GLCD 128x64 interfacing with 8051 Microcontroller

GLCD 128x64

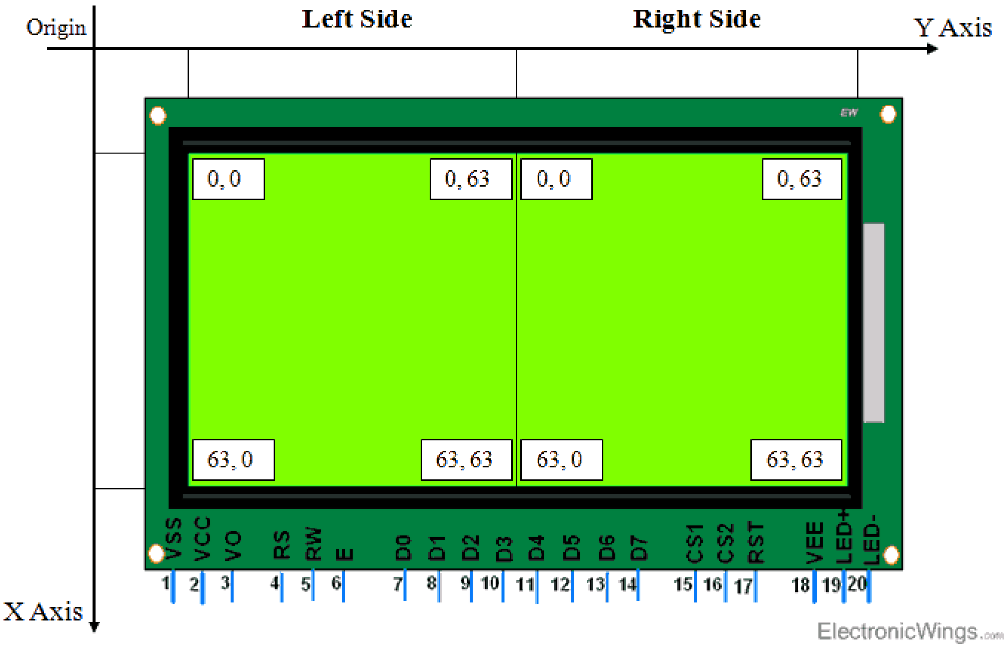
GLCD is a display device that can be used in embedded systems for displaying data and/or images/custom characters.

- Basically, a 128x64 Graphical LCD is a matrix of pixels.
- Each pixel is accessed by its X and Y address.
- We can simply visualize any pixel by making its value HIGH (1).

Hence, we can make any graphical design pixel by pixel using GLCD.

To get familiar with GLCD pins and their functions refer to **GLCD 128x64** (http://electronicwings.com/sensors-modules/glcd-128x64).

GLCD 128x64 Display Pinout and Pixels Structure



GLCD 128x64 Display Pinout and Pixels

Programming GLCD

Let's program the AT89S51 microcontroller to print text character on GLCD JHD12864E.

Initialization

To initialize the display, we need to do below steps,

- Send Display OFF command i.e. 0x3E
- Send Y address e.g. here 0x40 (Start address).

- Send X address (Page) e.g. here 0xB8 (Page0).
- Send Z address (Start line) e.g. here 0xC0 (from the 0th line).
- Now send Display ON command i.e. 0x3F.

GLCD_Init function

Input arguments: It has no input arguments.

Return type: It does not return any data type.

```
void GLCD_Init()         /* GLCD initialize function */
{
        CS1 = 1; CS2 = 1;        /* Select left & right half of display */
        RST = 1;                 /* Keep reset pin high */
        delay(20);
        GLCD_Command(0x3E);      /* Display OFF */
        GLCD_Command(0x40);      /* Set Y address (column=0) */
        GLCD_Command(0xB8);      /* Set x address (page=0) */
        GLCD_Command(0xC0);      /* Set z address (start line=0) */
        GLCD_Command(0x3F);      /* Display ON */
}
```

Command Write

To write command do the below steps

- Send command on data pins.
- Make RS = 0 (Command Register) and RW = 0 (Write Operation).
- Make High to Low transition on Enable pin of min. 1 us period.

GLCD_Command function

Input arguments: It has an input argument of Command.

Return type: It does not return any data type.

```
void GLCD_Command(char Command)  /* GLCD command function */
{
        Data_Port = Command;     /* Copy command on data pin */
        RS = 0;                  /* Make RS LOW to select command register */
        RW = 0;                  /* Make RW LOW to select write operation */
        E = 1;                   /* Make HIGH to LOW transition on Enable */
        _nop_ ();
        E = 0;
        _nop_ ();
}
```

Data Write

To write data do the below commands

- Send Data on data pins.
- Make RS = 1 (Data Register) and RW = 0 (Write Operation).
- Make High to Low transition on Enable pin of min 1 us period.

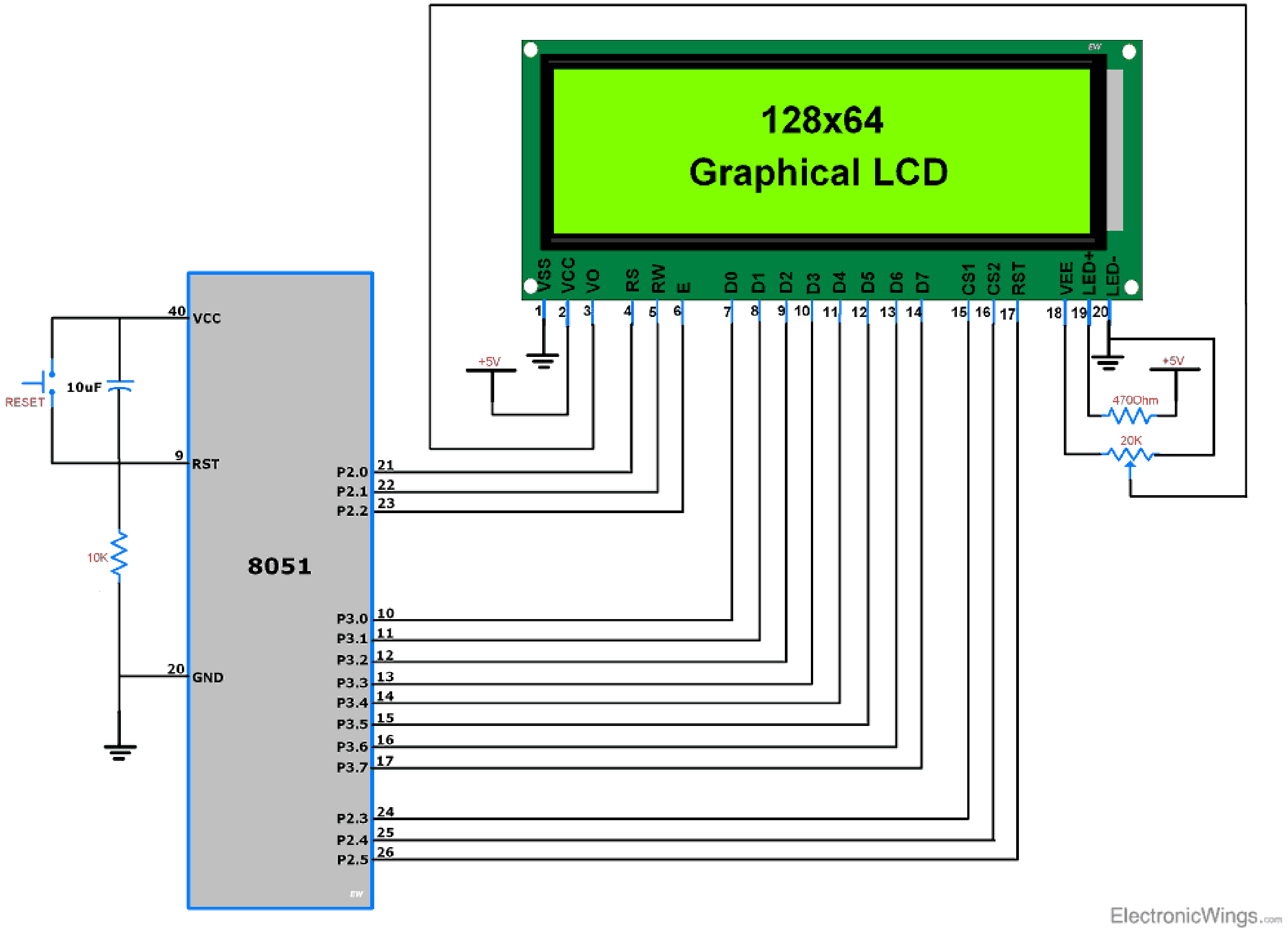
GLCD_Data function

Input arguments: It has input argument Data.

Return type: It does not return any data type.

```
void GLCD_Data(char Data)        /* GLCD data function */
{
        Data_Port = Data;          /* Copy data on data pin */
        RS = 1;                    /* Make RS HIGH to select data register */
        RW = 0;                    /* Make RW LOW to select write operation */
        E = 1;                     /* Make HIGH to LOW transition on Enable */
        _nop_ ();
        E = 0;
        _nop_ ();
}
```

GLCD 128x64 Pin Connection with 8051



GLCD 128x64 Interface with 8051

Code for Text Print GLCD 128x64 using 8051

```
/*
 *
 8051 GLCD 128x64 Text character font
 http://www.electronicwings.com
 *
 */

#include <reg51.h>          /* Include reg51 header file */
#include <intrins.h>        /* Include intrinsic header file */
#include "Font_Header.h"    /* Include font header file */

#define Data_Port P3        /* Define data port for GLCD */

sbit RS = P2^0;                     /* Set control bits pins */
sbit RW = P2^1;
sbit E = P2^2;
sbit CS1 = P2^3;
sbit CS2 = P2^4;
sbit RST = P2^5;


void delay(k)           /* Delay of msec with xtal = 11.0592MHz */
{
        int i,j;
        for (i=0;i<k;i++)
                for (j=0;j<112;j++);
}

void GLCD_Command(char Command) /* GLCD command function */
{
        Data_Port = Command;    /* Copy command on data pin */
        RS = 0;                 /* Make RS LOW to select command register */
        RW = 0;                 /* Make RW LOW to select write operation */
        E = 1;                  /* Make HIGH to LOW transition on Enable */
        _nop_ ();
        E = 0;
        _nop_ ();
}

void GLCD_Data(char Data)       /* GLCD data function */
{
        Data_Port = Data;       /* Copy data on data pin */
        RS = 1;                 /* Make RS HIGH to select data register */
        RW = 0;                 /* Make RW LOW to select write operation */
        E = 1;                  /* Make HIGH to LOW transition on Enable */
        _nop_ ();
        E = 0;
        _nop_ ();
}

void GLCD_Init()        /* GLCD initialize function */
{
        CS1 = 1; CS2 = 1;       /* Select left & right half of display */
        RST = 1;                /* Keep reset pin high */
        delay(20);
        GLCD_Command(0x3E);     /* Display OFF */
```

```
    GLCD_Command(0x40);      /* Set Y address (column=0) */
    GLCD_Command(0xB8);      /* Set x address (page=0) */
    GLCD_Command(0xC0);      /* Set z address (start line=0) */
    GLCD_Command(0x3F);      /* Display ON */
}

void GLCD_ClearAll()         /* GLCD all display clear function */
{
    int i,j;
    CS1 = 1; CS2 = 1;        /* Select left & right half of display */
    for(i=0;i<8;i++)
    {
        GLCD_Command((0xB8)+i); /* Increment page */
        for(j=0;j<64;j++)
        {
            GLCD_Data(0);    /* Write zeros to all 64 column */
        }
    }
    GLCD_Command(0x40);      /* Set Y address (column=0) */
    GLCD_Command(0xB8);      /* Set x address (page=0) */
}

void GLCD_String(char page_no, char *str)    /* GLCD string write function */
{
    unsigned int i,column,Page=((0xB8)+page_no),Y_address=0;
    float Page_inc=0.5;
    CS1 = 1; CS2 = 0;
    GLCD_Command(Page);
    for(i=0;str[i]!=0;i++)
    {
    if (Y_address>(1024-(((page_no)*128)+5)))
        break;
        if (str[i]!=32)
        {
            for (column=1;column<=5;column++)
            {
                if ((Y_address+column)==(128*((int)(Page_inc+0.5))))
                    if (column==5)
                      break;
                    GLCD_Command(0x40);
                    Y_address = Y_address+column;
                    CS1 = ~CS1; CS2 = ~CS2;
                    GLCD_Command((Page+Page_inc));
                    Page_inc=Page_inc+0.5;
                }
            }
        }
    if (Y_address>(1024-(((page_no)*128)+5
    break;
    if((font[((str[i]-32)*5)+4])==0 || str[i]==32)
    {
        for(column=0;column<5;column++)
        {
            GLCD_Data(font[str[i]-32][column]);
            if((Y_address+1)%64==0)
            {
                CS1 = ~CS1; CS2 = ~CS2;
                GLCD_Command((Page+Page_inc));
                Page_inc=Page_inc+0.5;
            }
            Y_address++;
        }
    }
    else
    {
        for(column=0;column<5;column++)
        {
            GLCD_Data(font[str[i]-32][column]);
            if((Y_address+1)%64==0)
                {
                    CS1 = ~CS1; CS2 = ~CS2;
                    GLCD_Command((Page+Page_inc));
                    Page_inc=Page_inc+0.5;
                }
                Y_address++;
            }
            GLCD_Data(0);
            Y_address++;
            if((Y_address)%64==0)
            {
                CS1 = ~CS1; CS2 = ~CS2;
                GLCD_Command((Page+Page_inc));
                Page_inc=Page_inc+0.5;
            }
        }
    }
    GLCD_Command(0x40);    /* Set Y address (column=0) */
}

void main()
{
    GLCD_Init();            /* Initialize GLCD */
    GLCD_ClearAll();        /* Clear all GLCD display */
    GLCD_String(0,"8051 Microcontroller");    /* Print String at 0th page of display */
    while(1);
}
```

GLCD128x64 String Output using 8051

GLCD128x64 Display Text Print

Programming 8051 to Display Image on GLCD 128x64

We are using the same functions that are used for displaying text except for GLCD_String function, which is modified here to print image data on GLCD.

- Here we are using AT89S51 Microcontroller to interface with GLCD. It has a limited 128 byte RAM.
- binary Image is of total size 128x64 pixels. So we need an array of 1024 byte [(128*64)/8=1024 bytes] to store the image in the microcontroller.
- Now the problem is, we can't define an array of size 1024 in RAM (Data memory) as it is only 128 byte. We have to save it in ROM (code memory). To define any variable in code memory we have to use the "code" keyword.

e.g.

```
unsigned char[ ]= "ABCDEFG";          /* array defined in Data memory (RAM) */
code unsigned char[ ]= "ABCDEFG" ;    /* array defined in Code memory (ROM) */
```

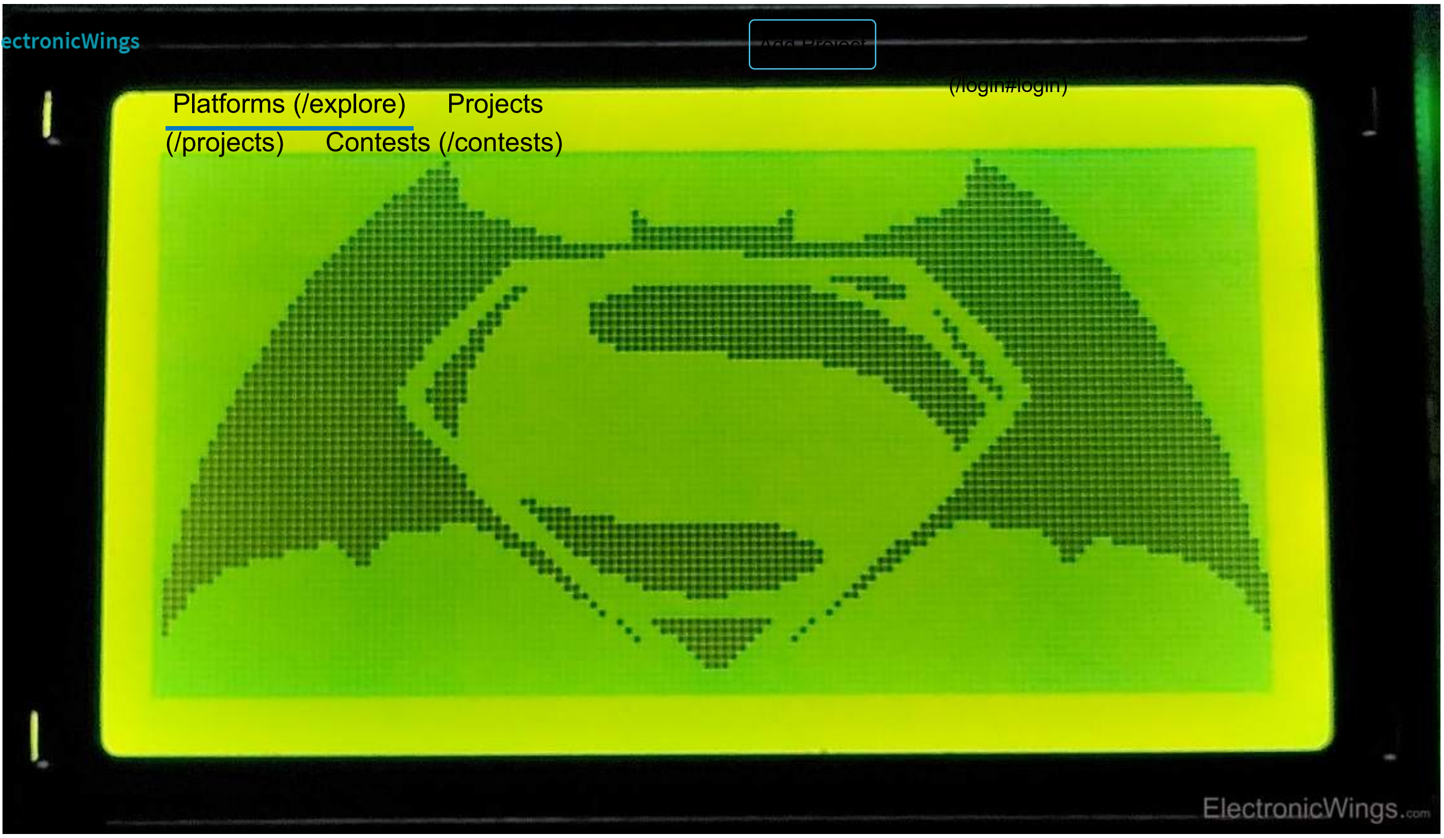
However, array defined in code memory will take more time to access, 8051 uses "movc A, A+DPTR" instruction, which normally takes more machine cycles to execute.

- The image array is defined in Image.h file.

Code for GLCD128x64 Bitmap Image Print

```
/*
 *
 8051 128x64 graphic LCD
 http://www.electronicwings.com
 */


#include <reg51.h>           /* Include reg51 header file */
#include <intrins.h>         /* Include intrinsic header file */
#include "Image.h"           /* Include image header file */
#define Data_Port P3         /* Define data port for GLCD */

sbit RS = P2^0;                       /* Set control bits pins */
sbit RW = P2^1;
sbit E = P2^2;
sbit CS1 = P2^3;
sbit CS2 = P2^4;
sbit RST = P2^5;

void delay(k)             /* Delay of msec with xtal = 11.0592MHz */
{
        int i,j;
        for (i=0;i<k;i++)
                for (j=0;j<112;j++);
}

void GLCD_Command(char Command)  /* GLCD command function */
{
        Data_Port = Command;     /* Copy command on data pin */
        RS = 0;                  /* Make RS LOW to select command register */
        RW = 0;                  /* Make RW LOW to select write operation */
        E = 1;                   /* Make HIGH to LOW transition on Enable */
        _nop_ ();
        E = 0;
        _nop_ ();
}

void GLCD_Data(char Data)      /* GLCD data function */
{
        Data_Port = Data;        /* Copy data on data pin */
        RS = 1;                  /* Make RS HIGH to select data register */
        RW = 0;                  /* Make RW LOW to select write operation */
        E = 1;                   /* Make HIGH to LOW transition on Enable */
        _nop_ ();
        E = 0;
        _nop_ ();
}

void GLCD_Init()          /* GLCD initialize function */
{
        CS1 = 1; CS2 = 1;        /* Select left & right half of display */
        RST = 1;                 /* Keep reset pin high */
        delay(20);
        GLCD_Command(0x3E);      /* Display OFF */
        GLCD_Command(0x40);      /* Set Y address (column=0) */
        GLCD_Command(0xB8);      /* Set x address (page=0) */
        GLCD_Command(0xC0);     /* Set z address (start line=0) */
        GLCD_Command(0x3F);     /* Display ON */
}
void GLCD_ClearAll()        /* GLCD all display clear function */

    int i,j;
        CS1 = 1; CS2 = 1;       /* Select left & right half of display */
        for(i=0;i<8;i++)
        {
                GLCD_Command((0xB8)+i); /* Increment page */
                for(j=0;j<64;j++)
                {
                        GLCD_Data(0);   /* Write zeros to all 64 column */
                }
        }
        GLCD_Command(0x40);             /* Set Y address (column=0) */
        GLCD_Command(0xB8);             /* Set x address (page=0) */
}


void GLCD_String(char *str)         /* GLCD string write function */
{
        int column,page,page_add=0xB8,k=0;
        float page_inc=0.5;
        CS1 = 1; CS2 = 0;
        for(page=0;page<16;page++)
        {
                for(column=0;column<64;column++)
                {
                        GLCD_Data(str[column+k]);
                }
                CS1 = ~CS1; CS2 = ~CS2;
                GLCD_Command((page_add+page_inc));
                page_inc=page_inc+0.5;
                k=k+64;
        }
        GLCD_Command(0x40);
        GLCD_Command(0xB8);
}

void main()
{
        GLCD_Init();                    /* Initialize GLCD */
        GLCD_ClearAll();                /* Clear all screen */
        GLCD_String(img);               /* Print image */
        while(1);
}
```

GLCD128x64 Image Print using 8051



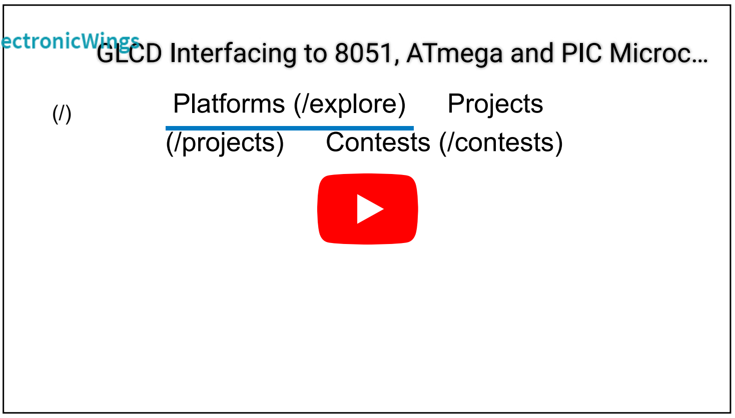
GLCD 128x64 bitmap Image Print

Animation on GLCD 128x64 using 8051

To make animation on GLCD 128x64 JHD12864E display do the below steps,

- Take two or more images in a manner that their sequence will create the illusion of motion.
- Convert it to Binary image data using Image2GLCD application.
- And print them on GLCD in series of a sequence. It will create animation.
- Note that provides sufficient delay in between images.

Animation Video GLCD128x64 using 8051



GLCD Interfacing to 8051, ATmega and PIC Microc...

Supporting Files

Source Code

- 8051_GLCD_Animation_Keil_Project Download | 863
- 8051_GLCD_Image_Keil_Project Download | 909
- 8051_GLCD_Text_Font_Keil_Project Download | 1110
- 8051_GLCD_Text_Font_Proteus_Simulation_File Download | 42

Attached File

- GLCD 128x64 Datasheet Download | 2600

Comments

Please login/signup to comment and reply.

About Us (/about)
Business Offering (/business-services)
Host Platform (/launch-platform)
Contact Us (/contactus)

Terms of Service (/terms-of-service)
Cookies Policy (/cookie-policy)
Privacy Policy (/privacy-policy)

Connect On:
Facebook(https://www.facebook.com/electronicwings)
LinkedIn(https://www.linkedin.com/company/electronicwin
Youtube(https://www.youtube.com/channel/UCNdqkukBtk4
Instagram(https://www.instagram.com/electronicwings_coi
igshid=1cip10jijttko)