ADXL345 Knock Pattern Sensor

Allison Pearce

St. Edmund's College

ap819@cam.ac.uk

I. INTRODUCTION

This paper describes a low-power knock pattern sensor built using an Atmel ATmega 644P microcontroller and an ADXL345 triple axis accelerometer. Knock pattern sensors have a number of useful applications. Shock sensors are often used to detect forcible entry into roadside telecommunications cabinets, and when a knock sensor is present, authorized personnel can enter a tap code to override the alarm. Knock patterns can also be used as an unlocking mechanism for a door or safe. Tap authentication has been investigated as a replacement for passwords in mobile devices [1]. Marques et al. claim that tap authentication is comparable to industry standards for security and usability with the added advantage of being inconspicuous. Tapping wireless devices together can also form part of a protocol for communication and information transfer [2]. For example, users can tap two phones together to share an image from one to the other.

The primary goal of this project was to make a functional knock pattern sensor that could compare a reasonably complex knock pattern to a previously recorded pattern. This required sensitive tap detection with minimal false positives and a method of representing and comparing knock patterns. The device also needed to be low power and simple to use.

II. PREPARATION

A. Device Selection

This project required a microcontroller and a sensor to perform knock detection. The Ateml ATmega 644P was selected for the microcontroller because [reverse engineer an answer here. It should probably be about this many words long. Additionally, a second reason would be good, because you want to sound compelling and get in the ballpark of 4000 words lolol]. The system's storage requirements were low, so the XXYB of the ATmega 644P were more than sufficient.

The ADXL345 triple axis accelerometer was chosen as the knock sensor for its sensitivity and ease of use. Alternatives included shock sensors and other types of accelerometers. An accelerometer was preferred to a shock sensor because previous implementations of knock pattern sensors reported that shock sensors were difficult to interpret. Compared to other accelerometers, the ADXL345 provides several unique and advantageous features. Chief among these is its built-in tap detection function. Taps are defined by several parameters that can be used to tune the sensitivity of the knock sensor, and detected taps can trigger an interrupt in the microcontroller. The ADXL345 is a digital accelerometer that uses SPI or I^2C to communicate with the microcontroller. It has one of the lowest power requirements. A power comparison of the ADXL345 and other accelerometers is shown in Table II-A.

The ADXL345 also provides programmable power modes than can be used to further reduce power consumption.

III. IMPLEMENTATION

A. System Overview

The device has two modes of operation, one for recording passcodes and one for comparing input knock patterns to the stored pattern. To enter recording mode, the user holds down a button until a red LED flashes. The user knocks the pattern, and a green LED flashes as each knock is registered. After five seconds of inactivity, the recording period times out, the green LED flashes the pattern back to the user for verification, and the device enters listening mode. In listening mode, the device sleeps until interrupted by a knock. It will record any subsequent knocks until five seconds passes without activity. It then compares the new pattern to the recorded pattern. A green LED flashes if the pattern is correct, otherwise a red LED flashes.

B. Hardware

In addition to the microcontroller and accelerometer, this project required a 5V 100mA power supply, a prototyping board, wires, a programming cable, a four-way switch, two LEDs, one (TODO)pf capacitor, and a button. The switch was necessary because the accelerometer communicates with the microcontroller over SPI, which uses the same pins as the programmer (MOSI, MISO, TODO finish elaborating). When programming the microcontroller, the switch is open(?), and when running the program with the accelerometer, the switch must be closed(?). The microcontroller, accelerometer, and accessories were connected as shown in Figure [TODO].

C. Software

The C program executed by the microcontroller consists of several initialization functions, a simple main loop, interrupt handlers, and functions to record and compare knock patterns.

The initialization functions establish the communication method between the ATmega 644P and the accelerometer, set values for the tap detection parameters and other ADXL345 registers, set parameters for the timer, and enable interrupts. The two most important parameters for tap detection are the tap threshold and the window. The tap threshold determines

Model	Voltage	Current
ADXL 345	2.0-3.6V	40-145μA
ADXL 345	2.0-3.6V	40-145 A
ADXL 345	2.0-3.6V	40-145 A

TABLE I.   ACCELEROMETER POWER REQUIREMENT COMPARISON

the minimum acceleration value that is considered a tap, and the window is the maximum time that the acceleration can remain above the threshold. Adjusting these parameters makes the system more or less sensitive to knocks. The other ADXL345 registers that were set and their values are described in Appendix TODO.

The bulk of the work in the software is triggered from the tap interrupt. Upon receiving the first tap after a period of inactivity, the tap interrupt starts a timer implemented using the ATmega 644P's built-in THING. Subsequent tap interrupts cause the timer value to be recorded and then clear and restart the timer. A timer interrupt is used to prevent overflow errors in the timer's counter. The timer interrupt is also responsible for calling the appropriate function after five seconds of inactivity: in record mode, it flashes the pattern back to the user, and in verify mode, it compares the input pattern to the recorded pattern and flashes the green or red LED depending on if it was correct.

Knock patterns are defined by the intervals between taps and stored as arrays of integers. When comparing patterns, the new pattern is immediately invalidated if it has a different number of knocks than the code pattern. The new pattern is accepted if it has the same number of taps as the code and each interval is within an acceptable range of the code's interval for that position in the sequence. The range can be changed to tune the sensitivity of the device, but 4ms was determined to be a reasonable default.

One optional features of the ADXL345 that was disabled in software is double tap detection. It is possible to make a distinction between single taps and double taps by defining additional parameters and flipping bits in the appropriate registers. Handling single and double taps differently offered no significant advantages for distinguishing different knock patterns, but it would have increased the complexity of how knock patterns are stored and compared.

D. Power Concerns

Low power consumption was a primary goal of the system achieved through a combination of power reduction strategies. The microcontroller enters ADC noise reduction sleep mode whenever five seconds pass without a knock interrupt. This sleep mode provided the greatest possible power reduction while still allowing the microcontroller to respond to knock interrupts. The accelerometer operates in a power saving mode, with all unnecessary functions turned off in the Power Reduction Register (PRR). TODO, TODO, and TODO were disabled. TODO and TODO remained on because TODO1 was necessary for THING1 and THING2 relied on uninterrupted function of TODO2. Though the ADXL345 offers a sleep mode, it was not used because [GREAT REASON OTHER THAN LAZINESS]. Another simple strategy that yielded a huge power reduction was adding pull-up resistors on all inputs to prevent them from floating.

E. Challenges

A difficult step early in the development process was configuring tap detection on the ADXL345. Most of the process is easy to follow, but I did not realize that the SOMETHING register had to be read in the main loop in order for the tap

Pattern Length	% Correct Pattern Accepted	% Incorrect Pattern Rejected
3	100	100
3	100	100
3	100	100
3	100	100
3	100	100
3	100	100
3	100	100
3	100	100
3	100	100
3	100	100

TABLE II.   SENSOR ACCURACY RESULTS

interrupt to fire until I happened to add a line reading from the register for debugging. Suddenly, tap detection worked.

Another step that was less straightforward that expected was determining the sleep mode. Based on the documentation for the ATmega 644P and the ADXL345, I expected TODO sleep mode to work because it says THING. In practice, however, I found that tap interrupts did not wake the microcontroller unless I used ADC noise reduction sleep mode. Someone with a more familiarity with Atmel sleep modes is unlikely to encounter this problem.

IV.   EVALUATION

The goal of the project was to make a low-power knock pattern sensor that was accurate and easy to use. All aspects of this goal were accomplished.

A. Power Consumption

Before any measures were taken to reduce power consumption, the device drew between XXA when idle and XXA when flashing the LEDs. After adding the sleep mode and changing the Power Reduction Register value, this was reduced to XXA when idle and XXA when flashing. Adding the pull-up resistors to prevent inputs from floating further reduced power consumption to between XXA and XXA.

B. Accuracy

The accuracy of the device can be affected to some degree by manipulating the sensitivity of the knock sensor. This is achieved by changing the tap threshold, tap window, and the size of the window used to compare knock intervals. A less sensitive knock sensor would register fewer false positive knocks and might perform better in a noisy environment, but it would require the user to be more deliberate about knocking, because lighter taps would not register. This might make it more challenging to input certain types of knock patterns. A more sensitive sensor, on the other hand, might interpret an accidental hand slip as a knock. Therefore it is better for accuracy if the user is aware of the device's settings and how they affect the expectations for knocking.

Reasonable defaults for the tap threshold, tap window, and comparison window were determined to be X, Y, and Z. Using these values, I tested ten different patterns of varying complexity. I recorded the pattern, input the correct pattern five times and an incorrect pattern five times, and then repeated the process again (so each correct pattern was verified ten times). The results of this experiment are shown in Table IV-B. Precision and recall were computer to be XX and YY, respectively.

This shows that the device is highly accurate. One limitation of this project is that there was limited experimentation involving one user recording the passcode and another user attempting to enter it. This scenario is less importance for tap sensing in mobile phone authentication or and other single-user use cases, but is something that should be explored for cases in which multiple users operate the same device.

C. Ease of Use

The device was not intended to be a complex system, and it provides all the functionality and feedback required with just one button, two LEDs, and the knock sensor. Resetting the code is a simple process, requiring only a button press and demonstration of the new pattern. The meanings of the various LED flashes are intuitive (green flashes to register knocks and replay the pattern, green or red flashes when reporting whether the pattern was correct).

V.   FUTURE WORK

The sensor described in this paper provides all the functionality needed to record and detect knock patterns, but several more sophisticated features could be added. One useful extension would be the ability to store multiple patterns in different categories. This would be useful if the sensor was used to open a door, for example. The owner of the building could have a master pattern and the option to add a one-time passcode for a contractor or a passcode that expires after a set time to be used for guests coming to a party. Other extensions of password modules could also apply, such as providing hints if the user forgets the code. A more sophisticated user interface could be created by adding an LCD screen and a keypad for user inputs other than knock patterns. However, these enhancements would significantly increase the memory and power requirements. Future work could also include incorporating the sensor with an application—attaching it to a servo-based locking mechanism and using it to unlock a door, or integrating it with a laptop or phone to replace the login password.

VI.   SUMMARY

I implemented a low-power knock pattern sensor that is simple to use and adaptable to a variety of applications. The system was built using an Amtega 644P microcontroller and an ADXL345 triple axis accelerometer. The device is highly accurate and the level of sensitivity can be customized to increase security or to reduce interference from noisy environments. The ADXL345 was an excellent choice for the knock sensor, especially because it provides an easily configured tap detection function. The code for storing and comparing knock patterns was both simple and effective, and did not seem to limit the complexity of knock patterns that could be recorded and verified.

REFERENCES

[1] Diogo Marques, Tiago Guerreiro, Luís Duarte, and Luís Carriço. Under the table: Tap authentication for smartphones. In *Proceedings of the 27th International BCS Human Computer Interaction Conference*, BCS-HCI '13, pages 33:1–33:6, Swinton, UK, 2013. British Computer Society.

[2] O.K. THORN. Interacting with devices based on physical device-to-device contact, July 9 2013. US Patent 8,482,403.