Design of Embedded Systems Advanced Course EDA385
2012-09-11

Labyrinth game

Supervisor: Flavius Gruian

Project Participants:
Fredrik Lundström mt08fl2@student.lth.se
Gustaf Gustafsson mt08gg7@student.lth.se
Robin Lefrant robin.lefrant.947@student.lu.se

Design of Embedded Systems Advanced Course EDA385
2012-09-11

Introduction

The Labyrinth game is a classic game developed by the Swedish company BRIO 1946. A marble should be transported through a labyrinth by tilting the ground of the labyrinth, and the player should avoid getting the marble in any of the holes along the path. If the marble fell into a hole the player has to start over by placing the marble in the beginning of the labyrinth path.

Our variant of the game will use a accelerometer to simulate the movements of the labyrinth ground (the tilt of the playfield) and the labyrinth with the marble will be shown at a ordinary VGA-connected screen.

Implementation

The labyrinth is graphically represented on a screen by the VGA connection. The labyrinth contain walls, holes and a marble that can move in the labyrinth. Different levels may be implemented. Scores can be counted and displayed on the 7 segments display on the board.

The game is controlled by a disc with a accelerometer mounted on it. When the disc is tilted the marble on the screen starts rolling in the tilt direction.
The signals from the accelerometer is sent to the Digilent board and processed.

The physical hardware that should be used is the Digilent Nexys 3 and the PmodACL - 3-axis Accelerometer. The playfield should be visible at a ordinary VGA connected computer screen.

For debugging an UART connection between the Nexys 3 and a computer should be used.

A block diagram for the system is illustrated in Figure 1 below.

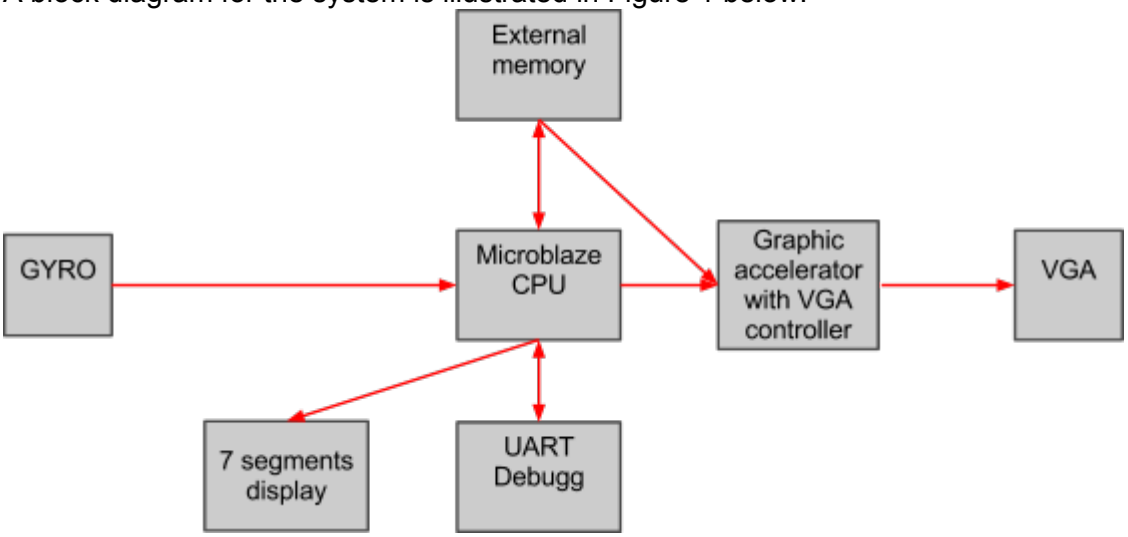

Figure 1. Block diagram for the system.

Hardware Design

The system part that we will implement in hardware is the VGA-controller. It would probably be too complicated and too slow to display something on the screen only with the CPU.

The VGA-controller will be implemented to operate at the resolution 640x480 pixels and the screen will be updated at the frequency of 60 Hz.

The hardware-part will also contain a MicroBlaze processor.

Software Design

All system-parts except the VGA-controller will be implemented in software and executed in the MicroBlaze processor.

The accelerometer-interface will be implemented using the existing libraries for the PmodACL.

The Physics accelerator is the system part that computes the position of the marble. This is made by applying the input from the accelerometer on a mathematical model. Input to this model is also the physical environment of the labyrinth i.e. the positions of the walls and holes. The mathematical model for the position computation is described under the heading: “Mathematical Model”.

The playfield is represented as a bitmap and is imported to the processor via the UART. The information in the bitmap is processed and the information is inserted into a bit-matrix that is easy and fast for the software to handle. The two values that the bit can take means either “free way” or “wall” or “hole”. Wall and hole have the same value and the value is interpreted as a hole if there is free way in all directions around it. The bit-matrix is placed in the BRAM and the original bitmap is placed in the external RAM to get accessible to the graphics accelerator.

The marble falls into the hole when the distance to the centrum of a hole is smaller than a specific value.

The debugging part of the system will be implemented using the UART communication.

Memory usage

The part that consumes most memory will probably be the bit-file with the playfield image. Because the resolution should be 640 * 480 with 8 bit colour the memory needed should be:
640*480*1(byte) = 307200 byte = 2457600 bit which is too much for the BRAM to handle (max 576000 bit). Therefore we will use the external RAM memory on the board.

Mathematical Model

The mathematical model to calculate the position of the marble is described below:

The PmodACL measures the acceleration a in the three directions x, y and z. In this case we just need x and y because the game is just defined for 2 horizontal components.

To get the position of the marble we first integrate the acceleration to get the velocity v and then integrate the velocity to get the position s.

$$v_x = \int a_x \, dt$$
$$v_y = \int a_y \, dt$$

$$s_x = \int v_x \, dt$$
$$s_y = \int v_y \, dt$$

Expected difficulties

Because the transfer speed from the external memory is slow, there can be some problem to recognize a wall or a hole by reading from the memory and do some calculations all the time. If the speed is too slow, we can for example make some simplifications of the wall and hole information and store that information in a array that can be stored in the BRAM. Another solution would be to store the wall dimensions and positions in a c file as constants.

Further improvements

If there is enough time when the implementation mention above is done, some further improvements will be done.

Keyboard

Support for a keyboard can be added to make it possible for the player to write his name. The name can then be stored together with the time for a game in a highscore list.
By using a keyboard command it could be possible to pause and see the highscore list. A pause function could be added using the buttons at the Spartan board too.

Levels

Some different levels (different playgrounds) can be added. It could also be possible to make a level and transfer it via the UART connection to the memory.

Sound

Some sounds can be added and be played for example when the game starts or the marble fall into a hole (different sound for right or wrong hole).
It could also be possible to play some annoying music during the game.

Time plan

The time of the project will be divided into some parts:

- Planning
- Implementation of the VGA interface (hardware).
- Implementation of the Accelerometer interface.
- Implementation of the UART interface.
- Implementation of a Physics accelerator (hardware or software) that governs the speed and direction of the marble movement depending on the lean angle of the disc.
- Implementation of the GUI.
- Writing the report and prepare the presentation.

The time plan per week is presented below. It can be changed during the project if we discover some parts to be more time consuming than expected.

Table 1.

	Fredrik	**Gustaf**	**Robin**
Week 1	Planning	Planning	Planning
Week 2	Accelerometer interface	Accelerometer interface	VGA test
Week 3	Graphical accelerator	Graphical accelerator	Graphical accelerator
Week 4	Physical model	Background from memory	Background from memory
Week 5	Collision handling	Hole falling detection	Collision handling

Week 6	Integration and testing	Integration and testing	Integration and testing
Week 7	Report writing Presentation	Report writing Presentation	Report writing Presentation