8086 Microprocessor Design Project

CMPE 310
Systems Design and Programming
Sabbir Ahmed

May 10, 2017

Contents

1 Introduction

This document provides detailed instructions to develop an 8086 microprocessor board using Cadence® OrCAD® Capture software. Included are the schematics of individual IC components and their description. Details of the ICs include decoding, programming specifications, and descriptions of IC pinouts.

1.1 Purpose

As per the project description, this document is to serve as the only documentation of the operational and functional specifications of the Intel 8086. The documentation is to be thorough and concise to provide information to design a similar board.

1.2 Scope and Organization of Document

The document will elaborate on the individual building blocks of the 8086 board. The integrated circuit (IC) chips used in designing the board will be discussed, along with brief, high-level overviews of their pinouts, their various connections and their functionalities. The connections and dependencies between the different components such as memory and IO devices will be discussed in detail.

The document is organized into sections that cover the individual components and their IC pinouts, functionalities, connections and role in the 8086 board. Schematics of the different components and their circuitry are included. Code snippets, including the VHDL (VHSIC Hardware Description Language) implementations of the decoding hardware and the Assembly implementations of the data and memory addressing, are also incorporated in the document.

The schematics in the document are included in Appendix A and the code snippets in Appendix B. The PCB Design layouts are included in Appendix C.

2 8086 Microprocessor

The 8086 microprocessor is an enhanced version of the 8085 microprocessor developed by Intel in 1978. It is a 16-bit microprocessor, with 20 address lines and 16 data lines to provide up to 1 MB of physical memory. The 8086 microprocessor described in the project will operate in its minimum mode.

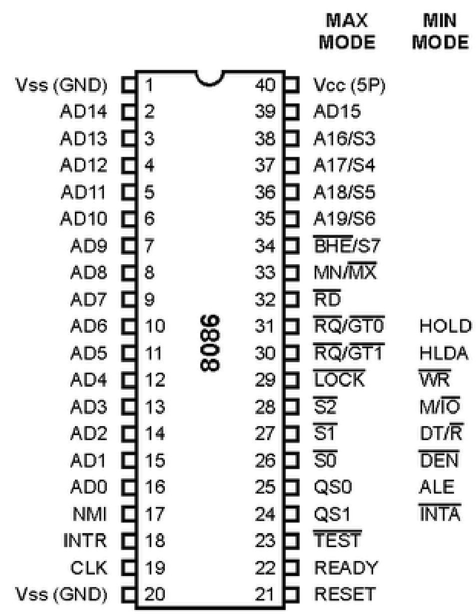

Figure 1: 8086 Microprocessor

2.1 Features

The 8086 microprocessor is known for its significant advancements since its predecessors. The most prominent features include, but are not limited to:

- 6 bytes of cache memory for faster processing
- Pipelining stages: Fetch Stage and Execute Stage
- Instruction queue
- 256 vectored interrupts
- Maximum and minimum modes of operation, suitable for multiple and single processors respectively

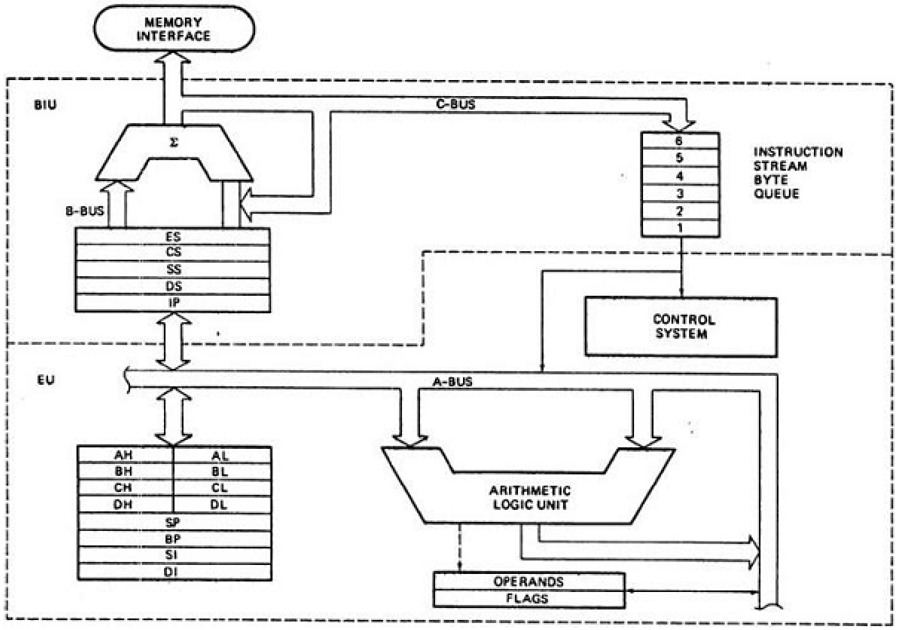

Figure 2: Architecture of 8086

2.2 Address and Data Buses

The 8086 CPU has a unidirectional address bus with 20 address lines and a bidirectional data bus with 16 data lines. [1] The address bus is used to select the desired memory or I/O device by generating a unique address which corresponds to the memory location or the location of I/O device of the system. The data bus is used to transfer data between the CPU and memory and the CPU and I/O devices.

The address bus is denoted as $A_{19} - A_0$ (20 lines) and the data bus $D_{15} - D_0$ (16 lines). The peripheral devices implemented with the 8086 in this document however consist of 8-bit data bus architectures. The data bus would therefore be multiplexed and more commonly denoted as $D_7 - D_0$ (8 lines).

2.3 Control Bus

The control bus of 8086 carries control signals which are used to specify the memory and I/O devices. [1] The bus is bidirectional and assists the CPU in synchronizing control signals to internal devices and external components. It is comprised of interrupt lines, byte enable lines, read/write signals and status lines.

3 Decoding

Discrete gate integrated chips are used throughout the board for decoding and demultiplexing address, data and control lines.

3.1 Programmable Logic Device - PAL16L8

Programmable Array Logic (PAL) is a type of Programmable Logic Device (PLD) used to implement logical functions. [2] PALs comprise of an AND gate array followed by an OR gate array, as shown in Figure 3.

16L8s were programmed to be used throughout the board to implement the various functionalities required in the system.

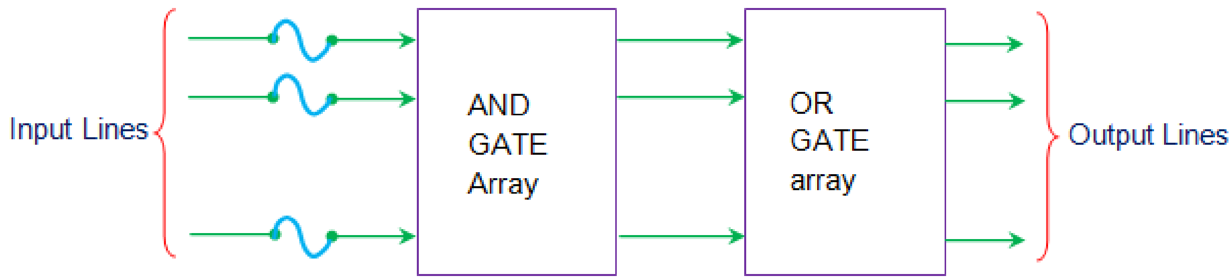

Figure 3: 8086 Microprocessor

3.2 Programming the PLD

VHSIC Hardware Description Language (VHDL) was used to program the different logical functions. The modules used for the nine (9) 16L8s used in the system are implemented in Appendix B.

4 Clock Generator - 8284A

The 8184A Clock Generator is an ancillary component to the 8086. This system clock is used to synchronize both internal and external operations using an external oscillator. The device is also used for READY and RESET synchronizations and TTL-level peripheral clock signal generation.

4.1 Clock Speed

The 8086 internal clock has a frequency of 5 MHz ($\frac{1}{3}$ of CLK). The external crystal typically oscillates at 15 MHz.

4.2 RESET Operation

Correct reset timing requires that the RESET input to the 8086 becomes a logic 1 in 4 clock cycles and remain high for at least 50 μS. The reset switch is implemented in a RC circuit with typical resistance of 100 $k\Omega$ and 10 μF.

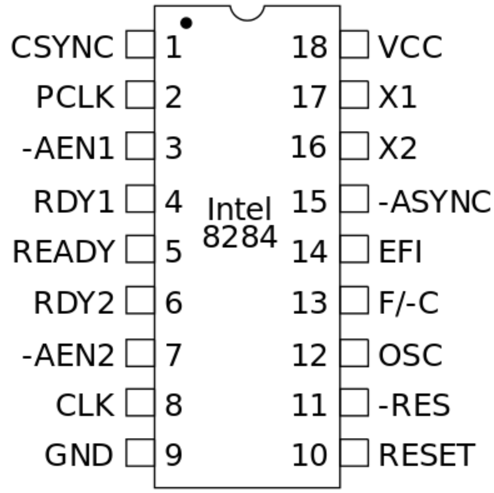

Figure 4: 8284A Clock Generator

5 Memory Architecture

The board implemented in this document consists of 128 kB of SRAM and 256 kB of CMOS flash memory.

5.1 Static Random Access Memory - CY7C199

Static RAM, or SRAM, is a type of volatile memory where bits are stored as long as power is supplied. They are ideal for cache memory because of their fast access times as low as 10 ns. CY7C199 SRAM chips are used to implement the SRAM on this system.

5.2 Addressing

The SRAM implemented in the system is composed of 32k $\times$ 8 CY7C199 SRAM chips decoded into two banks, with the lowest address at 0x00000.

5.3 CMOS Flash Memory - 28F010

Flash memory, or electrically erasable programmable ROM (EEPROM), is a type of non-volatile memory which maintains its state even after loss of power. Writing to this memory is much slower than a normal RAM, and it is used to store setup information. 28F010 CMOS chips are used to implement the flash memory on this system.

5.4 Addressing Flash Memory

The flash memory implemented in the system is composed of 128k $\times$ 8 28F010 CMOS chips decoded into two banks, with the highest address at 0xFFFFF.

6 Programmable Peripheral Interface

The 82C55A is a popular interfacing component that can interface any TTL-compatible I/O device to the microprocessor. The chip is used to interface the keyboard and the LCD controller in the system.

6.1 Addressing

The 3 82C55 chips are decoded at the following port addresses respectively:

- 0x000F (control register), 0x000D, 0x000B, 0x0009
- 0x000E (control register), 0x000C, 0x000A, 0x0008
- 0x0007 (control register), 0x0005, 0x0003, 0x0001

6.2 Assembly Implementation and Programming of the 82C55A

7 Programmable Keyboard/Display Interface - 8279

16 push button switches connected as four rows and four columns, 2 push button switches connected to the control and shift inputs as well as the reset switch connected to the IR0 pin of the 8259 interrupt controller and NMI keys are arranged as the 5×4 keyboard matrix into the 8279 keyboard controller for the system.

7.1 Addressing

The 8279 keyboard controller is decoded at 0x00F2 (command) and 0x00F0 (data).

7.2 Programming the Keyboard Interface

7.3 Command Words to Program the 8279

7.4 Assembly Implementation

The implementation of the keyboard controller is provided in Section B.2.2 of Appendix B.

8 Programmable Interval Timer - 8254

An 8254 programmable interval timer chip is integrated into the system. The component is required to solve the common timing control problems in the microprocessor. The chip also facilitates in other aspects of the system, such as accurate time delays, real time clocks, event counters, square wave generation, etc. The 8254 has 3 independent 16-bit programmable counters, each capable of counting in binary or binary coded decimal (BCD) with a maximum frequency of 1 MHz.

8.1 Programming

8.2 Addressing

8.3 Assembly Implementation

9 UART

9.1 16550 UART

9.2 Addressing the 16550

9.3 Programming the 16550

9.4 Assembly Implementation

9.5 MAX-235 and D-SUB-9

9.6 Device Descriptions and Implementations

10 Interrupt Controller - 8259

10.1 Implementing a Master Interrupt Controller

10.2 Addressing

10.3 Assembly Implementation and Programming

11 LCD Display

A 20×4 LCD display with no backlight and an integrated LCD controller are included in the board.

11.1 Addressing

The LCD controller decoded at addresses 0x00D6, 0x00D4, 0x00D2, and 0x00D0 is used to interface the LCD display.

11.2 Assembly Implementation

The implementation of the LCD controller is provided in Section B.2.1 of Appendix B.

12 LEDs and DIP Switches

2 common-anode seven-segment LEDs with a decimal point segment, 8 standalone LEDs and 8 DIP switches are also connected to and integrated into the system.

12.1 Seven-Segment LEDs

74LS374 latches decoded at 0x00CE and 0x00CF are used to interface the 7-segment LEDs.

12.2 LEDs

A 74LS374 latch decoded at 0x00CC is used to interface the 7-segment LEDs.

12.3 DIP Switches

A 74LS244 buffer decoded at 0x00CA is used to interface the 8 DIP switches.

13 External Headers

External headers are used to pull address, data and control lines and other port connections for external access.

13.1 30-Pin Headers with the 8255

30-pin headers are used to pull the port connections for external access. The connections are demonstrated on the schematic in Figure 9.

13.2 14-Pin Headers with the 8254

14-pin headers are used to pull the counters' outputs for external access. The connections are demonstrated on the schematic in Figure 11.

13.3 14-Pin Headers with the 8259

14-pin headers are used to pull the interrupt request lines for external access. The connections are demonstrated on the schematic in Figure 13.

13.4 60-Pin External Header with the Address, Data and Control Bus

30-pin headers are used to pull the address, data and control buses for external access. The connections are demonstrated on the schematic in Figure 6.

A Appendix A: Schematics

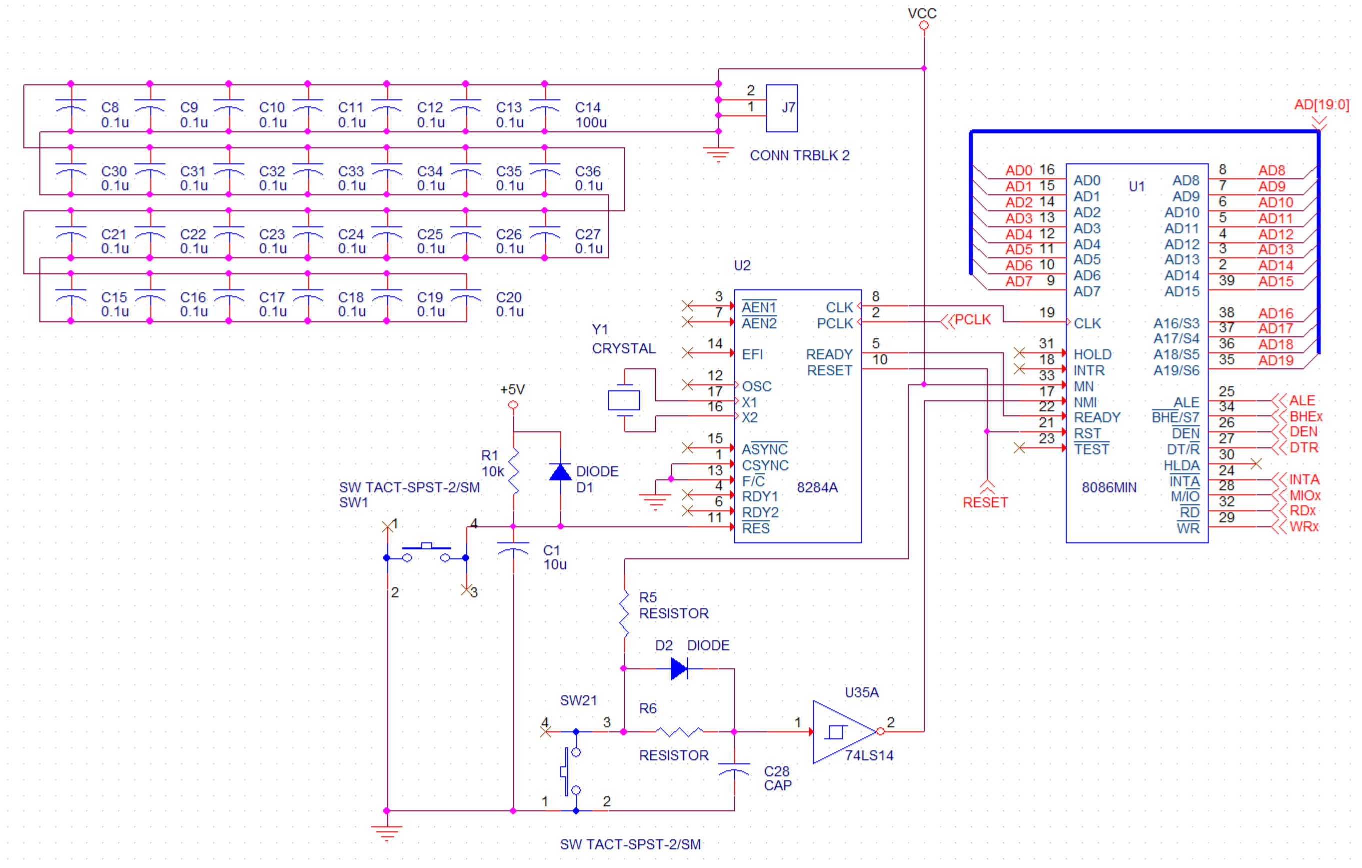

Figure 5: 8086 interfaced with the 8284A clock generator and its Reset RC Push Button Circuit, and the Power Bank of the Board

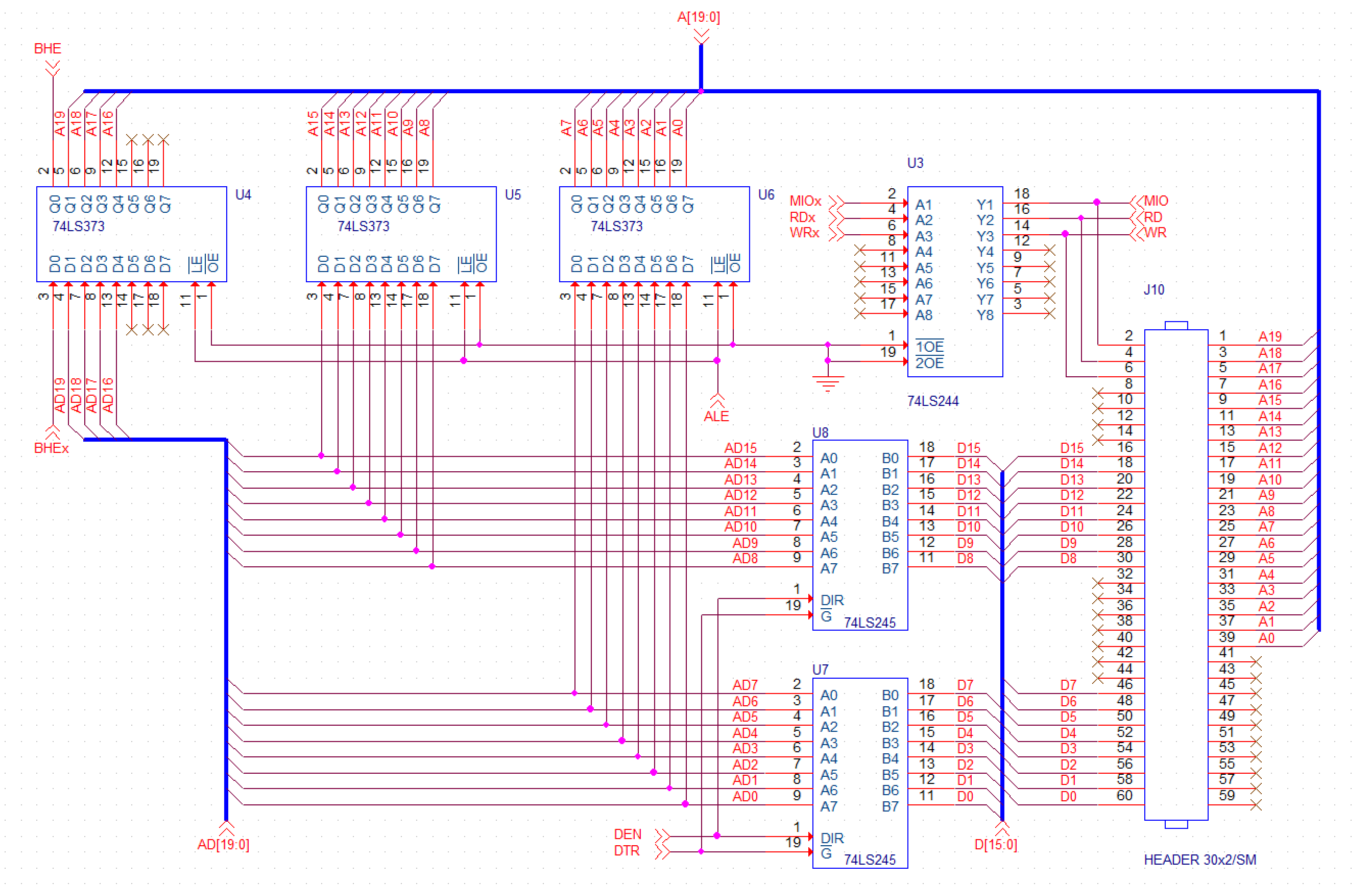

Figure 6: 8086 Demultiplexed with Address and Data Buses Pulled into Headers

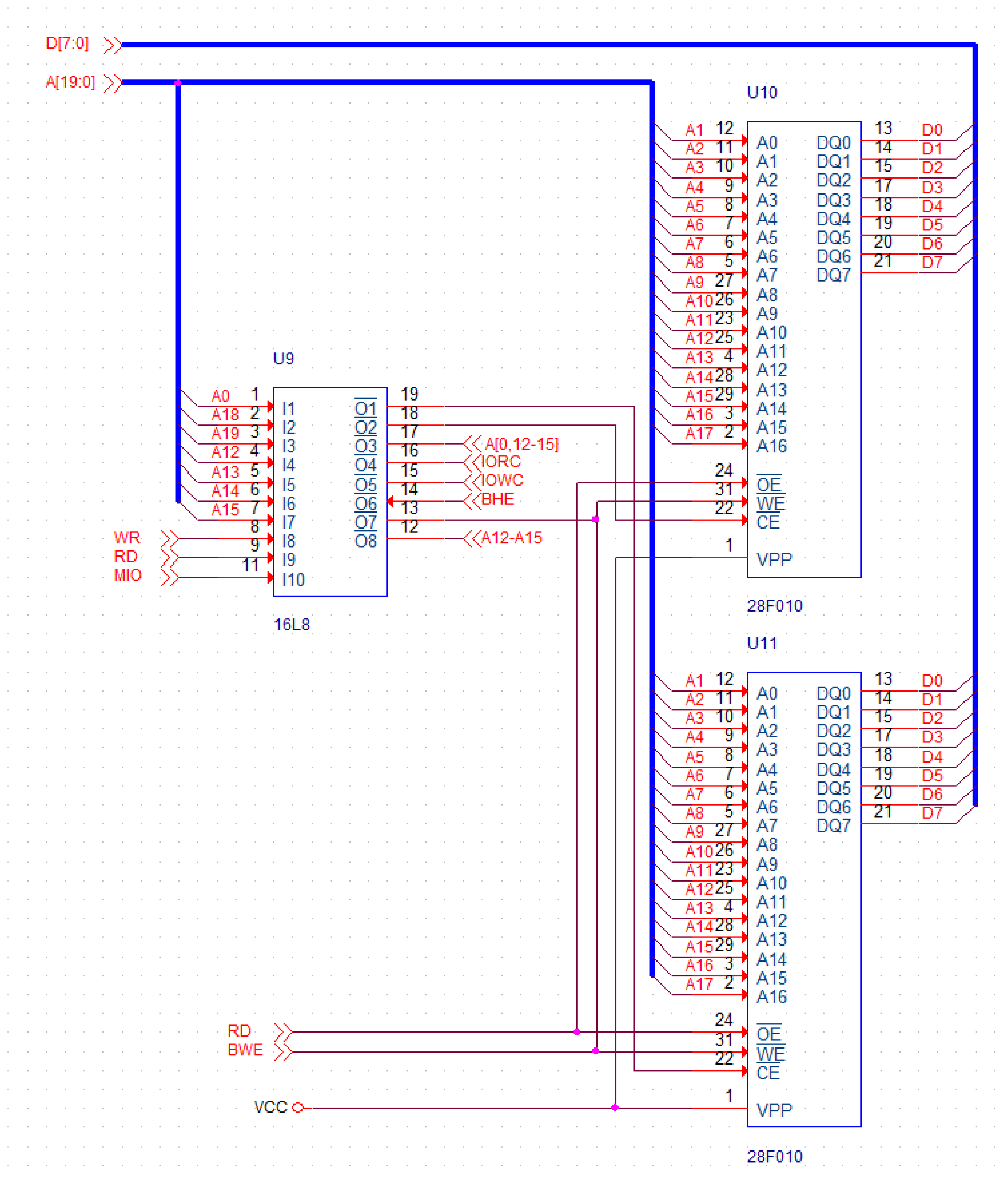

Figure 7: 256 kB of CMOS Flash Memory and 128 kB Static SRAM

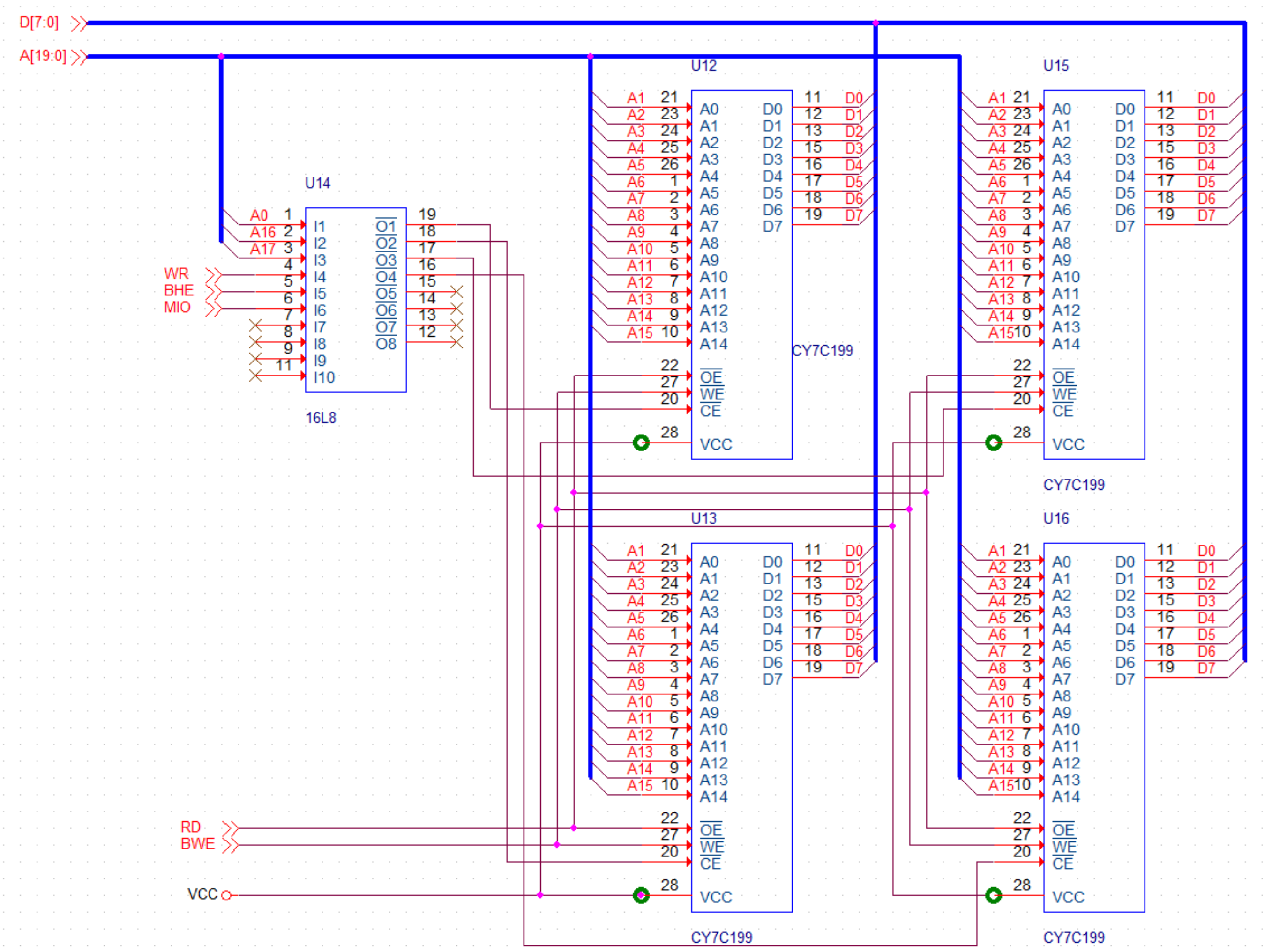

Figure 8: 128 kB Static SRAM

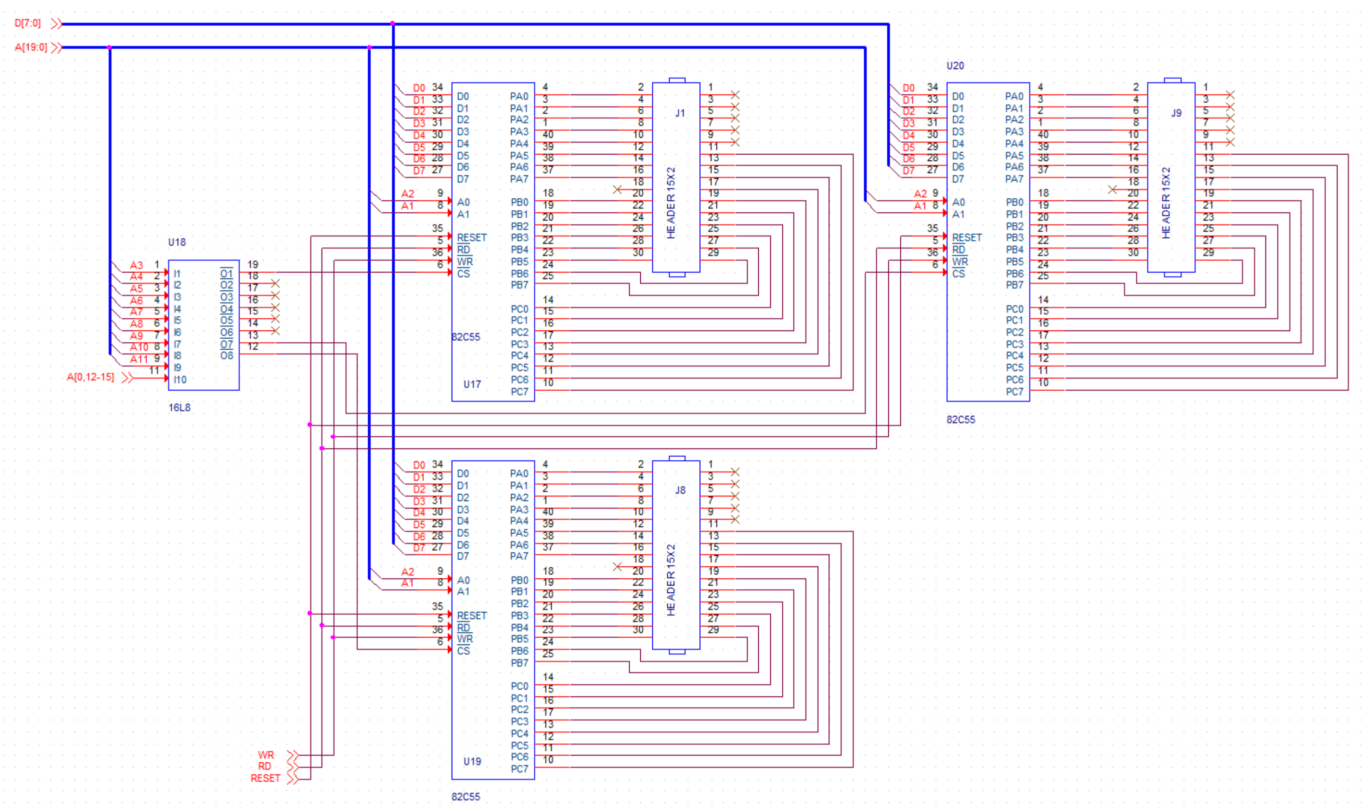

Figure 9: Programmable Peripheral Interface Chips with Port Connections Pulled into Headers

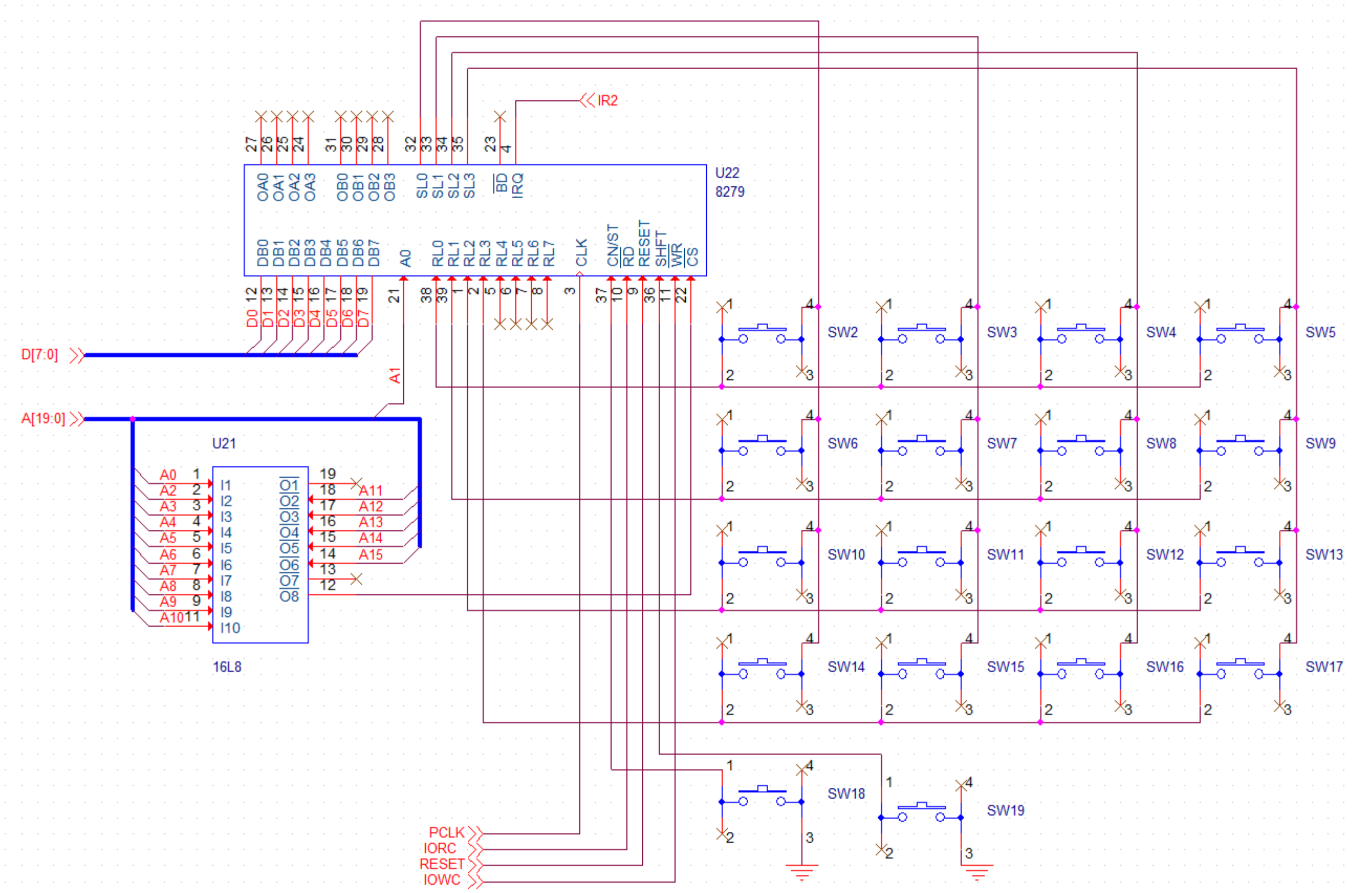

Figure 10: 5×4 Keyboard Matrix

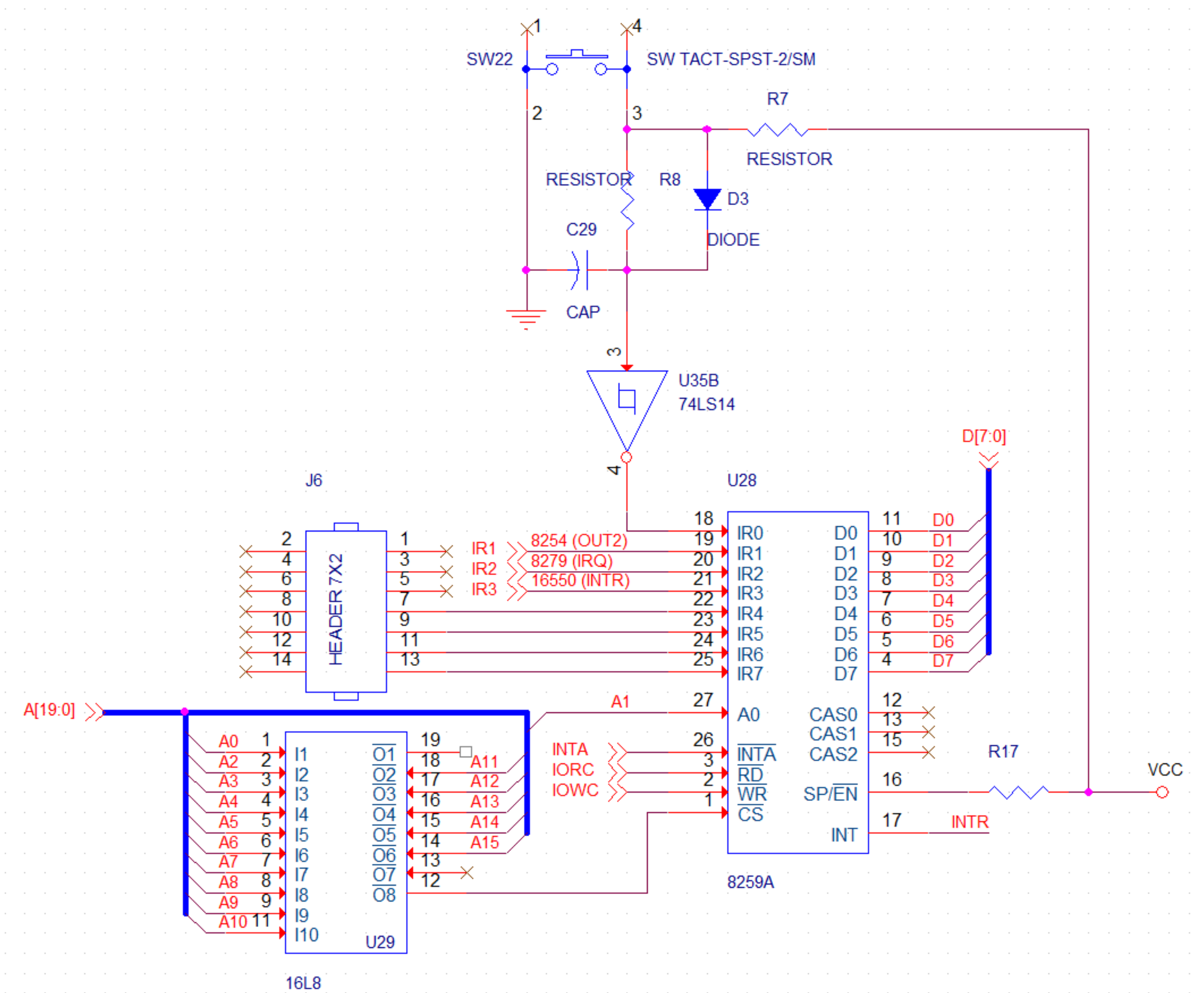

Figure 11: Programmable Interval Timer

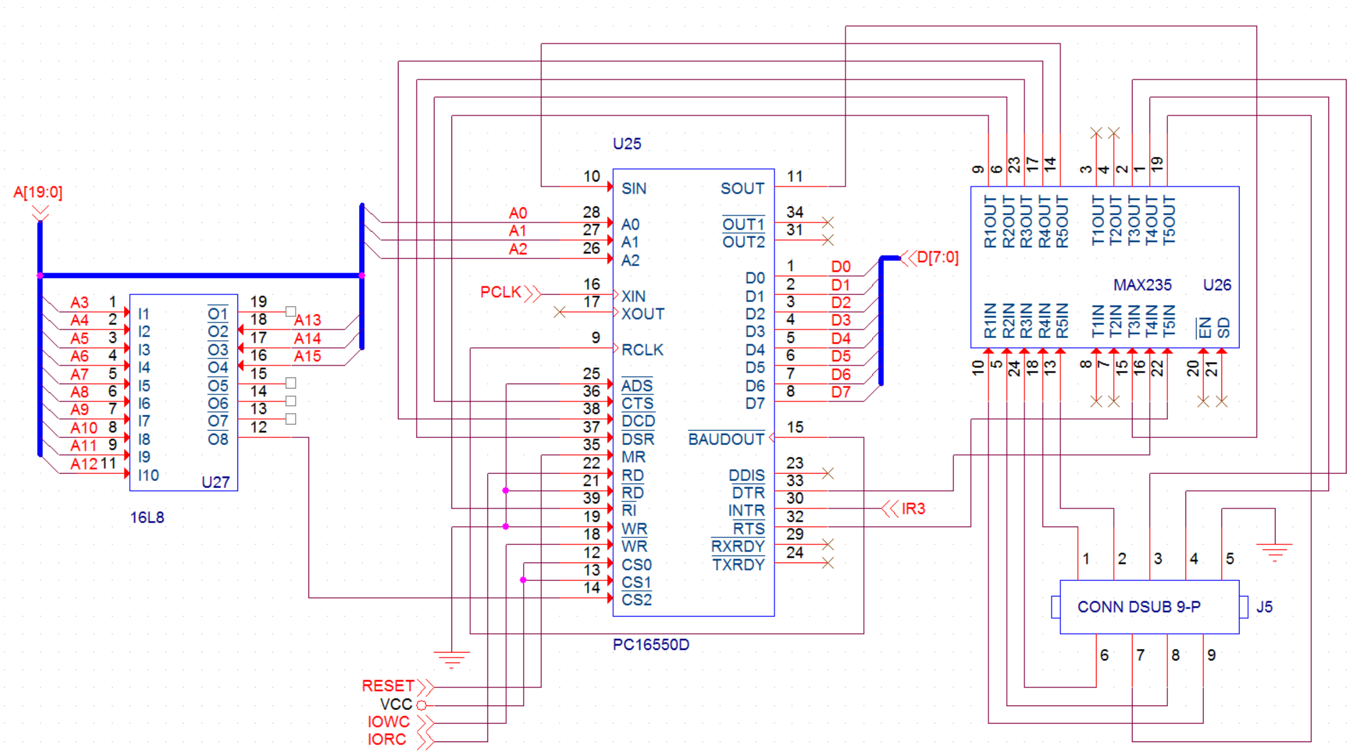

Figure 12: UART Connected for Serial Port Using a Line Driver/Receiver and a DSUB-9 connector

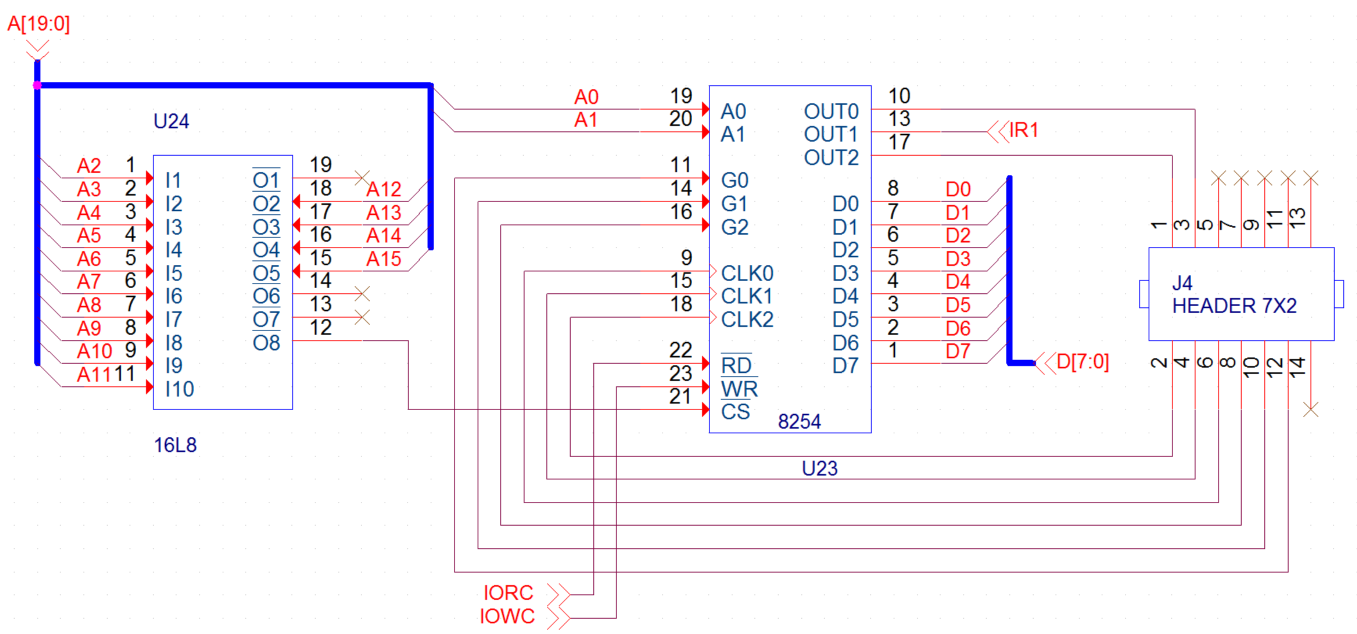

Figure 13: Programmable Interrupt Controller with Headers fro External Access

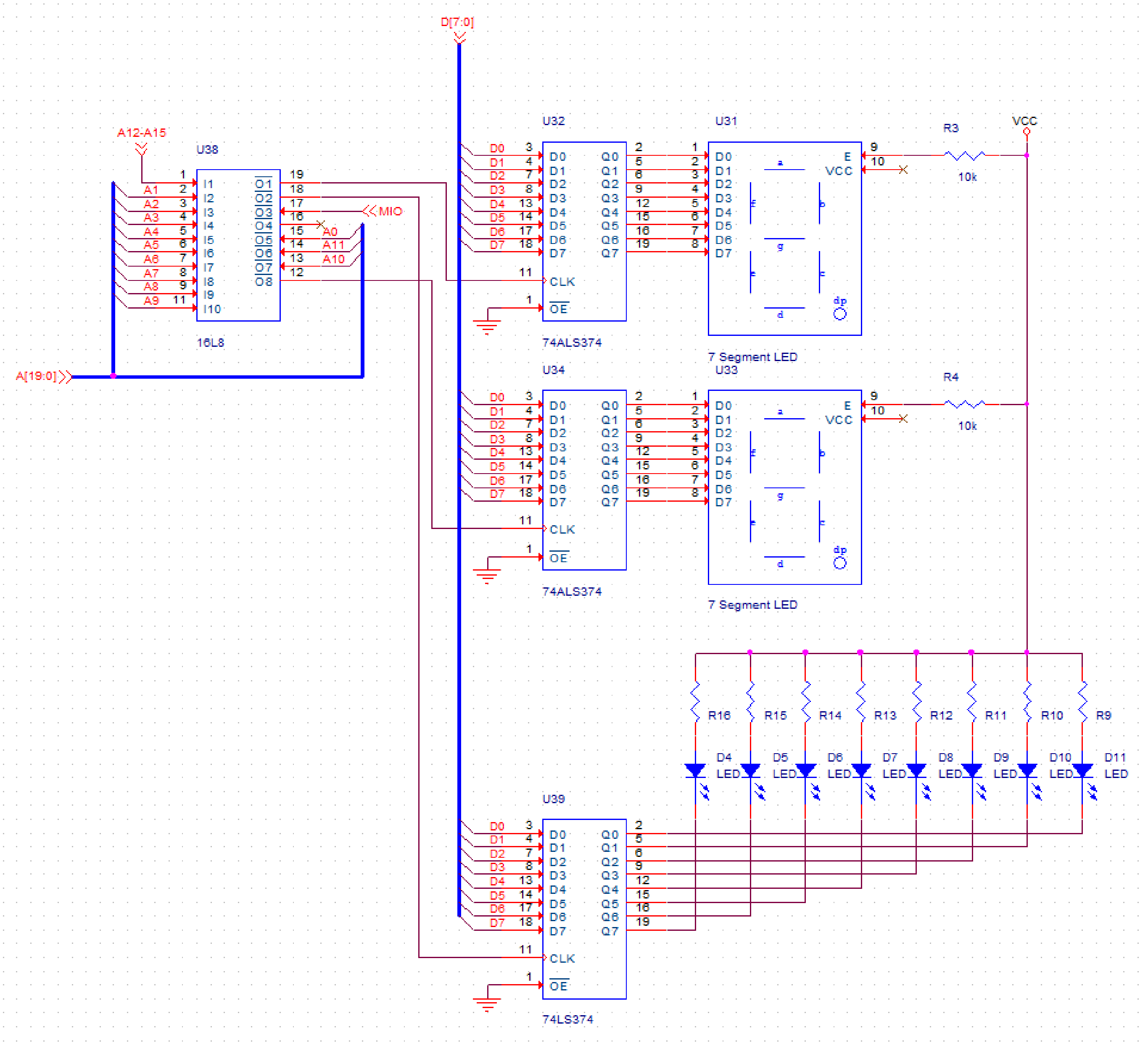

Figure 14: Common-Anode 7-Segment LEDs with 8 LEDs

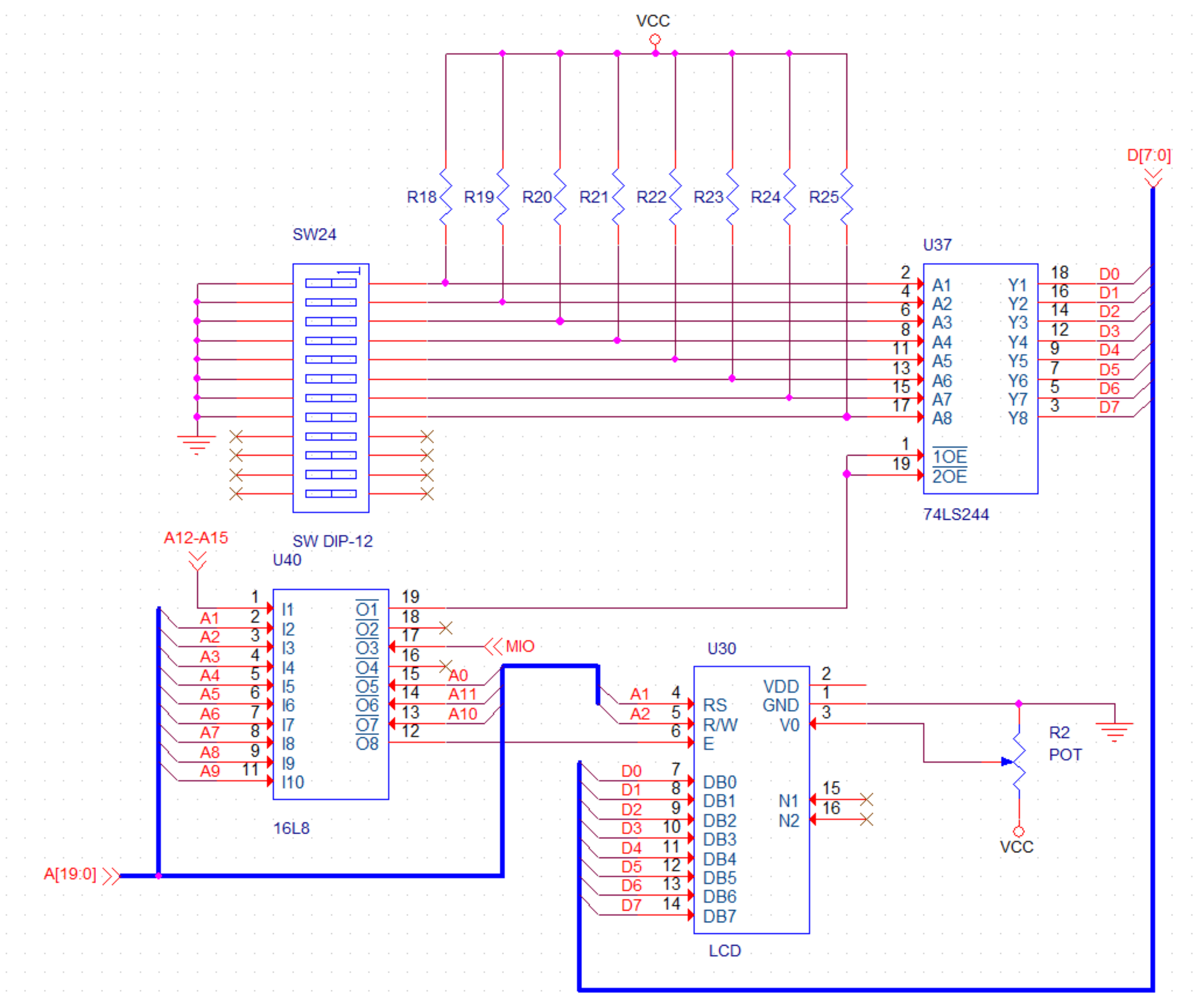

Figure 15: 20 character × 4 line LCD Display with an Integrated LCD Controller

B Appendix B: Code Implementations

B.1 Programmable Logic Devices

B.1.1 U9

```
--------------------------------------------------------------------------------
--
-- Module: DECODER_U9
-- Architecture used to decode address and control lines to the CMOS
-- flash memory
--
--------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U9 is
    port (
        A0, A12, A13, A14, A15, A18, A19, WR, RD, MIO, BHE: in STD_LOGIC;
        O1, O2, A12_A15, A0xA12_A15, IORC, IOWC: out STD_LOGIC
    );
end DECODER_U9;

architecture V1 of DECODER_U9 is
begin
    O1 <= A19 or A18 or A15 or A14 or A13 or A12 or not(A0);
    O2 <= A19 or A18 or A15 or A14 or A13 or A12 or A0;
    A12_A15 <= A15 or A14 or A13 or A12;
    A0xA12_A15 <= A15 or A14 or A13 or A12 or A0;
    IORC <= RD or MIO;
    IOWC <= WR or MIO;
end V1;
```

B.1.2 U14

```
--------------------------------------------------------------------------------
--
-- Module: DECODER_U14
-- Architecture used to decode address and control lines to the SRAM
--
--------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U14 is
    port (
        A0, A16, A17, WR, BHE, MIO: in STD_LOGIC;
        O1, O2, O3, O4: out STD_LOGIC
    );
end DECODER_U14;

architecture V1 of DECODER_U14 is
begin
    O1 <= not(A17) or not(A16) or A0 or not(WR) or not(BHE) or not(MIO);
    O2 <= not(A17) or A16 or A0 or not(WR) or not(BHE) or not(MIO);
    O3 <= A17 or not(A16) or A0 or not(WR) or not(BHE) or not(MIO);
    O4 <= A17 or A16 or A0 or not(WR) or not(BHE) or not(MIO);
end V1;
```

B.1.3 U18

```
--------------------------------------------------------------------------------
--
-- Module: DECODER_U18
-- Architecture used to decode address and control lines to the PPI chips
--
--------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U18 is
    port (
        A12_A15, A3, A4, A5, A6, A7, A8, A9, A10, A11, MIO: in STD_LOGIC;
        PPI1, PPI2, PPI3: out STD_LOGIC
    );
end DECODER_U18;

architecture V1 of DECODER_U18 is
begin
    A12_A15 <= A15 or A14 or A13 or A12;
    PPI1 <= A12_A15 or
            A11 or A10 or A9 or A8 or
            A7 or A6 or A5 or A4 or
            not(A3) or not(A0) or not(MIO);
    PPI2 <= A12_A15 or
            A11 or A10 or A9 or A8 or
            A7 or A6 or A5 or A4 or
            not(A3) or A0 or not(MIO);
    PPI3 <= A12_A15 or
            A11 or A10 or A9 or A8 or
            A7 or A6 or A5 or A4 or
            A3 or not(A0) or not(MIO);
end V1;
```

B.1.4 U21

```
-------------------------------------------------------------------------------
--
-- Module: DECODER_U21
-- Architecture used to decode address and control lines to the keyboard
--
-------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U21 is
    port (
        A0, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14,
            A15: in STD_LOGIC;
        O8: out STD_LOGIC
    );
end DECODER_U21;

architecture V1 of DECODER_U21 is
begin
    O8 <= A15 or A14 or A13 or A12 or
          A11 or A10 or A9 or A8 or
          not(A7) or not(A6) or not(A5) or not(A4) or
          A3 or A2 or A0;
end V1;
```

B.1.5 U24

```
--------------------------------------------------------------------------------
--
-- Module: DECODER_U24
-- Architecture used to decode address and control lines to the
-- programmable interval counter
--
--------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U24 is
    port (
        A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14,
            A15: in STD_LOGIC;
        O8: out STD_LOGIC
    );
end DECODER_U24;

architecture V1 of DECODER_U24 is
begin
    O8 <= not(A15) or not(A14) or not(A13) or not(A12) or
          not(A11) or not(A10) or not(A9) or not(A8) or
          not(A7) or not(A6) or not(A5) or A4 or
          not(A3) or A2;
end V1;
```

B.1.6 U27

```
--------------------------------------------------------------------------------
--
-- Module: DECODER_U27
-- Architecture used to decode address and control lines to the UART
--
--------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U27 is
    port (
        A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14,
            A15: in STD_LOGIC;
        O8: out STD_LOGIC
    );
end DECODER_U27;

architecture V1 of DECODER_U27 is
begin
    O8 <= A15 or A14 or A13 or A12 or
          A11 or A10 or A9 or A8 or
          not(A7) or not(A6) or not(A5) or A4 or
          A3;
end V1;
```

B.1.7 U29

```
------------------------------------------------------------------------------
--
-- Module: DECODER_U29
-- Architecture used to decode address and control lines to the interrupt
-- controller
--
------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U29 is
    port (
        A0, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14,
            A15: in STD_LOGIC;
        O8: out STD_LOGIC
    );
end DECODER_U29;

architecture V1 of DECODER_U29 is
begin
    O8 <= not(A15) or not(A14) or not(A13) or not(A12) or
          not(A11) or not(A10) or not(A9) or not(A8) or
          not(A7) or not(A6) or not(A5) or not(A4) or
          A3 or not(A2);
end V1;
```

B.1.8 U38

```
-------------------------------------------------------------------------------
--
-- Module: DECODER_U38
-- Architecture used to decode address and control lines to the LEDs
--
-------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U38 is
    port (
        A0, A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12_A15,
            MIO: in STD_LOGIC;
        O1, O2, O8: out STD_LOGIC
    );
end DECODER_U38;

architecture V1 of DECODER_U38 is
begin
    A12_A15 <= A15 or A14 or A13 or A12;
    O1 <= A12_A15 or
          A11 or A10 or A9 or A8 or
          not(A7) or not(A6) or A5 or A4 or
          not(A3) or not(A2) or not(A1) or A0;
    O2 <= A12_A15 or
          A11 or A10 or A9 or A8 or
          not(A7) or not(A6) or A5 or A4 or
          not(A3) or not(A2) or A1 or A0;
    O8 <= A12_A15 or
          A11 or A10 or A9 or A8 or
          not(A7) or not(A6) or A5 or A4 or
          not(A3) or not(A2) or not(A1) or not(A0);
end V1;
```

B.1.9 U40

```
--------------------------------------------------------------------------------
--
-- Module: DECODER_U40
-- Architecture used to decode address and control lines to the LCD and DIP
-- switches
--
--------------------------------------------------------------------------------


library ieee;
use ieee.std_logic_1164.all;

entity DECODER_U40 is
    port (
        A0, A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12_A15,
            MIO: in STD_LOGIC;
        O1, O8: out STD_LOGIC
    );
end DECODER_U40;

architecture V1 of DECODER_U40 is
begin
    A12_A15 <= A15 or A14 or A13 or A12;
    O1 <= A12_A15 or
          A11 or A10 or A9 or A8 or
          not(A7) or not(A6) or A5 or A4 or
          not(A3) or A2 or not(A1) or A0;
    O8 <= A12_A15 or
          A11 or A10 or A9 or A8 or
          not(A7) or not(A6) or A5 or not(A4) or
          A3 or A0;
end V1;
```

B.2 Assembly Implementations

B.2.1 LCD

```
; ----------------------------------------------------------------------------
; Instructions to initialize the LCD controller integrated in the system with
; the 82C55 Programmable Peripheral Interface (PPI) chip.
;
; To assemble and run with the Makefile provided:
;     make SCRIPT=lcd run
;
; ----------------------------------------------------------------------------

PORTA_ADDR  EQU 700H            ; set port addresses
PORTB_ADDR  EQU 701H
CMD_ADDR    EQU 703H

; macro to send a command or data to the LCD display
SEND MACRO PORTA_DATA, PORTB_DATA, DELAY
    MOV     AL, PORTA_DATA      ; PORTA_DATA to Port A
    MOV     DX, PORTA_ADDR
    OUT     DX, AL
    MOV     AL, PORTB_DATA      ; PORTB_DATA to Port B
    MOV     DX, PORTB_ADDR
    OUT     DX, AL
    OR      AL, 00000100B       ; set E bit
    OUT     DX, AL              ; send to Port B
    AND     AL, 11111011B       ; clear E bit
    NOP                         ; a small delay
    NOP
    OUT     DX, AL              ; send to Port B
    MOV     BL, DELAY           ; BL = delay count
    CALL    MS_DELAY            ; ms Time Delay
    ENDM

; Program to initialize the LCD display
START:
    MOV     AL, 0FH             ; program the 82C55
    MOV     DX, CMD_ADDR
    OUT     DX, AL
    MOV     AL, 0
    MOV     DX, PORTB_ADDR      ; clear Port B
    SEND    30H, 2, 16          ; send 30H for 16 ms
    SEND    30H, 2, 5           ; send 30H for 5 ms
    SEND    30H, 2, 1           ; send 30H for 1 ms
    SEND    38H, 2, 1           ; send 38H for 1 ms
    SEND    8, 2, 1             ; send 8 for 1 ms
    SEND    1, 2, 2             ; send 1 for 2 ms
    SEND    0CH, 2, 1           ; send 0CH for 1 ms
    SEND    6, 2, 1             ; send 6 for 1 ms
```

B.2.2 Keyboard

```
; ----------------------------------------------------------------------------
; Instructions to initialize the keyboard controller integrated in the system
; with the 82C55 Programmable Peripheral Interface (PPI) chip.
;
; To assemble and run with the Makefile provided:
;     make SCRIPT=keyboard run
;
; ----------------------------------------------------------------------------

KEYBRD_ADDR  EQU 00F2H          ; set 8279 addresses

; Program to initialze the keyboard with the 8279
START:
    MOV     DX, KEYBRD_ADDR     ; point to 8279 command address
    MOV     AL, 00000011B       ; set mode for left entry, 8-digit display,
                                ; decoded scan, N-key rollover
    OUT     DX, AL              ; send to 8279
    MOV     AL, 3EH             ; clock word for divide by 30
    OUT     DX, AL
    MOV     AL, C0H             ; clear display to display all zeroes
```

C Appendix C: PCB Layouts

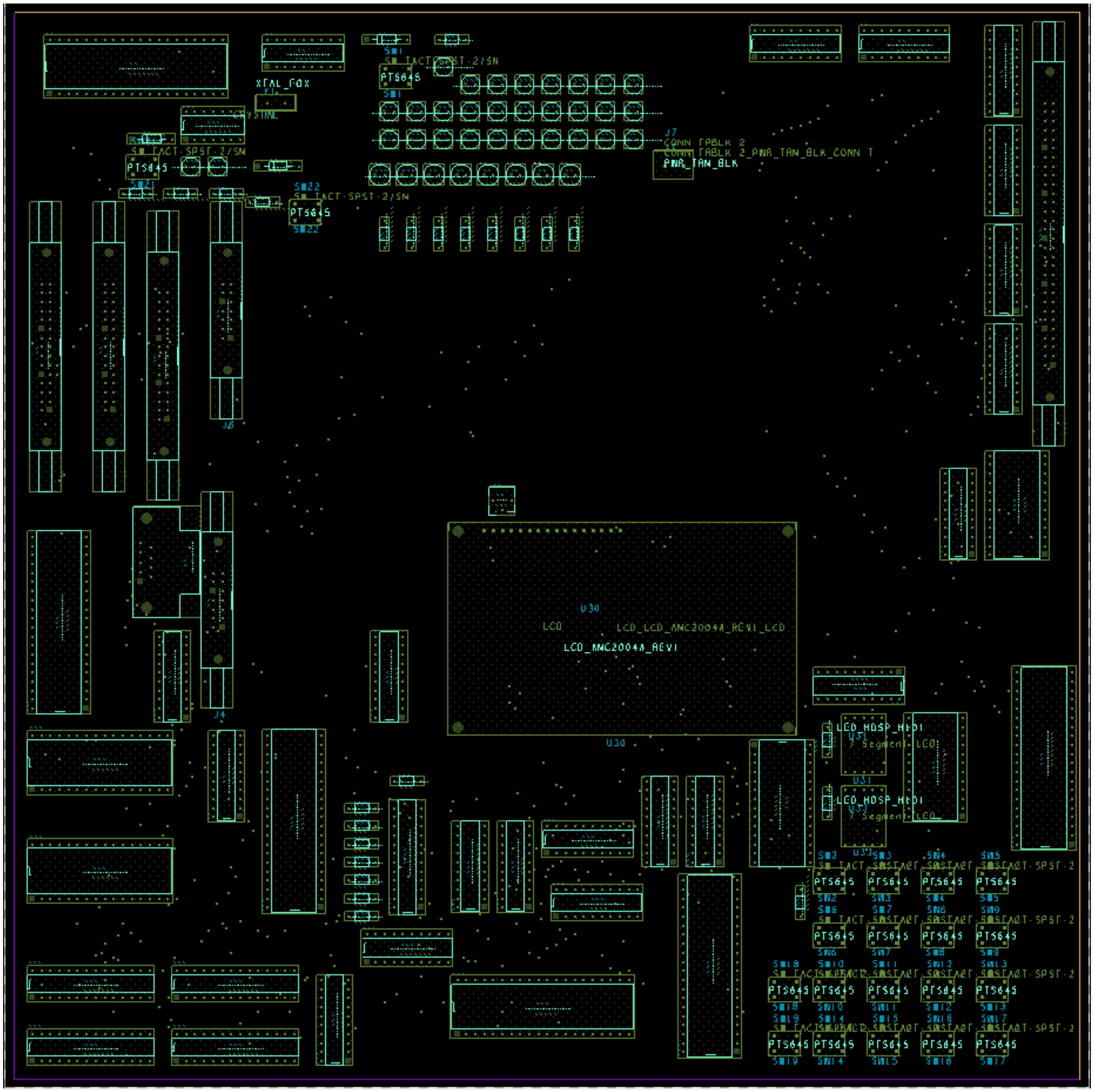

Figure 16: 8086 interfaced with the 8284A clock generator and its Reset RC Push Button Circuit, and the Power Bank of the Board

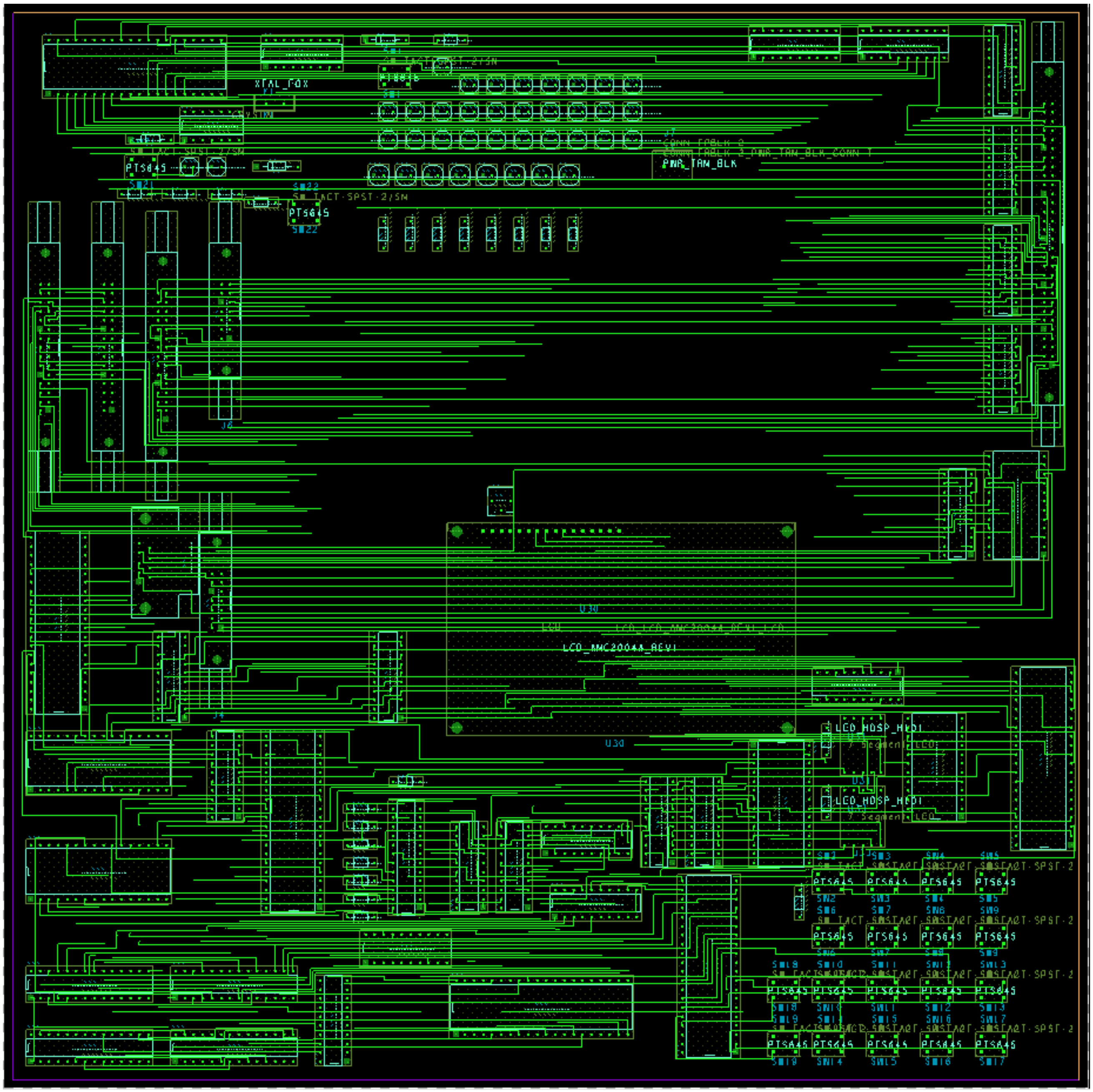

Figure 17: 8086 interfaced with the 8284A clock generator and its Reset RC Push Button Circuit, and the Power Bank of the Board

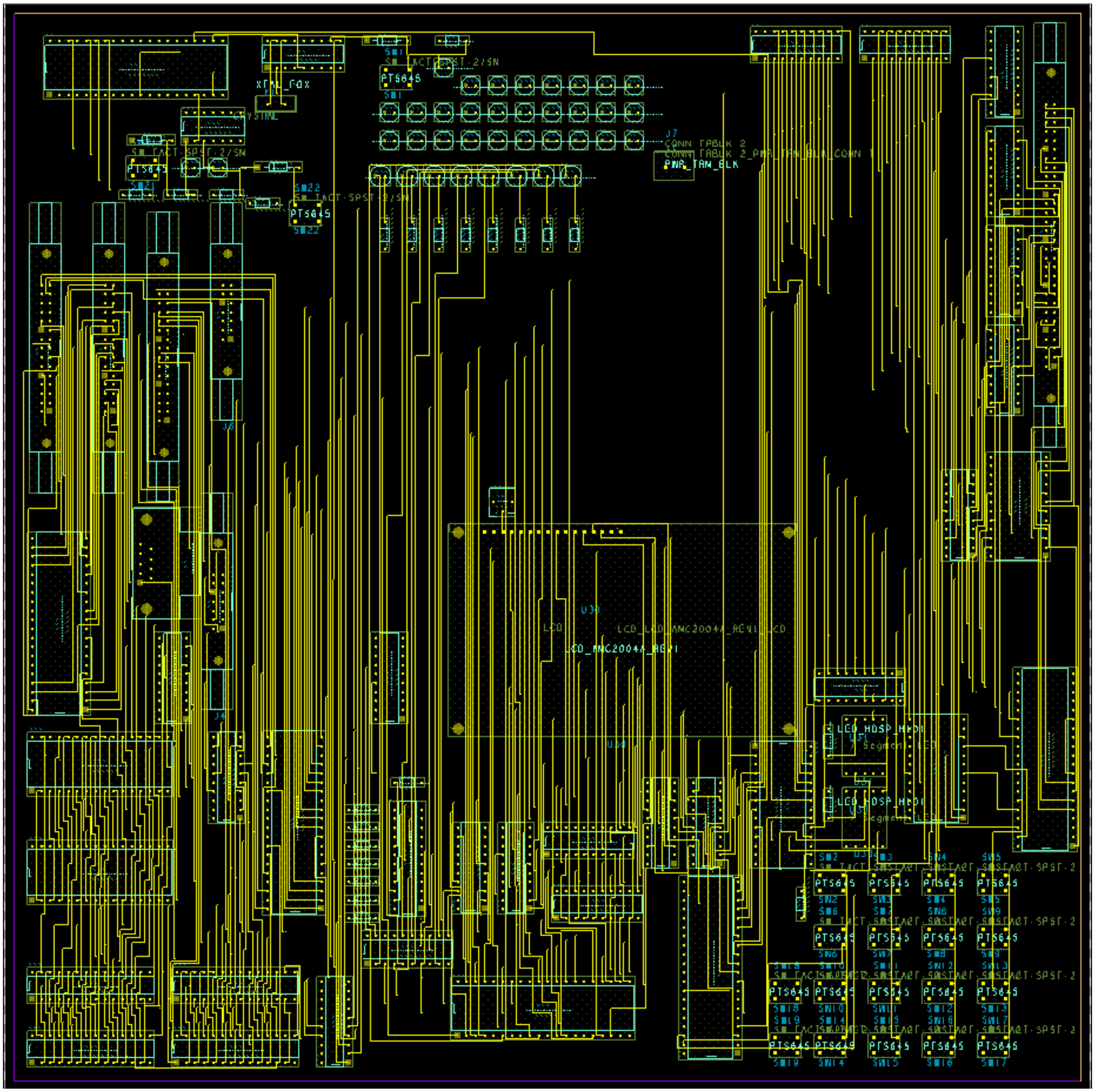

Figure 18: 8086 interfaced with the 8284A clock generator and its Reset RC Push Button Circuit, and the Power Bank of the Board

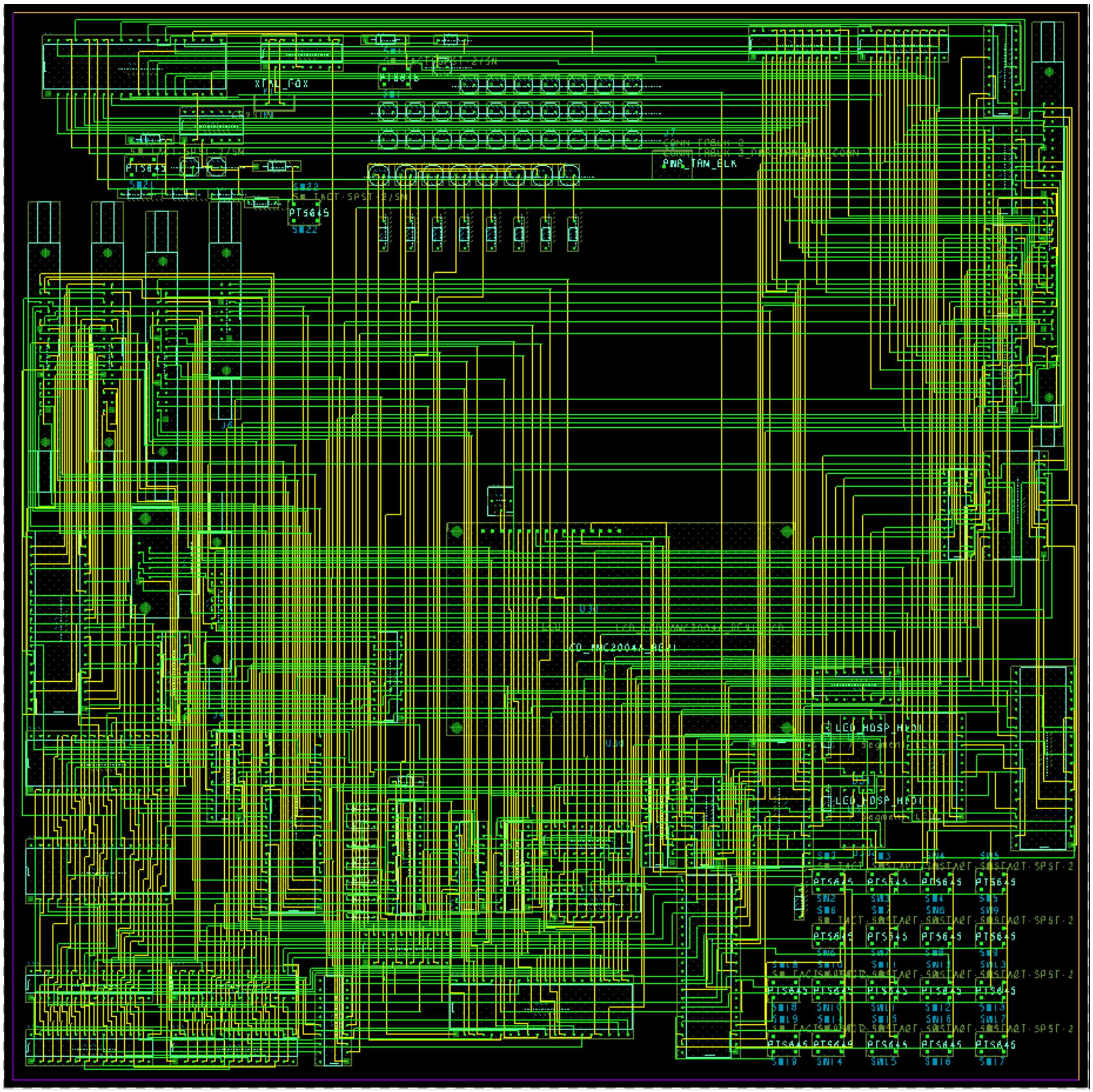

Figure 19: 8086 interfaced with the 8284A clock generator and its Reset RC Push Button Circuit, and the Power Bank of the Board

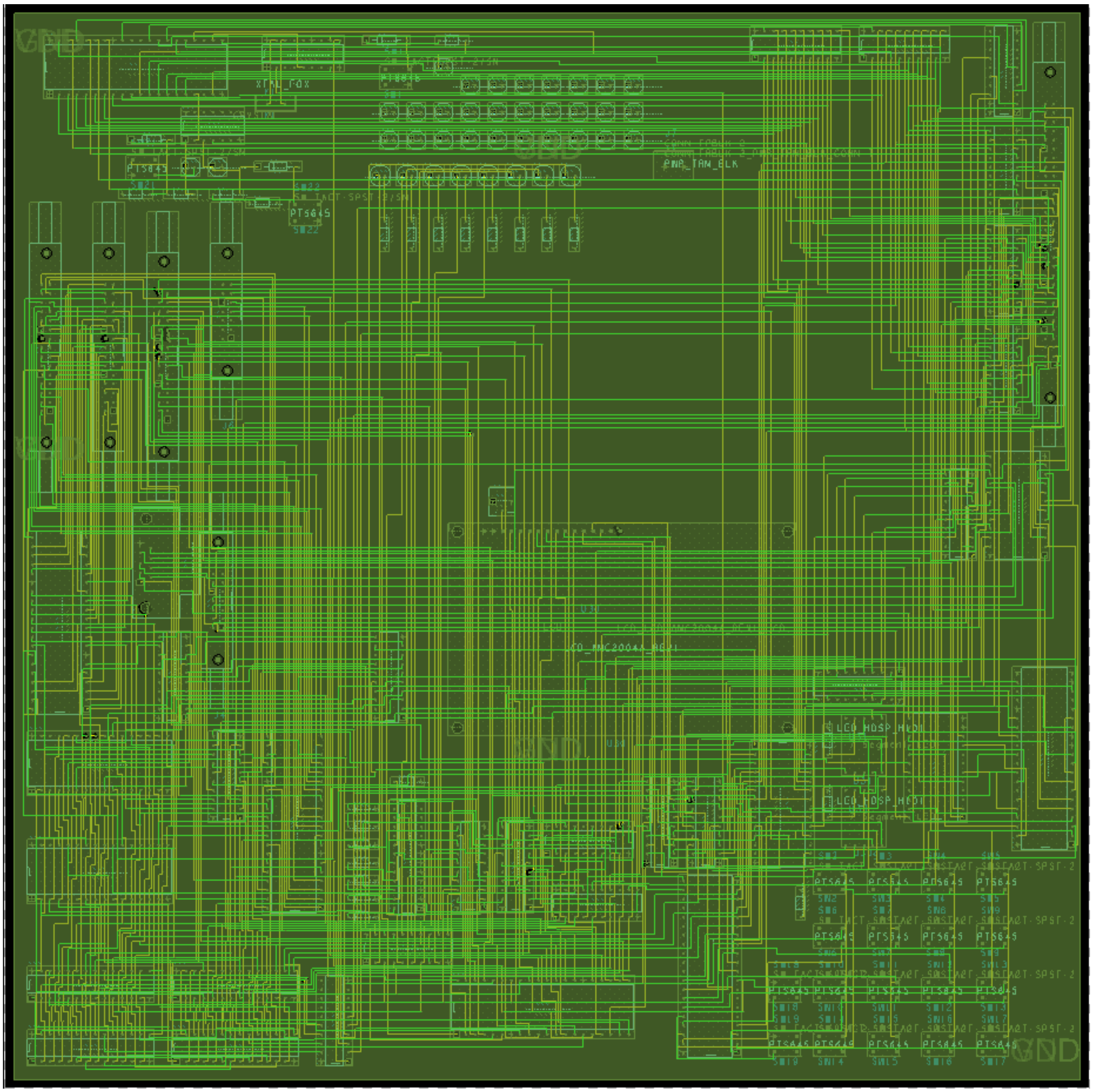

Figure 20: 8086 interfaced with the 8284A clock generator and its Reset RC Push Button Circuit, and the Power Bank of the Board

References

[1] DBHJDS
http://gradestack.com/Microprocessors-and/Architecture-of-8086-and/Address-Bus-Data-Bus-/19317-3912-38171-study-wtw

[2] Programmable Array Logic
https://www.electrical4u.com/programmable-array-logic/