Software Descriptions

**Introduction**

There are two major software components that run independently of each other and communicate via the network, each with many smaller components to complete the tasks. The card reader software encapsulates everything required to read the card reader data, communicate with the server, and unlock the door strike. The server waits for data from the card reader, checks to see if there is a match in its list, and replies to the card reader with unlock or remain locked.

**State Chart for Card Reader**

![State Chart for Card Reader](image)

**Card Reader Implementation Details**

Card Swipe Component:

The physical card reader outputs data serially in 5-bit “words” that consist of 4-bits of useful data and one parity bit. For reading convenience it is wired to a CPLD to better interface with the ATmega32. The CPLD shifts the serial data from the card reader and filters out the useful data to output in parallel to the ATmega32. A flag is set when appropriate data is shifted in place to be read.
Checking the Master List:

Programmed into nonvolatile memory will be a list of “Master IDs” (TA’s or Faculty) so that in the case of network failure, access can still be granted. The ATmega32 compares the data from the card swipe with its list and grants or denies access.

Communication with Server:

The ENC28j60 Ethernet controller performs the actual communication over the internet. It communicates with the ATmega32 with SPI. For simplicity UDP/IP will be used instead of TCP/IP because data are small enough to send all at once and if packets are lost over the network the card can simply be swiped again. (Note: The Server IP needs to be fixed or each card reader must be reprogrammed each time)

Unlocking the Door:

The door strike is wired to a 12V source and an NFET acting as a switch between ground. The ATmega32 can simply toggle a bit to close the switch on the NFET and power the door strike.

State Chart for Server

![State Chart]

Server Implementation Details

Communicating with Card Reader:

The server spins in a loop waiting to receive UDP packets from a card reader. When packets are received the source address is saved so that it can later reply.

ID List:

The list of acceptable UFIDs will be stored in an external file. Because each card reader will have a separate list of UFIDs, identifiers will be given to each list. The final data structure will be much
like a hash-table (each list will be its own vector/array). The card reader will send along with the UFID, its identifier so that the correct list can be compared.

**Additional Components not in Main Flow Chart**

Programming Mode:

In addition to the main functionality of the card reader system, a secondary mode of operation is also needed to setup the ID lists. When the server enters programming mode, the option to append UFIDs to an existing room (list or master list on card reader) or create a new room, or erasing lists. Once in this mode, cards swiped on the programming card reader will have their IDs stored to the appropriate list. The reason why this mode is not part of the main flow chart is because ideally this function would run concurrently with the main server software. In other words, entering programming mode should not stop the server from communicating with the other card readers. Implementing this would require some sort of interrupt request on receiving packets from the network and the feasibility of this is unknown.