Amrita Vishwa Vidyapeetham

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

A REPORT

ON

”IMAGE DISPLAYER ”

SUBMITTED BY

NMS.YASWANTH (AM.EN.U4ECE18135)
P.NARASIMHA (AM.EN.U4ECE18139) K.SRIKAR (AM.EN.U4ECE18164) SIDHARTH.K.S (AM.EN.U4ECE18149)

UNDER THE GUIDANCE OF

Asst.Prof.SenthilMurugan
Asst.Prof.ChinmayiR.
Asst.Prof.SaritaP.S.

(Academic Year: 2020-2021)

Amrita Vishwa Vidyapeetham Amrita School of Engineering, Kollam

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Certificate

This is to certify that project entitled

"IMAGE DISPLAYER"

has been completed by

Mr. NMS.YASWANTH (Roll No. AM.EN.U4ECE18135)
Mr. P.NARASIMHA (Roll No. AM.EN.U4ECE18139) Mr. K.SRIKAR (Roll No. AM.EN.U4ECE18164)
Mr. SIDHARTH.K.S (Roll No. AM.EN.U4ECE18149)

of TE COMP I/II/Second Shift in the Semester - I of academic year 2020-2021 in partial fulfillment of the Third Year of Bachelor degree in "Electronics and Communication Engineering" as prescribed by the Amrita Vishwa Vidyapeetham University.

Prof. Guide Name
Seminar Guide

Senthil Murugan
Chinmayi R

Place: Amritapuri, Kollam.
Date: 13/12/2020

ACKNOWLEDGEMENT

It gives me great pleasure and satisfaction in presenting this mini project on "Image Displayer".

Though we have taken efforts in the making of the project,it wouldn't have been successful without the help extended by our Microcontrollers Lab faculty.

I have furthermore to thank Electronics Department HOD **Dr.Ravishankar** *and* **Asst.Professors** *to encourage me to go ahead and for continuous guidance. I also want to thank them all for their assistance and guidance for preparing report.*

I would like to thank all those, who have directly or indirectly helped me for the completion of the work during this mini project.

NMS.YASWANTH (AM.EN.U4ECE18135)
P.NARASIMHA (AM.EN.U4ECE18139)
K.SRIKAR (AM.EN.U4ECE18164)
SIDHARTH.K.S (AM.EN.U4ECE18149)

Dept. of Electronics And Communication Engineering

Contents

List of Figures

List of Tables

Abstract

Our circuit describes a mini digital photo album project which loads bitmap images (BMP) from an SD card inserted into the SD card reader module.

The project was developed using the LPC2148 microcontroller and PG-128x128- A lcd screen. Simulation was performed using proteus 8.1 software and code was written and compiled in embedded c using keil uvision 4.0 written below is a sort of documentation describing the components of the system and the source code.

Keywords-*Orientation Field Estimation, Enhanced Feedback, Minu- tiae, Fuzzy Feature Match(FFM).*

Chapter 1

INTRODUCTION

This project describes a Image displayer project which loads bitmap images (BMP) from an SD card inserted into the SD card reader module. It reads the data in a contiguous manner and as such the card needs to be set up with the image files prior to reading them off. The microcontroller displays the images on the SD card on a 128x128 graphic lcd. It also employs a slideshow function that triggers the system to read the next image in a preset amount of time and a power button that puts the system in a low power state.

Literature review:

Image Displayer is a conveyable device that concedes permanent storage, retrieval, presentation, and organization of digital pictures and short animations acquired from a computer, digital camera or the Internet. Transferring pictures from the camera is as uncomplicated as the flashcard from the camera, establishing it into the digital photo album. Also, any picture on a computer can be collected in a digital photo album. Using the internet, long-distance picture distribution can be made seamless. It can be practiced as a picture frame by using presentation mode, which displays a slide show of pictures. It also resolves the predicament of printing pictures at less cost. Also, it is much more prosperous and easier to use than a computer.

It is contemplated that the digital photo album transforms the design digital photos are inspected. Also, digital photos are surveyed on a camera or mobile phone or immediately thereafter when offloaded to a computer or a backup device or when printed. Then a user can generate numerous backup storage devices or memory cards, restraining thousands of pictures. Adopting the digital album of the immediate enlightenment, a user can safely and efficiently store all of these backup devices and memory cards and illustration collection of photos at a moment's intimation and in a single portable location. So, a user faces the problem of storing all the photos in a particular location, such as his/her computer, and endangering the possibility of having the crash of the computer and squandering all the photos.

Chapter 2

Theory of Working

2.1 Image Data Basics

A digital image in its essence consists of a two-dimensional array of numbers. The color or gray shade displayed for a given picture element (pixel) depends on the num- ber stored in the array for that pixel. A grayscale image is set up in such a way that each pixel takes on a value between zero and the number of gray scales or gray levels that the camera can record. Color images are similar to gray scale except that there are three bands, or channels, corresponding to the colors red, green, and blue. Thus, each pixel has three values associated with it.The simplest type of image data is black and white. It is a binary image meaning each pixel is either 0 or 1. For this implementation we work with binary images as the screen used for display is a monochrome one.

2.2 SD card basics

The MMC/SDC can be attached to the most microcontrollers via a generic SPI interface or some GPIO ports. Therefore the SPI mode is suitable for low cost embedded applications with no native host interface available

Power ON or card insersion - After supply voltage reached 2.2 volts, wait for one millisecond at least. Set SPI clock rate between 100 kHz and 400 kHz. Set DI and CS high and apply 74 or more clock pulses to SCLK. The card will enter its native operating mode and go ready to accept native command.

Software reset - Send a CMD0 with CS low to reset the card. The card samples CS signal on a CMD0 is received successfully. If the CS signal is low, the card enters SPI mode and responds R1 with In Idle State bit (0x01). Since the CMD0 must be sent as a native command, the CRC field must have a valid value. When once the card enters SPI mode, the CRC feature is disabled and the CRC is not checked by the card, so that command transmission routine can be written with the hardcorded CRC value that valid for only CMD0 and CMD8 with the argument of zero. The CRC feature can also be switched with CMD59.

Initialization - In idle state, the card ac- cepts only CMD0, CMD1, ACMD41,CMD58 and CMD59. Any other commands will be rejected. In this time, read OCR register and check working voltage range of the card. In case of the system supply voltage is out of working voltage range, the card must be rejected. Note that all cards work at supply voltage range of 2.7 to 3.6 volts at least, so that the host controller needs not check the OCR if supply voltage is in this range. The card initiates the initialization process when a CMD1 is received. To detect end of the initialization process, the host controller must send CMD1 and check the response until end of the initialization.

.

2.3 SPI command set

Each command is expressed in abbreviation like GO_IDLE_STATE or CMD, is the number of the command index and the value can be 0 to 63. Following table describes commands used for generic read/ write and card initialization

Command Index	Argument	Response	Data	Abbreviation	Description
CMD0	None(0)	R1	No	GO_IDLE_STATE	Software reset.
CMD1	None(0)	R1	No	SEND_OP_COND	Initiate initialization process.
ACMD41(*1)	*2	R1	No	APP_SEND_OP_COND	For only SDC. Initiate initialization process.
CMD8	*3	R7	No	SEND_IF_COND	For only SDC V2. Check voltage range.
CMD9	None(0)	R1	Yes	SEND_CSD	Read CSD register.
CMD10	None(0)	R1	Yes	SEND_CID	Read CID register.
CMD12	None(0)	R1b	No	STOP_TRANSMISSION	Stop to read data.
CMD16	Block length[31:0]	R1	No	SET_BLOCKLEN	Change R/W block size.
CMD17	Address[31:0]	R1	Yes	READ_SINGLE_BLOCK	Read a block.
CMD18	Address[31:0]	R1	Yes	READ_MULTIPLE_BLOCK	Read multiple blocks.
CMD23	Number of blocks[15:0]	R1	No	SET_BLOCK_COUNT	For only MMC. Define number of blocks to transfer with next multi-block read/write command.
ACMD23(*1)	Number of blocks[22:0]	R1	No	SET_WR_BLOCK_ERASE_COUNT	For only SDC. Define number of blocks to pre-erase with next multi-block write command.
CMD24	Address[31:0]	R1	Yes	WRITE_BLOCK	Write a block.
CMD25	Address[31:0]	R1	Yes	WRITE_MULTIPLE_BLOCK	Write multiple blocks.
CMD55(*1)	None(0)	R1	No	APP_CMD	Leading command of ACMD<n> command.
CMD58	None(0)	R3	No	READ_OCR	Read OCR.

Chapter 3

Components Required

3.1 Power LED

The LED lights up to signify that the system is active and functional if system is pow- ered down the led turns off.

3.2 Buttons

There are three buttons implemented each triggering their own interrupts the Play/Pause button as is obvious by its name controls whether or not the slide show is running. The Next button opens the next image on the memory card. The power button turns off all peripherals and powers down screen. The buttons were selected so that the power button is the button on p0.14 and the others are on the joystick so as to make it easier to test directly on the kit.

3.3 SD Reader

The SPI mode is an alternative operating mode that is defined to use the MMC/SDCs without a native host interface. The memory card is initialized with hex arrays con- sisting of 128x128 sized bitmaps meaning one line is 128/8 = 16 bytes long and total image is 128 lines or 2Kb the SD card operates by dividing its space to sectors sized 512 bytes, so every picture takes up 4 sectors. The SD reader functions by reading an image in to a char array from the memory card given starting sector address.

3.4 Display Unit

The display unit in this case is a graphic 128x128 LCD screen its power supply or Vdd is driven by the microcontroller an interface has been implemented to display images and text on the lcd.

3.5 Slide Show

The slide show is implemented with a timer interrupt that triggers the system to dis- play next image every 5 seconds

Chapter 4

Block Diagram

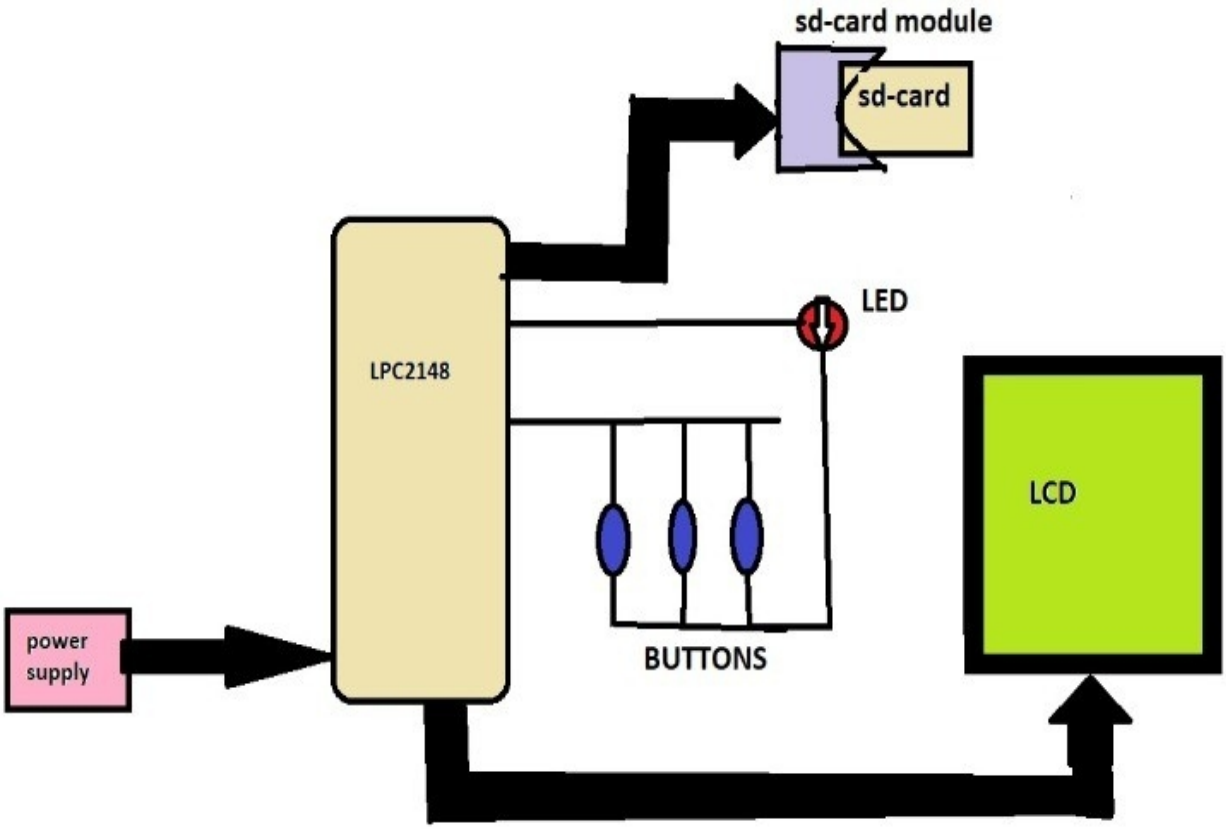
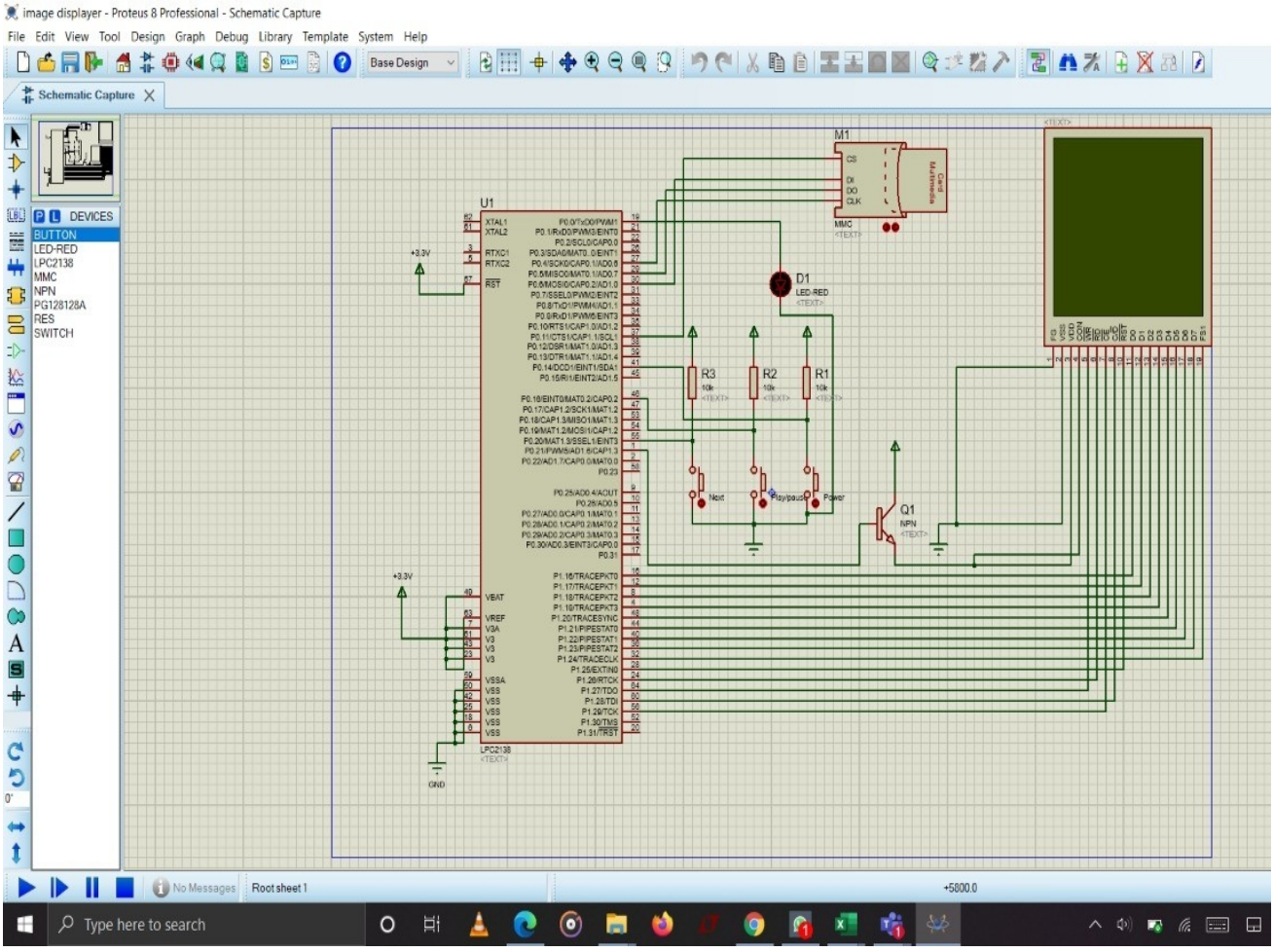


Figure 4.1: Image Displayer block diagram

Chapter 5

Circuit Diagram



Chapter 6

Results and Discussions

Results:

• Analyzed and implemented interrupt services in ARM LPC2148 and learnt to prioritize requests from external devices.

• Successfully constructed a circuit system to control external peripheral devices such as LCD display device through inputs given via buttons .

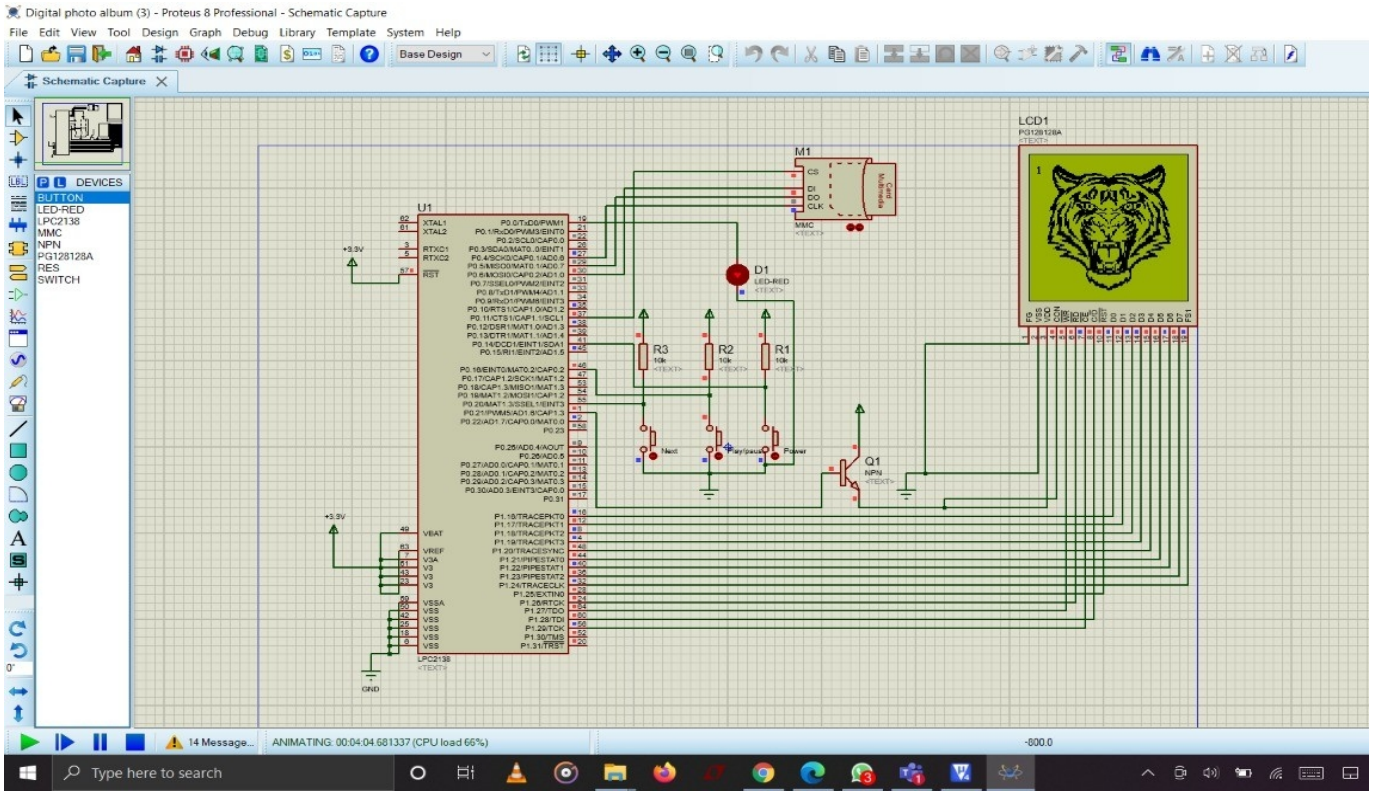


Figure 6.1: Output1

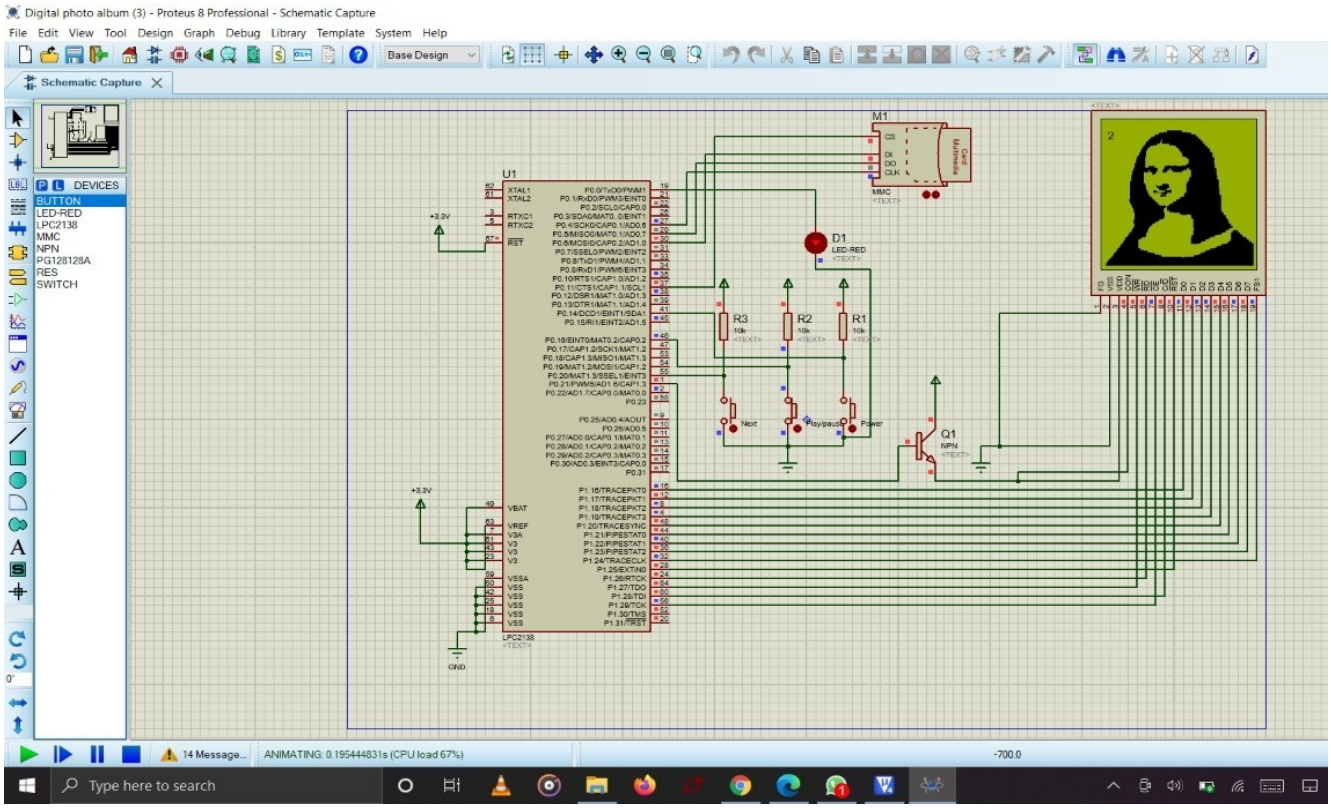


Figure 6.2: Output2

Discussions:

• Recreating the above circuit on a miniature scale does not have any industrial value but the concept of interrupt handling has it's weightage in the field of communi- cation and media.

• Implementing the circuit on software platform is cost-effective, real time results can't be properly observed as prices of components varies over places.

• There are 12 types of BITMAP images we have given to the sd-card, these above images are the some of the examples of those output figures.

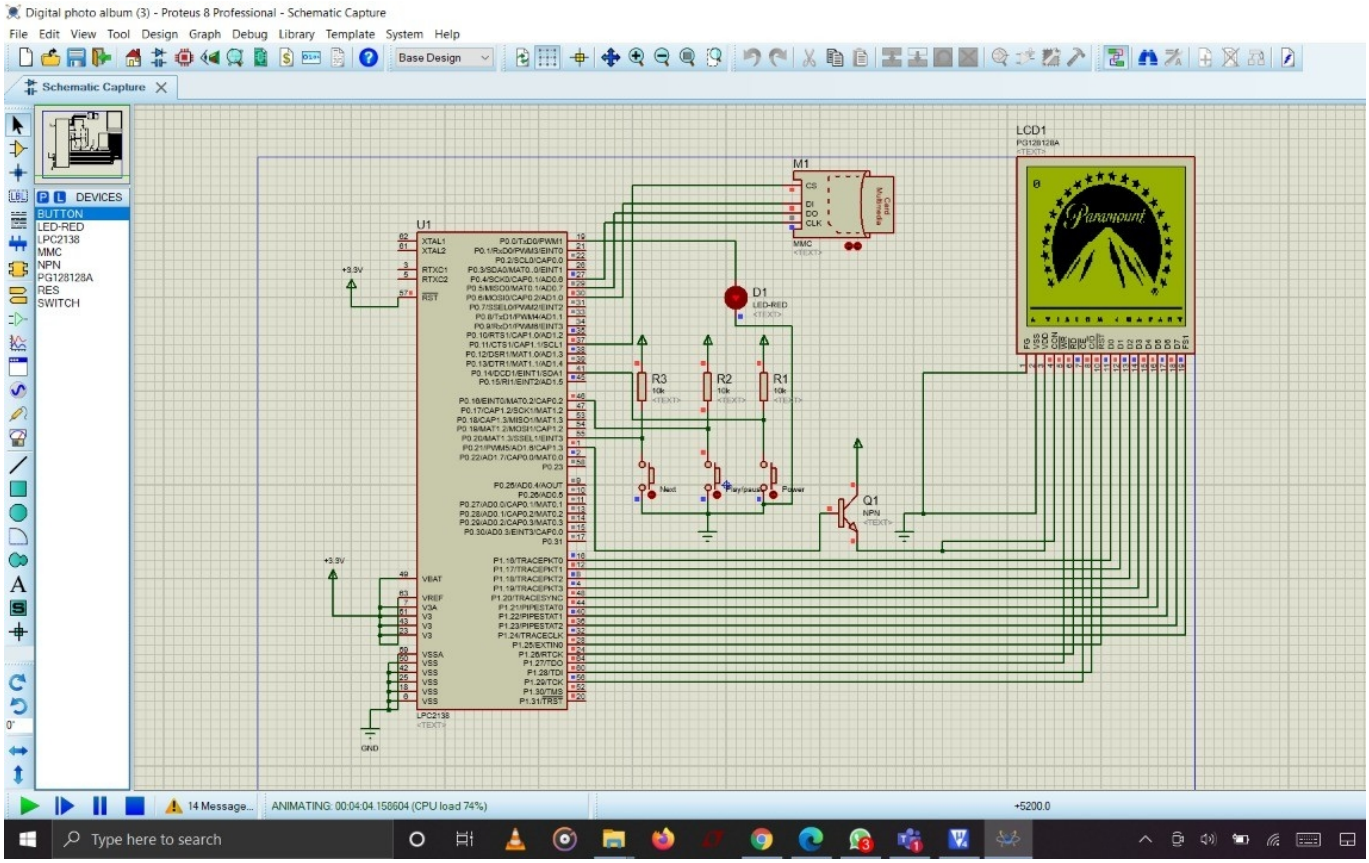


Figure 6.3: Output3

Chapter 7

Conclusion and Future Work

Image displayers are very widely used now a days and they will be used for displaying images the For the Restaurents, and they also used for showcase in the home, it will be used as a memory for all the people

The overall cost for making this Image displayer will be approx:1500/- to 2000/-

Image displayers are handy for all sorts of things:

Creating a separate album for a family vacation

Creating a special gift for someone

Documenting a special holiday

Documenting a specific family tradition

Capturing a sports season or extra-curricular event

Documenting major life events such as adoption, graduation, birthday, wedding, birth, or death

Documentation

8.1 System.c

PLL (Phase Locked Loop) - Is an electronic circuit consisting of Current controlled oscillator(CCO), phase detector and frequency multiplier. The task of the PLL is to generate the required clock for the CPU from the crystal oscillator base frequency which in this case is 12MHz.

void clock_init(void) – initializes the system clock to count at 48 MHz

void feedSeq(void) - The byte sequence (0xAA-0x55) is required to change the PLL config. to protect it from changing inadvertently by spurious system activity or external processes.

void setupPLL0(void) – sets up system to operate at frequency 4 times faster than the crystal oscillator.

void connectPLL0(void) - checks whether PLL has locked on to the desired frequency.

8.2 Timer.c

Timer0 is set up with a vectored irq on slot3. Prescale is set to 48000 , at system time of 48MHz the prescale means that the counter increments every milli second, the match register is set to 5000 meaning an interrupt is triggered every 5000 milli seconds.

void timer_init(void) – initializes timer and installs it as an interrupt

__irq void T0ISR(void) – the interrupt routine executed upon interrupt, it flips the next_pending flag

8.3 Buttons.c

Pins P0.14, P0.16 and P0.20 are set up as external interrupts also the global flags ‘next_pending’, ‘power’ and ‘slide_show’ are initiated here to be used in all other files.

void button_init(void) – initializes pins as external interrupts and install them in vectored slots on the interrupt vector slot. Power button is set up to wake the micro controller from power down mode

__irq void next_button_ISR (void) – turns on next_pending flag and if slide_show is active it also resets the timer

__irq void play_button_ISR (void) – toggles slide_show and depending on the value of slide show it sets

the timer to either reset or start .

__irq void power_button_ISR (void) – toggles power, if powering off it disables all peripherals and puts the microcontroller in power down mode. If powering on it enables SPI and timer0 and resumes regular operations

8.4 Lcd.c

The PG-128x128-A lcd has the following available pin connections

Pin #	Symbol	Description
1	FGND	Frame ground
2	Vss	Power supply(GND)
3	Vdd	Power supply(+)
4	Vo	Contrast Adjust
5	WR	Data write
6	RD	Data read
7	CE	Chip enable
8	C/D	Command / data select
10	RST	Reset
11 - 18	DB0 - DB7	Data bus line
19	FS	Font select

void lcd_init (void) – initializes p1.16 – p1.29 as outputs and initializes screens output mode and turns it on.

void wr_xd (data, command) – writes data and sends command

void wr_auto (unsigned char data) – automatically writes data to lcds Data bus line.

void chk_busy (unsigned char autowr) – converts Data bus line pins to input and checks if buffer is clear

void clrram (void) – clears the screen by writing 0 to every pixel byte by byte.

void disp_img (addr, xl, yl, img) – draws img of size xl , yl starting at addr.

void draw_string(x, y, str) – draws string at x,y (lcd_init sets display mode as img xor text so text is overlaid over the image in contrast).

8.5 Spi.c

The Serial Peripheral Interface (SPI) is a synchronous serial communication interface. The communication is Carried out by using Four lines (serial data out (SDO), serial data in(SDI), serial clock (SCK) and Slave select (SSEL)).

SPI_init(void) – Initializes pins and starts the SPI module as a master on the bus.

SPI_write(char data) - Sends a data byte on the SPI bus as a master.

SPI_read(void) - Reads a data byte from the SPI bus

8.6 SD.c

SD_init(void) - initializes the SD card in SPI mode.

SD_sendCommand(cmd, arg) - sends a standard command to SD/MMC

SD_readImage(startBlock, img) - reads a single image(2048 Bytes) from SD/MMC.

8.7 Main.c

Almost all of the functionality is done using the interrupts but since it would be a very bad idea to run bulky tasks from with in interrupts they are mainly used to set flags letting the main function know what to do next.

Void inititalize(void) – initializes system clock, buttons, lcd, timer, spi module and sd card.

Void next(void) – retrieves next img from sd card and displays it on the lcd along with its index.

Chapter 9

Appendix

Entire code link:

https://github.com/NMSYASWANTH/IMAGE-DISPLAYER-MC_PROJECT

Snipshots of some functions code:

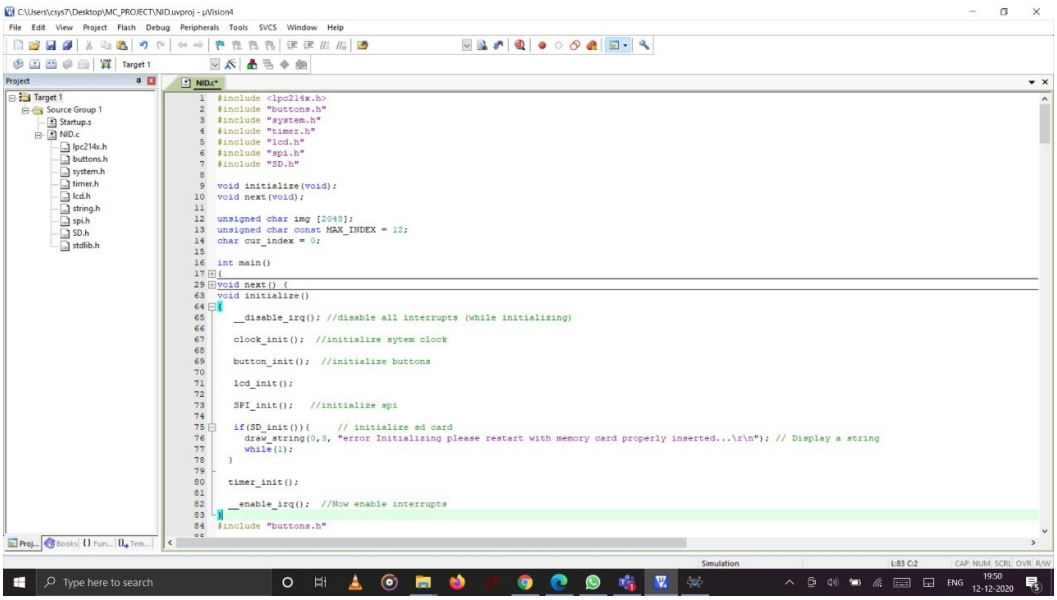
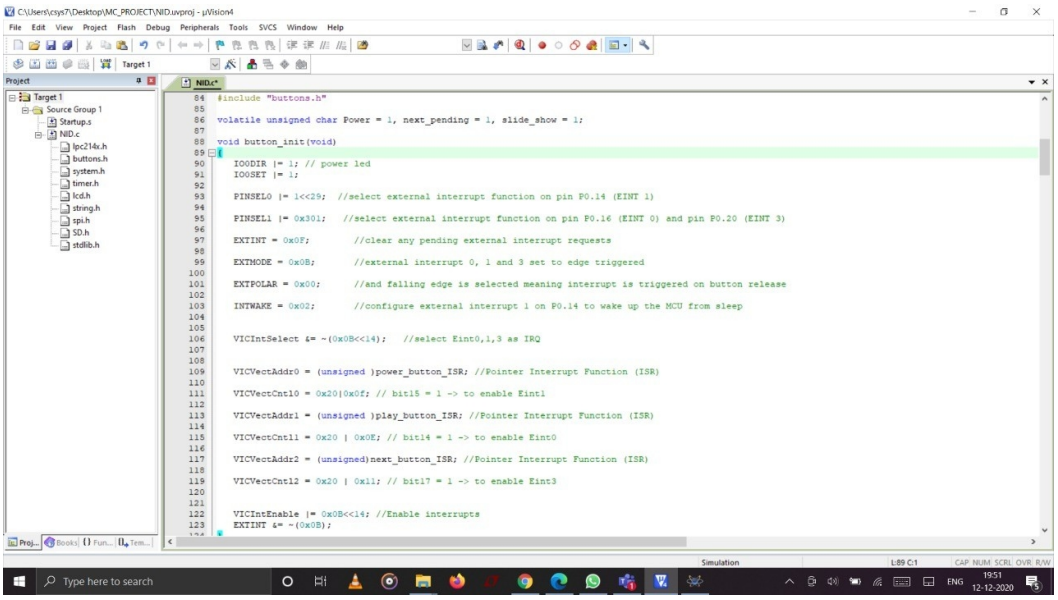
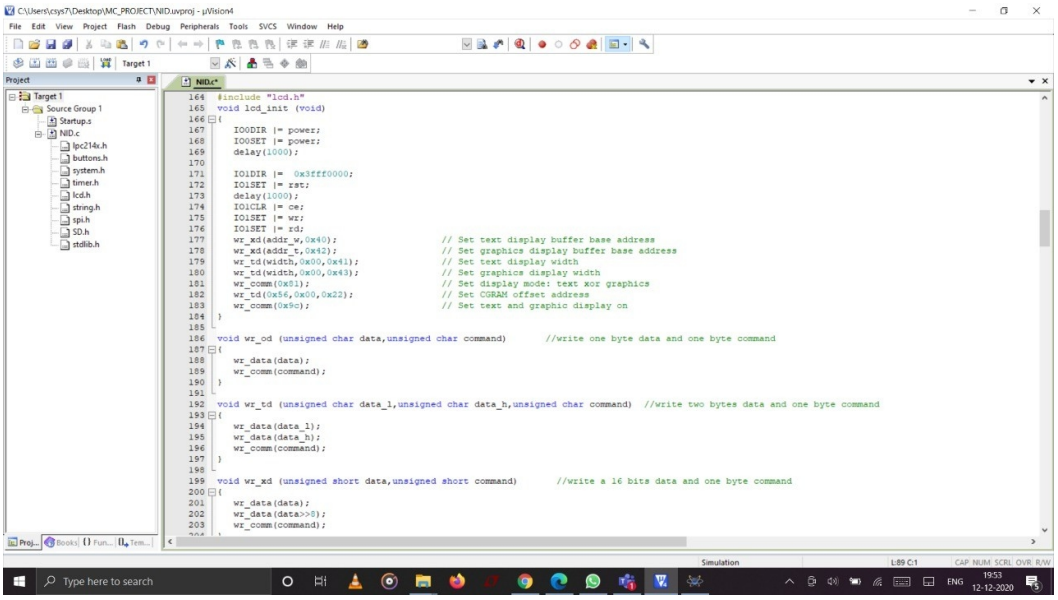


Figure 9.1: main code



```
#include "buttons.h"

volatile unsigned char Power = 1, next_pending = 1, slide_show = 1;

void button_init(void)
{
   IO0DIR |= 1; // power led
   IO0SET |= 1;

   PINSEL0 |= 1<<29;  //select external interrupt function on pin P0.14 (EINT 1)

   PINSEL1 |= 0x301;   //select external interrupt function on pin P0.16 (EINT 0) and pin P0.20 (EINT 3)

   EXTINT = 0x0F;        //clear any pending external interrupt requests

   EXTMODE = 0x0B;       //external interrupt 0, 1 and 3 set to edge triggered

   EXTPOLAR = 0x00;      //and falling edge is selected meaning interrupt is triggered on button release

   INTWAKE = 0x02;       //configure external interrupt 1 on P0.14 to wake up the MCU from sleep

   VICIntSelect &= ~(0x0B<<14);   //select Eint0,1,3 as IRQ

   VICVectAddr0 = (unsigned )power_button_ISR; //Pointer Interrupt Function (ISR)

   VICVectCntl0 = 0x20|0x0f; // bit15 = 1 -> to enable Eint1

   VICVectAddr1 = (unsigned )play_button_ISR; //Pointer Interrupt Function (ISR)

   VICVectCntl1 = 0x20 | 0x0E; // bit14 = 1 -> to enable Eint0

   VICVectAddr2 = (unsigned)next_button_ISR; //Pointer Interrupt Function (ISR)

   VICVectCntl2 = 0x20 | 0x11; // bit17 = 1 -> to enable Eint3

   VICIntEnable |= 0x0B<<14; //Enable interrupts
   EXTINT &= ~(0x0B);
```

Figure 9.2: buttons code



```
#include "lcd.h"
void lcd_init (void)
{
   IO0DIR |= power;
   IO0SET |= power;
   delay(1000);

   IO1DIR |=  0x3fff0000;
   IO1SET |= rst;
   delay(1000);
   IO1CLR |= ce;
   IO1SET |= wr;
   IO1SET |= rd;
   wr_xd(addr_w,0x40);                  // Set text display buffer base address
   wr_xd(addr_t,0x42);                  // Set graphics display buffer base address
   wr_td(width,0x00,0x41);              // Set text display width
   wr_td(width,0x00,0x43);              // Set graphics display width
   wr_comm(0x81);                       // Set display mode: text xor graphics
   wr_td(0x54,0x00,0x22);               // Set CGRAM offset address
   wr_comm(0x9c);                       // Set text and graphic display on
}

void wr_od (unsigned char data,unsigned char command)       //write one byte data and one byte command
{
   wr_data(data);
   wr_comm(command);
}

void wr_td (unsigned char data_l,unsigned char data_h,unsigned char command)  //write two bytes data and one byte command
{
   wr_data(data_l);
   wr_data(data_h);
   wr_comm(command);
}

void wr_xd (unsigned short data,unsigned short command)       //write a 16 bits data and one byte command
{
   wr_data(data);
   wr_data(data>>8);
   wr_comm(command);
```

Figure 9.3: lcd code

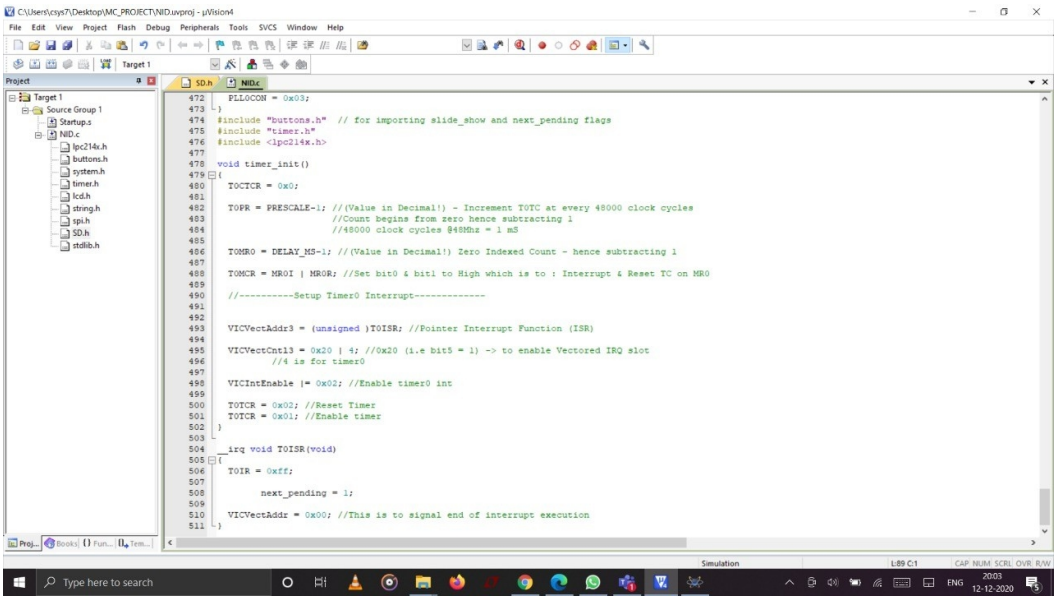


Figure 9.4: timer code

Chapter 10

REFERENCES

[1] Engineers garage – connecting sd card to avr

[2] Image processing in C – Dwayne Philips