Team Name: The Marmosets
Team Members: Kyle Pinto, Efaz Shikder, Hemit Shah
Team Number: 24

Embedded Systems Project Proposal

Background

Students living in dorms are bound by their contracts to not modify (i.e. damage) any parts of their room. This prevents them from implementing commercially available smart technologies for homes in their dorms such as smart lights and locks.

We propose creating an affordable and contract-friendly smart dorm system to help students transition into the modern era of living. This system would comprise of an automatic smart lock accessible wirelessly (by creating a server on the pi), and potentially an automatic light control system that is based on the same framework as the lock mechanism.

Function

The smart lock would comprise of a laser and photodiode based system alongside a servo that would control the existing deadbolt in a student dorm. The laser would be mounted off the door and the photodiode on the door such that when the door is closed, the laser will hit the photodiode and the system will determine that the door is closed by means of a state machine. Additionally, a servo motor will be used to turn the deadbolt between the locked and unlocked positions. These functions will be controlled using a state machine.

The auto-locking system would be dependent on the user selecting an auto-lock on or off state. In the auto-lock on state, the system would automatically lock the door within a user-defined period of time after the door is closed.

If off, the system would only lock and unlock if the user manually selects these lock states. The user would have the option of remotely locking and unlocking their door by using an additional Raspberry Pi to receive input. The “remote control” could sense user input using a simple push button, and would also display the status of the door lock using LED lights. Since the Pi is able to connect to the campus wifi network, instructions can be sent wirelessly between the electronic door lock and the user controller. Alternatively, the “ecelinux4” server may be used to transfer instruction files (.txt format) between the Pis.

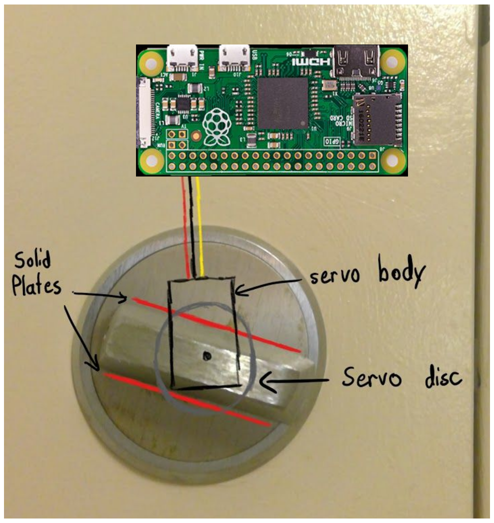
Additionally, since it may be useful for the user to set specific times to lock and unlock their doors we will create a lock.cfg file that holds the input parameters on times to auto-lock and auto-unlock the door on certain dates or repeating weekdays that will be read by the auto-lock program. This program will also write to a lock.stat file when the user locks and unlocks the door and over time analyze what times the user comes into and leaves their room on average as this data may be useful for the user as well.

Considering that this system could have critical failures that may leave the room locked while the student is outside without any method to get inside, we will have to implement a watchdog and run the program as a service that starts on system startup. It will ensure that if the system crashes, the lock will revert to its previous state recorded in the log file.

Equipment Required

Equipment	Justification	Source	Cost
(3) Raspberry Pis	Run program, communicate button presses for manual controlling of auto-lock	ECE150 Materials	$0
(3) Laser diodes	laser beam creation	ECE150 Materials	$0
(3) Photodiodes	laser beam detection	ECE150 Materials	$0
(1) Servo motor	lock and unlock door	RidgidWare / Radioshack	~$15
(2) Push Buttons	user input on remote control	Personal Collection	$0
(2) LED lights	display status of lock	Personal Collection	$0
Resistors & Capacitors	Signal conditioning I/O, current regulation, etc.	ECE150 Materials & Personal Collection	$0

Physical Apparatus and Design



The rough diagram on left displays the proposed physical apparatus that will be used to control the deadbolt in student dorms. The solid plates labelled in the diagram will be mounted on the servo disc such that they will encapsulate the deadbolt on either side and allow the servo to exert torque on it to lock or unlock the door. This servo will be connected to a nearby Raspberry Pi. Both will be mounted using an easily removable substance such as double sided tape.

The servo that will be used will be a standard-sized hobby servo motor. Some possible options are the Futaba S3004 or the Hi-Tec HS-422 standard size servo motors, which are available in hobby and electronics stores. These servos contain plastic gearings, but if a more robust motor is needed, a servo with a metal gearing may be used instead. One challenge associated with using a servo motor is its dependency on an adjustable PWM signal. In order to direct the servo to a desired position, the controller must be able to modulate and output a PWM signal that is within the required frequency range. Fortunately, the Raspberry Pi contains multiple GPIO pins that are capable of outputting PWM signals. Additionally, there exist multiple stable library releases that enable control of advanced GPIO functions such as these. WiringPi and PIGPIO are two popular libraries that enable PWM functionality of the GPIO pins using the c programming language, and one of these will be used to control the servo motor. It should also be noted that the servo motor will be powered by an external power source, presumably a 9V battery converted to 5V on separate board. This will prevent the Pi from being damaged if the servo draws too much power.

Potential Extensions

The automatic light control framework would be based on the on the same servo motor framework as well as the laser and photodiode system. The double laser and photodiode system would detect if a user has entered the room or left the room by means of a state machine and set the light states to be on or off depending on these states. This automatic control would also be dependent on the user selecting if it should be on or off.

Additionally the auto-light system would record whenever the user manually turns the light on and off and write this information to a log file that would be used for future reference. An example application of this would be that if the user turns on the light on average past 8 every day (past sundown when there's no natural light), over a period of a week if the program detects this as a consistent change it would record this as a new permanent setting and turn on the light automatically after 8 everyday. Or another method would be to use a sensor to detect the ambient light and turn on the lights when it goes past a set brightness level set by the user.

The auto-light system could also be used as an alarm aid. The user could write their alarm time to a configuration file that the auto-light program would read and use to turn on the lights at the alarm time every morning.