**E-ink display photoframe**

**Introduction**

In this project we display information on e-ink display module using Arduino nano –BLE module. E-ink display has screen has a resolution of 800x600 and 4 colors, black, white, and two shades of grey. The screen also has a card reader which we'll be using. Simply uploading pictures on the card won't do the job however. The display only supports BMP files and the smaller the file size the faster it'll be loaded which will save a lot of battery.

**Components**

* Arduino nano –BLE
* 1.54 inch E-Paper Display –
* Connecting wires

**Application**

shelf label, industrial instruments, home and appliances, automotive, mobile devices such as E-Paper, E-Book, E-Reader and Smart Watch.

**Objective**

During this activity ,you will help students to achieve following objectives

1. Understanding the principle and operation of 1.54inch E –ink display module
2. Design algorithm and flowchart to view photoframe on display
3. Programming 1.54 inch E –ink display module with arduino nano
4. Interfacing 1.54inch E –ink display module with arduino nano

**Program**

#include <SPI.h>

#include "1\_54in\_epaper.h"

#include <stdlib.h>

Epd epd; *// initiate e-paper display [epd]*

**unsigned** **char** image[1024]; *// memory for display*

Paint paint(image, 0, 0); *// setup for text display*

#define COLORED 0 *// background color handler (dark)*

#define UNCOLORED 1 *// background color handler (light)*

String tot\_val = ""; *// storage variable for serial data*

String header\_txt[3] = {"1.54in e-Paper","Simple Demo","Maker Portal"}; *// header text*

**int** initial\_space = 10; *// initial white/dark space at the top of the display*

**const** **int** init\_row\_lines = **sizeof**(header\_txt)/**sizeof**(String); *// counting rows to avoid header overlap*

**int** row\_line = 0; *// for iterating*

**int** row\_height = 24; *// row height (based on text size)*

**const** **int** max\_lines = 8; *// max lines that can print at font-20 (for this demo)*

String line\_array[max\_lines-init\_row\_lines]; *// More lines than this and the display will overflow*

**void** setup() {

Serial.begin(9600); *// start serial handling for Bluetooth handling*

clear\_func(); *// clear display and set header*

}

**void** loop() {

**char** val = Serial.read(); *// read character from BLE device (iPhone)*

**if** (**int**(val)!=-1){ *// make sure it's a valid integer pin*

tot\_val+=val; *// if the input is over 1 character, this handles that*

**if** (val=='\n'){ *// wait for newline*

**int** str\_len = strlen(tot\_val.c\_str()); *// get length of string*

tot\_val[str\_len-1] = '\0'; *// blank out the newline character from BLExAR*

**if** (row\_line<(max\_lines-init\_row\_lines)){

line\_array[row\_line] = tot\_val; *// append to string array*

header\_print(); *// print header text*

update\_txt(row\_line);

} **else** {

**for** (**int** qq=0;qq<(max\_lines-init\_row\_lines);qq++){

line\_array[qq] = String(" "); *// reset array too*

}

clear\_func(); *// clear display upon overload*

row\_line = 0; *// reset iterator*

line\_array[row\_line] = tot\_val; *// append to string array*

header\_print(); *// print header*

update\_txt(row\_line); *// update with last received text*

}

tot\_val = ""; *// clear serial input variable*

row\_line+=1; *// iterate to ensure new text prints to new line*

}

}

}

**void** header\_print(){

paint.SetWidth(200); *// set display width*

paint.SetHeight(24); *// set initial vertical space*

*// loop through header text and print values*

**for** (**int** jj=0;jj<init\_row\_lines;jj++){

paint.Clear(jj%2); *// background*

paint.DrawStringAt(0, 4, header\_txt[jj].c\_str(), &Font20, **int**(!**bool**(jj%2))); *// text*

epd.SetFrameMemory(paint.GetImage(), 0, initial\_space+(jj\*row\_height), paint.GetWidth(), paint.GetHeight());

}

}

**void** clear\_func(){

epd.LDirInit(); *// initialize epaper*

epd.Clear(); *// clear old text/imagery*

epd.DisplayPartBaseWhiteImage(); *// lay a base white layer down first*

header\_print(); *// print header text*

epd.DisplayPartFrame(); *// display header text*

}

**void** update\_txt(**int** row\_ii){

**for** (**int** ii=0;ii<=row\_ii;ii++){

paint.Clear(**int**(!**bool**(ii%2))); *// clear background*

*// below: draw at opposite to background (light vs dark)*

paint.DrawStringAt(0, 4, line\_array[ii].c\_str(), &Font20, ii%2);

epd.SetFrameMemory(paint.GetImage(), 0, initial\_space+((ii+init\_row\_lines)\*row\_height), paint.GetWidth(), paint.GetHeight());

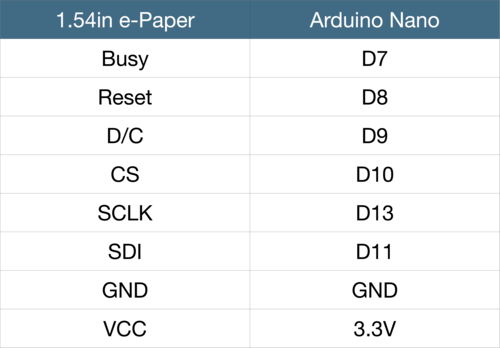
}

epd.DisplayPartFrame(); *// display new text*

}

**Hardware**

Connection between 1.54inch e-ink display module and arduino nano are as follows



This wiring will work with any ATmega328P-based Arduino uno and nano

