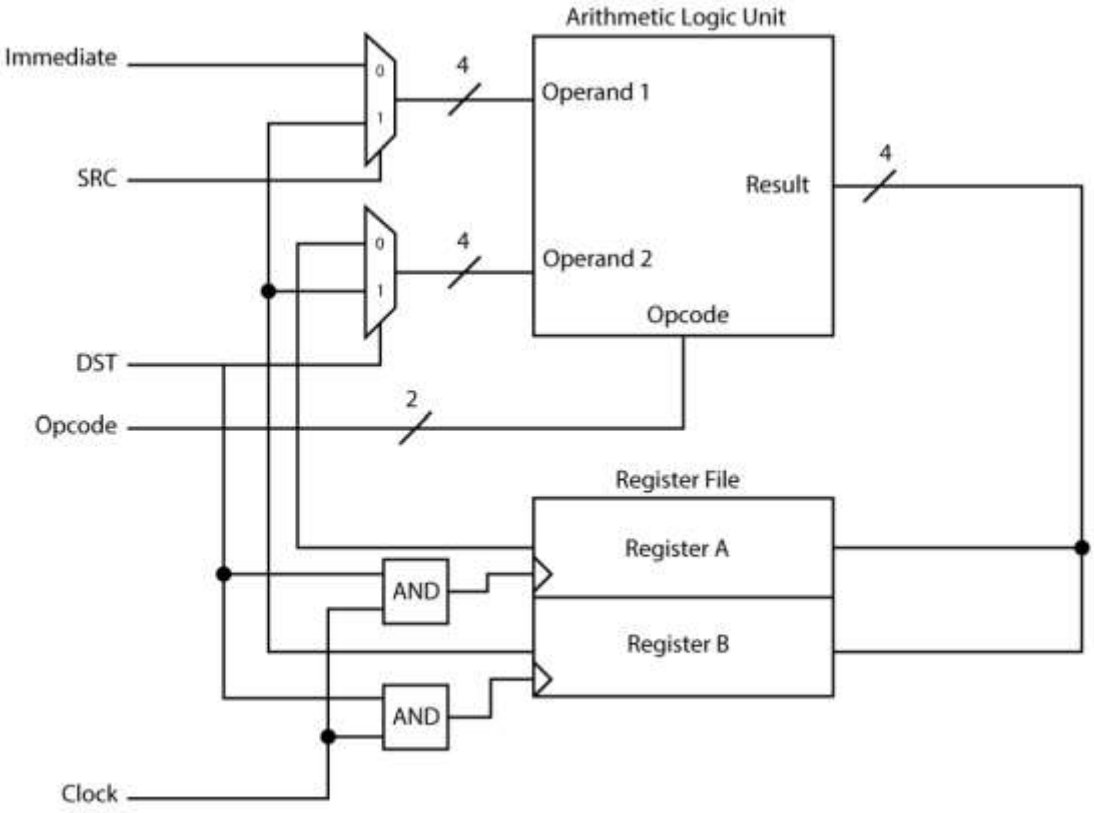
CPU Project

Group Members: Kaleb Badgett, Seth McDaniel, Ivan Ortiz, and Joel Zell

We created a 4 bit cpu with the objective of computing multiplication between two numbers (0-10). It had to be able to complete four different functions: Addition, Subtraction, Exclusive OR, and Move. This is the documentation of this project.

Below is the schematic:



*Note there should be an inverter before one of the and gates (between DST and the AND gate)

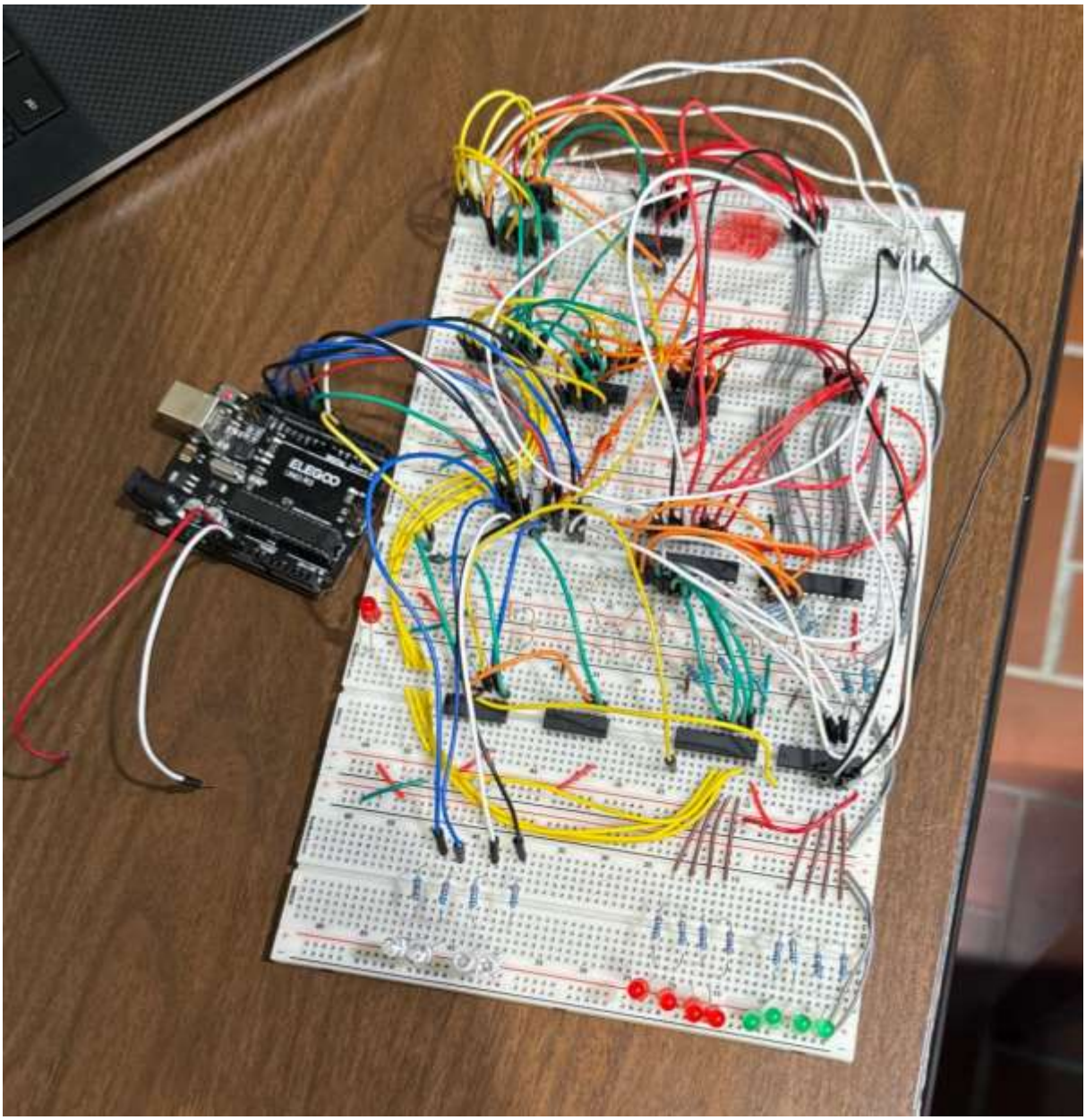
Two Muxes, 2 PLDs, 2 octal flip flop chips, 3 hex inverters, and one AND chip were used in the physical implementation of the project.

In this project, we created machine code using operation (op) code, the source bit, the destination bit, and the immediate value, in that order. This 4-bit processor took an 8-bit input and used that to create the desired outputs. The binary was then converted to hexadecimal to be used as machine code. To imitate a ROM to use our machine code, we used an Arduino.

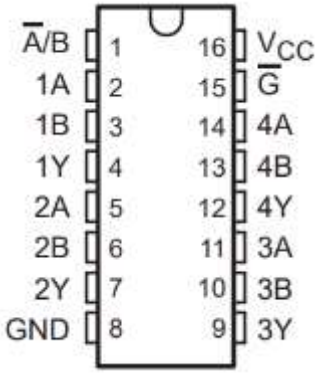
The op code is as follows:
00 ADD
01 SUB
10 XOR
11 MOV

For example, 1100 1101 became 0xCD. This meant that the value D (12) was to be stored in register A.

Below, you will find a photo of our final design.



Muxes:



What is high and low?

On the first mux that used selector SRC, 0 selected Immediate, and 1 selected register B.
On the second mux that used DST, 0 selected register A, and 1 selected register B.

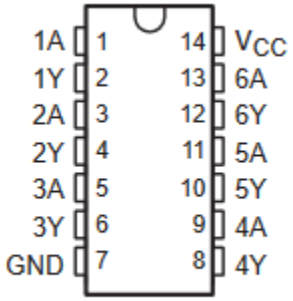
What pins are what bit?

Both Muxes:
Pin 1 (!A/B) was the selector on both muxes.
Pins 4, 7, 9, and 12 (1Y, 2Y, 3Y, and 4Y respectively) were the outputs of the muxes that led to the hex inverters.
Pin 8 (GND) was the ground pin for both muxes.
Pin 16 (VCC) was the power pin for both muxes.
Pin 15 (!G) was the enable signal that was connected to ground.

Mux 1:
Pins 2, 5, 11, and 14 (1A, 2A, 3A, and 4A respectively) were all for the immediate value.
Pins 3, 6, 10, and 13 (1B, 2B, 3B, and 4B respectively) were all for the values coming from register B.

Mux 2:
Pins 2, 5, 11, and 14 (1A, 2A, 3A, and 4A respectively) were all for the values coming from register A.
Pins 3, 6, 10, and 13 (1B, 2B, 3B, and 4B respectively) were all for the values coming from register B.

Hex Inverters:



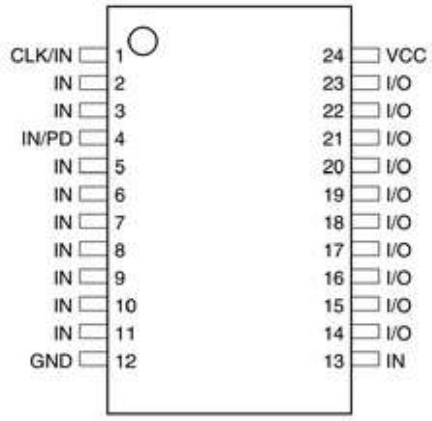
Both Inverters:
Pin 7 (GND) was the ground pin.
Pin 14 (VCC) was the power pin.

Inverter 1:
Pins 1, 3, 5, and 13 (1A, 2A, 3A, and 6A respectively) were the input pins that took the 4-bit values from mux 1 and inverted them in order to make those values usable later on.
Pins 2, 4, 6, and 12 (1Y, 2Y, 3Y, and 6Y respectively) were the outputs of the inverter that led to the ALU (PLDs).

Inverter 2:
Pins 1, 3, 5, and 13 (1A, 2A, 3A, and 6A respectively) were the input pins that took the 4-bit values from mux 2 and inverted them in order to make those values usable later on.
Pins 2, 4, 6, and 12 (1Y, 2Y, 3Y, and 6Y respectively) were the outputs of the inverter that led to the ALU (PLDs).

Inverter 3 (AND gate):
Pin 1 (1A) was the input from DST.
Pin 2 (1Y) was the output that led to the AND gate for later use.

PLDs (ALU):



Adder/Subtractor PLD:
Pin 12 (GND) is the ground pin.
Pin 24 (VCC) is the power pin.
Pins 1-10 were the input pins for the opcode (2 bits, pins 1 and 2), the mux 1 values (4 bits, pins 3-6), and the mux 2 values (4 bits, pins 7-10).
Pins 14-17 were the output pins (4 bits) that led to the second PLD.
Pins 18-21 were our pin nodes that acted as our carryout's for the addition.

C:\WINCUPL\WINCUPL\ADDERSUBALUPLD.PLD

```
Name     CPU DESIGN ;
PartNo   00 ;
Date     4/22/2024 ;
Revision 01 ;
Designer Engineer ;
Company  Texas Tech ;
Assembly None ;
Location  ;
Device   g22v10 ;

/* *************** INPUT PINS *********************/
PIN  1   =     op0                    ; /*                              */
PIN  2   =     op1                    ; /*                              */
PIN  3   =     a0                     ; /*                              */
PIN  4   =     a1                     ; /*                              */
PIN  5   =     a2                     ; /*                              */
PIN  6   =     a3                     ; /*                              */
PIN  7   =     b0                     ; /*                              */
PIN  8   =     b1                     ; /*                              */
PIN  9   =     b2                     ; /*                              */
PIN  10   =    b3                      ; /*                              */

/* *************** OUTPUT PINS *********************/
PIN  14   =     f0                    ; /*                              */
PIN  15   =     f1                     ; /*                              */
PIN  16   =     f2                     ; /*                              */
PIN  17   =     f3                     ; /*                              */

/* *************** PINNODES     *********************/
PINNODE   18  =   c0                   ; /*                              */
PINNODE   19  =   c1                    ; /*
PINNODE   20  =   c2                    ; /*
PINNODE   21  =   c3                    ; /*
```

```
/* Logic */
c0 = op0;
c1 = (a0 & (b0 $ op0)) # (a0 & c0) # ((b0 $ op0) & c0);
c2 = (a1 & (b1 $ op0)) # (a1 & c1) # ((b1 $ op0) & c1);
c3 = (a2 & (b2 $ op0)) # (a2 & c2) # ((b2 $ op0) & c2);


f0 = ((!op0 & !op1) & (a0 $ b0 $ c0)) # ((op0 & !op1) & (a0 $ (b0 $ op0) $
c0));
f1 = ((!op0 & !op1) & (a1 $ b1 $ c1)) # ((op0 & !op1) & (a1 $ (b1 $ op0) $
c1));
f2 = ((!op0 & !op1) & (a2 $ b2 $ c2)) # ((op0 & !op1) & (a2 $ (b2 $ op0) $
c2));
f3 = ((!op0 & !op1) & (a3 $ b3 $ c3)) # ((op0 & !op1) & (a3 $ (b3 $ op0) $
c3));
```

Waterfall PLD (XOR and MOV):
Pin 12 (GND) is the ground pin.
Pin 24 (VCC) is the power pin.
Pins 1-15 were the input pins for the opcode (2-bits, pins 1 and 2), the mux 1 values (4-bits, pins 3-6), the mux 2 values (4-bits, pins 7-10), and the Adder/Subtractor PLD values (4-bits, pins 11-15).
Pins 16-19 were the output pins (4-bits) that led to the registers.

C:\WINCUPL\WINCUPL\WATERFALL.PLD

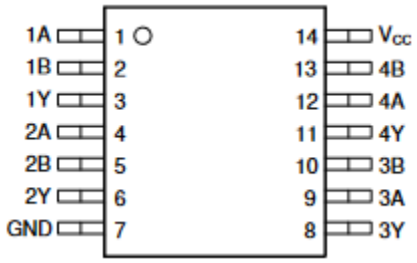
```
Name     WATERFALL ;
PartNo   00 ;
Date     4/24/2024 ;
Revision 01 ;
Designer Engineer ;
Company  Texas Tech ;
Assembly None ;
Location  ;
Device   g22v10 ;

/* *************** INPUT PINS *********************/
PIN  1   =   op0                      ; /*                              *
PIN  2   =   op1                      ; /*                              *
PIN  3   =   a0                      ; /*                              */
PIN  4   =   a1                      ; /*                              */
PIN  5   =   a2                      ; /*                              */
PIN  6   =   a3                      ; /*                              */
PIN  7   =   b0                      ; /*                              */
PIN  8   =   b1                      ; /*                              */
PIN  9   =   b2                      ; /*                              */
PIN  10  =   b3                      ; /*                              */
PIN  11  =   addsub0                      ; /*
PIN  13  =   addsub1                      ; /*
PIN  14  =   addsub2                      ; /*
PIN  15  =   addsub3                      ; /*

/* *************** OUTPUT PINS *********************/
PIN  16  =   f0                      ; /*                              */
PIN  17  =   f1                      ; /*                              */
PIN  18  =   f2                      ; /*                              */
PIN  19  =   f3                      ; /*                              */
```

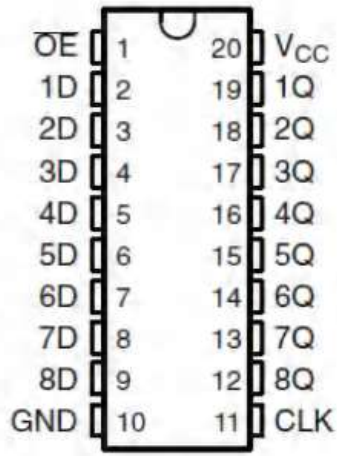
```
/****************** LOGIC ********************/
f0 =  ((op0 & op1) & a0) # ((!op0 & op1) & (a0 $ b0)) # ((!op1) & addsub0);
f1 =  ((op0 & op1) & a1) # ((!op0 & op1) & (a1 $ b1)) # ((!op1) & addsub1);
f2 =  ((op0 & op1) & a2) # ((!op0 & op1) & (a2 $ b2)) # ((!op1) & addsub2);
f3 =  ((op0 & op1) & a3) # ((!op0 & op1) & (a3 $ b3)) # ((!op1) & addsub3);
```

AND Gate:



Pin 7 (GND) is the ground pin.
Pin 14 (VCC) is the power pin.
Pins 1, 2, 4, and 5 (1A, 1B, 2A, and 2B respectively) were the inputs that took DST, !DST, and CLK.
Pins 3 and 6 (1Y, and 2Y respectively) were the output pins that led to the clock signals of the registers.

D Flip Flops:



Register A:
Pin 20 (VCC) was the power pin.
Pin 10 (GND) was the ground pin.
Pin 11 (CLK) was the clock signal.
Pin 1 (!OE) is an enable signal that was grounded.
Pins 2, 3, 4, and 5 (1D, 2D, 3D, and 4D respectively) were the inputs from the ALU that were to be stored as the register A values.
Pins 19, 18, 17, and 16 (1Q, 2Q, 3Q, and 4Q respectively) were the output pins that went to the green LEDs and back to the second mux.

Register B:
Pin 20 (VCC) was the power pin.
Pin 10 (GND) was the ground pin.
Pin 11 (CLK) was the clock signal.
Pin 1 (!OE) was an enable signal that was grounded.
Pins 2, 3, 4, and 5 (1D, 2D, 3D, and 4D respectively) were the inputs from the ALU that were to be stored as the register B values.
Pins 19, 18, 17, and 16 (1Q, 2Q, 3Q, and 4Q respectively) were the output pins that went to the red LEDs and back to both muxes.

Wire Coloring:

Color organization varied by section due to lack of wires of required length.

Arduino (ROM) Code:

Below is the code we used for a subtraction computation.

```
//My pin setup

int pins[]={6,7,8,9,10,11,12,13};// I/O pins

int clock = 5;

//9,7,8,6 --> immediate

//13 - > OP1

//12 - > OP2

//11 - > SRC

//10 - > DST

void setup(){

pinMode(clock,OUTPUT);

  for(int i=0; i<8; i++)

  {

    pinMode(pins[i], OUTPUT);

  }
```

```
}

void loop() {

 int program_counter = 0;

 byte rom[]= {

   0xC4,

   0xD5,

   0x43,

   0x54,

 };

 for (int i=0; i<4;i++)

 {

    for(int j=0; j<8; j++)

    {

      digitalWrite(pins[j], bitRead(rom[program_counter],j));

    }

    program_counter++;

    delay(1000);

    digitalWrite(clock,1);

    delay(1000);

    digitalWrite(clock,0);

    delay(1000);

 }

 delay(600000000);

}
```

Below is the code we used in our multiplication algorithm. We multiplied 5x9.

```
//My pin setup

int pins[]={6,7,8,9,10,11,12,13};// I/O pins

int clock = 5;

//9,7,8,6 --> immediate

//13 - > OP1

//12 - > OP2

//11 - > SRC

//10 - > DST

void setup(){

pinMode(clock,OUTPUT);

 for(int i=0; i<8; i++)

 {

   pinMode(pins[i], OUTPUT);

 }

}


void loop() {

 int program_counter = 0;

 byte rom[]= {

   0xC0,

   0xD0,

   0x15,
```

```
  0x00,
  0x10,
  0x00,
  0x18,
  0x00,
  0xE0,//remove this and reverse wires****this moves B to A
  0xD0,
  /*remove this and reverse wires****this moves the value of A "0" to the register,
  this value is stored in our heads because we can not switch the two in one clock cycle,
  the outputs from reg b to output a and reg a to output b(current output has a going to "a"
  and reg b going to "b")*/
  0x10,
  0x12,
};
for (int i=0; i<12;i++)
{
   for(int j=0; j<8; j++)
   {
     digitalWrite(pins[j], bitRead(rom[program_counter],j));
   }
   program_counter++;
   delay(1000);
   digitalWrite(clock,1);
   delay(1000);
```

```
    digitalWrite(clock,0);

    delay(1000);

}

delay(600000000);
```

Overall:

What skills were used?

We had to use technical skills, active listening, data analysis, problem solving, computer skills, teamwork, programming, communication, and organizational skills throughout the duration of this project.

What struggles did we run into?

We ran into several issues while completing this project.

1. Incorrect wiring.
   Upon attempting the first few trials using our CPU, three of our functions would not work due to the enable signal on our muxes not being grounded. We had to search the breadboard for issues and fix this.

2. Breadboard malfunction.
   We worked for a number of days on this project, but kept running into the same problem. Our results were not correct. We tried to troubleshoot many possible issues, but after several days, Seth McDaniel found the issue. One of the rails on one of the breadboards was out and would not transfer any sort of signal. Once Seth found this problem, he fixed it and we were able to resume work the following day.

3. PLD program error.
   After fixing the error with the breadboard, some of our functions still were not working properly. We found an error in the wincupl code used on one of the PLDs and fixed it.

4. Ensuring machine code was correct.
   One error we had was within the machine code we wrote for the arduino. After further investigation, we were able to fix this problem.

Did we have a group leader?

We did not have a designated group leader.

Did we need a group leader?

While having a designated group leader may have helped when it came to project management, we were able to complete all project requirements by ensuring each of us aided wherever and whenever needed.

What did we learn from the project?

Through this project, we learned about the inner workings of a CPU, a ROM, and an ALU. As we faced different issues we learned how to overcome those problems, which then gave us a deeper understanding of the parts we were working with.

What did we learn from working as a group?

While working as a group, we learned how to divide responsibilities to ensure we met the schedule and even completed the project a couple days early. We also learned how to overcome issues as a group rather than individually. This was done by ensuring multiple people were working on the project at once which, for the most part, allowed us to find issues quicker than we may have otherwise.