

CMPE 121 Laboratory Project

Expanded-bus MC68HC11E1 based system

Jason R. Heimann

December 10, 2004

1 Introduction

A MC68HC11E1 based system was built as a laboratory project for a micro-controller system design class. The design, development and implementation of the system proved to be a valuable experience that underscored the material covered in lecture. A core design was suggested by the professor, including a serial port and a self-diagnostic mode of operation. An expansion to the core design was developed and built, allowing the display of scrolling messages and icons on an LED array.

2 Design

The design of my particular 'HC11 system focused on meeting the “minimum requirements” with good electrical and mechanical design and construction, and allowing maximum flexibility for future expansions. Expecting to use the board for many different applications over its lifetime, I designed the system to use as little board space as possible, and to allow for the maximum addressable amount of ROM and RAM. I also wanted to keep all “outside-world” interfaces such as the RS232 connector, reset button, power connector and DIP switches, at the edges of the board for easier access.

Computer software including Orcad was used to aid in the layout and wiring design, providing a relatively easy way to visualize the constructed board without doing any physical work! Wiring design goals included keeping bus lines equal in length, preventing runs of many wires in parallel and in the same area (to make wiring easier). With the software, I found that rotating my ROM chip by 90° and some other chips by 180° created a “neat and tidy” wiring layout.

In building the 'HC11 system, I now had a powerful tool with which to build a proof-of-concept for a new automotive indicator (i.e. brake light). I wanted a new way to control a matrix of lights that allowed for the production of selected dynamic patterns. I designed two 8-bit expansion ports into the system and produced an 8x8 LED matrix on a daughter-board, controlled by the two new ports.

3 Implementation

3.1 Hardware

The system hardware, as wired, is shown in the Appendix. Sheet 1 includes the MC68HC11E1 microcontroller, address latch, RAM, ROM and associated decoding logic. Also included on Sheet 1 is the power input circuit: a 5V, 1A regulator with a reverse-polarity protection diode and bypass capacitors. It should be noted that all IC's in the system are bypassed by at least one capacitor of 0.1 μ f placed as close to the IC as possible.

The core design of the system closely mimics the "Expanded Mode" design provided by Motorola in their 'HC11 Reference Manual. Port A is duplexed as a DIP switch input (for baud rate selection) and a timing signal output (a 1.0 Hz clock signal). Port D is used to implement the SCI interface's TxD and RxD lines, as well as two lines for hardware handshaking.

The system expansion required the addition of a few glue logic IC's (74HC00 and 74HC08) to provide decoding, as well as two 8-bit edge-triggered latches. With this addition of two output-only ports, I now had the signals required to drive a LED matrix. A daughter-board with matrix was hand-built and interfaced to the output ports through current limiting resistors, and NUN switching transistors were used to decrease the current output required of the latches.

3.2 Software

I began with my design of the self-diagnostic software. First, external memory locations are written to and read from in an interleaved fashion to check for address lines that may be "stuck" at one logic level. Second, all internal and external RAM is "exercised" to ensure all data registers are devoid of stuck bits. Any errors found during these exercises are reported to a PC through the serial port, with specific information (such as an address line number or register location) that will lead the user to the specific source of the problem.

I also implemented a chronological clock using the output capture functionality of the 'HC11. After pre- and post-scaling the internal clock, a 0.5 Hz internal signal is produced via a repeating interrupt that toggles an external pin, producing a 1.0 Hz signal. The average period of this clock was measured with a Digital Storage Oscilloscope, and found to be 499.9 μ S. This gives an error of -0.1 μ S per second; the clock will "lose" about 0.26 seconds per month.

An interrupt-driven SCI interface was designed to free up other CPU resources while many characters are queued for delivery. Baud rates of 300 to 9600 APS are attainable with minimal overhead; two 1k FIFOs were implemented for input and output buffering, allowing maximum throughput at the port while requiring the fewest CPU cycles necessary for transmission / reception.

The software required to drive the LED matrix can be as simple as one line of code, or as complicated as one can imagine. To reduce development time

and achieve an acceptable minimum proof of concept I wrote a small bit of code that updates the display in cooperation with other processes in the main event loop. A user-programmable matrix holds the byte codes to display on the LED matrix. When the program matrix exceeds the size of the LED matrix, the byte codes are “scrolled” across the display. Additional self-diagnostic software was written to allow checking for “bad” port lines and burnt-out LEDs / NPNs.

4 Conclusions

Despite the long, hard hours spent in the lab, and the frustrating periods of debugging, this was a very rewarding project. I learned the importance of planning, calculating and executing every detail of the project first on paper, if possible in computer simulation, before doing any construction (“look before you leap!”). My experiences with Orcad were valuable and will no doubt make a good bullet point on my future resume. I also came to appreciate the power and versatility of the microcontroller, seeing its role in the designs of various students. Spending time working with fellow 'HC11 designers in the lab provided me with many insights on design and construction, and allowed me to contribute (even if in a small part) to others' projects.

Of all my colleagues I would like to thank Andrey Yurovsky for his eagerness to help out whenever I had a problem with my code. He helped me form a better understanding of the instructions and control registers implemented in the 'HC11. I would also like to thank Prof. S. Petersen for his help in and out of the lab. His sagely advice and excellent detective skills made the learning process a little less bumpy.