8-Bit CPU Curriculum and Kit

Written and Designed by: Jeremy Ho, Justin Merlia, Tyler Tetens, and Eric Walter

Acknowledgments

We would like to thank Dr. Stanley for being our advisor throughout the design and development of this kit. Dr. Stanley has given us fantastic advice on how to better this project in many ways. We would also like to thank Dr. Woodley for giving us a structure as to how we should be going about creating a product and keeping us true to deadlines. Finally we would like to thank the Electrical and Computer Engineering department at the Missouri University of Science and Technology. The department has been instrumental in helping us develop into the professionals we are.

Table of contents

8-Bit CPU Kit Overview

This kit is intended to help anyone understand the architecture of a basic processor by guiding them through the process of building one step-by-step. It is modular to make sure that connecting various components can be built and explained individually while showing what they are doing in real time using LEDs. This kit will give an explanation of each individual component, it's functions in the cpu, and why it is vital to the end creation.

In order to utilize this kit successfully the creators assume that the readers have a basic understanding of electronics and circuits. Inside this curriculum there is a basic layout of the fundamentals of electronics and transistors. For further readings the creators suggest **Digital Fundamentals** by: Thomas L. Floyd.

Basic Computer Architecture Review

A computer is comprised of many different parts. The part this curriculum and kit are focused on is the Central Processing Unit. The CPU is the brain of the computer. A CPU is comprised of many different modules. The modules we are going to focus on are a clock, arithmetic logic unit, random access memory, program counter, output register, input register, Two arithmetic registers, instruction register, and control logic. Each of these modules will be detailed below.

There are many different architectures that a computer can be made from. Some types of popular computer architectures are Harvard, Von Neumann, and RISC. The CPU that we will be using uses a SAP-1 design. This stands for Simple as Possible.

A computer architecture is just a set or rules and connections. These rules and connections come together in an organized fashion to run multiple pieces of code. On a hardware level there are many transistors that are linked together via wires. When a transistor turns on and off they produce a signal that is read by a different set of transistors and wires. This continues until an output is produced. On a software level there are certain transistors that are set in the EEPROM (The EEPROM will be described down below). These transistors are the instructions for all the other transistors to do thing.

Within a computer there are many different operations that can happen. This is determined by its architecture. For this CPU we are focusing on sixteen different instructions. This might seem like a lot of things to do but in reality very little can be programmed. A modern CPU uses well over a thousand instructions to do what you see on a display, such as your phone or computer display.

Intro to Electronics

It is important important to understand the principles behind circuits and electronic components before building a CPU. The CPU we are building consists of mainly resistors, capacitors, transistors, and diodes. It is important to understand why the components are used in their respective circuits to achieve a bigger goal. The long black rectangles are commonly referred to as IC's or Integrated Circuits. These rectangles with pins on them have many transistors inside of them in various configurations.

When dealing with electronics there are three fundamentals. These fundamentals are current, voltage, and resistance. When hooking two or more components together such as a LED, resistor and power supply it is important to give what is necessary.

A very important formula to know is **Ohm's Law: Voltage(V) = Current(I) * Resistance(R)**, to determine current , voltage, and resistance of a circuit. Voltage is the difference in charge between two points in the circuit. Current is the rate of at which the charge is flowing. Resistance is how much the material resists the current. Finding these three fundamentals tells you how your circuit is working. If voltage is not traveling to the appropriate destination your circuit will not work. If you provide to little or too much voltage your circuit has a potential for damage and/or will not work.

For this kit we are dealing with five volts (5V) throughout the entire circuit. Each component utilizes this voltage. As you add more components to the CPU how much current is drawn will change. This is due to more charge being needed to fulfill the needs of the components.

The ICs used are composed of many transistors. A transistor is a current gat this is controlled from a third pin. A transistor comes in various shapes and sizes and types. We will be working with bipolar junction transistors or BJTs. These BJTs come in two types, NPN and PNP.

Intro to Binary Numbers

Binary numbers are a base-2 number system, meaning the only two symbols used are 0's and 1's. This system is commonly used in computers as it simplifies the design of computers. The numeral system we are used to is in base-10 otherwise known as the decimal system 0 through 9. Hexadecimal or base-16 is also commonly used in computing and

represented by 0-9 and A-F. Below is a table showing the conversions of numbers in decimal to binary to hex.

Decimal	Binary	Hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Hexadecimal is primarily used to read binary. It is easier to read FA then it is to read 11111010. Binary is what machines use. A 1 is represented by a high signal commonly 3.3v and a 0 is represented by a low signal or 0v. When counting in binary each 0 or 1 is a bit. Four bits is a nibble. Two nibbles is a byte. In other words 8 bits = 1 byte. This 8-bit CPU uses 8 bits in

order to transmit data. This data sometimes has to be calculated. In order for binary numbers to be counted three simple rules are followed:

0 + 0 = 0

1 + 0 = 1

1 + 1 = 0 and a carry out bit of 1.

Subtraction of binary works similarly to addition except that you perform the two's complement of subtractor before you start addition. To perform two's complement you flip all the 0's to 1's and 1's to 0's then you add 1.

Example:

Two's complement of 1111.

Flip all the bits: 1111 = 0000

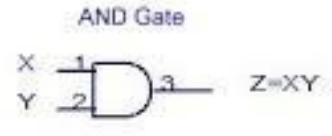
Add 1: 0000 + 1 = 0001

Intro to Logic Gates

Transistors used in the computers perform many operations. These transistors are configured in ways to provide many types of gates. The gates we will be focusing on are the six fundamentals. AND, NOT, OR, NAND, NOR, and XOR gate. The gates we use can have various inputs. We will be focusing on two input for all five of the gates and a one input NOT gate. Each gate below is described with its symbol and its truth table. The truth table describes each gates operations. For instance the AND gate describes how when only both inputs are 1's the output is a 1. For each other configuration of its inputs the output is a 0.

AND Gate

2 Input AND Gate

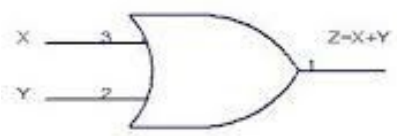


TRUTH TABLE

INPUTS		OUTPUT
X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate

2 Input OR Gate

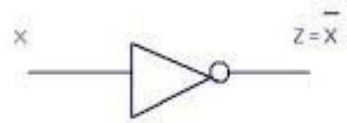


TRUTH TABLE

INPUTS		OUTPUT
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

NOT Gate

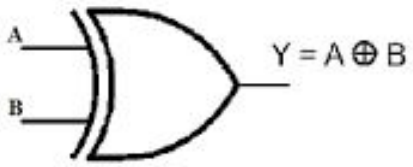
NOT Gate



TRUTH TABLE

INPUT	OUTPUT
X	Z
0	1
1	0

XOR Gate



INPUT		OUTPUT
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

NAND Gate

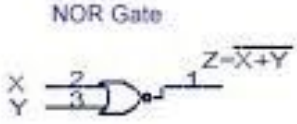
2 Input NAND Gate

TRUTH TABLE

INPUTS		OUTPUT
X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate

2 Input NOR Gate



TRUTH TABLE

INPUTS		OUTPUT
X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

These six gates provide many of the functionality of a CPU in many different configurations. These symbols will be shown later in how to build the circuit.

Important Parts in your CPU

Arithmetic Logic Unit (ALU)

The ALU is the part of the computer that executes mathematical operations. For our SAP architecture we will have two functions. We will implement addition and subtraction operations.To perform these operations we will be using full-adders. A full-adder takes in two bits of binary and adds them together. Multiple chains of full-adders together create the addition and subtraction operations. These full adders are made for us as a 74LS283.

The ALU uses two registers to perform its operations. The A and B Registers are connected to the ALU. Reading and writing to these registers performs the operations of addition and subtraction in our CPU.

A and B Registers

These are general purpose registers that will temporarily store data within the computer's volatile (temporary) memory. These registers are potentially the most important part of the CPU. Without these values operating no mathematical operations can run while the rest of the CPU will still perform with errors. These registers read and write out data to the eight bit bus. All of the chips writing out to the bus are connected via a tri-state buffer. This buffer provides a key role to only enable certain modules to write to the bus during certain operations. This ensures data accuracy and provides the top down run methodology used in this SAP architecture.

Control Logic

The control logic is the brain of the operation. It determines what happens next in each instruction and what the processor recognizes. It receives and sends data to allows communication between each components.

Instruction Register

The instruction register holds the instruction currently being executed or decoded as well as an address that the instruction will operate on. This component serves an important purpose. The first four bits constantly feed the op code in the control logic. The last four bits feed into the computer so that the address can be transferred into the MAR or Program Counter.

Random Access Memory (RAM)

The RAM stores the program that the computer is actively using and any data that is needed by the program so that the computer is able to retrieve the information it needs quickly and efficiently.

Memory Address Register (MAR)

The MAR stores the memory address where the data will be taken from in the CPU. This only holds the location of the data that needs to be accessed. This chip constantly feeds into the memory unless the computer is being programmed. The multiplexer used allows the computer to select from two four bit inputs. These inputs are the MAR or user input via DIP switches.

EEPROM

This is short for Electrically Erasable Programmable Read-Only Memory. An EEPROM is used by the user to erase, create, and modify programs to determine what the computer can do. The EEPROM is used in the Control Logic and Output Register to provide read and write access for controlling what is currently executing.

Program Counter

The program counter is a register used by the computer to show the location or address of the current instruction to be executed. After fetching a new instruction, the program counter is increased by 1. It can be shifted to different places to perform subroutines, but at the end of these subroutines it is returned to the value before entering.

In order for the program counter to work it needs to count. One of the most essential parts of a computer is its ability to count. To count jk flip flops are used. Flip flops change their state when a single from the clock module is received. Whenever the falling edge (the downward portions of the square wave signal) is detected the state change occurs. In order for a state change to happen though the current state needs to be saved. To save the current state the flip flops using an array of logic gates that store one bit. By using multiple flip-flops we can connect them together the result is a binary counter.

The program counter also performs a few more instructions for the CPU. the PC has the ability to load in the JMP opcode which allows the computer to jump to a specific instruction.

Clock

A clock is needed to execute instructions and each instruction requires a number of clock cycles before it can process another instruction. This part also synchronizes all the components on the board for the computer to work properly. 555 timers are used in order create the pulse necessary. Three timers are used in conjunction with logic gates to provide different pulses. The first mode is just constant output while the other mode is a single pulse. A single pulse is necessary in order to provide a simple walk step through ability. This step through ability will allow the user to walk through the example code pre-loaded in the EEPROM

Output Register

The output register is similar to the A and B registers. This register doesn't write out the data to leds in binary though. The output register is attached to four seven-segment displays. These displays show the user the current output in decimal or hex. This can be changed via how the EEPROM associated with the output register is programmed.Initially in order to decode a binary number to decimal. Decoding this usually requires a lot of logic gates in a complex array. Luckily EEPROM can be reprogrammed in order to do perform like logic gates. This is the only register where a single chip is used instead of discrete logic to simplify the CPU.

Parts List

Described Below is a complete list of the components provide in the kit you purchased.
Not included: Soldering Equipment or Personal protective equipment.

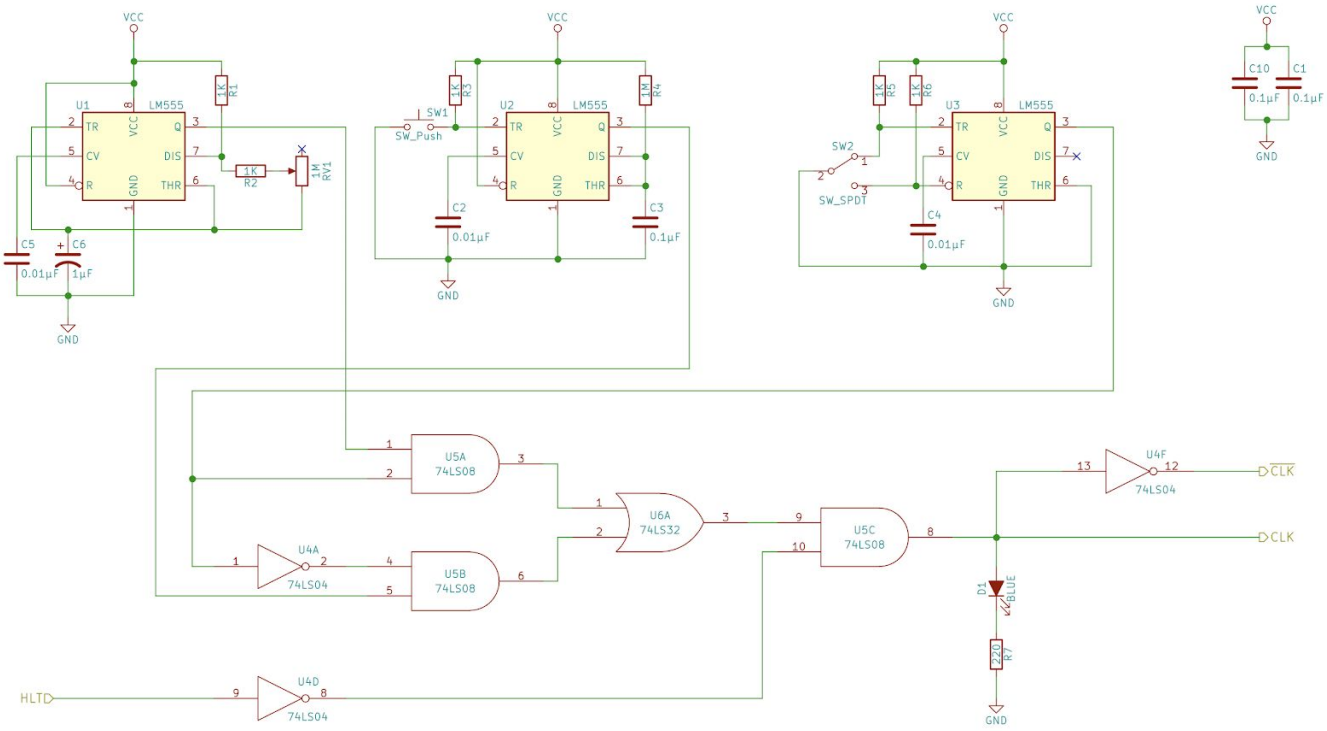
Description	Quantity
PCB modules	11
1K Ohm Resistor	10
10K Ohm Resistor	9
100K Ohm Resistor	1
470 Ohm Resistor	24
1M Ohm Resistor	1
1M Ohm potentiometer	1
0.01 Microfarad Capacitor	6
0.1 Microfarad Capacitor	16
1 Microfarad Capacitor	1
555 timer IC	4
74LS00	2
74LS02	1
74LS04	5
74LS08	3
74LS32	1
74LS76	1
74LS86	2
74LS138	1
74LS139	1

74LS157	4
74LS161	2
74LS173	8
74189	2
74LS245	6
74LS273	1
74LS283	2
2816C EEPROM	3
Double-Throw toggle switch	3
Momentary 6mm tactile switch	3
8-position DIP switch	1
4-position DIP switch	1
Red LED	44
Yellow LED	8
Green LED	12
Blue LED	21
7 segment display common cathode	4
5v Wall wart adapter	1
Barrel wire connector	1
Right angle connector 8 pin Male	20
Right angle connector 8 pin Female	20

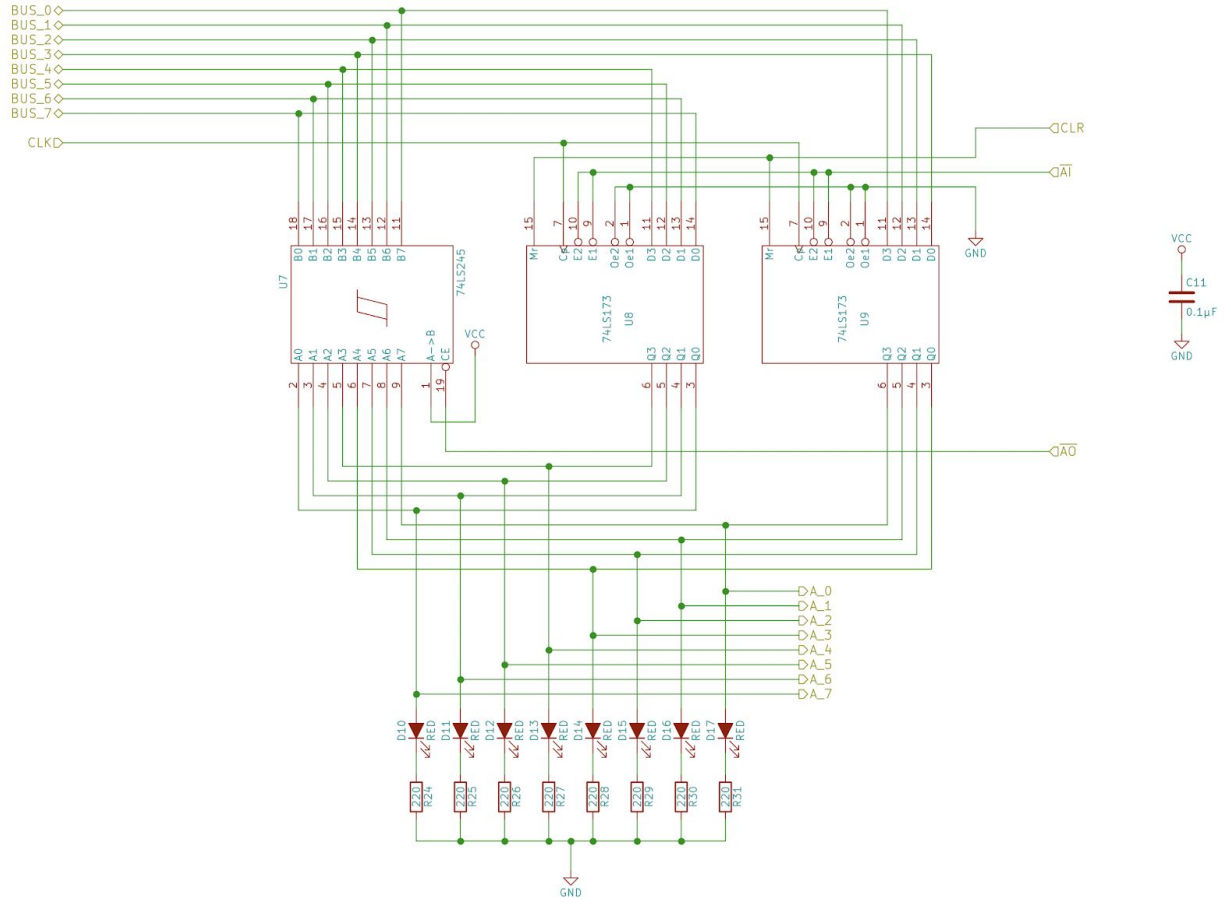
If you would like to reprogram your EEPROM you can purchase an additional Arduino UNO and two 74HC595 Shift registers as a kit. This will enable you to program the EEPROMs used in the output and control logic.

Schematics

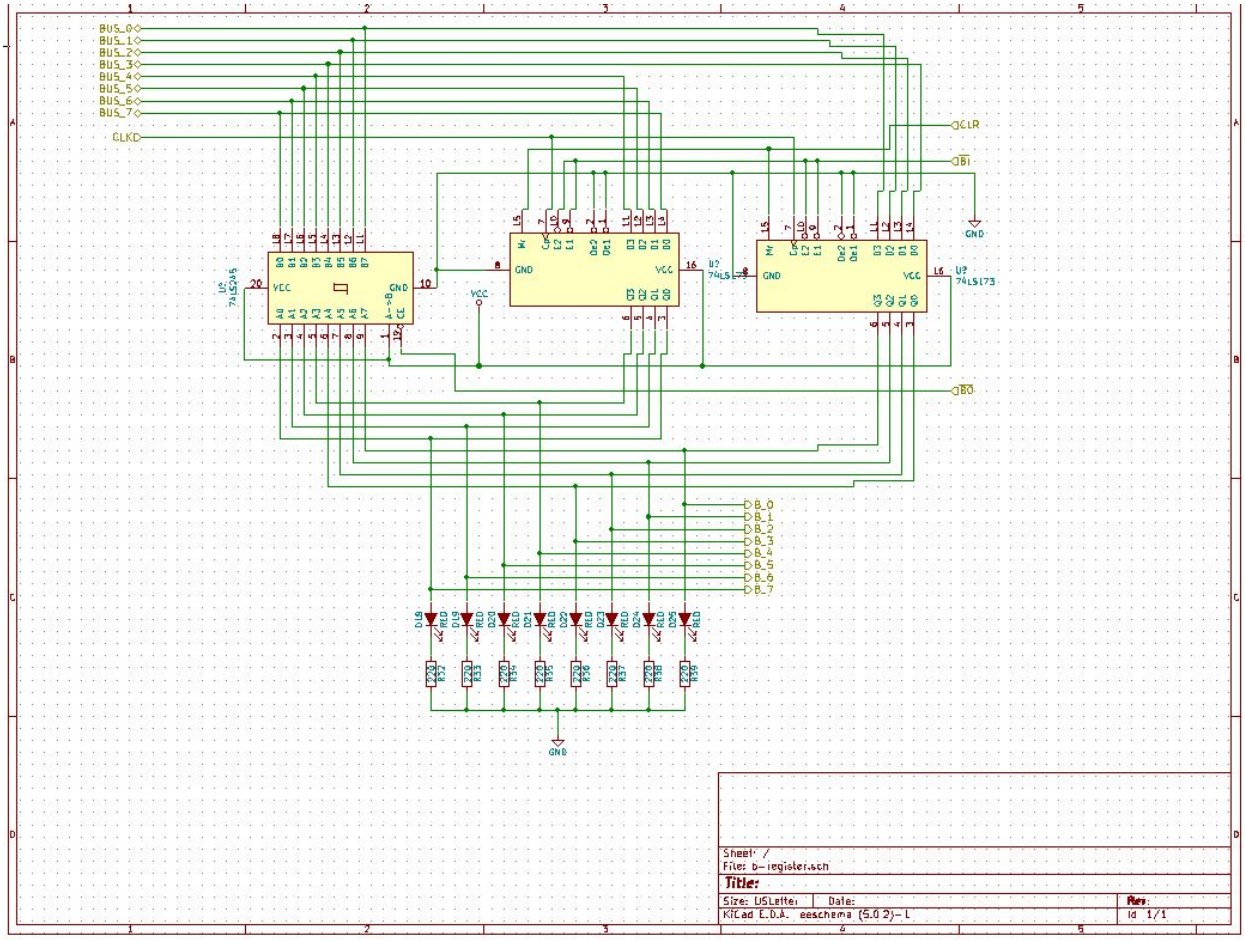
Clock:



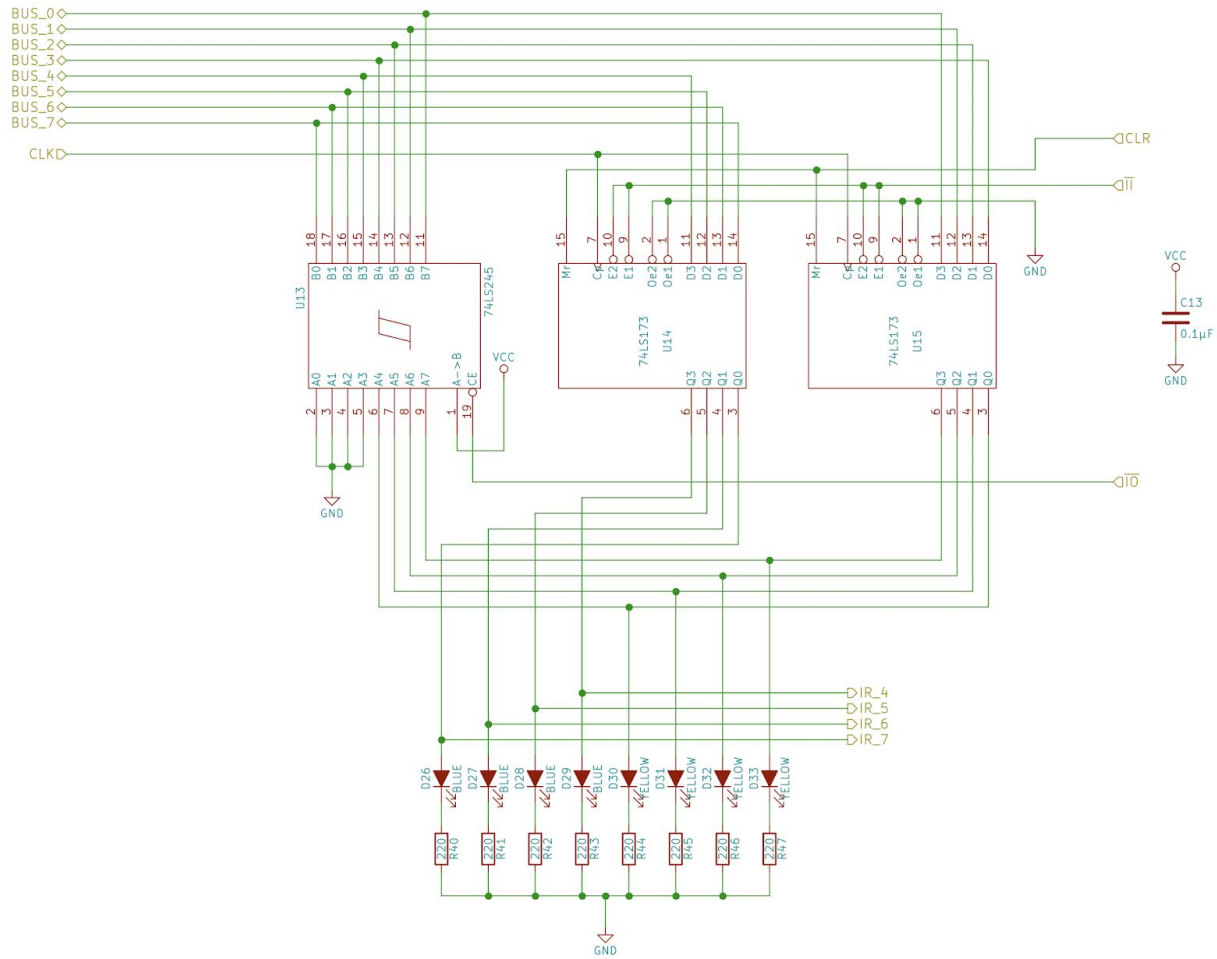
A Register:


BUS_0
BUS_1
BUS_2
BUS_3
BUS_4
BUS_5
BUS_6
BUS_7
CLK
CLR
AI
AO
GND
VCC
C11
0.1µF
U7
74LS245
U8
74LS173
U9
74LS173
A_0
A_1
A_2
A_3
A_4
A_5
A_6
A_7
D10 RED
D11 RED
D12 RED
D13 RED
D14 RED
D15 RED
D16 RED
D17 RED
220 R24
220 R25
220 R26
220 R27
220 R28
220 R29
220 R30
220 R31

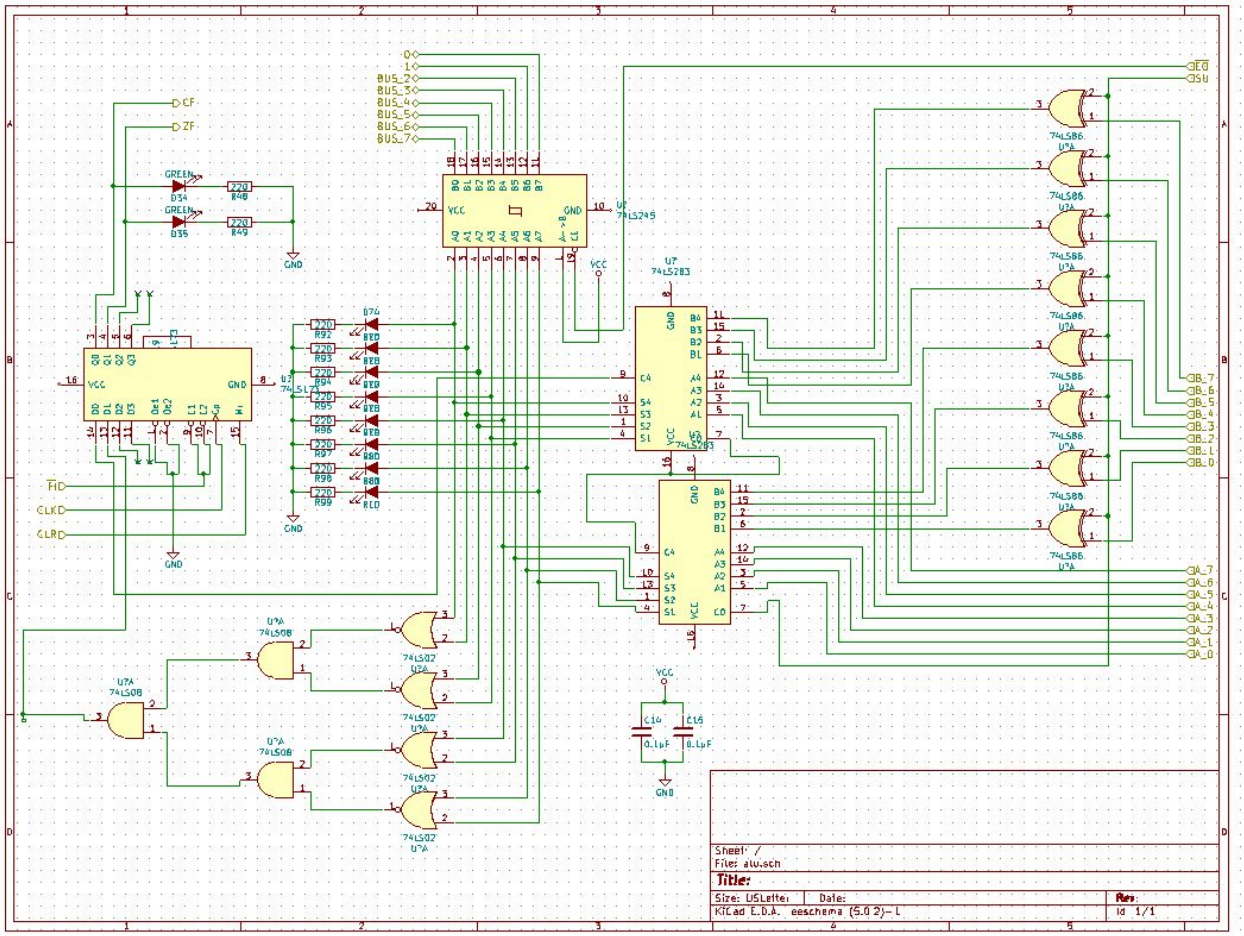
B Register:


BUS_0
BUS_1
BUS_2
BUS_3
BUS_4
BUS_5
BUS_6
BUS_7
CLKD
CLR
BI
BO
GND
VCC
74LS245
74LS173
B_0
B_1
B_2
B_3
B_4
B_5
B_6
B_7
Sheet: /
File: b-register.sch
Title:
Size: USLetter
Date:
KiCad E.D.A. eeschema (5.0.2)-1
Rev:
Id 1/1

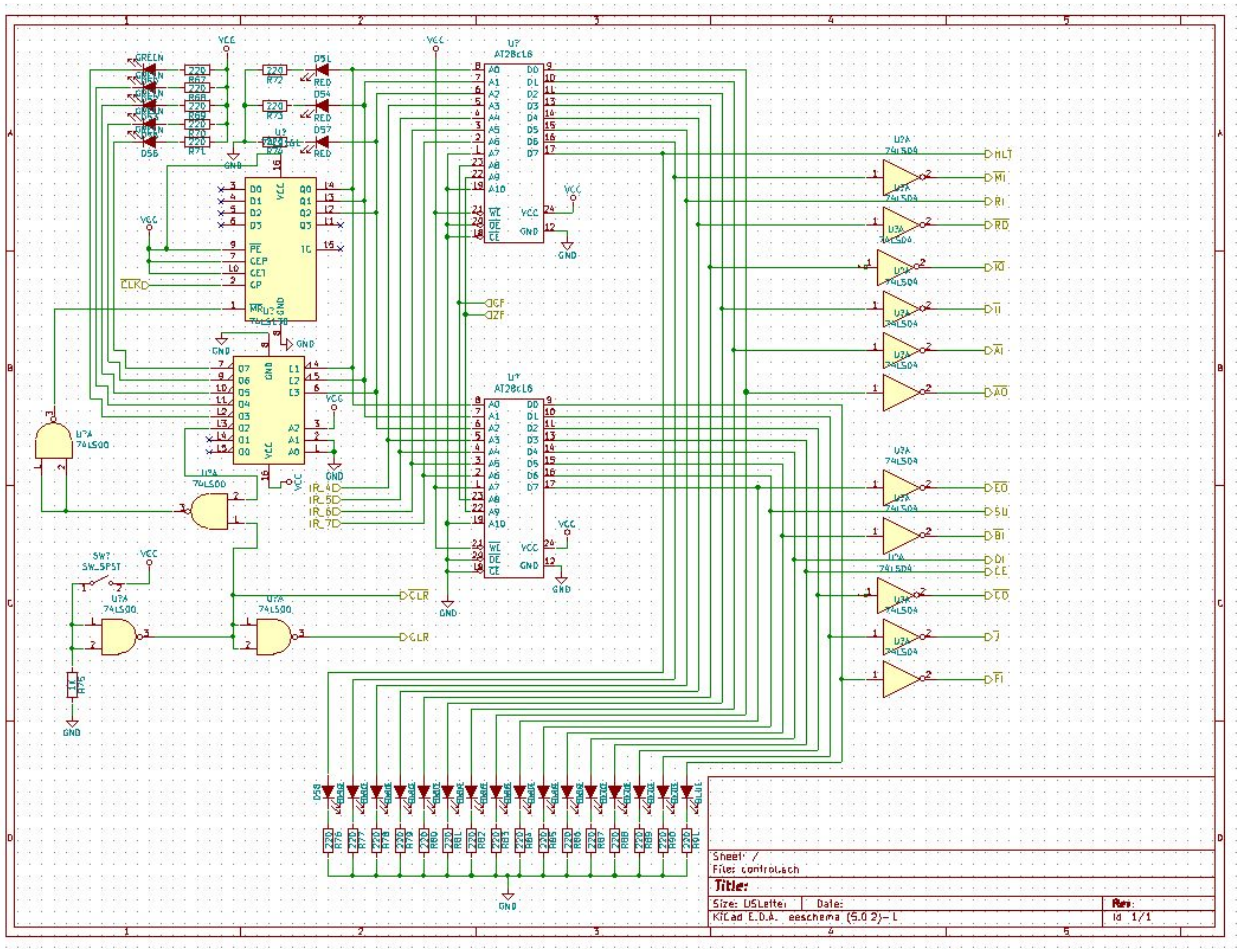
Instruction Register:



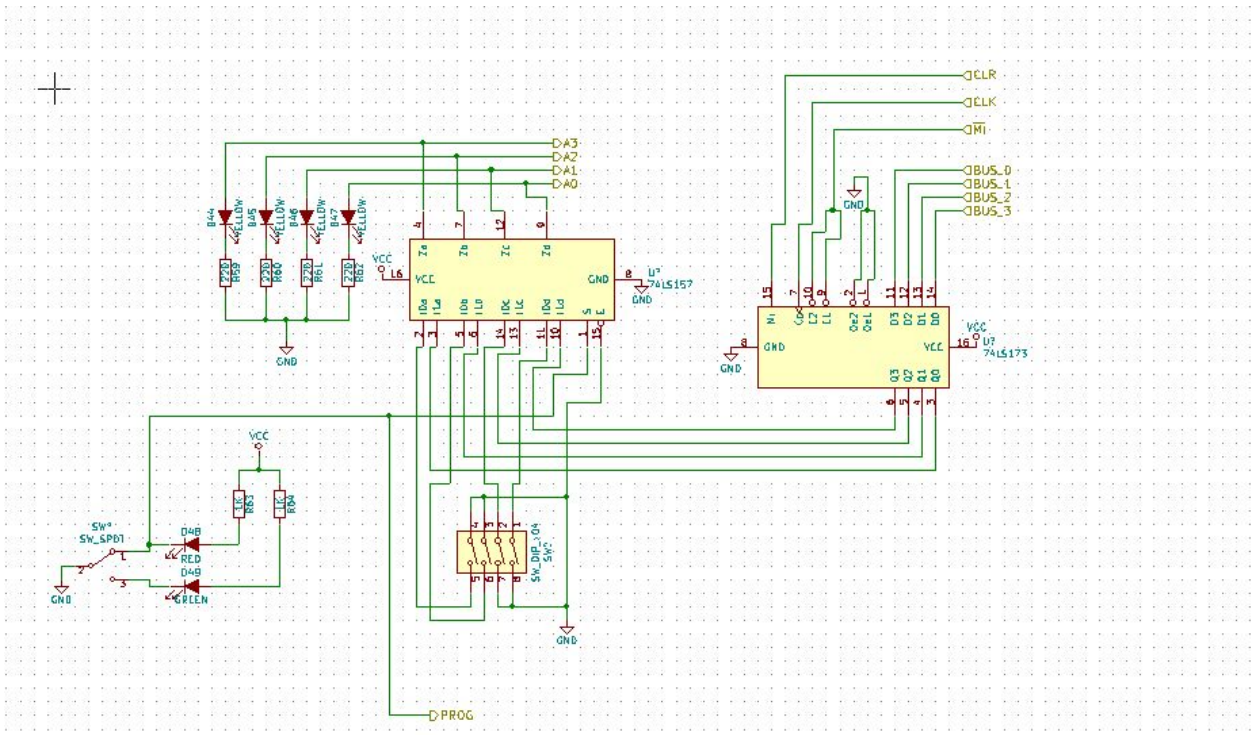
Arithmetic Logic Unit:



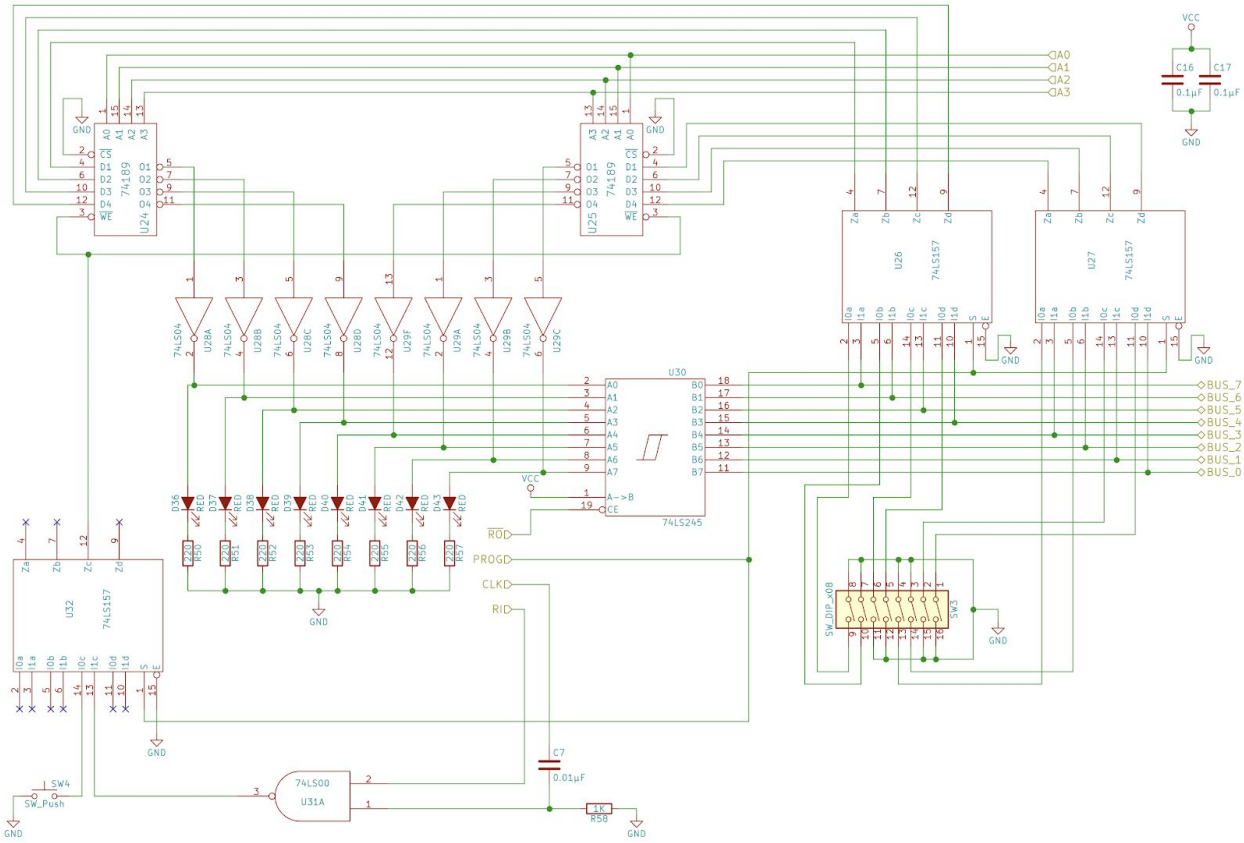
Control Logic:



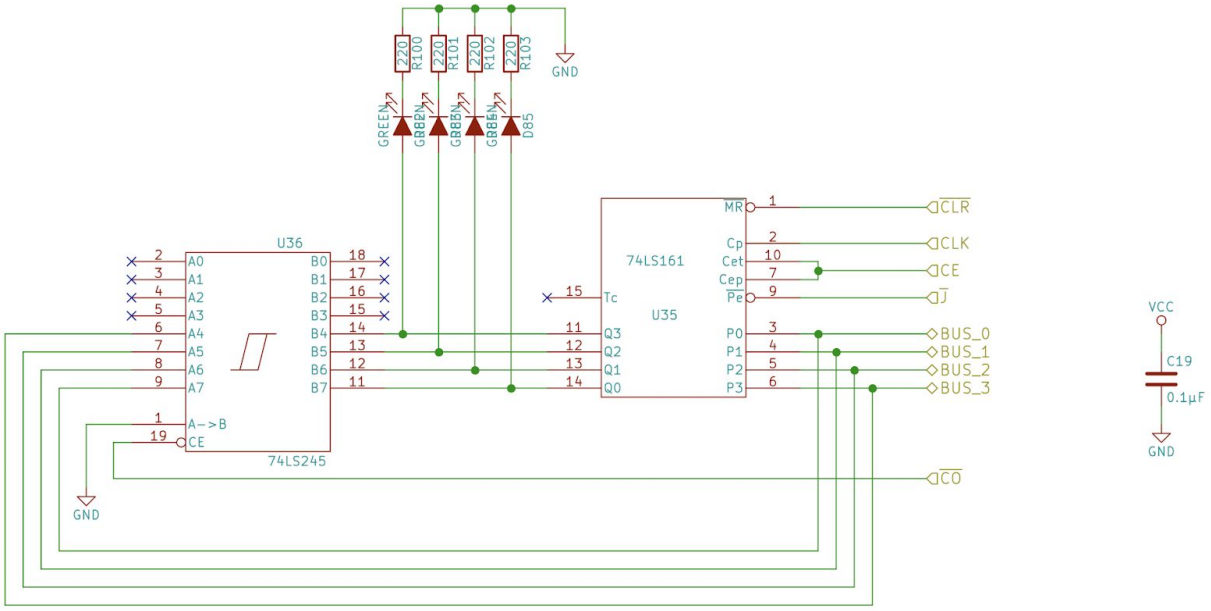
Memory Address Register:



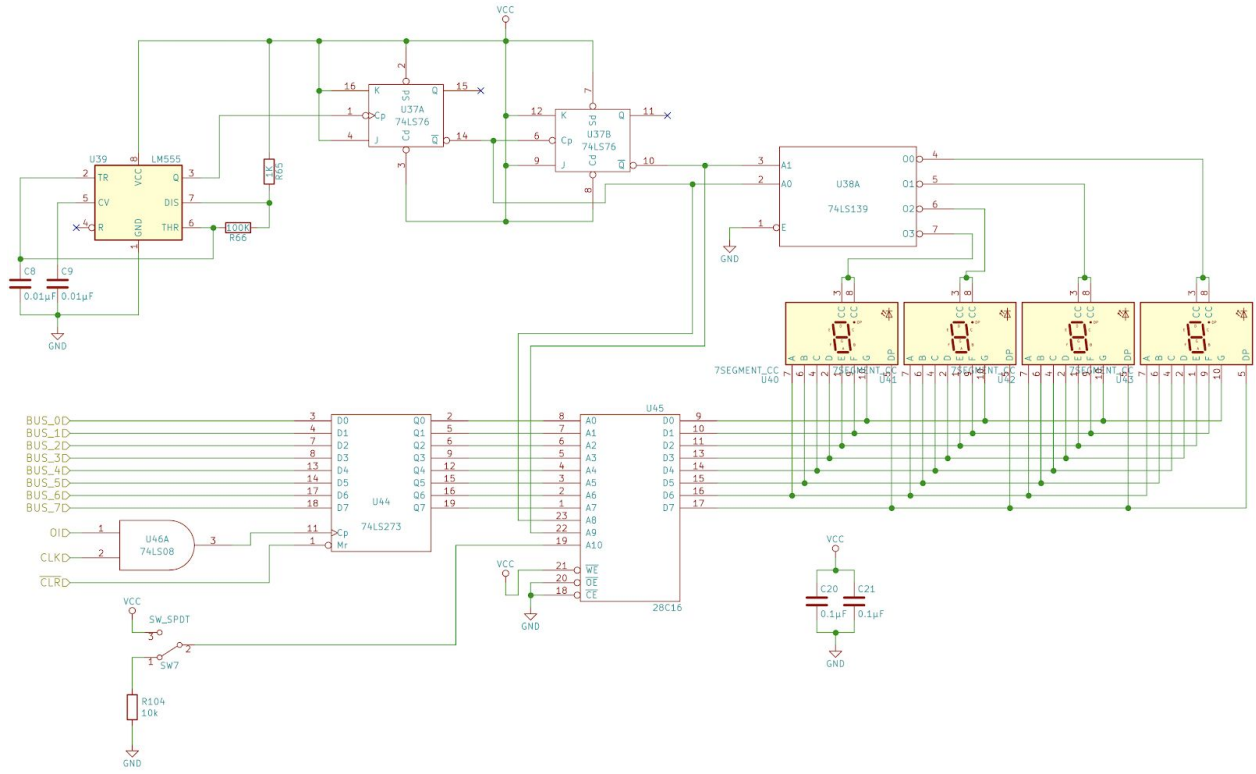
Random Access Memory:



Program counter:



Output Register:



Example Code:

This section is reserved for future use in describing in detail, the preloaded program.
This program is stored in the EEPROM and will be loaded upon providing power to the modules.