Security Count Down Alarm

nPoints BY NTS PRESS

Guided Project Instructions

Figure 1 A Pad Combo Lock

Related Core Concepts:

Keyboard

Seven Segment Display

Buzzer

"You will tackle the task of creating a simplified security system which will keep track of a password"

Learn It!

Home security is an integral part of everyday life. There is a lot of decision making and logic used in security in order to create an infrastructure of complex and redundant encryption. At its core, a security system is meant to prevent an intruder from gaining access to something valuable. The network built for home security combines a wide array of electronics with a single control center. Most people recognize the keypad and sensors on doors and windows as a staple in home security but stop short of realizing the breadth of a truly integrated system with the ability to link you to your home through your phone, monitor the level of interaction with all entrances and even link you to the right points of contact in the case of an emergency. In this module you will tackle the task of creating a simplified security system which will keep track of a password, enter an armed mode and keep track of incorrect password entries and displays that information to users. This module will prepare you for larger systems with more digital I/O and more complex decision making.

Build It!

This module will build on the knowledge gained from the keyboard core concept and add logic to decipher what key was pressed on the keypad. We will also explore the use of Functional Global variables to store pieces of data.

Task 1: Modify the keypad code to use a formula to determine a keypad press. This can be done multiple ways, one example is using the iteration count of the for loop in the code to determine the row you are on then search the received Boolean array for a true value and correlate the two numbers.

Task 2: Create a second loop that pulls the returned character from the first loop and determines a state to enter into based on if the character is a number or letter. Have these states can be cases in a case structure and for now display a string talking about the state you are in. For example if you press a number, have your string indicator read: "You pressed a number".

Task 3: Open the Keypad storage VI and examine it. This structure is called a functional global variable. The purpose of this VI is to store the individual button presses as well as the Password information entered from the keypad. The driving factor in this VI is the uninitialized shift registers. These allow for the VI to be called and execute once storing whatever data is written to them into shift register memory. The enum allows for calling Vis to specify a task for the Keypad Storage.vi to carry out.

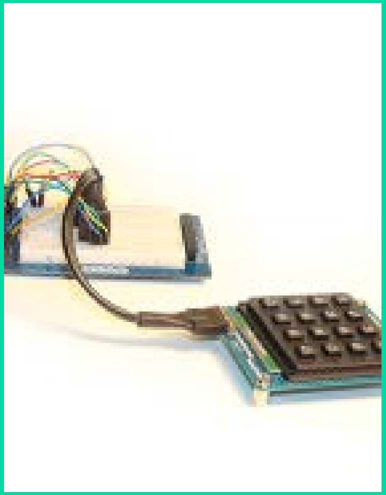


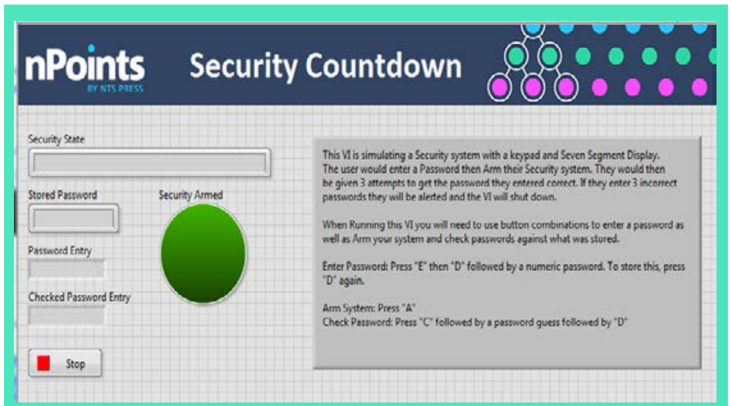
Figure 2 Keyboard Wiring

Task 4: Use the Keypad Storage.vi to store a single button press then modify the second loop to have a case that checks if the letter E then D are pressed. This can be done by storing a previous button press then using a compare VI to check f the current button press is "D" and the stored is "E". Make this case the "Enter Password" case. Either handle code dealing with entering a password here or create a full state machine where this case points to another that does handle the input of a password (several keypad presses).

Task 5: Modify your code to have a case (state) that sets the count of the incorrect password check (Attempt Counter) to 3 and sets the "Security Armed" Boolean to true. Call this the "Armed" state. Update your seven segment display with the Attempt Counter value.

Guiding Questions:

- Can you think of a way to stop your code from reading multiple button presses if the user only pressed a keypad button once?
- What are the different ways to communicate information between loops? Is there a preferred way to do this, discuss pros and cons of the different methods.
- How do you think the character capitalization function on CPU keyboards is implemented? Is there a way to do this with your keypad?



This VI is simulating a Security system with a keypad and Seven Segment Display. The user would enter a Password then Arm their Security system. They would then be given 3 attempts to get the password they entered correct. If they enter 3 incorrect passwords they will be alerted and the VI will shut down.

When Running this VI you will need to use button combinations to enter a password as well as Arm your system and check passwords against what was stored.

Enter Password: Press "E" then "D" followed by a numeric password. To store this, press "D" again.

Arm System: Press "A"
Check Password: Press "C" followed by a password guess followed by "D"

Figure 3 Countdown Alarm Example Front Panel

Task 4: Code the case where the user enters a password by using the Keypad Storage.vi. Make sure to check that the user is not entering in blank data. Include in your code a means of the user indicating they are done entering data. For example, specify a character that if received will exit from the enter password case.

Task 5: Add a case that users can get to by pressing the "C" key that will be a replica of the case where they enter a password, in this case however the password entered will be compared to the password that was stored to see if they entered it correctly. If there is a match then indicate to the user that they got the password correct and set the "Security Armed" Boolean to false. If the entered password does not match the stored password then let the user know then decrement the Attempt Counter. Have a check of the Attempt Counter, if it is 0 then indicate to the user they have exceeded the maximum attempts, send a tone to your buzzer for a short duration then stop your code.

Expand it!

- Add functionality to allow for multiple passwords to be stored. You can use another key press to execute this functionality. When checking passwords, make sure to check against all stored passwords.

Research It!

Functional Global Variable

http://labviewwiki.org/Functional_global_variable

Keyboard Scanning

http://www.emc.com.tw/twn/database/Sa2/Gp/An/Gf/AN-009.pdf