

08

**Fall**

**System Design And**

**Modeling Specifications**

Version 1.1

11/16/2014

Approved and Authored by:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Sean Koppenhafer | Computer Engineer |  | Cameron Tribe | Electrical Engineer |
|  |  |  |  |  |
| Travis Berger | Electrical Engineer |  | Jaime Rodriguez | Electrical Engineer |

**Table of Contents:**

**1 Introduction ……………………………………………………………………………………… 2**

**2 Block Diagram …………………………………………………………………………………… 2**

* 2.1 Level (0) **………………………………………………………………………………..** 2
* 2.1.1 Level (0) Table **…………………………………………………………………**…… 3
* 2.2 Level (1) **………………………………………………………………………………..** 4
* 2.2.1 Level (0) Table **…………………………………………………………………**…… 5

**3 Unified Modeling Language Views ………………………………………………………….. 6**

* 3.1 State Machine View **…………………………………………………………………..** 6
* 3.1.1 State Table/Description **…………………………………………..………………..** 7
* 3.2 Use-Case View **………………………………………………………………………..** 8
* 3.2.1 Program Mode **………………………………………………………………………** 8
* 3.2.1 Unlock Mode **………………………………………………………………………...** 9
* 3.2.1 Reset Mode **………………………………………………………………………….** 9

**1 Introduction**

The Purpose of this document is to give a visual description of the system design from high to low level diagrams and model the behavior seen in the system using a Unified Modeling Language, in different views.

**2 Block Diagrams**

**2.1 Level (0):**

The diagram below gives a high-level view of each sub component being utilized in the system with a description of its corresponding inputs and outputs.

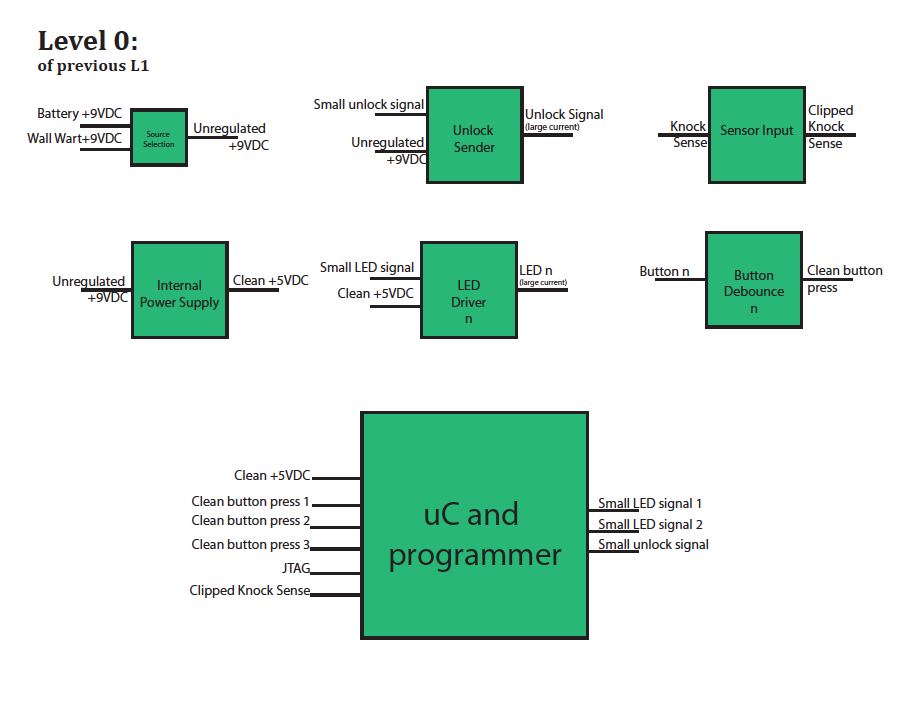
****

Figure 1. Level (0) of Block Diagram of nocLock System

**2.1.1 Level (0) Table:**

The table below gives a brief description of the functionality of each corresponding input and output described in the Level (0) block diagram.

|  |  |  |
| --- | --- | --- |
| **Input Name** | **Input Type** | **Description** |
| Battery +9VDC | Power | A standard 9V battery can be used as a backup in case of power outage. |
| Wall Wart +9VDC | Power | In normal operation, the nocLock uses either a 9VDC AC adapter with a current rating of at lease 1500mA. |
| Button 1 | Digital | Button 1 will be used to initiate the controller and tell it that the user intends to enter a sequence of knocks, It is also how the user tells the system when the knock sequence has ended. |
| Button 2 | Digital | Button 2 will indicate that the user intends to write a new knock sequence. It is also used to navigate through the new knock write sequence. |
| Button 3 | Digital | Button 3 will indicate the user intends to completely erase all stored knocks and start over. This is intended for debugging purposes as a back door to start over; This button will not be easily accessible. |
| Knock Sense | Analog | This analog voltage will indicate when a knock has happened and will include some protection circuitry to reduce any voltages higher than the tolerance of the microcontroller. |
| Programmer | Data | The programmer input is a connection to another programmer, to allow the microcontroller to be programmed. This input will be composed of several lines. |
| **Output Name** | **Output Type** | **Description** |
| LED 1  (High current) | Digital | This will control a LED of a single color to indicate where the user is in the program and it is used for visual queues of confirmation or denial. |
| LED 2  (High Current) | Digital | This will control a LED of a single color to indicate where the user is in the program and it is used for visual queues of confirmation or denial. |
| Unlock Signal  (High Current) | Digital | The unlock signal will most likely be driving by an inductive load that requires a large amounts of current to operate. |

Table 1. Level (0) input and output descriptions

**2.2 Level 1:**

In the Diagram below we see the combination of the high-level to low-level view as the entire system has been connected with the sub-components seen in the Level (0)

Diagram.

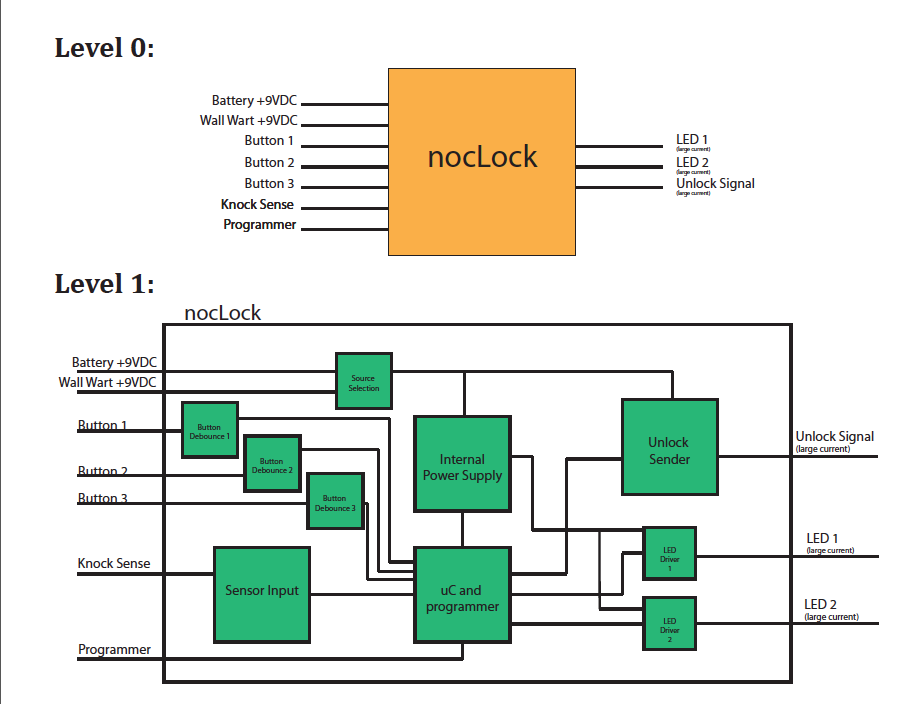
****

Figure 2. Level (1) of Block Diagram of nocLock System

**2.2.1 Level 1 Table:**

Brief description of what is going on

**Missing table Level (1)**

Table 2. Level (1) description of input output functionality

**3 Unified Modeling Language Views**

The Purpose of the UML Views seen in this section are to describe the functionality expected in the nocLock system, Give a visual representation and description of the behavior seen in each state as well give an explanation of each use- case scenario a potential user might encounter.

**3.1 State Machine View:**

The State machine below helps to visual the behavior expected in the system. While supporting the functionality and decomposition of the sub-components seen the block diagrams above.

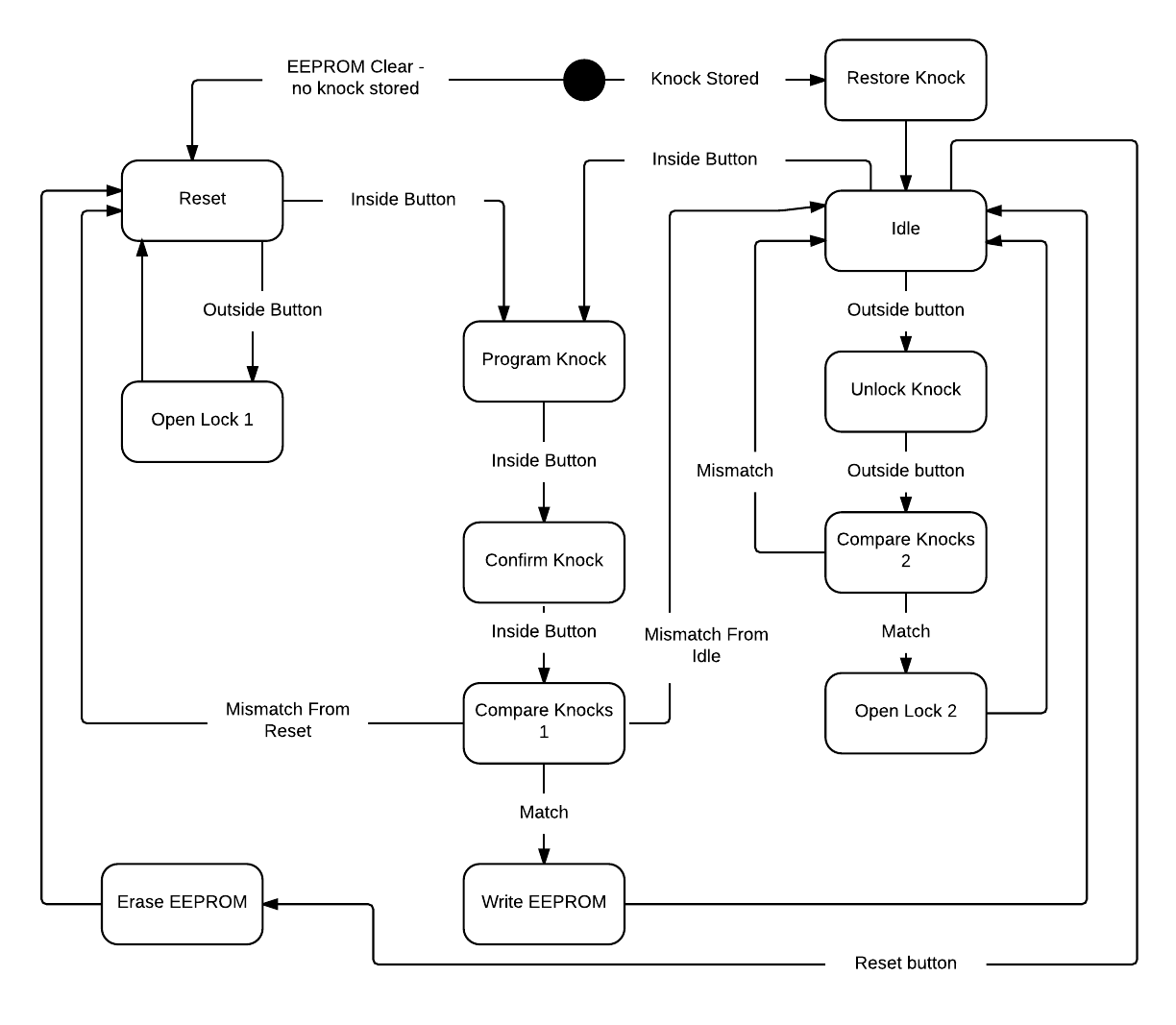


Figure 3. State Machine View of nocLock System

**3.1.1 State Table/Description:**

The Table below states the corresponding inputs and outputs of the state machine above, and gives a brief description of those ports functionality.

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Inputs** | **Outputs** | **Function** |
| Reset | - Outside button  - Inside button | - Red LED on  - Green LED off  - Solenoid closed | No current knock is stored. Allows for the door to be opened without inputting a knock. |
| Open Lock 1 | - None | - Red LED off  - Green LED on  - Solenoid open | Opens the lock for 5 seconds and turns the green LED on to signal that the door is open. Returns to reset after 5 seconds. |
| Program Knock | - Inside button  - 5V analog ADC “knock” value | - Red LED on\*  - Green LED on\*  - Solenoid closed | Checks the ADC value for a knock (defined as a value above a certain thresh hold). Stores the time of each knock in milliseconds. When the inside button is pressed, the sequence is over. |
| Confirm Knock | - Inside button  - 5V analog ADC “knock” value | - Red LED on\*  - Green LED on\*  - Solenoid closed | Briefly, only the green LED is on to signal that the knock was recorded and now the user must confirm their knock. This function does exactly what Program Knock does, recording the sequence to a new area. |
| Compare Knocks 1 | - Program sequence and confirm sequence | - If records match, green LED on  - If no match, red LED on  - Solenoid closed | Compares the program knock and the confirmation knock. If the knocks fall within a certain thresh hold of each other, the knocks match. If they do not match, return to either reset or idle, depending on which one entered the program state. |
| Write EEPROM | - Program knock sequence | - Red LED off  - Green LED on  - Solenoid closed | Writes the newly programmed knock sequence to EEPROM in case the micro controller loses power. |
| Restore Knock | - None | - Red LED on  - Green LED off  - Solenoid closed | Restores a knock sequence from EEPROM after a micro controller restart. |
| Idle | - Inside button  - Outside button  - Reset button | - Red LED on  - Green LED off  - Solenoid closed | Sits idle and waits for one of the three buttons to be pushed |
| Unlock Knock | - Outside button  - 5V analog ADC “knock” value | - Red LED on\*  - Green LED on\*  - Solenoid closed | Much like Program Knock, listen for an ADC knock value above the thresh hold and record the time. When the outside button is pressed, the sequence is done. |
| Compare Knocks 2 | - Programmed knock sequence and unlock sequence | - Red LED on  - Green LED off  - Solenoid closed | Compare the programmed sequence to the unlock sequence in the same way as compare knocks 1. If the knock times fall within a thresh hold of each other, the box is opened. |
| Open Lock 2 | - None | - Red LED off  - Green LED on  - Solenoid open | Opens the lock for 5 seconds and turns the green LED on to signal that the door is open. Returns to idle after 5 seconds. |
| Erase EEPROM | - None | - Red LED on  - Green LED off  - Solenoid closed | Erase the programmed knock sequence in EEPROM and return to reset. This is a debugging feature only. |

Table 3. State Machine input, output functionality descriptions

\* Table note - we are using a bi-colored LED, Having green and red on at the same time creates yellow.

**3.2 Use-Case View**

This view defines the operation of the nocLock system and all

potential use-case scenarios a user can encounter.

**3.2.1 Program Mode:**

|  |  |
| --- | --- |
| Use-Case | Program Mode |
| Actor | User |
| Summary | User programs a unique knock sequence that will be used to unlock the device. The user opens the device and enters program mode by pushing the program button. The user enters the new knock sequence presses the program button and then enters the new knock again to confirm. If the knock sequence is the same for both entries the yellow LED will turn green if it is different for both entries it will turn red. |
| Pre-Conditions | 1. The device is open allowing access to program button. |
| Normal Flow of Events | 1. User opens device. 2. User presses program button. 3. LED turns yellow prompting user to enter new knock sequence. 4. User enters new knock sequence. 5. User presses program button. 6. LED turns green to confirm entry of knock sequence. 7. LED turns yellow prompting user to confirm new knock sequence. 8. User enters knock sequence from step 4. 9. User presses program button. 10. LED turns green validating new knock sequence is programmed. |
| Variations | 1a. User cannot open device  10a. LED turns red notifying user that the knock sequence was incorrect and not programmed to memory. |
| Post Conditions | 1. New knock sequence is stored in memory. |

Table 4. Program Mode description of operation

**3.2.2 Unlock Mode:**

|  |  |
| --- | --- |
| Use-Case | Unlock |
| Actor | User |
| Summary | User enters the correct knock sequence and opens the device. To enter the knock sequence user must press the action button, enter the knock sequence, and press the action button again to submit knock entry. If the knock is correct the device will open. If not the device will remain locked. |
| Pre-Conditions | 1. Knock sequence has been programmed. |
| Normal Flow of Events | 1. User presses action button. 2. LED flashes yellow prompting user to enter knock sequence. 3. User enters knock sequence. 4. User presses action button to submit knock sequence. 5. LED flashes green and device unlocks for five seconds. |
| Variations | 2a. No knock is stored, green LED flashes and device unlocks after action button is pressed.  5a. LED turns red notifying user that the knock sequence was incorrect and device stays locked. |
| Post Conditions | 1. User should close device door. |

Table 5. Unlock Mode description of operation

|  |  |
| --- | --- |
| Use-Case | Reset |
| Actor | User |
| Summary | Hitting the reset button resets the device. EEProm is erased and the device is restored to original settings. |
| Pre-Conditions | 1. The device is open and the user has access to the reset button. |
| Normal Flow of Events | 1. User presses the reset button. 2. Device is reset. |
| Variations | N/A |
| Post Conditions |  |

**3.2.3 Reset Mode:**

Table 6. Reset Mode description of operation