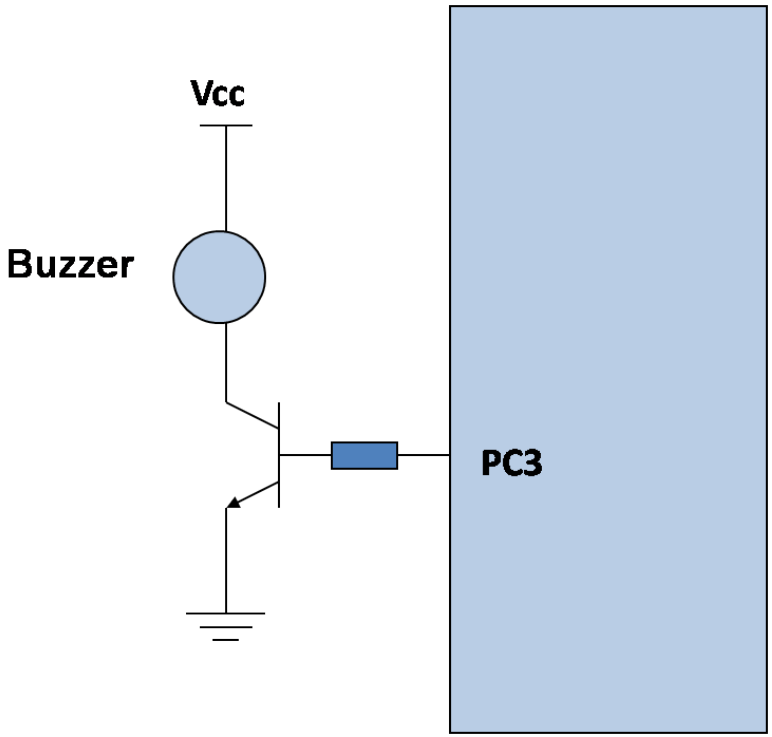
1. Buzzer is connected to Port C pin 3



2. To turn ON buzzer: send logic HIGH on pin 3 of Port C

Buzzer Interfacing in Firebird V

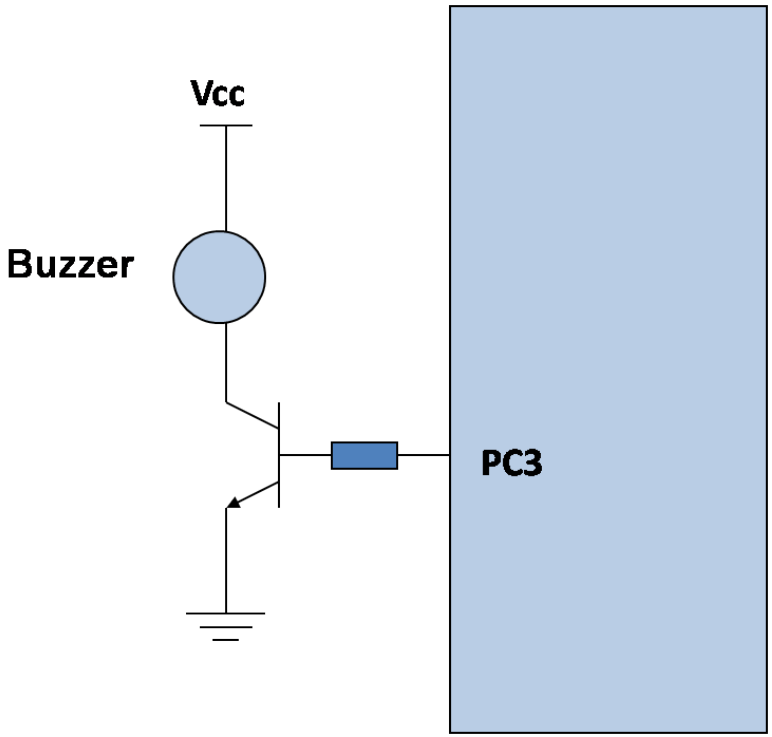
1. Buzzer is connected to Port C pin 3



2. To turn ON buzzer: send logic HIGH on pin 3 of Port C
3. To turn OFF buzzer:

Buzzer Interfacing in Firebird V

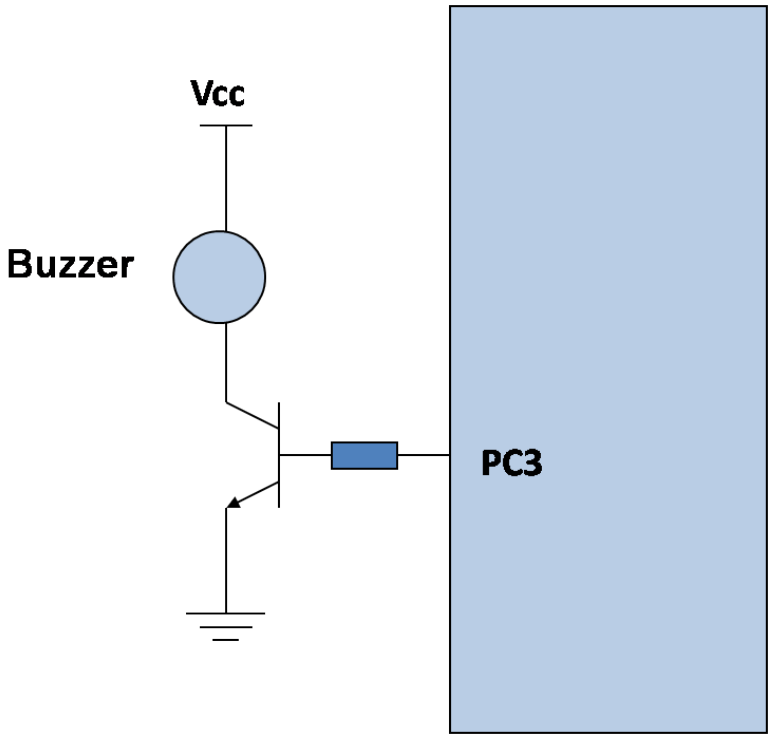
1. Buzzer is connected to Port C pin 3



2. To turn ON buzzer: send logic HIGH on pin 3 of Port C
3. To turn OFF buzzer:

Buzzer Interfacing in Firebird V

1. Buzzer is connected to Port C pin 3



2. To turn ON buzzer: send logic HIGH on pin 3 of Port C
3. To turn OFF buzzer: send logic LOW on pin 3 of Port C

Buzzer Program

Buzzer Program

1. Configure PC.3 pin as Output.

Buzzer Program

1. Configure PC.3 pin as Output.

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC =

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

   PORTC =

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

   PORTC = 0x08; // 0000 1000

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

   PORTC = 0x08; // 0000 1000

3. To turn OFF the buzzer set PC.3 output LOW

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

   PORTC = 0x08; // 0000 1000

3. To turn OFF the buzzer set PC.3 output LOW

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

   PORTC = 0x08; // 0000 1000

3. To turn OFF the buzzer set PC.3 output LOW

   PORTC =

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

   PORTC = 0x08; // 0000 1000

3. To turn OFF the buzzer set PC.3 output LOW

   PORTC = 0x00; // 0000 0000

Buzzer Program

1. Configure PC.3 pin as Output.

   DDRC = 0x08; // 0000 1000

2. To turn ON the buzzer set PC.3 output HIGH

   PORTC = 0x08; // 0000 1000

3. To turn OFF the buzzer set PC.3 output LOW

   PORTC = 0x00; // 0000 0000

AVR Programming Tools

AVR Programming Tools

1. Software Required

AVR Programming Tools

1. Software Required

AVR Programming Tools

1. Software Required

   ATMEL STUDIO 6

AVR Programming Tools

1. Software Required

   ATMEL STUDIO 6

- Integrated Development Environment (IDE)
- Supports Developing and Debugging of AVR and ARM based microcontroller application
- Download Link: `www.atmel.com/Microsite/atmel_studio6/default.aspx`

AVR Programming Tools

1. Software Required

   ATMEL STUDIO 6

   - Integrated Development Environment (IDE)
   - Supports Developing and Debugging of AVR and ARM based microcontroller application
   - Download Link: `www.atmel.com/Microsite/atmel_studio6/default.aspx`

2. Techniques to load the program

AVR Programming Tools

1. Software Required

   ATMEL STUDIO 6

   - Integrated Development Environment (IDE)
   - Supports Developing and Debugging of AVR and ARM based microcontroller application
   - Download Link: `www.atmel.com/Microsite/atmel_studio6/default.aspx`

2. Techniques to load the program

AVR Programming Tools

1. Software Required

   ATMEL STUDIO 6

   - Integrated Development Environment (IDE)
   - Supports Developing and Debugging of AVR and ARM based microcontroller application
   - Download Link: `www.atmel.com/Microsite/atmel_studio6/default.aspx`

2. Techniques to load the program

   - hex file can be loaded into microcontroller using

AVR Programming Tools

1. Software Required

   ATMEL STUDIO 6

   - Integrated Development Environment (IDE)
   - Supports Developing and Debugging of AVR and ARM based microcontroller application
   - Download Link: `www.atmel.com/Microsite/atmel_studio6/default.aspx`

2. Techniques to load the program

   - hex file can be loaded into microcontroller using
     a. Bootloader/USB

AVR Programming Tools

1. Software Required

   ATMEL STUDIO 6

   - Integrated Development Environment (IDE)
   - Supports Developing and Debugging of AVR and ARM based microcontroller application
   - Download Link: `www.atmel.com/Microsite/atmel_studio6/default.aspx`

2. Techniques to load the program

   - hex file can be loaded into microcontroller using
     a. Bootloader/USB

AVR Programming Tools

1. Software Required

   ATMEL STUDIO 6

   - Integrated Development Environment (IDE)
   - Supports Developing and Debugging of AVR and ARM based microcontroller application
   - Download Link: `www.atmel.com/Microsite/atmel_studio6/default.aspx`

2. Techniques to load the program

   - hex file can be loaded into microcontroller using
     a. Bootloader/USB
     b. AVR Programmers viz. AVR MKII, AVRDude, Pony-Programmer, etc..

Syntax for C-Program

Syntax for C-Program

#include

Syntax for C-Program

#include

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

Syntax for C-Program

#include

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

Pin Configuration

Syntax for C-Program

#include

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

Pin Configuration

```
void buzzer_pin_config (void)
{
     DDRC  =
     PORTC =
}
```

Syntax for C-Program

Syntax for C-Program

Main-Program

Syntax for C-Program

Main-Program

```
int main (void)
{
   buzzer_pin_config();
   while(1)
    {
      buzzer_on();
      _delay_ms(1000);
      buzzer_off();
      _delay_ms(1000);
    }
}
```

Syntax for C-Program

Main-Program

```
int main (void)
{
   buzzer_pin_config();
   while(1)
    {
      buzzer_on();
      _delay_ms(1000);
      buzzer_off();
      _delay_ms(1000);
    }
}
```

Functions

Syntax for C-Program

Main-Program

```
int main (void)
{
   buzzer_pin_config();
   while(1)
    {
      buzzer_on();
      _delay_ms(1000);
      buzzer_off();
      _delay_ms(1000);
    }
}
```

Functions

```
void buzzer_on (void)
{
     PORTC = ;
}
```

Syntax for C-Program

Main-Program

```
int main (void)
{
   buzzer_pin_config();
   while(1)
    {
      buzzer_on();
      _delay_ms(1000);
      buzzer_off();
      _delay_ms(1000);
    }
}
```

Functions

```
void buzzer_on (void)
{
     PORTC = ;
}
```

```
void buzzer_off (void)
{
     PORTC = ;
}
```

Input IO device Interfacing Task

Input IO device Interfacing Task

- Boot (Interrupt) switch is connected to **Port E pin 7 (PE.7)**

Input IO device Interfacing Task

- Boot (Interrupt) switch is connected to **Port E pin 7 (PE.7)**
- Task is to switch ON the buzzer as long as the switch is pressed; buzzer should turn OFF when switch is released.

Thank You!

Post your queries on: support@e-yantra.org