CM10194 – Computer Systems Architecture: Coursework

Overview

Our project is a prototype for a high-security RFID door lock system that would be used in an area where multi factor authentication (e.g. government building) is necessary. The program involves 2 Arduinos, a master and a slave. The master runs all the hardware components, including an RFID scanner, 2 buttons, a piezo speaker, a screen and a servo (to represent the door lock mechanism). The slave Arduino controls the security represented as a bubble machine, and does all the verification. They're both connected through Bluetooth serial, using HC-05 modules.

Once a face is detected, a timer is started, and the Arduino prompts the user to scan an RFID card. If correct they are then asked to input an answer to the question "What is the answer to the universe?" in binary, if correct again a door will open, this is displayed as a servo. If they scan 3 wrong RFID cards, input the answer to the question wrong 3 times, or run out of time then an alarm is sounded, and the bubble machine turns on. We needed communication in this project because one Arduino is designed for validation of information and one is designed for the input of information. In the final product the one designed for validation (slave), would be in a very secure location so it cannot be tampered with[1].

Software

Face Detection: For the prototype, we've decided to go with facial detection, as it is significantly easier to implement over facial recognition, given the time constraints. In order to do this, we've created a python 3 program using a library called Open CV. This has inbuilt face detection algorithms. Of which we've decided to use the HAAR algorithm (Cascade)[2].

PC-Arduino Coms: For the face detection to be linked with the Arduino, the python program must send a serial command to the master Arduino. This is done using a library called PySerial, and allows for the main program to be run only when a face has been detected for long enough[3].

Master Program: The master program controls most of the components used in the system, including the RFID scanner[5], the TFT screen[7,8], the buttons, and the "door lock", which is a servo in the prototype. The master also has a 30 second, non-intrusive timer that starts when a face has been detected[9].

Master-Slave Coms: The master must communicate with the slave every time an rfid tag is scanned, or a password is entered. This is then checked with the correct/expected output(s), and another signal is sent back to the master[1].

Slave Program: The slave is mostly checking the expected outputs against the actual outputs. However, it's also in charge of controlling the bubble machine whenever the timer runs out, or they get 3 passwords or RFID tags wrong.

Hardware

PC: The computer simply needs python 3.X with the previously mentioned libraries installed, as well as a webcam, and a spare USB port to plug the master in to. In a final version of this program, the PC-Arduino coms might be done through Bluetooth/another wireless communication protocol

Master: The master contains: An RC522 RFID receiver; a TFT screen; 2 buttons (used to represent binary) ; a piezo speaker; a 9g servo and a HC-05 Bluetooth module in master mode. In a final version of this system, a pin pad would replace the buttons, so the pin can be harder to crack (> 256 options)

Slave: The slave contains: A relay connected to a bubble machine and a HC-05 Bluetooth module in slave mode. In a final version of this system, the bubble machine would be replaced with something higher security, such as a taser

Communications: We decided to go for a Bluetooth serial connection between the two Arduinos, because in the final version of this system, one of the modules would be in a different area. This would probably use WiFi/ethernet, however using this for our prototype would have been a challenge due to issues regarding the network used on campus. Using Bluetooth also made development of the project easier, as the Arduinos didn't need to be right next to each other while testing and developing features. We used HC-05 modules because after some research, they seemed to be the best, especially in terms of documentation and support[4].

Difficulties

PC-Arduino Coms: The main issue we had here was involving the fact that we couldn't see the serial display for the master Arduino while the python program was communicating with it.

Master-Slave Coms: The main issue we had when it came to getting the Arduinos to communicate was setting up and connecting the two HC-05 modules together. Once they were together, the process of using them was the same as typical serial if they were wired together

Connecting the TFT Screen and RFID Reader Simultaneously: When we were trying to connect the two devices to the master at the same time, we noticed that some of the pins on each device can't be reassigned using code (MOSI, MISO and SCK) pins were preassigned and need to be used in order for the device to function. After some research, we found out that the 6 female ICSP pins on the Arduino board can be used as secondary pins for such features. Therefore, all we needed to do was plug them into there. We also had to use some of the analogue AX pins for our components, which doesn't cause any extra issues[6].

Fritzing Schematic: While making the schematic, we noticed that the application we were using (Fritzing) did not have all the parts we used, specifically the HC-05 modules and the TFT screen, to solve this we ended up designing the parts ourselves and manually adding them to fritzing

Connecting the Bubble Machine: Due to the bubble machine being an off the shelf product, it wasn't inherently designed to work with the Arduino. This led to power issues whenever the device was plugged in for controlling, due to the machine containing 4 AA batteries (6v power), and a button, which was removed for the process of wiring it up. The solution to this was using a relay and connecting the machine along the closed portion of the switch, with the coil being controlled by a digital out pin. This meant it could then be programmed simply by using digitalWrite(X,HIGH/LOW), just like an LED.

Inherent Issues: Due to time constraints, as well as the limitation of people/knowledge, we wouldn't have been able to complete the full project, as facial recognition is very complex and would take a while to train. However, we made a working prototype that represents the goal of the system.

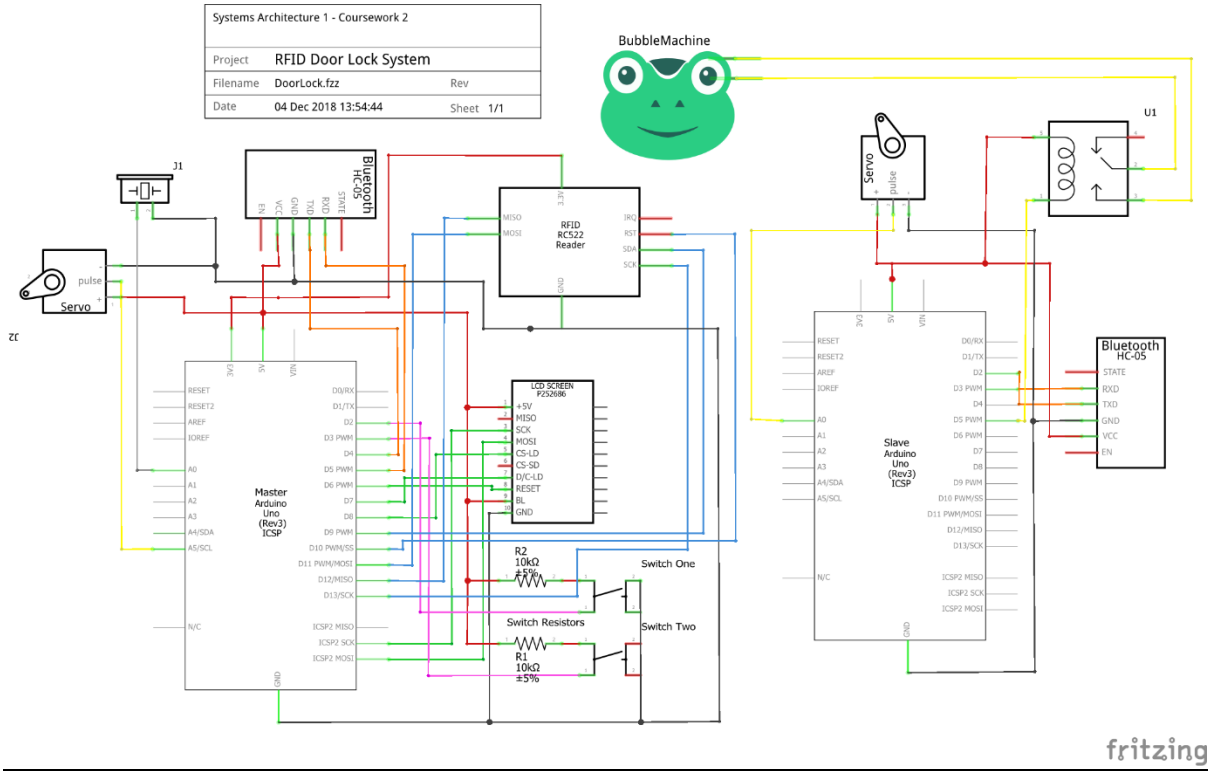
Conclusion

To conclude we think we have created a functioning prototype which could be extended upon to create the final product. We enjoyed doing this project and found it suitably difficult. We think we worked well together and had good communication. We took this as an opportunity to explore things we were interested in, specifically Face detection and RFID cards. This opportunity led to us learning a lot about the subjects and wanting to do more.

CM10194 – Computer Systems Architecture: Coursework

Appendix

Images



References

1) https://www.arduino.cc/reference/en/language/functions/communication/serial/
2) https://docs.opencv.org/3.4.3/d7/d8b/tutorial_py_face_detection.html
3) https://playground.arduino.cc/interfacing/python
4) https://howtomechatronics.com/tutorials/arduino/how-to-configure-pair-two-hc-05-bluetooth-module-master-slave-commands/
5) https://playground.arduino.cc/Learning/MFRC522
6) https://www.arduino.cc/en/Tutorial/AnalogInputPins
7) https://www.arduino.cc/en/Guide/TFT#toc9#
8) https://www.arduino.cc/en/Reference/SPI
9) https://www.forward.com.au/pfod/ArduinoProgramming/TimingDelaysInArduino.html

(libraries used included in separate file)