ECSE 444 Group 14 Initial Report

Kai Fan Zheng
260962377
McGill University
Montreal, Canada
kaifan.zheng@mail.mcgill.ca

Niilo Vuokila
260926706
McGill University
Montreal, Canada
niilo.vuokila@mail.mcgill.ca

Noshin Saiyara Chowdhury
260971544
McGill University
Montreal, Canada
noshin.chowdhury2@mail.mcgill.ca

Sarah Ajji
260925797
McGill University
Montreal, Canada
sarah.ajji@mail.mcgill.ca

I. Overview of the System

The system solves the problem of encrypted inter-device communication. Encryption of messages between devices is necessary so malicious third parties cannot get access to sensitive data or personal messages. Below, in Figure 1, is a diagram overview of the solution to our problem

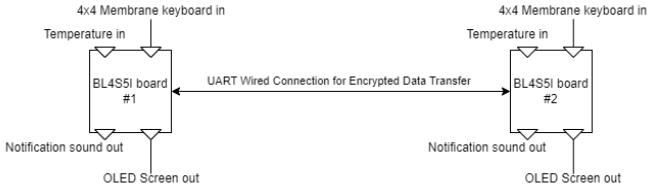

Figure 1. Diagram overview of the system

The system will consist of 2 interconnected boards, each with 2 inputs and 2 outputs. Both will have a user-facing 4x4 membrane keyboard for inputting a message to transmit. Additionally, the system will have a possibility of transmitting temperature data when requested by the other board. This functionality will be mapped to a specific request string that the other board sends. When received, the temperature data is sent and displayed on the requesting board.

The system has a slightly altered design from the proposal. Because of minimal/nonexistent documentation for both the Bluetooth and Wi-Fi on-board chips, we have elected to use a wired UART transfer instead.

II. Description of Design Approach

A. RSA Cryptosystem

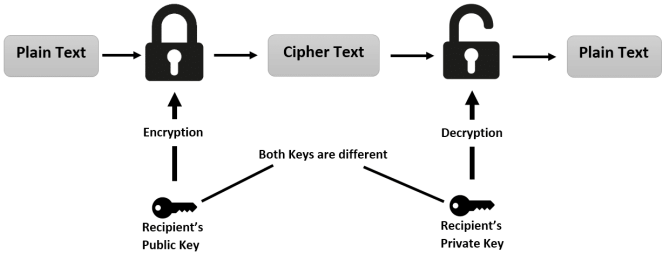

Figure 2. RSA Cryptosystem Overview [1]

For secure communication between the boards, we will be encrypting our messages using RSA Encryption system. Therefore, the message can only be read correctly by the receiver who it is meant for. The system works with the idea that large modular multiplications are hard to reverse, so encryption is based on modular arithmetic. A secret or private key is generated from two (sufficiently large) prime numbers e.g., p and q. The product of p and q, p*q, will form the mod base, n which is part of the public key, which is visible to everyone, and is used to encrypt the message for the receiver. The other component k of the public key is generated by finding a number which is co-prime to both n and (p-1) *(q-1). That is k for the public key, which is the power to which the message should be raised and then mod by n to form the encrypted message. The secret key is the modular inverse of k mod (p-1) *(q-1), which is only known to the receiver and used to decrypt the message. Since n is such a substantial number, it is difficult to obtain p and q from the public key, as well as the modular inverse of k in mod of (p-1) *(q-1) to find the secret key for decryption, therefore making the system secure to outsiders trying to access the message.

B. UART Communication Driver

Since we were not able to establish a communication between the two boards via WIFI in time for the initial demo, we decided to transfer data using UART. UART4 was configured for both boards and a connection was established between the boards where one board acts as a transmitter and the other acts as a receiver. The communication between the boards is established using two functions: *'sendMessagePage'* and *'getMessagePage'*. *'sendMessagePage'* transmits the keypad reading to the UART while *'getMessagePage'* receives the reading from the UART. We tested the communication by implementing *'testSend'* and *'testGet'* functions. *'testSend'* polls for a button press then transmits a char to the UART (48 if the button is pressed and 49 if it is not). *'testGet'* turns the LED on when the data is received through the UART.

C. Temperature and humidity sensors functionality

The temperature and humidity sensor used will be the on-board HTS221 sensors. The sensors will be read only when a request from the other board is received for a reading. The requests will be mapped to a specific input sequence of characters. Reading will be performed using manufacturer-provided board support packages (BSPs).

D. Speaker Output (ringtone)

This component has been completed. The ringtone consists of sinusoidal values for 4 different values in an array of size 22932 bytes, which fits in memory. Thus, there is no need to use the flash memory on the board. Each sinusoidal frequency

also contains one harmonic, to add a more pleasing sound to the tones instead of a bare sinusoid. Since the whole sound data is stored in a single array, DMA is used to playback the sound in normal mode. In this way, the only impact on taking up CPU resources is when the ringtone is requested to be played. The ringtone can be played in a non-blocking manner at any point through '*play_ringtone'*, provided that '*generate_ringtone*' has been called before this.

E. 4x4 Membrane Keyboard Functionality

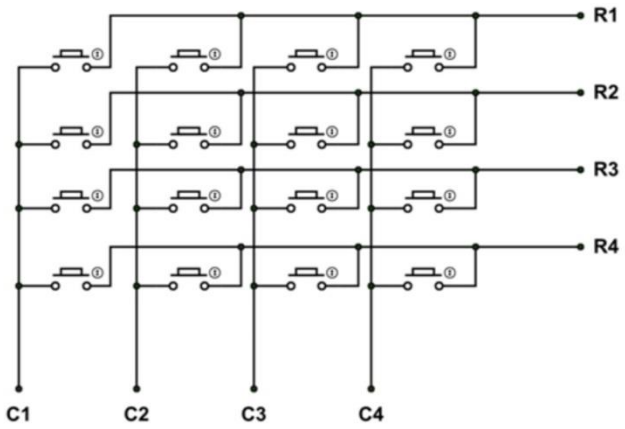

Figure 3. 4x4 Keyboard Diagram [2]

This component has been completed. The 4x4 membrane keyboard functions by polling the row inputs and asserting the column outputs to see the correct button that was pressed. Firstly, then program will be C1, C2, C3, C4 to high voltage output, and read R1, R2, R3, R4 to check if any of the pin read the high voltage signal. For example, if R1 pin reads the high voltage signal, means that one of the buttons in row one is pressed. Once the program detects which row the button locates, the program will set all the column pins to 0 one by one. For example, if program detects the button locates on R1, if setting the C1 to low voltage, and that make R1 read low voltage signal as well, that means the button locates on R1, C1, which is the top left button on the diagram.

F. OLED Screen Output

The OLED screen has a module for I2C communication protocol, so we enabled two i2c pins, D15 and D14. We encapsulate an external open source 128 * 64 OLED screen driver, so that we can pass "printToScreen() " Method prints the string to the screen, and the "clearScreen()" method clears the screen. The font we choose is 7*10, therefore the screen can display 6 lines at once, and each line can display up to 18 characters.

III. Discussion of Results

A. RSA Cryptosystem

The cryptosystem can encode a message and decode a message as expected.

B. UART Communicatication Driver

Currently running into the problem of "HAL_UART_Receive()" does not receive data properly, creating a new project to run tests by receiving serial data from computer and send data to the board to debug the problem.

C. Speaker Output

The speaker output sounds as expected. There may still be some changes made to the melody but overall, it works as intended.

D. 4x4 Membrane Keyboard

The keyboard driver can detect which button is pressed and return a char to represent the key.

E. OLED Screen

The OLED screen driver can print a string to the screen and clear everything on the screen.

IV. Plan for Further Implementation

The plan for further implementation consists of separately completing all components and assembling them at the end before the final demonstration.

For now, a schedule with dates looks as follows:

Date	**Goal**
Nov 22nd	Have UART transfer working, demonstrate inter-device communication with user inputs
Nov 24th	Have all components tested and working individually
Nov 28th	Assemble all components into a working system and add additional functionalities if possible

V. References

[1] javainterviewpoint, "Java RSA Encryption and Decryption Example," Java Interview Point, 11 March 2019. [Online]. Available: https://www.javainterviewpoint.com/rsa-encryption-and-decryption/. [Accessed 21 November 2022].

[2] C. 101, "4x4 Keypad Module," Components 101, 22 March 2018. [Online]. Available: https://components101.com/misc/4x4-keypad-module-pinout-configuration-features-datasheet. [Accessed 21 November 2022].

[3] N. Mohideen, "OLED with Blue Pill using STM32CubeIDE," Micropeta, [Online]. Available: https://www.micropeta.com/video19. [Accessed 21 November 2022].