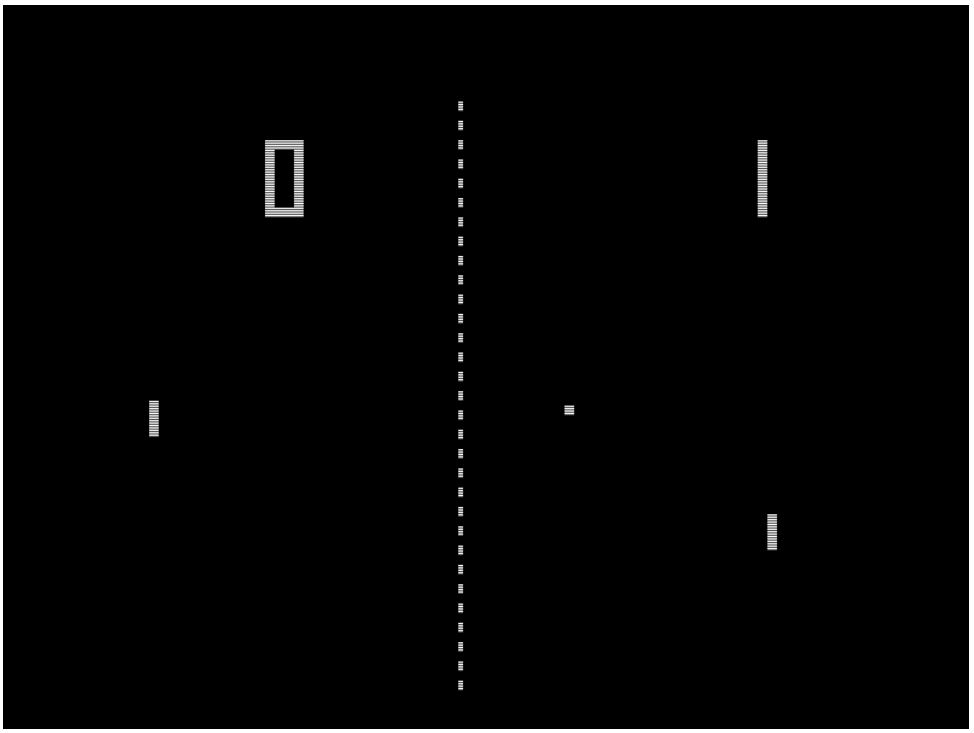
C11

Graphics & Games

This exercise is an introduction to working with a graphical LCD display on the `Il Matto` development board. The focus of the exercise is to learn to use functions from a graphical library to build a computer game with animation.



Schedule

Preparation Time	:	3 hours
Lab Time	:	3 hours

Items provided

Tools	:	
Components	:	
Equipment	:	
Software	:	`avr-gcc`, `Eclipse`

Items to bring

Il Matto

Identity card

Laboratory logbook

Toolkit

Version: December 9, 2013

Steve R. Gunn
Electronics and Computer Science
University of Southampton

Before entering the laboratory you should read through this document and complete the preparatory tasks detailed in section 2.

> **Academic Integrity** – *If you wish you may undertake the preparation jointly with other students. If you do so it is important that you acknowledge this fact in your logbook. Similarly, you will probably want to use sources from the internet to help answer some of the questions. Again, record any sources in your logbook.*

You will undertake the exercise working with your laboratory partner. During the exercise you should use your logbook to record your observations, which you can refer to in the future – perhaps to write a formal report on the exercise, or to remind you about the procedures. As such it should be legible, and observations should be clearly referenced to the appropriate part of the exercise. As a guide the ✎ symbol has been used to indicate a mandatory entry in your logbook. However, you should always record additional observations whenever something unexpected occurs, or when you discover something of interest.

For each Task you should create a new directory so that you have a working version of every program at the end of the lab. Remember to place comments in your code.

You will be marked using the standard laboratory marking scheme; at the beginning of the exercise one of the laboratory demonstrators will mark your preparatory work and at the end of the exercise you will be marked on your progress, understanding and logbook.

Notation

This document uses the following conventions:

✎ An entry should be made in your logbook

1 Introduction

This lab introduces you to working with a graphical LCD on an embedded platform.

1.1 Outcomes

At the end of the exercise you should be able to:

- Interface to a graphical TFT display.
- Link with pre-built libraries.
- Have some fun.

2 Preparation

Study and familiarise yourself with the provided graphical LCD library code. Use your preparation time to prepare code for the laboratory and think how you might customise your game. Consider whether you will use any interrupts.

3 Laboratory Work

Open the Eclipse application on the computer you are using and when asked for the workspace you wish to use, make sure the checkbox for *use as default* is not ticked and type `H:\ELEC1201\Labs\C11`.

Remember to create a new C project (File -> New -> New project.. -> C/C++ -> C Project), for each task of the exercise and name it with the appropriate part of the exercise for easy future reference, e.g. `C11_3.1`. This time you should select "AVR Cross Target Application" -> "Empty Project" as "Project type" and add in a new C source file.

In the following tasks you should make use of the LCD graphics library that you have been provided with. The available functions can be found in the header file `lcd.h` given in section A. You will need to link your project with the library by adding `-llcd` to the end of the linker command. The library is configured to have the LCD connected across ports A and C as for the pacman program from X2R.

3.1 Hello World

Write a program to display the message "Hello World" on the LCD display.

3.2 Simple Graphics

Write a program to draw a yellow rectangle of width 200 pixels by height 100 pixels in the centre of the display. Note that there are two functions provided for drawing rectangles. The first fills the

rectangle with a constant colour whilst the second fills the rectangle with the colours stored in an array. Remember that the function is inclusive of the co-ordinates, so that if the left and right co-ordinates are the same the rectangle will be one pixel wide.

3.3 Simple Animation

Write a program to draw a yellow square of width 5 pixels in the centre of the display. Now extend your program to move the square in a diagonal direction. When the square reaches the edge of the display it should bounce off in an elastic collision. You will need to erase the old square before drawing the new one. Additionally you should add a small delay of around 4ms to slow down the speed. Think carefully about the best place to put this delay in the loop to get the best animation.

3.4 User Input

Write a program to draw a green rectangle of 3 pixels high by 50 pixels wide. Add functionality to allow the horizontal position of the rectangle to adjusted by using an input device. You are free to choose any method you like, but you could connect two switches or wires to two inputs to control left or right movement or you could use the rotary encoder from lab C6. The program should ensure that the rectangle cannot be moved beyond the edge of the screen.

3.5 Making a Game

Combine your programs from section 3.3 and section 3.4 to implement the game of pong. Here the yellow square becomes the ball and the green rectangle the bat. The ball should bounce around the screen and when it reaches the edge with the bat on, it should only bounce if the bat is in the way. Otherwise the game should stop and print the message "game over" in the middle of the screen.

4 Optional Additional Work

4.1 Customisation of your Game

Use your imagination to provide a more personalised version of the game. Here are some ideas:

- Make the graphics more colourful and less like rectangles.
- Add a two player mode with an additional two inputs.
- Add a scoring display.
- Connect two Il Matto's together using the UART interface and when the ball reaches the top of the screen it should leave and enter the screen on the other Il Matto to create two player mode.
- Add sound to the game.

References

[1] Atmel Corporation. ATMEGA164PA/324PA/644PA/1284P Datasheet. Datasheet 8152G-AVR-11/09, 2011. URL http://www.atmel.com/Images/doc8152.pdf.

A LCD Library

You can link your project with the static LCD library from the command line by invoking the following command. Here the `-L` option specifies the path to the library file `liblcd.a`. It is important to make sure the `-llcd` command is at the end of the line otherwise you will likely receive undefined references.

```
avr-gcc -mmcu=atmega644p -L../lcdlib -o pong.elf pong.o -llcd
```

LISTING 1: lcd.h

```
/*  Author: Steve Gunn
 * Licence: This work is licensed under the Creative Commons Attribution License.
 *           View this license at http://creativecommons.org/about/licenses/
 */

#include <stdint.h>

#define LCDWIDTH    240
#define LCDHEIGHT   320

/* Colour definitions RGB565 */
#define WHITE       0xFFFF
#define BLACK       0x0000
#define BLUE        0x001F
#define GREEN       0x07E0
#define CYAN        0x07FF
#define RED         0xF800
#define MAGENTA     0xF81F
#define YELLOW      0xFFE0

typedef enum {North, West, South, East} orientation;

typedef struct {
    uint16_t width, height;
    orientation orient;
    uint16_t x, y;
    uint16_t foreground, background;
} lcd;

extern lcd display;

typedef struct {
    uint16_t left, right;
    uint16_t top, bottom;
} rectangle;

void init_lcd();
void set_orientation(orientation o);
void clear_screen();
void fill_rectangle(rectangle r, uint16_t col);
void fill_rectangle_indexed(rectangle r, uint16_t* col);
void display_char(char c);
void display_string(char *str);
```