Lab 4:

Project Time Timesheet

Hazael Magino - VV12417
Blake Webb - HJ69411
Abner Ben - AB74294
Airik Jenkins
CMPE 310: System Design & Programming
University of Maryland. Baltimore County
May 4, 2024

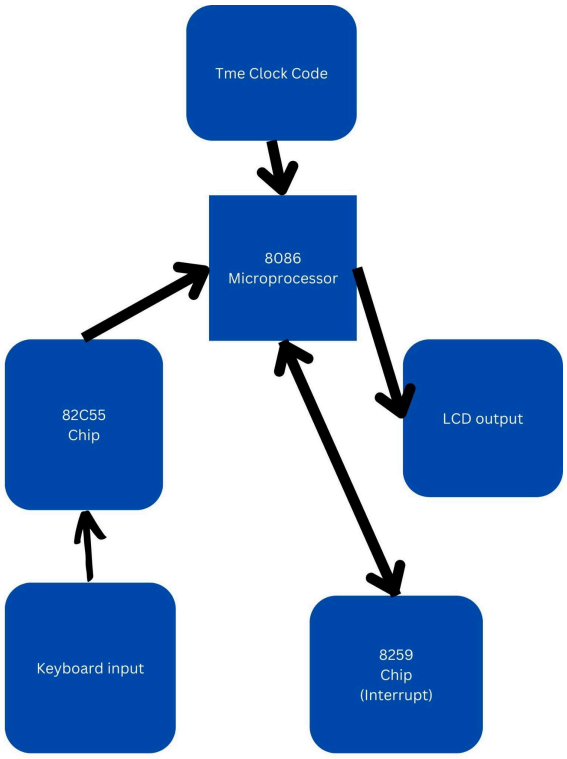
1. Introduction

This project centers around the development of a Clock Timing Sheet Program/Device designed specifically for efficient tracking of clock-in and clock-out times while also handling wage calculations. The system encompasses both hardware and software components, and this report delves into the intricate details of its implementation.

By exploring the inner workings of this design, we aim to provide a comprehensive understanding of how the clock timing sheet program/device operates, ensuring accurate timekeeping and streamlined processes.

2. Goals

The goals of this project were to Develop a way to handle user inputs of times and name, be able to calculate the worker wages based on saved input and display necessary information for users to interact with the system. To do this tasks were divided into sections for group members to take on. One member would handle the implementation of the code which would take care of any programming associated with the chipsets and main handling of the software. Member two would handle the hardware of the 8086 microprocessor and the clock input for the rest of the implementation. Member 3 would handle the keyboard input and interrupt processes. Members would handle implementation of multiple inputs at once with increasing the scalability with the 82C55. Below is a rough diagram of how the chipsets should interact with each other



3. Requirements

Some requirements for this project that we were designed to address with this project were:

- Design the code necessary for the device
- Detailed schematics of the project components
- 8086 in minimum mode and its associated bus buffering/demultiplexing logic. NMI should be connected to a push-button switch with appropriate timing logic. The data bus, address bus and all control signals should be connected to headers after buffering and demultiplexing for external access to design expansion boards.
- 256KB of CMOS flash composed of 128K x 8 28F010 CMOS Flash Memory, decoded into two banks, highest address FFFFFH.
- 128KB of SRAM composed of 32K x 8 CY7C199 Static SRAM, decoded into two banks, lowest address 00000H.
- 8284A Clock Generator along with crystal and reset subcircuits
- 3 8255 chips, decoded at the following addresses (all in hex), all port connections should be pulled to headers for external access.
  PPI 1: Port addresses FFFF (control register), FFFD, FFFB, FFF9
  PPI 2: Port addresses FFFE (control register), FFFC, FFFA, FFF8
  PPI 3: Port addresses FFF7 (control register), FFF5, FFF3, FFF1
- 8259 decoded at FFF6 (command) and FFF4 (data). IR0 connected to a push button switch with necessary circuits, IR1 connected to 8254 counter 2 output, IR2 connected to the 8279 IRQ output and IR3 connected to the 16550 INTR output. All other IR lines connected to headers for external access.
- 1 8254 decoded at FFDE (command), FFDC, FFDA, FFD8, with all counter 1 and counter 3 pins connected to headers. Counter 2 output connected to IR1 of the 8259, gate and clock connected to header.
- 1 8279 decoded at FFF2 (command) and FFF0 (data) which will be operated in the decoded mode. Connect 20 push button switches connected as four rows and five columns. Also connect two switches to the control and shift inputs. In your layout arrange these 22 keys as well as reset, switch connected to the IR0 input on the 8259 and NMI keys to form a 5x5 keyboard matrix layout. CLK input to the 8279 should be the PCLK signal from the 8284A.
- 1 20 character x 4 line LCD display with no back-light and an integrated LCD controller decoded at addresses FFD6, FFD4, FFD2 and FFD0 decoded at FFCE and FFCF, 8 LEDs connected to a 74374 latch decoded at FFCC.
- A power terminal block to provide power to the board. A 100uF decoupling capacitor should be connected next to the power terminal block. Place a 0.1uF decoupling capacitor next to each of the major chips in your project.
- Other misc. components that are required for the above circuits e.g. resistor packs, caps, etc.

Hardware Schematics

8086:

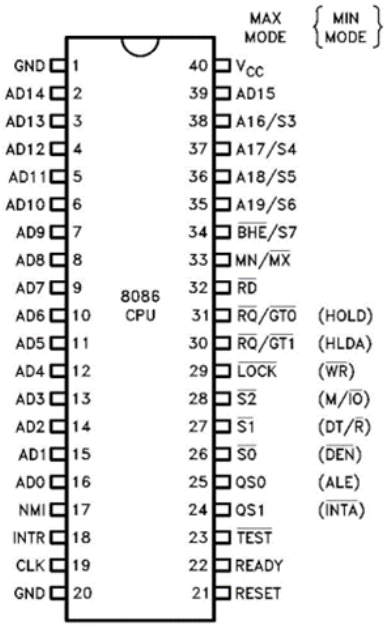
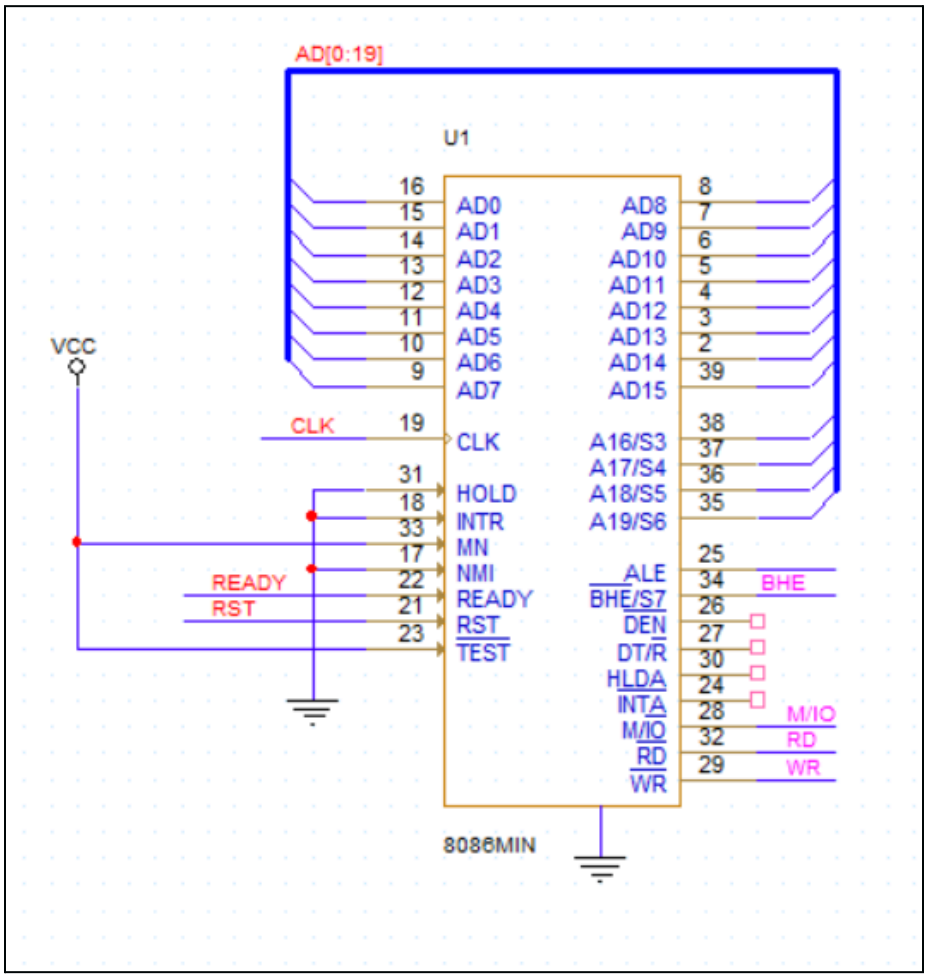


Figure 1: Intel 8086 Microprocessor Pin Configuration

The 8086 microprocessor, which is well-known for its adaptability and processing capacity, is carefully included into our system to program essential time management features. Serving as the central processing unit (CPU), the 8086 performs complex time computations with efficiency, keeping careful track of employee sign-in (1) and sign-out (0) events, and accurately computing overtime hours. Working in ideal conjunction with the 8284 clock generator, the 8086 receives clock pulses and synchronizes its actions with the clock generator's Ready (RDY) and Reset (RST) signals. Additionally, it is essential to data communication because it feeds the 82C55 peripheral interface adapter with the Read (RD) signal, which facilitates efficient data transfer between peripherals and the CPU. The 8086 microprocessor is intricately integrated into our system to provide strong and reliable time management capabilities that improves productivity and efficiency in a variety of operational scenarios. Figure 2 below illustrates a detailed schematic of the microprocessor, providing a visual representation of the described configuration.

Figure 2: 8086 Microprocessor Schematic



8284:

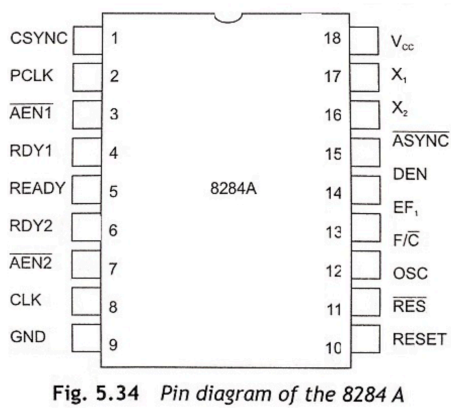
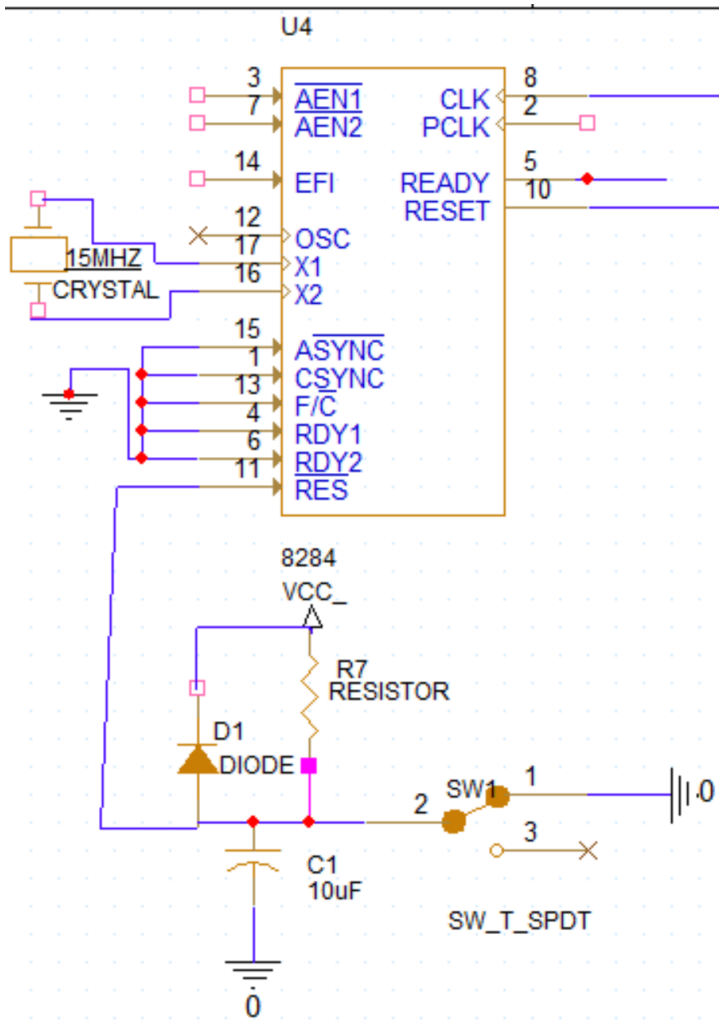


Fig. 5.34 *Pin diagram of the 8284 A*

Figure 3: 8284A Clock Pin Configuration

The 8284 Clock Generator is an integral component designed to synchronize the operations of the 8086 microprocessor and its associated peripheral chips. As depicted in Figure 2, the chipset's pinout configuration is outlined, highlighting the primary outputs: the CLK for the microprocessor and the PCLK for peripheral devices. In the schematic design for this project, the CLK output is driven by a 15 MHz Crystal Oscillator, which establishes the clock configuration for the chipset terminals X1 and X2. The circuit configuration grounding pins 15, 1, 13, 4, and 6 to ensure stable operation.The OSC terminal (pin 12) is unnecessary in this single-clock system, eliminating the need for multiple clock sources. The reset mechanism is configured by a minimalistic RC circuit, comprising a diode connected to a 5V supply in parallel with a resistor. This configuration allows for a gradual power disengagement, facilitated by a capacitor in series with the ground. Additionally, a reset switch is incorporated, establishing a connection to the ground when activated. This systematic reset protocol is crucial for the harmonization of the entire device upon initialization. Figure 4 below illustrates a detailed schematic of the clock generation device, providing a visual representation of the described configuration.

Figure 4: 8284 Clock Generation Schematic



8259:

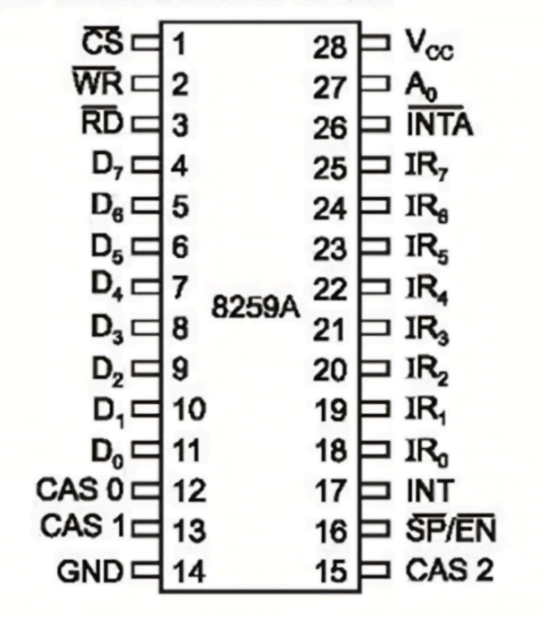
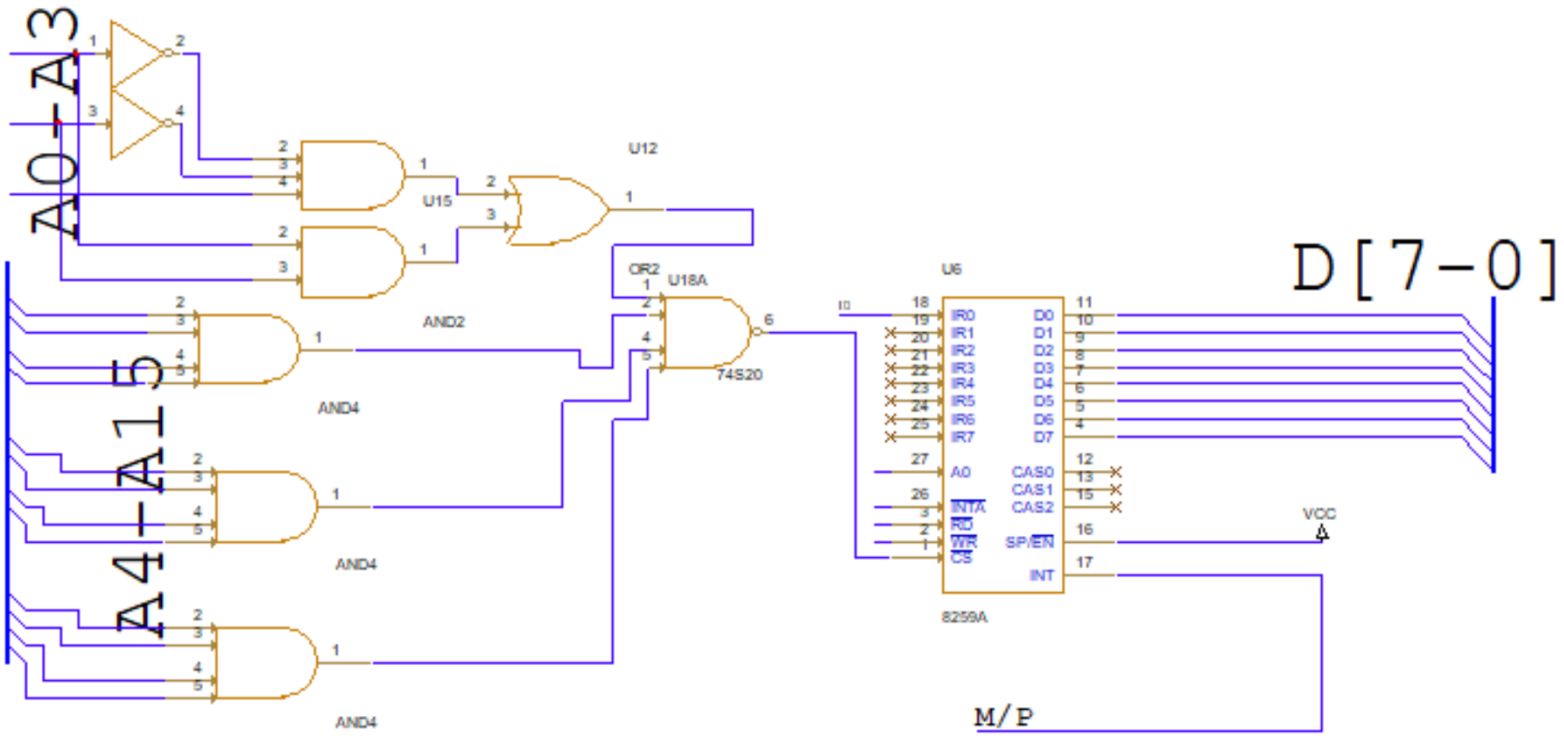


Figure 5: 8259 Pin Configuration

The 8259A is a programmable interrupt controller. This controller is necessary for handling vectored priority interrupts to the microprocessor. It can handle bidirectional connections to any 8 bit bank. It can also be used in expanded models with primary and secondary implementations with up to 8 secondary units. The inputs of the Chip select (pin 1) and write(pin 2) are manually decoded while other connections go directly to the microprocessor. The main functionality with this chip's usage will be to handle keyboard input. Each time a key is pressed or some functionality occurs that requires the microprocessor, this chip will send a signal to divert attention to a special process occuring. With our design utilization of INT and IR1 would be needed because the keyboard was the only peripheral interfacing with the microcontroller other than the display. The other IR lines would be unnecessary unless development of a multi-level cascade would be needed. D7-D0 would be inputs from the keyboard fed in from the 82C55 to the microprocessor. The CS on the 8259 would be decoded at FFF6 (command) and FFF2 (data). Here the decoding would be done using two inverters connected to 2 ands and a or gate which feeds into a 4 input and where 3 other and gates take in (A4-A15). Below is figure 6 of the schematic design we implemented.

Figure 6: 8259 Schematic



82C55:

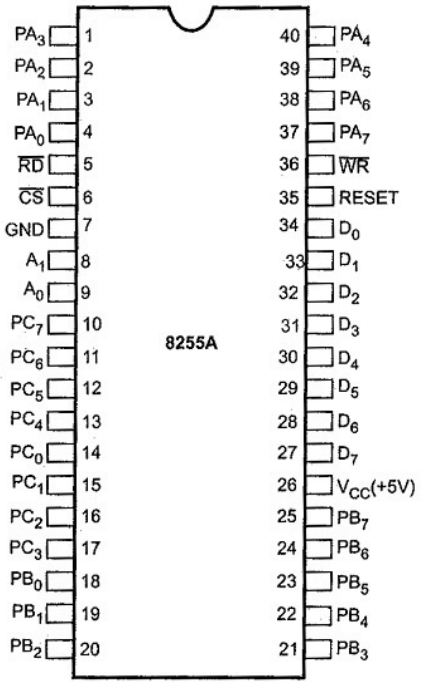


Figure 7: 82C55 Pin out diagram

The 82C55 is a popular interfacing chip with other components. It can interface any TTL-compatible I/O device to the microprocessor. It is commonly used to interface to the keyboard and/or parallel printer ports in the PCs. Here the chip is divided into different sections of usage. Aside from the normal data ports and normal read,writes, reset, and chip select, the chip possesses different ports. These ports have specific I/O assignments based on A0 and A1. Each port corresponds to a unique combination of A1 and A0. Port C is divided into halves for sections A and B if extra port space is needed. In our design the schematic of this design was heavily utilized because of its capabilities to interface with a keyboard. In the hardware design of this chip the team decided to utilize ports A and B as sections of usage for the 6x6 matrix which was meant to be understood as the keyboard. 6 pins were connected to each port. Port A has 6 connections and port B has 6 connections meant to illustrate the matrix. The matrix in question is merely just the alphabet as well as the number inputs of 0-9. Port C for this design would be used as the output of the data to an LCD display. How this would be decoded would be using a 16L8 from A2-A10. The O1 pin would be used as the write for this chip and the 08 would be used as the CS of the 82C55. This chip and keyboard and matrix (keyboard) would be powered with a power block connected to VCC with capacitors of 10 and 100nf. Lastly if we were to ever scale this system design up the following figure 10 would be used to show multiple input devices. Below are the figures for both these designs.

Figure 8: 82C55 (single)

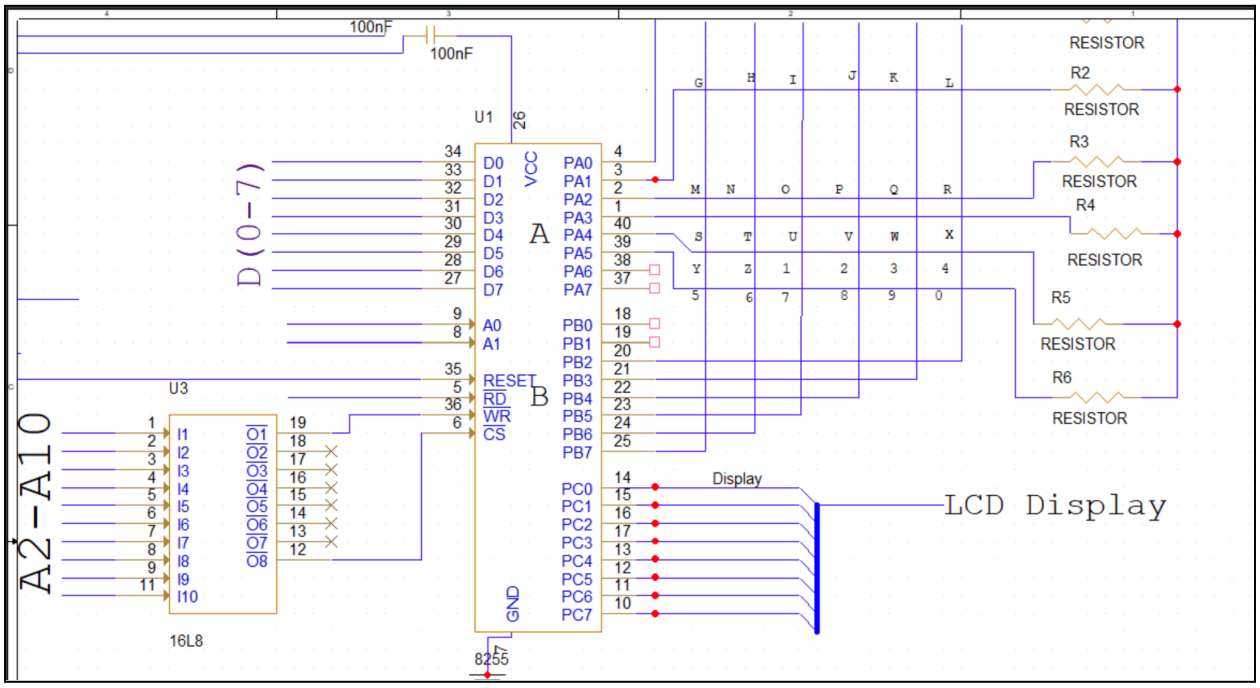
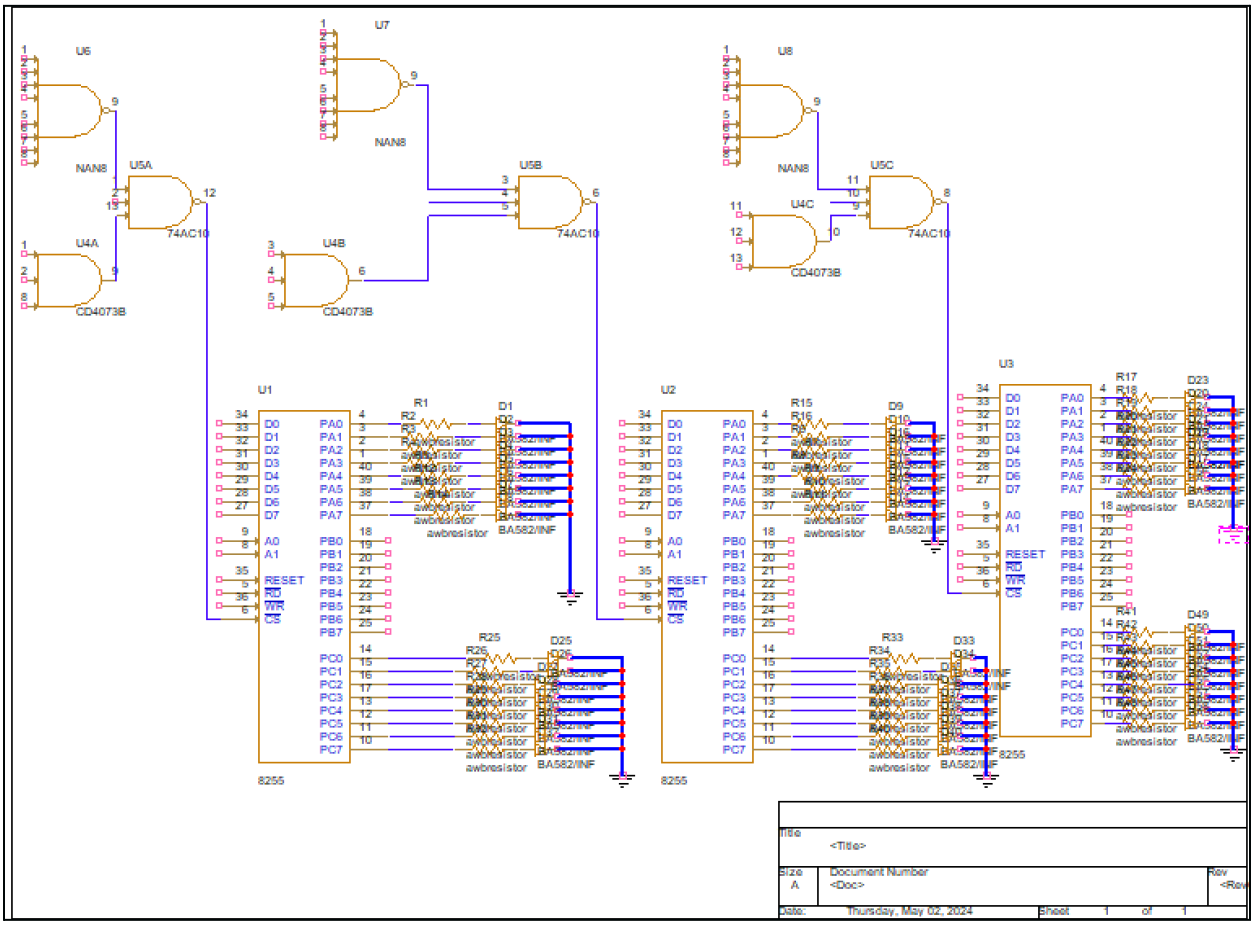


Figure 9: 82C55 (multiple devices)



Code

In our project's code structure, the main menu serves as the central control hub, facilitating user interactions and navigation through various functionalities. Implemented as a while loop, the main menu offers a dynamic interface where users can seamlessly choose between signing in or out, accessing payout calculations, or opting to exit the program. This modular design not only enhances user experience by providing clear and intuitive pathways but also promotes code organization and maintainability.

To facilitate efficient development and debugging, the codebase is divided into several files, each housing distinct segments or functions. This modular approach offers numerous benefits, including enhanced readability, easier debugging, and improved collaboration among team members. By compartmentalizing code into logical units, such as user authentication, payout calculations, and menu navigation, we ensure that each component remains focused and easily manageable. Furthermore, this separation of concerns minimizes the risk of unintended side effects and promotes code reusability, fostering a more robust and scalable software architecture. Overall, the utilization of multiple files in our codebase exemplifies our commitment to best practices in software development, enabling smoother debugging processes and facilitating future enhancements with ease.

The sign-in and sign-out functionalities are pivotal aspects of our system, serving to efficiently record employee attendance data. When initiated, these processes prompt users to input their first name, last name, and the respective time of sign-in or sign-out in the standard hour:minutes format. These inputs are then captured and stored in variables within the program's memory space before being transferred and persisted in either a file or a designated memory location for future access and reference. This approach ensures a comprehensive record of employee attendance, facilitating effective work hour tracking, productivity monitoring, and report generation as needed, while upholding data integrity and security standards.

The time calculation functionality within our system operates seamlessly by leveraging the sign-in and sign-out data previously recorded for each employee. This process involves subtracting the recorded sign-out time from the sign-in time to determine the duration of the employee's work session. By accessing this data and performing the necessary arithmetic operations, our system accurately computes the total hours worked by the employee.

Once the duration of the work session is calculated, it is multiplied by the bi-hourly rate, which accounts for compensation on a per-half-hour basis. This calculation ensures precise remuneration for the time spent working, aligning with industry standards and fair compensation practices.

The final result of this computation, represented as an integer, is then displayed on the screen, providing both the employee and administrative users with clear and transparent information regarding the employee's earnings for the given work period. This streamlined process not only simplifies payroll management but also ensures accuracy and fairness in compensating employees for their time and effort.

Contributions

Hazael Magino: 82C55 Schematic, 8284 Schematic, 8259 Schematic
Abner Ben: 8086 Schematic
Blake Webb: NASM code
Airik Jenkins:

Future Plans

Optimized Code

In reflecting on our project and considering future optimizations, transitioning to 16-bit NASM code could offer significant performance enhancements. By implementing this change, we aim to streamline our codebase, improve memory efficiency, and enhance overall execution speed. This transition not only aligns with current industry standards but also reflects our commitment to continuous improvement and staying abreast of evolving technologies. Moreover, the adoption of 16-bit NASM code presents an opportunity to explore novel approaches to code optimization, paving the way for a more robust and efficient software architecture.

Card Scanning

Integrating a card scanning device emerged as a promising solution to streamline our sign-in process. While the inclusion of such technology seemed beyond the scope of our course and technical expertise, its potential impact on user experience and efficiency cannot be overstated. By leveraging a card scanning device, we eliminate the reliance on manual keyboard input, thereby expediting the sign-in process and minimizing user friction. This innovation not only aligns with industry trends but also underscores our commitment to delivering a seamless and user-centric solution. Moving forward, exploring avenues to incorporate this technology will be paramount to enhancing both the functionality and user experience of our project.

Clock

The integration of a clock would greatly improve our system. By allowing users to set and adjust the time manually through a limited keyboard setup, we aim to streamline operations and enhance user convenience. This feature not only eliminates the need for manual time input but also ensures accuracy and consistency in time tracking, fostering a more efficient workflow. Moreover, the implementation of a digital internal clock will make our system seem less primitive.

Conclusion

In conclusion, this cumulative project helped us develop all that we have learned about various chipsets and apply our assembly programming skills to help develop and build a rough sketch of a system that not only tracks clock-in and clock-out times but also accurately calculates worker wages.The implementation of the 8086 microprocessor and 82C55 chipset provides a solid foundation for handling multiple inputs and ensuring scalability for larger models of this project. The assembly programming has been pivotal in orchestrating the device's operations, while another member's expertise in hardware has fortified the system's reliability. The keyboard input and interrupt processes, overseen by the third member, have enhanced the user interface, making it intuitive and user-friendly as well. Despite the obstacles faced with the software needed to build schematics and the uniqueness of this device, the team was able to handle and build a concrete design for this clock device. In the future we would like to allow card scanning, optimize code, add an internal clock, and increase the security of the system. Overall this project focused and enhanced our skills with chipset design and assembly programming.