Wireless-Keyless Electronic Door Lock
Andrew Frey
Etec 471-Morton
9, March 2010
Introduction:

The Wireless-Keyless Electronic Door System is a door lock system designed to help the user to never have to worry about forgetting their keys again. The lock system has two main parts: the keypad/transmitter and the receiver/lock. The keypad will attach in a place of convenience close to the door, and the lock will be installed into the door that you wish to secure. If you want to unlock the door, simply enter choose the unlock option and enter your code and the door will unlock. If you want to unlock/lock the door, chose the lock option, enter your security code, and the door will lock without the use of traditional keys.

Description and Sketch:

The Wireless Electronic Door System will be a self contained apparatus consisting of two parts. The first part is the keypad and LCD display. This section will be placed in a place that is most convenient to the user using either small screws or double-sided tape. The second part will be the lock that will replace the current lock in the door. The keypad/ LCD section will not exceed (5 x 3 x 3)” and the lock section will not exceed (5 x 3 x 3)”.

The lock system utilizes one microcontroller and one CPLD. On the transmitting side that includes the keypad and the LCD display there will be a MC9S12C32. On the receiver side that will be the door lock, there will be a Xilinx CoolRunnerII CPLD. Below, figure 1 is a conceptual design for what the final product would look like.

Transmitting Side:

The MC9S12C32 (figure 2) has 32k of flash and 2k of RAM. It also has several power saving modes. Since my device is battery powered, the power saving modes are very important
to increase battery life. The resources used on the 9S12 are the serial peripheral interface (SPI) port and several of the general purpose I/O pins. Shortage of memory will not be a problem on this side of the system.

The transmitting side will handle the user interface. It will take the input from the user and perform the desired function. If they wish to lock/unlock the door, then they must input the correct code and the signal will be sent through the SPI to the RF transceiver to the receiving side.

The major components aside from the MCU are a keypad, LCD display, and the RF module. To get the required 3.3V at the RF modules when the LCD, keypad and MCU all use 5V, I will use a voltage regulator. The pins on the 9S12 that I use are laid out in the block diagram of figure 2. The 9S12 will communicate with the keypad and LCD display through general purpose I/O pins. The RF modules will communicate via the SPI port.

Receiving Side:

The Xilinx CoolRunnerII CPLD that I am using is the XC2C256. It has 256 macrocells and 16 function blocks. On this CPLD I will be using a maximum of 10 of the I/O pins for the RF transceivers and the door lock. This device is designed for low-power applications.

After the signal is sent by the transmitting side, the CPLD will receive the word through the RF transceiver. Then, using VHDL, the word will be changed from a serial word to a parallel to be decoded. If the word received matches the word that is programmed to be valid, then the CPLD will send a signal to the door lock through a driver that will lock/unlock the door.

The major components aside from the CPLD are an RF module, switching regulator and the lock itself. The CPLD will receive the signal using the I/O pins and will then also utilize the I/O pins to drive the motor. On the next page, in figure 2, I have diagrams for my MCU and CPLD.
Software Description:

Transmitting Side:

The 9S12 will be programmed using the C language. I will be using the Micro-COS kernel to control the tasks. On the transmitting side, there will be several modules. The first will have to do with the input from the keypad. When there is a key pushed, it will determine if the key press if valid and if it is a valid input, then the character will be sent to the LCD display module and to the module that will determine if there has been a correct code entered. The LCD module will simply output an asterisk for each key pressed and will output the status of the door, if it has been successfully locked or unlocked. There will also be a module that will receive an input that is deemed to be valid, and will output the lock/unlock signal.
Receiving Side:

Here, there will be a module that will receive the signal and decode it. Since the CPLD processes logic parallel, and the data is coming in through the serial port, there will be a serial to parallel converter so the signal can be processed. If the signal received is valid, it will send an output to a separate module that will active the driver and lock/unlock the door handle.

User Interface:

The user interface consists of programming unlock/lock codes into the system and the interaction of the user entering their code into the keypad to lock or unlock the door. The initial programming for someone wanting to use the system will start when they first turn on the system. A prompt and will ask the user what they wish their pass code to be. After this is programmed into the keypad, the home menu will be displayed. From the home menu, there will be three options. You can add/delete another user code, lock the door, or, if you start typing your code, you can unlock the door. If you choose to add/delete a user code or lock the door, you will be prompted for your security code. Below is a graphical representation of the user interface.

The first time the lock is powered up, you will be prompted to enter the first security code for this system as shown below:

```
ENTER SECURITY CODE: * * * *
```

The user will enter four digits and they will be shown as asterisks on the display. Only numbers 0-9 will be allowed to be input. Once an acceptable code is entered, the screen will display the following:

```
CODE ACCEPTED
```

Once the initial code is input, the screen will display the main menu for the lock system as shown below:

```
UNLOCK CODE: * * * *
# LOCK ADD/DEL
```

If the user starts typing their code, then it defaults to unlocking the door. If the user enters a valid unlock code, the display reads as follows:

```
DOOR UNLOCKED
```

<table>
<thead>
<tr>
<th>E</th>
<th>N</th>
<th>T</th>
<th>E</th>
<th>R</th>
<th>S</th>
<th>E</th>
<th>C</th>
<th>U</th>
<th>R</th>
<th>I</th>
<th>T</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>O</td>
<td>D</td>
<td>E</td>
<td></td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>O</th>
<th>D</th>
<th>E</th>
<th>A</th>
<th>C</th>
<th>C</th>
<th>E</th>
<th>P</th>
<th>T</th>
<th>E</th>
<th>D</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>U</th>
<th>N</th>
<th>L</th>
<th>O</th>
<th>C</th>
<th>K</th>
<th>C</th>
<th>O</th>
<th>D</th>
<th>E</th>
<th></th>
<th>*</th>
<th>*</th>
<th>*</th>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>L</td>
<td>O</td>
<td>C</td>
<td>K</td>
<td></td>
<td>*</td>
<td>A</td>
<td>D</td>
<td>D</td>
<td>/</td>
<td>D</td>
<td>E</td>
<td>L</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D</th>
<th>O</th>
<th>O</th>
<th>R</th>
<th>U</th>
<th>N</th>
<th>L</th>
<th>O</th>
<th>C</th>
<th>K</th>
<th>E</th>
<th>D</th>
</tr>
</thead>
</table>

5
If the # key is pressed from the main menu, meaning the user has chosen the “lock” function, the screen will display:

ENTER CODE: * * * *

The asterisks are output when the user enters their code.

If a valid code is entered, then the screen will read:

DOOR LOCKED

If a code is entered that is not recognized by the lock system, then the following message is displayed:

INVALID CODE!

It then will return to the main menu.

If the user selects the * key, meaning they want to add or delete a user, then the display will look like this:

ENTER SECURITY CODE: * * * *

Only a user with an active, valid code can advance past this screen. If an invalid code is entered, the “Invalid Code” message above is displayed and the main menu is shown again.

If a valid code is entered, then we move onto the following screen:

1 ADD CODE 2 DEL CODE 3 MAIN

Here, the user can choose to add a code, delete a code, or return to the main menu.

If they chose to add a user code, the following sequence takes place:

ENTER NEW CODE: * * * *

After a code is entered, the display will read:

CONFIRM NEW CODE: 1 YES 2 NO

If the user confirms, then the following is displayed:

USER ADDED!

And then the screen will go back to the add/delete/main screen. If the user doesn’t confirm, it will also revert to the add/delete/main screen.
If from the add/delete/main screen, the user selects “del code” then the sequence is similar to add code, with a couple changes. The first screen will look like this:

```
ENTER CODE FOR DELETION: * * * *
```

Once a valid code is input, the screen will read:

```
CONFIRM DELETE CODE: 1 YES 2 NO
```

If the deletion is confirmed, the screen will display:

```
CODE DELETED!
```

And will revert back to the add/delete/main screen. If the user doesn’t confirm the deletion, it will return to the add/delete/main screen.

If at any point during the add or delete process, an invalid code is entered, the “Invalid Code!” message will be output and then it will revert back to the add/delete/main screen.

On the next page is a state diagram for the lock system.
Initial Programming
First code entered and saved

Inactivity

Button Pressed

Home Menu:
*: Add/Delete User
#: Lock Door
Unlock door: Enter Code

Enter Authorization Code

Incorrect Code

1: Add User
2: Delete User
3: Home Menu

Enter Authorization Code

Incorrect Code

Signal Sent
Door Unlocked

Signal Sent
Door locked

Enter User Code to be deleted

Confirm
1: Yes
2: No

Enter New User Code

Confirm
1: Yes
2: No

Incorrect Code

Code Accepted

Code Accepted
After an inactivity period of 3 minutes, the system will enter a low power sleep mode to conserve power.

The FCC requires RF signal energy to be in the range of 9kHz to 3,000,000MHz. The RF modules I will utilize will operate at 2.4GHz and have been FCC tested and approved. They will be sending a unique 8-bit word that will be sent by the 9S12C and decoded by the CPLD.

Development Plan:

Due to the amount of time it can take to receive parts in the mail, the first part of my plan is to order the necessary parts that I require. After they show up I can begin to test the parts to make sure that they can function as I need them to. I can then start to hook up my circuits and once I verify they are working, I can start on the software. The C program and the VHDL module can be adjusted and worked on throughout the development. Below is my weekly schedule for Winter quarter.

<table>
<thead>
<tr>
<th>Week</th>
<th>Task to be accomplished</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Order/purchase parts</td>
</tr>
<tr>
<td>2</td>
<td>Receive and test parts</td>
</tr>
<tr>
<td>3</td>
<td>Start circuit construction</td>
</tr>
<tr>
<td>4</td>
<td>Circuit construction (kepad)</td>
</tr>
<tr>
<td>5</td>
<td>Circuit construction (LCD)</td>
</tr>
<tr>
<td>6</td>
<td>Circuit construction, have keypad and LCD integrated, start on RF modules</td>
</tr>
<tr>
<td>7</td>
<td>Test circuit</td>
</tr>
<tr>
<td>8</td>
<td>Finish circuit construction</td>
</tr>
<tr>
<td>9</td>
<td>Start software</td>
</tr>
<tr>
<td>10</td>
<td>Be able to communicate with MCU and CPLD</td>
</tr>
</tbody>
</table>

And now the schedule for Spring quarter:

<table>
<thead>
<tr>
<th>Week</th>
<th>Task to be accomplished</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Program password into MCU and CPLD</td>
</tr>
<tr>
<td>2</td>
<td>Write keypad module/test keypad/LCD interaction</td>
</tr>
<tr>
<td>3</td>
<td>Write LCD module/test/hardware review</td>
</tr>
<tr>
<td>4</td>
<td>Begin RF testing, Hardware documents due</td>
</tr>
<tr>
<td>5</td>
<td>Interface with RF modules</td>
</tr>
<tr>
<td>6</td>
<td>Test software/hardware interaction</td>
</tr>
<tr>
<td>7</td>
<td>Test software/hardware interaction</td>
</tr>
<tr>
<td>8</td>
<td>Test project/adjust/code review</td>
</tr>
<tr>
<td>9</td>
<td>Test project/adjust/code review/start practicing presentation</td>
</tr>
<tr>
<td>10</td>
<td>Final Presentation</td>
</tr>
</tbody>
</table>

On top of the basic hardware, consisting of the LCD, keypad and MCU, and software, I will also have to get a servo to control the lock. This will be attached to the CPLD through the general purpose I/O and the driver. To present it, I will have the keypad programmed. I will then demonstrate what happens when incorrect codes are entered and also will display the lock.
working correctly. I will also add and delete extra user codes. I will be using a 9S12C as my MCU. Also, my power saving techniques will be displayed.

**Electrical Specifications:**

FCC Regulations: frequency energy between 9kHz and 3,000,000MHz

- My RF Modules at 2.4GHz conform to 47 CFR sec 15

Operating Temperature: -10 to 55°C

Number of Passwords: Max of 8

Max Range: 20ft

**Transmitting Side:**

MCU supply: 5V

Batteries to be used: 4 Duracell AA batteries (2850 mAh)

Estimate Battery Life: 60 Days

**Receiving Side:**

CPLD supply: 3.3V

Batteries to be used: 3 Duracell AA batteries (2850 mAh)

Estimated Battery Life: 60 Days

**Preliminary Parts List:**

**Transmitting Side:**

<table>
<thead>
<tr>
<th>Part</th>
<th>Cost</th>
<th>Lead Times</th>
<th>Power Dissipation</th>
<th>Sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF Module</td>
<td>$25.02</td>
<td>3-5 Days</td>
<td>26.2mW</td>
<td>SparkFun</td>
</tr>
<tr>
<td>Resistors</td>
<td>$0.20</td>
<td>1-2 Days</td>
<td>50mW</td>
<td>ET341 Store room</td>
</tr>
<tr>
<td>Keypad</td>
<td>$21.20</td>
<td>2-3 Weeks</td>
<td>-</td>
<td>Grayhill</td>
</tr>
<tr>
<td>LCD Module</td>
<td>$9.11</td>
<td>10-14 Days</td>
<td>8.25mW</td>
<td>DigiKey</td>
</tr>
<tr>
<td>MC9SC32</td>
<td>$8.13</td>
<td>3-4 Weeks</td>
<td>75mW</td>
<td>Freescale</td>
</tr>
<tr>
<td>Batteries(AA)</td>
<td>$4.99</td>
<td>3-5 Days</td>
<td>-</td>
<td>RadioShack</td>
</tr>
<tr>
<td>Voltage Regulator</td>
<td>$2.00</td>
<td>1-2 Weeks</td>
<td>500mW</td>
<td>Mouser</td>
</tr>
<tr>
<td>Total</td>
<td>$70.65</td>
<td></td>
<td>606.25mW</td>
<td></td>
</tr>
</tbody>
</table>
### Receiving Side:

<table>
<thead>
<tr>
<th>Part</th>
<th>Cost</th>
<th>Lead Time</th>
<th>Power Dissipation</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoolRunnerII</td>
<td>$16.25</td>
<td>10-14 Days</td>
<td>.4mW</td>
<td>Xilinx</td>
</tr>
<tr>
<td>Switching</td>
<td>$2.00</td>
<td>2-3 Weeks</td>
<td>700mW</td>
<td>Mouser</td>
</tr>
<tr>
<td>Regulator</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resistors</td>
<td>$0.20</td>
<td>1-2 Days</td>
<td>50mW</td>
<td>ET341</td>
</tr>
<tr>
<td>Electric Lock</td>
<td>$64.00</td>
<td>1-2 Weeks</td>
<td>120mW</td>
<td>Lowes</td>
</tr>
<tr>
<td>RF Module</td>
<td>$25.02</td>
<td>3-5 Days</td>
<td>21.2mW</td>
<td>SparkFun</td>
</tr>
<tr>
<td>Batteries(AA)</td>
<td>$4.99</td>
<td>3.5 Days</td>
<td>-</td>
<td>RadioShack</td>
</tr>
<tr>
<td>Total</td>
<td>$112.46</td>
<td></td>
<td>891.6mW</td>
<td></td>
</tr>
</tbody>
</table>