THAT Home Automation Topology (THAT)

Digital Thermostat Module
Electronic Access Module

Design Proposal

2009/12/13

Project By:
Nick Viera
Chris Miller

Advised By:
Dr. James Irwin
Dr. Aleksander Malinowski

Table of Contents

1. Project Introduction

The terms “home automation” and “building automation” are often used to describe a wide array of products and systems. These products aim to provide the user(s) with better, more-intelligent control over their environment as it relates to a home, building, or other indoor space. Unfortunately, most “automation” products are either inexpensive, simplistic, and severely limited in functionality, or they are very expensive, complex, and functionally chaotic.

THAT Home Automation Topology, also known as THAT System, describes a new, comprehensive, IP/Ethernet-based home automation system. THAT System is designed to be as modular and economically feasible as possible, while retaining a rich, usable feature set.

The scope of this project involves the creation and definition of THAT System, as well as the design and construction of two (2) modules for use with THAT System. Nick Viera and Chris Miller will co-develop the framework that defines THAT System. In addition, Nick Viera will develop a THAT-compatible “Digital Thermostat Module.” Chris Miller will develop a THAT-compatible “Electronic Access Module.”

2.1. Modules and System Interaction

THAT System is defined to be used in (but not limited to) two (2) basic system configurations: minimalistic and comprehensive. The minimalistic configuration describes a “system” consisting of only a couple THAT modules. These modules operate essentially as “stand-alone” devices, with little or no communication between modules.

The comprehensive configuration describes a system consisting of a few to many modules, with a “master” computer. In this system the computer is the master arbitrator of the system, using information from some modules to dictate the behavior of other modules. The computer shall run comprehensive control software writing in Python or similar cross-platform compatible software. Should the computer fail or otherwise disappear from the network, certain modules controlling critical systems shall revert to self-contained, “fail-safe” modes.

Possible modules to be designed for THAT System are listed in Figure 4-1. THAT System will, however, work with many different modules above and beyond those listed in Figure 4-1 due to its modular, open-ended design.

2.2. Overall Design Goals

- Modularity on the lowest feasible level.
- Standardization of hardware to the largest extent possible.
- Standardization of communication to the largest extent possible.
- Standardization of master-to-modules system control, including “fail-safe” modes.
- Form follows function.
- Design integrity takes precedence over design cost.
- Open Source Software for most functionality
- Open Hardware for most functionality
- “Freemium” philosophy for advanced functionality

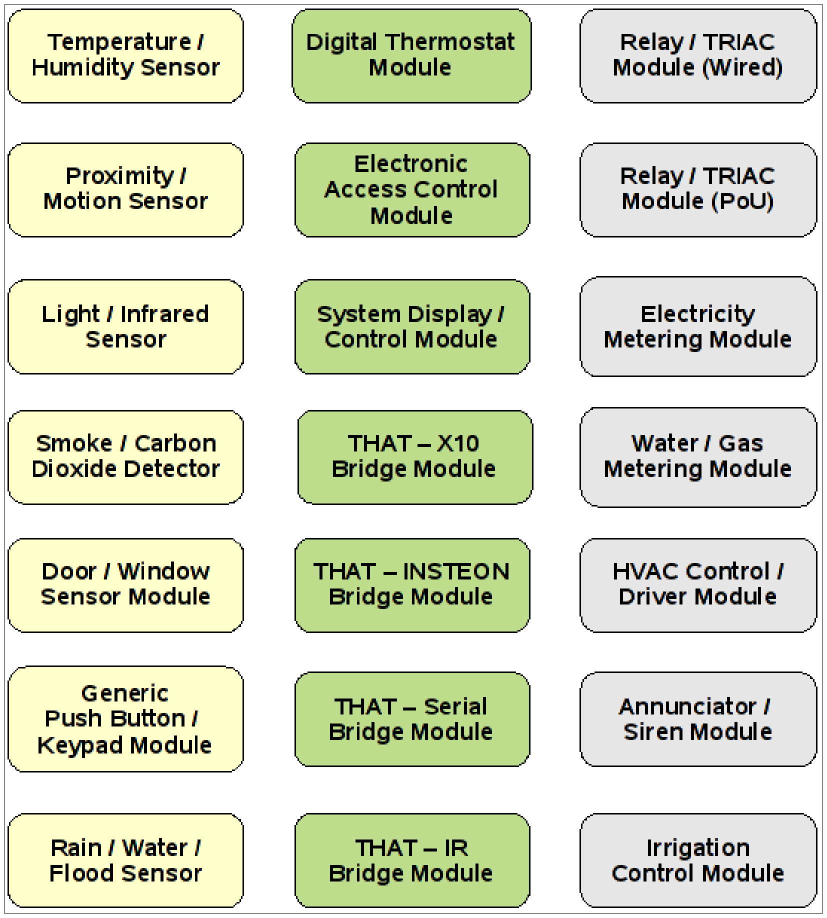

Temperature / Humidity Sensor
Digital Thermostat Module
Relay / TRIAC Module (Wired)
Proximity / Motion Sensor
Electronic Access Control Module
Relay / TRIAC Module (PoU)
Light / Infrared Sensor
System Display / Control Module
Electricity Metering Module
Smoke / Carbon Dioxide Detector
THAT – X10 Bridge Module
Water / Gas Metering Module
Door / Window Sensor Module
THAT – INSTEON Bridge Module
HVAC Control / Driver Module
Generic Push Button / Keypad Module
THAT – Serial Bridge Module
Annunciator / Siren Module
Rain / Water / Flood Sensor
THAT – IR Bridge Module
Irrigation Control Module

Figure 4-1: Possible Modules for THAT System

2.3. Hardware Requirements

- Link protocol: 10BASE-T, 100BASE-TX Ethernet
- Data port(s): 8P8C Modular jack
- External power port: 2.1mm barrel jack
- Primary power supply: IEEE 802.3 Power over Ethernet (PoE)
- Secondary power supply: 12-48 VDC or 9-30 VAC [1]

[1] The presence of the secondary AC/DC power supply, such as that from a “wall-wart” or other point-of-use power supply, shall take precedence over (and disable) the sourcing of any power from the PoE supply.

2.4. Communication Requirements

- Data transport protocol: TCP
- Internet protocol: IP [2]
- Data port: 8428
- Host type: Server [3]
- Provisions for master-control of a THAT system using a software client. [4]

[2] Initial THAT modules shall support IP, version 4. While modules can support IP, version 6, this support is not required for THAT conformance.

[3] All modules shall default to being servers on the network, listening on port 8428. Modules are allowed to optionally be set to communicate as clients and/or using other TCP ports. However, such behavior shall never be the default mode of operation.

[4] A “master” control software suite is planned for THAT system. The software shall be written using the Python programming language and should act as a client on the network to send and receive data with as many modules as necessary. Additionally, the software will provide more robust control of THAT system through its position as a “master” data controller.

3. Digital Thermostat Module

3.1. Introduction

The Digital Thermostat Module (version 1.0), code-named COPTA, is an advanced control module for use with THAT System. As such, the COPTA design shares the basic goals and requirements defined for THAT System. This module shall be developed by Nick Viera in collaboration with Chris Miller.

COPTA is an advanced, digital thermostat with an advanced feature set and programmable control. It is designed for use with most standardized residential and light-commercial, single or split-unit HVAC systems. COPTA shall perform most basic functions as a stand-alone device or it can be integrated into a larger THAT System to provide maximum flexibility and functionality.

3.2. Hardware Specifications

The COPTA hardware shall be based on an 8-bit microcontroller system. Communication in conformance with THAT will be made possible using a hardware Ethernet controller designed for 10/100Base-TX data rates.

Most of the hardware for COPTA falls into the category of general digital I/O. However, some aspects of the design will utilize more advanced communication interfaces, such as synchronous 2-wire serial (I2C) and 3-wire serial (SPI) interfaces. These interfaces shall be used for communication between the microcontroller and the temperature, humidity, and Ethernet ICs.

The power supply for this module shall comply with THAT System power specifications. All on-board outputs for basic HVAC control will be in the form of mechanical relays. Other control outputs shall be made possible using an external relay module.

The direct user interface, consisting of six (6) pushbuttons, four (4) LEDs, and the LCD screen, shall allow for direct control and programming of the thermostat. The following list includes hardware design goals for the Digital Thermostat Module.

- Graphic, back-lit, monochromatic LCD display.
- Integral temperature and humidity sensors.
- Integral outputs for simple 24VAC HVAC system control (Heat, A/C, Fan).
- Support for advanced 24VAC HVAC system control (w/ separate relay module).
- LED indicator lamps for easy system status notification.
- Real time clock with calendar.
- Non-volatile memory for storing "permanent" system settings.
- Battery-backup for temporary settings and RTC.
- Infrared Receiver

3.3. Software Specifications

The base firmware for COPTA shall be written using C and assembly programming languages, as necessary. The firmware shall implement both the functionality necessary for thermostat operation, and the Ethernet / TCP stack necessary for communication with other THAT modules.

Additional software residing on a computer, PDA, cell phone, etc. can allow for the remote control of the thermostat module. Such software could provide further options for advanced control and integration of the module into a larger system of THAT modules. Overall software design goals are listed below:

- Modularity in design.
- Configurable support for single and multi-stage A/C and heat-pump systems.
- Configurable support for controlling dynamically variable HVAC systems.
- Configurable support for reading external temperature/humidity sensors.
- “Learning” of codes from Infrared remotes.

3.4. Design Concept

The COPTA module shall be a stand-alone unit capable of directly replacing any “standard” wall-mount thermostat. To achieve this goal, the unit shall not be too large in size and shall be as aesthetically pleasing as possible. The initial physical concept design for COPTA is a unit approximately 4.5” in diameter, and less than 2” deep.

An example packaging concept for the digital thermostat prototype is shown in Figure 7-1. The overall packaging design goals are listed below:

- Form Factor: Wall mount, round.
- Physical size: Approx. 4.5” diameter, < 2” depth.
- Main user interface: 6-button direct input, LCD/LED output.

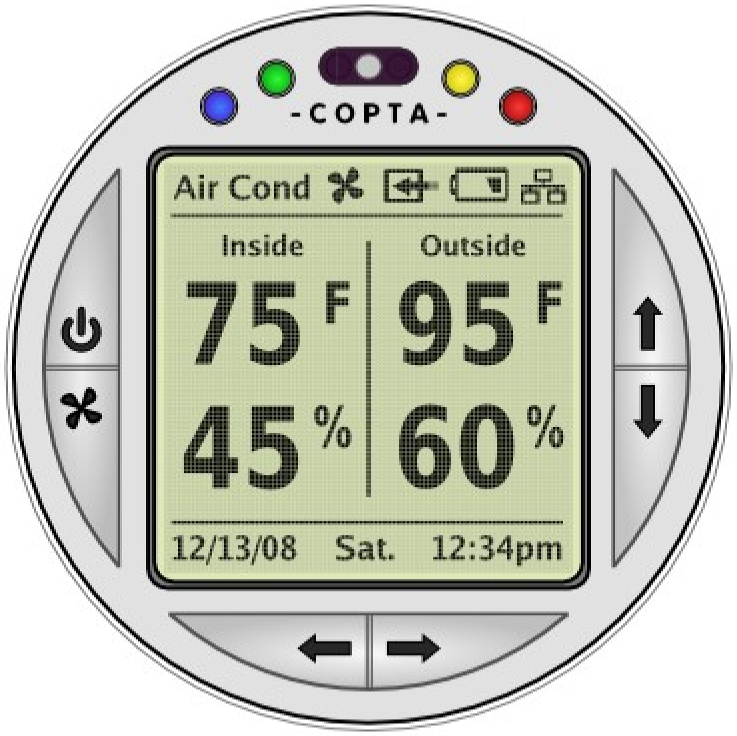
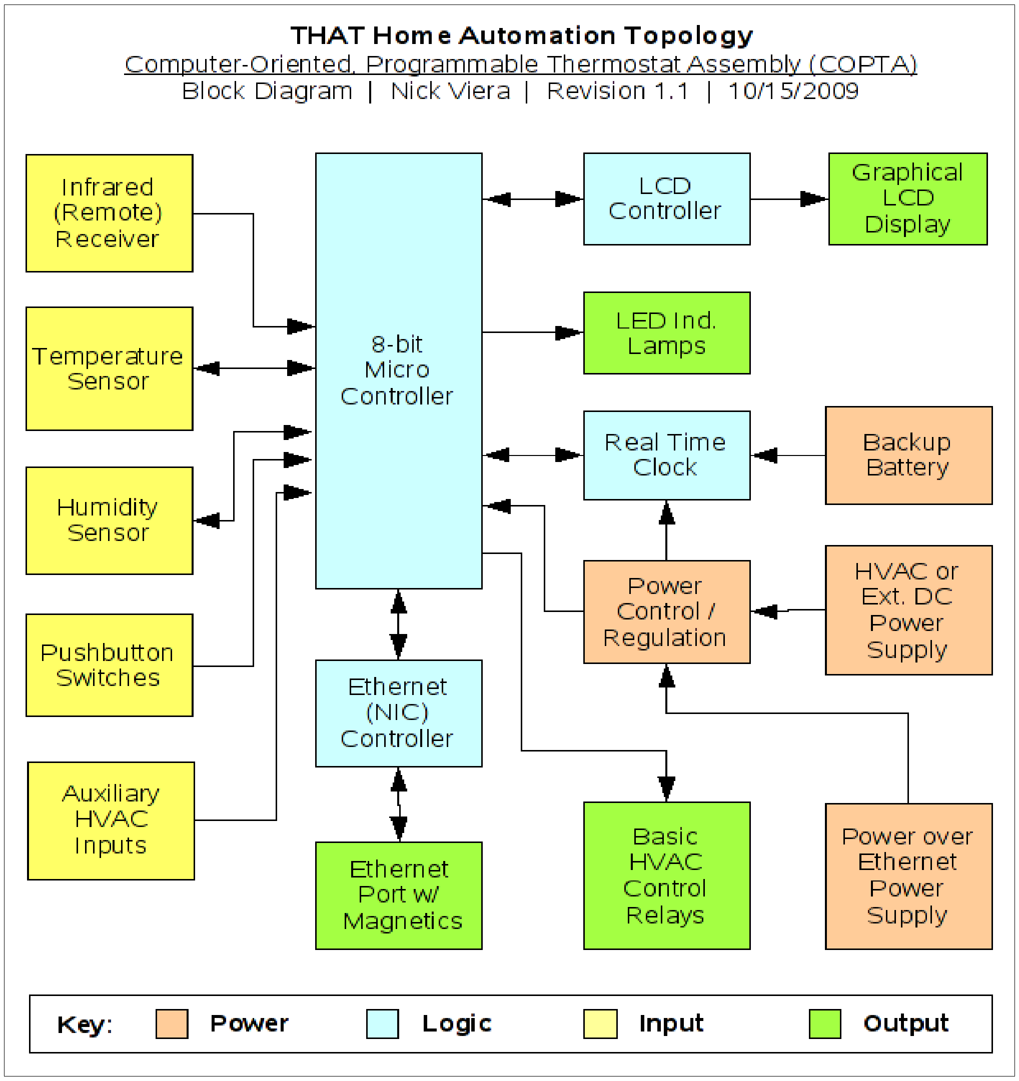


Figure 7-1: Prototype Design Concept

3.5. System Block Diagram

The overall hardware block diagram for the module is shown in figure 9-1.


THAT Home Automation Topology
Computer-Oriented, Programmable Thermostat Assembly (COPTA)
Block Diagram | Nick Viera | Revision 1.1 | 10/15/2009
Infrared (Remote) Receiver
Temperature Sensor
Humidity Sensor
Pushbutton Switches
Auxiliary HVAC Inputs
8-bit Micro Controller
Ethernet (NIC) Controller
Ethernet Port w/ Magnetics
LCD Controller
Graphical LCD Display
LED Ind. Lamps
Real Time Clock
Backup Battery
Power Control / Regulation
HVAC or Ext. DC Power Supply
Basic HVAC Control Relays
Power over Ethernet Power Supply
Key:
Power
Logic
Input
Output
Figure 9-1: System Block Diagram

4. Electronic Access Module

4.1. Introduction

The Electronic Access Module (EAM) will be a flexible entry and security system, equipped with programmable control, that shares the basic goals and requirements defined for THAT System. It will be designed for residential and light-commercial buildings. The EAM will be able to function as a stand-alone device or in a larger THAT System to provide maximum flexibility and functionality. This module will be developed by Chris Miller in collaboration with Nick Viera.

4.2. Hardware Specifications

The EAM hardware will be based on an 8-bit microcontroller running in the speed range of 10-20 MHz. Most of the hardware will fall into the category of general digital I/O. Additionally, 3-wire serial (SPI) will be used for communication between the microcontroller and the Ethernet ICs. Communication in conformance with THAT will be made possible using a hardware Ethernet controller designed for 10/100Base-TX data rates. See Figure 12-1 for the hardware block diagram.

- Platform: 8-bit Atmel AVR microcontroller (most likely Atmega328)
- Ethernet: Hardware ethernet interface controller such as Microchip's enc28j60 IC
- Monochromatic VFD or backlit LCD display
- Ten (10) passcode pushbuttons
- One (1) doorbell pushbutton
- Wireless receiver for basic lock/unlock remote control
- Wireless key-chain dongle support (AES encrypted)
- Three (3) LED indicators for quick system status notification.
- Three (3) built-in relays for controlling access hardware
- Real time clock with calendar.
- Non-volatile memory for storing "permanent" system settings.
- Battery-backup for temporary settings and RTC.

4.3. Software Specifications

EAM firmware will be consist of mainly C code. The firmware will implement both the functionality necessary for the entry and security functionality as well as the Ethernet / networking stack necessary for communication with the client PC.

- C and AVR assembly languages as needed.
- Configurable support for access input modules (doorbell, keypad, wireless dongle, card reader, etc.)
- Configurable support for access output modules (relays for door strikes, keyless deadbolts, and magnetic door closers)
- Configurable support for security input modules (window / door, smoke / CO, motion, water, etc.)
- Configurable support for security output modules (siren, phone / SMS / email notifier, X10 / Insteon, etc.)

4.4. Design Concept

The initial physical concept design for EAM slave is a unit approximately 3" high, 6" wide, and 1" deep. An example packaging concept for the EAM prototype is shown in Figure 11-1. The overall packaging design goals are listed below:

- Form Factor: Wall mount, rectangular.
- Height x Width x Depth: 3" x 6" x 1"
- Main user interface: 10 (+1) pushbutton, VFD output.

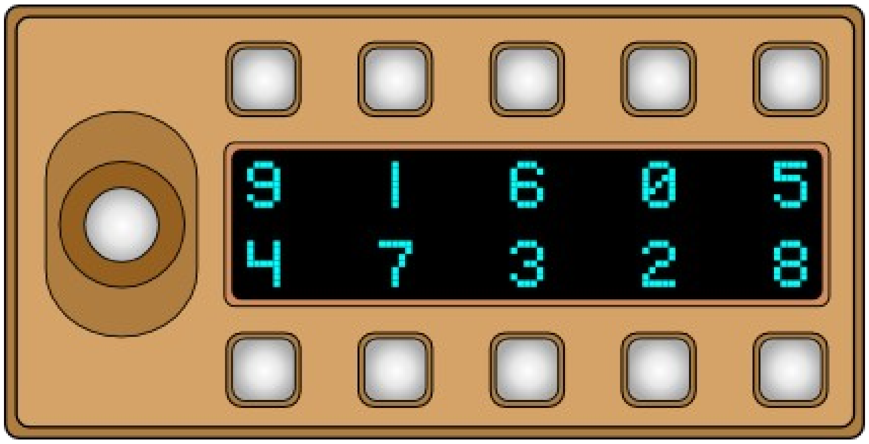


Figure 7-1: Prototype Design Concept

4.5. System Block Diagram

The overall hardware block diagram for the module is shown in figure 12-1.

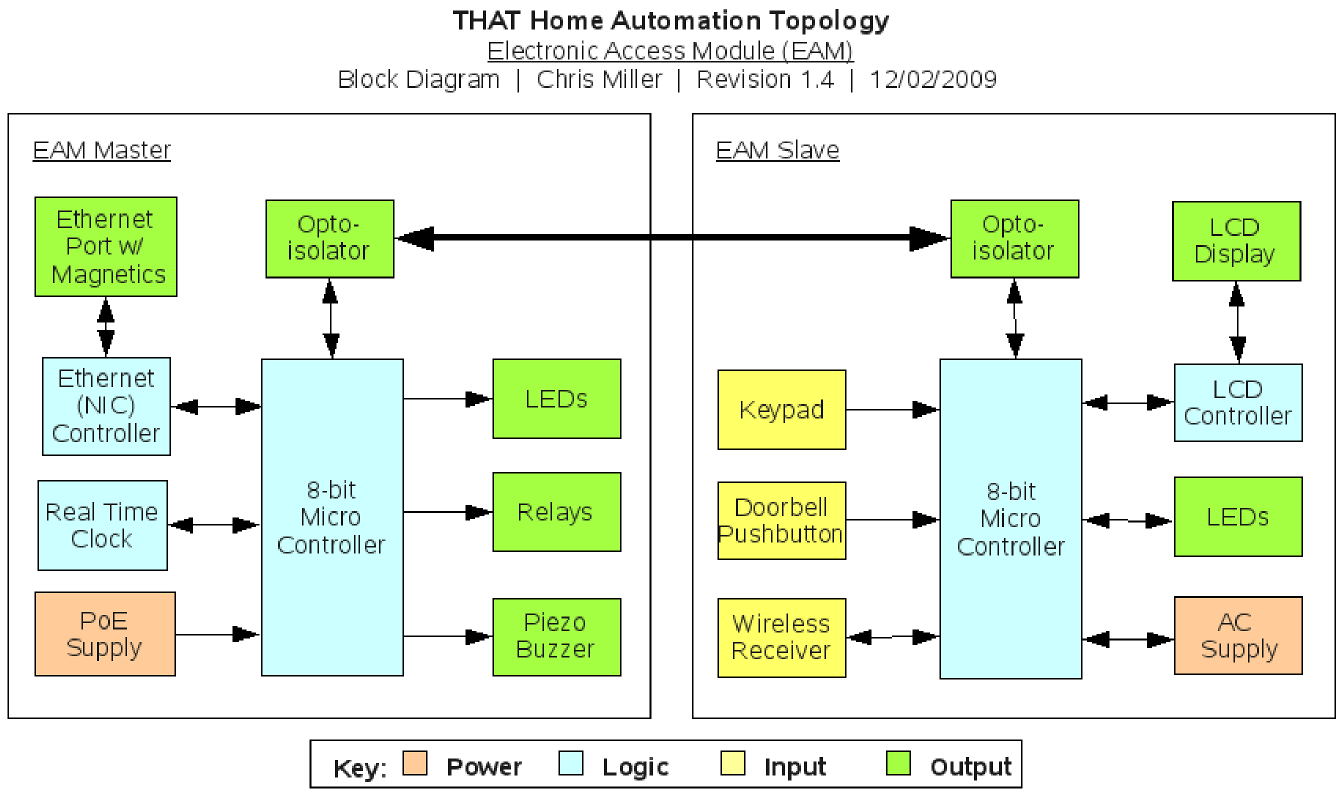


Figure 12-1: System Block Diagram

5. Project Organization

5.1. Development Process

The scope of this project involves the research and development of THAT System and the two modules into at least a basic prototype stage. However, as time permits, the project will be advanced into a fully functional prototyping stage with a goal towards small to moderate-quantity production. The general development process is outlined below:

- Project concept development
- Initial research into project concept
- Preliminary specifications and goals
- Preliminary hardware and software design
- Basic prototype development and continued research
- Device refinement and expansion
- Fully-functional prototype development and 'Alpha' testing
- Pre-production refinement and debugging
- Pilot production and 'Beta' testing
- Minor refinements and other pre-production tasks
- Production (small to large scale)

5.2. Project Timeline

We plan to follow a fast-paced development cycle so that we can achieve our goal of building functional prototypes of our modules before the week of March 13th, 2010 (the university's Spring Break.) Should we be unable to meet this goal, we will still have remaining time in the spring semester to make substantial progress on our project.

January 2010: Complete preliminary hardware design and begin acquisition of all parts. Begin hardware integration. Begin device firmware / software design. Continue research.

February 2010: Continue development and refinement of hardware and firmware designs. Continue expanding functionality of devices while improving hardware and firmware design.

March 2010: Finish hardware design and build first “complete” prototype modules. Continued development and debugging of device hardware, firmware, and system software based on experiences from alpha testing of the prototype devices.

April 2010: Freeze the design of the modules for the scope of Senior Project. Complete comprehensive documentation of the current state of the project and of the prototype modules. Prepare necessary project presentations.

May 2010: Finish any remaining documentation tasks, present the project to our faculty and peers. Graduate. Continue project development on an individual basis.