|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 🔑 **Essential** 🔑 | | | |  | 🏆 **Enhanced** 🏆 | | | |
|  |  |  |  |  |  |  |  |  |
| \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ |  | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ |

**Score: \_\_\_\_\_**

**09 – A Microprogrammed Machine**

**Activities**

COMP256 – Computing Abstractions

Dickinson College

Spring 2022

Prof. Grant Braught

**Name:**

**Introduction:**

Today’s class introduced the Machine Abstractions topic including several levels of abstraction at which we will view the machine itself (Physical Machine, Microprogram Machine, Machine Language Machine). The Knob & Switch computer, a visualization of a simple computer that allows you to experience these levels in a hands-on way was also described. Today’s activities start by having you work with the Knob & Switch computer at the Physical Machine and Microprogram Machine levels of abstraction. They then give you the chance to take a deeper look at the history of a few of the important early physical machines that we’ve heard of before (Colossus and ENIAC).

**The Knob & Switch Datapath:**

A datapath is the part of the computer’s central processing unit that performs computations. The *Register Bank* contains registers that provide a small amount of temporary storage for the data on which the computations will be performed. The *Arithmetic and Logic Unit (ALU)* contains the circuitry that performs the computations (e.g. addition and subtraction). The data moves from the Register Bank to the ALU and back on *Data Busses*. A Data Bus is essentially a collection of wires, one for each bit, that move the data from place to place.

The Knob & Switch (K&S) computer provides a simulation of the *datapath* of small manually programmable machine. It has been designed to give you some insight into the way that modern computers perform computation. The calculations performed by the K&S Datapath are configured by setting knobs in the datapath to choose the registers containing the data to be used, the operation to be performed and the register into which the result will be stored.

You can open the K&S datapath simulation using the link below and use it to answer the following basic questions.

* <https://dickinson-comp256.github.io/Knob-And-Switch-Computer/datapath.html>

🔑 1. Consider the configuration of the K&S datapath shown below.

![A screenshot of a cell phone

Description automatically generated]()

Describe in a sentence or two what computation the K&S would perform in this configuration. Be sure to indicate what the inputs are, where they come from, what operation is performed, what the output will be and where it will be stored.

🔑 2. Put the values 4321 and -1234 into registers and configure the K&S datapath to add them together and store the result in a different location than the inputs. Give a screen shot of the K&S datapath after it has executed this operation.

🔑 3. Change the configuration of the K&S datapath from question #2 so that the result of the ALU operation is stored into the same register as one of the inputs. What happens?

**ALU Flags:**

You may have noticed that there are several check boxes in the ALU box (Zero, Negative, Unsigned Overflow and Signed Overflow). These are *flags* that indicate properties of the result produced by the ALU when it performs an operation. For example, the Zero flag will be set (checked) if the result of the ALU operation is zero. The Negative flag is set if the ALU result is negative. The Unsigned Overflow flag is set if the operation produces overflow in unsigned binary. The Signed Overflow flag is set if the operation produces overflow in Two’s Complement.

🔑 4. For this question, configure your K&S datapath as shown below.

Graphical user interface, diagram

Description automatically generated

a. Give values for R0 and R1 that would cause the zero flag to be set. Be sure to test your answer by executing the datapath. It is okay if other flags are also set as long as the zero flag is set.

b. Give values for R0 and R1 that would cause the negative flag to be set. Be sure to test your answer by executing the datapath. It is okay if other flags are also set as long as the negative flag is set.

🏆🏆 c. Give values for R0 and R1 that would cause the Signed Overflow flag to be set but will not cause the Unsigned Overflow flag to be set. Hint: Thinking in two’s complement binary is the best way to solve this problem. Show your values in both base 10 and binary and explain in a few sentences why they set the Signed Overflow flag and do not set the Unsigned Overflow flag. Be sure to test your answer by executing the datapath.

**Bitwise Operations:**

You will have noticed that the K&S ALU can perform four different operations (+, |, & and -). The + and - perform addition and subtraction of the binary values as we have learned about in class. In addition to addition and subtraction computer ALUs usually also perform a range of *bitwise operations*. In a bitwise operation, the bits in the corresponding locations of the inputs are operated on in pairs. For example, consider the 4-bit examples of the bitwise OR (|) and bitwise AND (&) operations shown below:

Bitwise OR (|) Bitwise AND (&)

1011 A 1011 A

| 0010 B & 0010 B

1011 A|B 0010 A&B

Notice that each bit of the result is obtained by applying the operation (OR or AND) to the corresponding bits of A and B. For example, the yellow highlighted bit of the result in the bitwise OR example is a 1 because the two bits above it are 1 (A) and 0 (B) and 1 OR 0 is 1. Similarly, the yellow highlighted bit of the result in the bitwise AND example is a 0 because the two bits above it are 1 (A) and 0 (B) and the 1 AND 0 is 0. The other 3 bits of each result are computed in the same way, using the bits directly above them.

🔑 5. This question gives you a little practice with these bitwise operators.

a. Give the result of the following 8-bit bitwise operations (you can use the K&S in Base 2 mode to check your answer, just enter leading 0’s to get to 16 bits):

1010 1101 0101 0110

& 1100 1001 | 1100 0101

b. What result do you obtain if you perform a bitwise & or a bitwise | operation if A and B are the same number? Hint: Try it! Set the datapath to show the values in binary (Number Base: 2), enter a value in to one of the registers and it to both the A and B bus.

**Knob & Switch with Main Memory:**

The K&S with Main Memory adds a main memory (i.e. a RAM) to the machine and also adds some switches to the datapath. The main memory provides a larger storage area that can hold more data, and as we will eventually see program instructions. The new switches have been added to make it possible to move data between the main memory and the registers.

You can open the K&S with Main Memory using the link below.

* <https://dickinson-comp256.github.io/Knob-And-Switch-Computer/dpandmem.html>

🔑 6. Enter a value into main memory location 510 and then configure the K&S datapath so that the value you entered will be transferred into register R2. In doing so, be sure that you have the switches set so that there is only one value incoming to the C bus. Include a screen shot of the main memory and data path as your answer for this question.

🔑 7. Enter a value into register R3 and configure the datapath so that the value you entered is transferred to main memory address 710.

* Do not assume that any register contains the value 0.
* The execution should store the value into memory but may not change the value in any other registers.

Hint: Find a way to move the value in R3 through the ALU without it being changed. At first it may appear that there is not a way to do this. But there is, and you found it in an earlier question!

Include a screen shot of the main memory and data path as your answer for this question.

**Memory Addresses:**

🔑 8. Every location in the main memory has a *memory address*, given by the number beside the cell. The K&S will show us these addresses in base 10. But we know that they, like everything else in the computer, must be expressed in binary. Because all of the addresses are positive, we use unsigned binary values for these addresses.

a. What is the address of the first (top most) memory location? Give your answer in both unsigned binary and base 10.

b. What is the address of the last (bottom most) memory location? Give your answer in both unsigned binary and base 10.

c. How many total memory locations are there in the K&S? Express this both as an integer and as a power of two (e.g. 2x where you give x a value).

d. The K&S uses 5 bits to represent its memory addresses. Why does it use that number of bits?

e. If the designer of the K&S had wanted to have 64 different memory locations, how many bits would have been needed for the memory addresses?

f. If the K&S had been designed to use 8 bits for its memory addresses, how many memory locations could there be?

**K&S Microinstructions:**

The Microprogrammable K&S adds the ability to program the machine using microinstructions. Each *microinstruction* is a pattern of 1’s and 0’s that represent the configuration of the machine’s knobs and switches. Stated another way, a micro-instruction is an abstract representation of a particular configuration of the machine. You need to know what the individual bits mean to write the microinstruction, but you do not need to know how those 1’s and 0’s actually do the configuration. These exercises introduce you to the writing microinstructions for the K&S.

You can open the Microprogrammable K&S using the link below.

* <https://dickinson-comp256.github.io/Knob-And-Switch-Computer/micromachine.html>

Each microinstruction for the K&S has six fields (e.g. A Addr, B Add, ALU op, etc) as shown below:

A screenshot of a cell phone

Description automatically generated

Each field will contain a pattern of bits (1’s and 0’s) that specify the configuration of the corresponding element of the K&S data path. More specifically:

* Each Knob (A Addr, B Addr, ALU Op, C Addr) has 4 positions. So two bits will be used to represent its position as Up (00), Right (01), Down (10) and Left (11). For example, the B Addr field for the knob shown below would be encoded as 10.

A picture containing drawing

Description automatically generated

* The Switch Pos field uses 4-bits to indicate the position of each of the four switches in the data path as Open (0) or Closed (1). The switches are represented in counterclockwise order starting at the top and going counterclockwise. For example, the switch positions below would be encoded into a micro instruction as 1010.



* The RW Addr field is 5 bits and gives the unsigned binary value of the memory address of the location that will be involved in the instruction. For example, if memory address 1210 is being used then the RW Addr field will be 011002.

🔑 9. Consider the micro instruction shown below:

A screenshot of a cell phone

Description automatically generated

Using the screen short of the K&S shown below, draw in the knob and switch positions and indicate the memory location corresponding to the above micro instruction.

Graphical user interface, application

Description automatically generated

🔑 10. Consider the K&S machine configuration shown below:

A screenshot of a cell phone

Description automatically generated

Fill in the table below with the bits of the micro-instruction that will produce the machine configuration shown above.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  | **A Addr** | **B Addr** | **ALU Op** | **Switch Pos** | **C Addr** | **RW Addr** |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

**K&S Microprograms:**

As you can see there is room for you to enter five microinstructions into this version of the K&S. The machine will execute these micro-instructions one after the other from top to bottom. Thus, you can write programs for the K&S to perform multiple step calculations using microinstructions. Programs written in microinstructions are called *microprograms*.

🔑 11. Write a microprogram for the K&S that will perform the following computation:

* Add the value in main memory location 5 to the value in main memory location 20 and store the result in main memory location 16. Hint: You’ll need to move these values into registers, compute the result and then move the result back to main memory.

Be sure to enter values into the K&S memory at locations 5 and 20 and then run your program to test that it works. Place a screen shot of the microprogram memory containing your microprogram in the box below as your answer to this question.

**Let’s Add a Little History for Perspective:**

Many of the world’s most important inventions are an almost inevitable product of the discoveries and problems of their times. For each of these there were many different people in many different places all working at solving similar problems using similar ideas. Thus, it can be hard to say who was the inventor of X (e.g. steam engine, automobile, lightbulb, etc.) or which was the first X. Computers are no different.

🏆 12. Using your favorite search engine identify at least four candidates for the world’s first electronic computer. For each machine identify the year it was created, the primary inventors and the location where it was constructed.

🏆 13. Watch the video: *1946 ENIAC Computer History Remastered* from the Computer History Archives Project and answer the questions below:

* <https://www.youtube.com/watch?v=bGk9W65vXNA> (9:37)

a. What does ENIAC stand for?

b. How was the ENIAC programmed?

c. What was the original problem that ENIAC was designed to solve?Top of Form

d. What were some of the media that were used for input and output with the ENIAC?

e. What were a few of the later computers designed by ENIACs engineers?

f. What were a few of the commercial computing companies that were created by those that created the ENIAC?

Though, not exactly the same, working with the first version of the K&S that you used will have given you a little bit of a feel for how machines like the ENIAC were programmed. In the ENIAC, there would have been many simple computational units similar to the K&S datapath. Each one would have been configured to perform some operation. These units would have then been connected together to perform a sequence of operations (i.e. a program).

🏆 14. Watch the video *Bottom of FormColossus: The World's First Electronic Computer* from The Open University and answer the questions below:

* <https://www.youtube.com/watch?v=EdxBO9jfU8k> (5:32)

a. What was the problem that the Colossus was designed to solve?

b. How is the Colossus programmed?

c. What types of media were used for input and output for Colossus?

🏆🏆 15. Explain how the ALU in the Knob and Switch computer is related to the circuits that you built in Lab #3.

Optional: To help me improve and scope these activities for future semesters please consider providing the following feedback.

a. Approximately how much time did you spend on this activity outside of class time?

b. Please comment on any particular challenges you faced in completing this activity.