Documentation and Techincal report for Semestral work Display and keyboard on PCIe bus A0B36APO

Zdenek Brabec, brabezd1@fel.cvut.cz
Vladyslav Gorbunov, gorbuvla@fel.cvut.cz

May 12, 2016

Contents

1 Introduction

This paper aims to introduce our work and implementation of Semestral work for subject A0B36APO. The Semestral Work is to implement simple pager on given Altera device connected to the PC through PCIe bus. The first section is User Guide where all necessary commands for the device can be found. The next section is Technical report, where one can find few words about our implementation.

2 User Guide

2.1 Client Application

Compile source codes and then execute main file with pager id, server ip address and port arguments with following code in Listing 2.1.

```
gcc main.c -o main
./main 123 localhost 55556
```

The pager id is limited to max 4 digits. You need to have running server as its decribed on assignment site[1].

2.2 Key schema

The key schema is same as for mobile phones. The Figure 1 showing the actual device. First row is for keys 7, 8 and 9, second row is for keys 4, 5 and 6, thrid row is for keys 1, 2 and 3 and fourth row is for keys *, 0 and #. The side buttons stands for ENTER (the bottom one) and CANCEL (the top one).

Figure 1: Altera board

[1] https://cw.fel.cvut.cz/wiki/courses/a0b36apo/en/tutorials/10/start

2.3 Device states

The device begins in *Initial state*. The app immediately asks server for any new messages and continues to do it every 5 seconds.

If new message appears the divice displays notice as well as acoustic signalization. User can read this message by pressing ENTER key. In the first row is displayed ID of sender and in second row it is message body. Pressing another ENTER is for confirm server about message read. The device shows messages until every message for this pager id is read. If there is no more message, the device will turn into *Initial state*.

In *Initial state* user can navigate to *New Message state* pressing any key. In this state user can navigate back to *Initial state* either pressing CANCEL button or after 30 seconds of inactivity (no button is pressed). First, user entry the target pager id following it by pressing ENTER key. Next entry is for message itself with maximum of 6 digits. The message is sent by another ENTER key press.

3 Techical report

3.1 Workflow

We used provided project boilerplate from assignment sites, where PCIe card connector was made. Next we familiarized ourselfs with the device. First, we created methods handling LCD display write. Following task was a handler funtion for keyboard, with user-friendly key debouncing. Next we introduce algorihm itself – the loop system for switching our system states.

3.2 Implementation

The following diagram in Figure 2 describes it the best. The program starts with device initialization – the LCD display opening procedures as well as defining initial constants. The algorihm is in fact just a Finite State Machine. The initial state stands for basic loop where on a display is "NO NEW MESSAGES" and application doing two things: checking for any key press and every 5 seconds asking server if new message arrived.

The keyboard is operating in a way where

1. single column is disabled
2. nothing is disabled

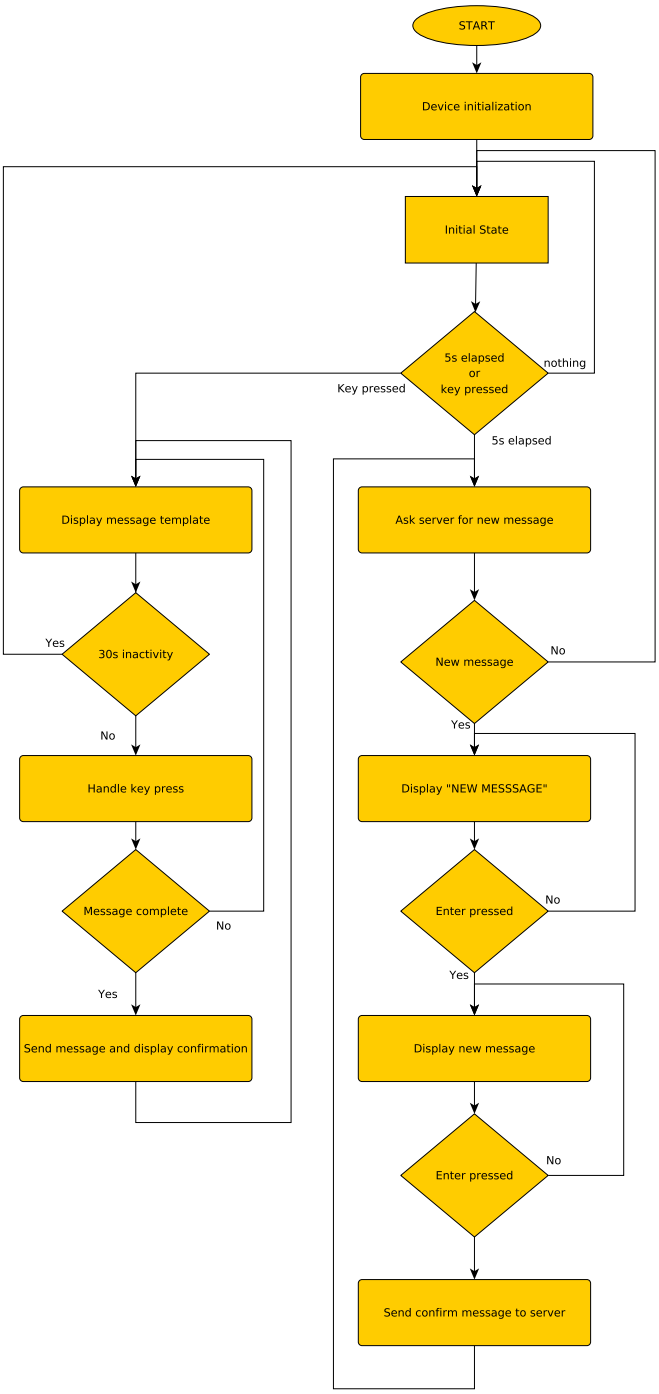
The keypress detection algorithm works as following: First is checking for any key press (no columns are disabled), thus it decodes (from board registers) in which row the pressed key is. Next are one by one disabled columns. For pressed key stands that when key's column is disabled, nothing is read in board registers (0xFF). This is happening really fast – the main algorithm is not blocked as multithread approach was not used. But not that fast to ensure user friendly key debouncing.

Every 5 seconds in *Initial state* is client application asking a server if any new message arrived. If it does, the display shows "NEW MESSAGE, PRESS ENTER TO READ". If ENTER is pressed, automata switches to another state – the message displaying. The message read is confirmed by pressing another ENTER key. This means confirmation message is sent to server, user is noticed that everything went well and automata switch to state where is asking server for any new messages.

If in *Initial state* is pressed any key, automata switches to state for writing new message. In this state the application is checking for 30 seconds of inactivity (no key is pressed). This is done by comparison of system time. If this occurs, automata switches to the *Initial state*. Same thing

can be achieved using CANCEL button. If not, this state is expecting target Id to be entered. The Id must be from 1 to 4 digits long. The Id is confirmed by pressing ENTER key. Then it is message body to be entered. This is restricted to maximum of 6 digits. Another ENTER key sends the message and confirmation is displayed. Automata switches to state where new message to be written.

Figure 2: Simplified algorithm diagram



4 Conclusion

The task is succesfully completed. The most challenging task was to implement proper server communication handling through TCP/IP protocols. Althogh the task and its completation was great exprecience, the development environment could be better. The slow classrom computers with awful preinstalled software were bit depressing.