Università degli studi di Padova
Department of Information Engineering
Academic year 2021/2022

Master degree in ICT for Internet and Multimedia

AX3: Snake on Aurix board

ICT for Industrial Application

Arash Ahmadian - 2049580
Alexander Bonora - 2057200
Marcello Raimondi - 2007715
Francesco Trolese - 2050691

Contents

1 Introduction

This section offers an overview of the objectives and the components which were used in order to develop the project. A brief description will explain the aim of this work. Then a more detailed list of the physical components and their usage is presented.

1.1 Purpose of the project

The aim of this project was initially to develop the renowned game known as Snake on an AURIX TC375 board connected to a LED panel and a joystick. During the development of the project a necessary change was made in order to turn on the LED panel and the AURIX TC375 board was swapped with an AURIX 397 TFT Board. Once the communication was established we programmed the snake game logic in C language in a way that they could be shown on the LED panel and controlled by the joystick.

1.2 System components

The list below offers an overview of all the hardware components needed to implement this project. The most relevant ones will be described in details in section 2.

AURIX 397 TFT
The AURIX TFT Board was used to establish a master-slave connection to the panel and to read the input from the joystick, the high output voltage was 5V.
Infineon offers a wide range of development tools to work on their board including the Eclipse based IDE AURIX Development Studio, which is a comprehensive environment that includes C-Compiler and Multi-core Debugger.

LED panel
Our LED panel is made up of 2 different components, an 8x8 LED panel and a Serially Interfaced, 8-Digit LED Display Driver (MAX7219). The driver uses SPI (Serial Peripheral Interface) to simplify the synchronous communication with any board.

Joystick
The joystick is composed of two potentiometers, each of these is connected to 5V and grounded. Both these potentiometer send a voltage between 0 and 5V in order to indicate the current position of the joystick in the x and y axes. These voltages are then read by the board.

2 Implementation

This section offers a detailed overview on the code of our program, of the components we have used and how they are connected to the Aurix board.

The logic of the program is split between multiple files. `Cpu0_Main.c` contains the main of the program (that runs on one of the cores of the Aurix TFT board. `game.c` contains the logic of the Snake game, `ADC_queued_scan.c` contains the functions to initialize and read the joystick position, `EVADC.c` provides the code to read the Aurix board potentiometer value and `SPI.c` contains the APIs to drive the LED matrix.

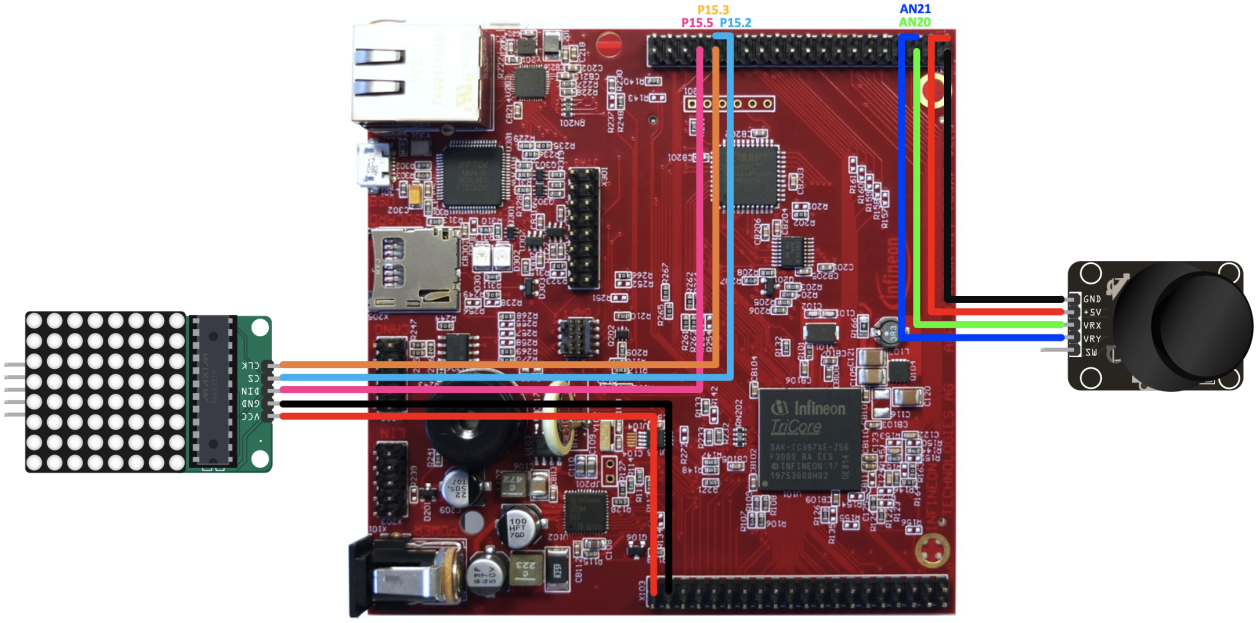

Figure 1: Wiring diagram of the project

2.1 Joystick

The values of the two potentiometers are read from the VRX and VRY pins in the joystick. Once read, they are converted into digital values through the EVADC (Enhanced Versatile Analog-to-Digital Converter) module of the board. This module is configured to measure both analog signals from the joystick in sequence using a queued request. Our implementation is based on Infineon's example project 'ADC Queued Scan 1 for Aurix TC397 TFT'[1]. Importantly, due to the inaccuracy of the joystick, the potentiometers in the rest position return slightly different values each time the program is executed, therefore a calibration function had to be implemented. The purpose of this function is to determine, at the start of the program, which values are returned by the potentiometers when the joystick stays in the rest position.

2.2 Led matrix

The Led matrix is controlled through a master-slave communication implemented by 'SPI CPU 1 for KIT Aurix TC397 TFT' [2]. This library is able to ensure full duplex communication even though in this case we only need to send information from the board to the matrix.

The data is sent from the board in form of 16-bits integer numbers which are then parsed by the display driver. This driver is responsible for the multiplexing of leds when they need to be turned on at the same time. The first part of the integer describes the address of the driver that specifies how the the the other part of the integer gets interpreted. The driver itself has different modes for decoding the input data but the one used here is the "direct decoding" in which we sent the direct cohoridnates of the leds that we want to turn on.

In order to orchestrate the switching on and off of the LEDs a custom library is been created that sets up the intensity and frequency of the led matrix and then sends the data to the led matrix.

2.3 Game logic

The game logic in the file `game.c` was adapted from a project on Github that implements the Snake game on an Arduino board[3] and ported to execute on our Aurix board. The main functions, used to orchestrate all the components and to manage the game logic are two.

- `initGame()` takes care of initialising the components (joystick, board potentiometer and led matrix) and calibrating the rest position of the joystick. This function is only invoked once at the start of the program.
- `runGame()`, after invoking `initGame()` described above, enters an infinite loop in which the following operations are performed:
  1. Generation of an apple in the gameboard if the one present has been eaten.
  2. Reading of the joystick position and computation of the new snake direction.
  3. Update of the snake in the gameboard and evaluation of the game state (normal/win/gameover).
  4. Dump of the gameboard on the console. This step was developed for debugging purposes only, as it allows to test the correctness of the game logic without using the LED matrix. It is only executed when a flag indicating debugging is active.

3 Issues and development

The development for this project was far from linear or devoid of problems. Here we present a summary of all the technical difficulties that were found during development

3.1 Voltage

Due to the power requirements of both the LED panel and the joystick, the Aurix board could not be powered by simply connecting it to the 3.3V micro-usb port from a pc but had to be powered with an external power supply set to 7.5V.

3.2 Board

The board model AURIX 375 was not able to output enough power to substain the led panel, especially when more leds had to be turned on and multiplexed together. Due to this the model had to be changed into AURIX 397 TFT.

3.3 Libraries

While Infineon provides a plethora of options to obtain a master-slave communication with other peripherals, it does not provide any library able to control the led panel. Such a library had to be implemented from scratch in order for the project to work.

References

[1] Infineon. *ADC_Queued_Scan_1 for KIT Aurix TC397 TFT*. URL: `https://www.infineon.com/dgdl/Infineon-AURIX_TC3xx_ADC_Queued_Scan_1_KIT_TC397_TFT-Training-v01_00-EN.pdf?fileId=5546d46274cf54d50174da20c0cf2219` (visited on 06/28/2022).

[2] Infineon. *SPI_CPU_1 for KIT Aurix TC397 TFT*. URL: `https://www.infineon.com/dgdl/Infineon-AURIX_SPI_CPU_1_KIT_TC397_TFT-Training-v01_00-EN.pdf?fileId=5546d46272e49d2a0172e6fcc3a9021e` (visited on 06/28/2022).

[3] ondt. *Arduino snake*. URL: `https://github.com/ondt/arduino-snake` (visited on 06/28/2022).