

CMPS 320 Task 6: Disk Controller

Assigned 22 February; pretask due 27 February; actual due 12 March
55 points, plus 20 extra credit

Description

This task considers the process of reading external memory on a hard drive. The previous tasks considered solid-state internal memory organizations with single pieces of data or data in contiguous blocks. Here the organization is based on the physical parameters of head, track, and sector to locate a cluster on a spinning disk, and then to read the bytes within the cluster. Clusters can be located anywhere on the disk and chained to define a file. Accessing a file involves following the chain from the specified start to its defined end and reading the bytes along the way. Think back to linked lists from CMPS 350.

Specification

Create one circuit in LogicDog 2.0 with your user name as the filename.

Be sure to look at the entire document and address Part 3 before you start and after you are done. There is a pretask.

Part 1: Disk Implementation

Description

The disk organization is as follows:

- four read/write heads; i.e., two per disk platter, one per side.
- eight tracks per platter.
- eight sectors per track.
- eight bytes per sector; i.e., the cluster size.

The disk structure is represented in a ROM chip, where the descriptor is the address for accessing any byte on the disk:

bit	10	9	8	7	6	5	4	3	2	1	0
meaning	sector			track		head		offset			

For example, 011.010.01.100 accesses byte 4 of sector 3 of track 2 from head (platter) 1. The corresponding address in ROM is \$34C, which derives from this breakout: 011.0100.1100. The dots in the first breakout refer to the table boundaries above, and in the second to hex nibbles; i.e., 011 is \$3, 0100 is \$4, and 1100 is \$C. The offset is the byte index into the cluster.

All clusters contain eight bytes. The first seven are the actual data; the eighth is either (1) a descriptor that specifies the location of the next cluster in the chain, or (2) the sentinel value \$FF that indicates there is no next cluster. (There is therefore no sector 7, track 7, head 3 on the disk.) In practice, the last cluster would ordinarily have a variable size because files are rarely evenly divisible by the cluster size, but for our purposes, the operating system would deal with ignoring the unused bytes. Note how this structure is analogous to a linked list: a start node (head), subsequent nodes chained with pointers, and a final node (tail) pointing to null.

The disk contents in `cmps320-task-6-disk.dat` are organized as follows. Each row is a cluster. The leftmost two columns are its respective decimal and hex address in ROM (which are not in the file), and the right columns are the eight bytes of data in it starting from that address. The rightmost column is the descriptor to the next cluster. Most of the disk is unused and filled with \$00. All actual data values are ASCII characters. Only the first part of the file is shown here.

Address	Data								Next
0 \$000	41	42	43	44	45	46	47	FF	
8 \$008	00	00	00	00	00	00	00	00	
16 \$010	41	42	43	44	45	46	47	07	
24 \$018	00	00	00	00	00	00	00	00	
32 \$020	4F	50	51	52	53	54	55	0B	
40 \$028	00	00	00	00	00	00	00	00	
48 \$030	00	00	00	00	00	00	00	00	
56 \$038	48	49	4A	4B	4C	4D	4E	04	
64 \$040	00	00	00	00	00	00	00	00	
72 \$048	00	00	00	00	00	00	00	00	
80 \$050	00	00	00	00	00	00	00	00	
88 \$058	56	57	58	59	5A	5B	5C	FF	

Definition

Input

- a two-bit input pin called `head`.
- a three-bit input pin called `track`.
- a three-bit input pin called `sector`.
- a one-bit input pin called `load`.
- a button called `clock`.

Output

- four hex-digit displays labeled *Current Address* as a group, which display the current head, track, sector, and byte offset values. Label each with its respective name.
- three hex-digit displays labeled *Next Address* as a group, which display the next head, track, and sector in the chain. Label each with its respective name.
- a green LED called `done`.
- a blue LED called `advance`.
- a TTY terminal with one row and 32 columns. (This configuration has no particular significance.)

Memory

- a 2048×8-bit ROM device, which is loaded with the contents of `cmps320-task-6-disk.dat`.

Behavior

Enter the descriptor in terms of head, track, and sector, then enable `load` to move there and click `clock` to read its first byte. Disable `load`. The next six clock ticks (probably 12 mouse clicks) will read the next six bytes and display them to the terminal in order. The seventh clock tick will read the next descriptor and illuminate `advance`. If it is `$FF`, the process ends, `done` illuminates, and subsequent ticks will have no effect; otherwise, the drive will move to the location the descriptor specifies, and this process will repeat for the next six clocks ticks again. *This second step is extra credit.*

Here are some examples:

<u>Example</u>	<u>Cluster 1</u>	<u>Cluster 2</u>	<u>Cluster 3</u>	<u>Cluster 4</u>	<u>Output</u>
1	S0 T0 H0 (\$00)				ABCDEFGH
2	S0 T1 H0 (\$04)	S0 T2 H3 (\$0B)			OPQRSTUVWXYZ[\
3	S0 T0 H2 (\$02)	S0 T1 H3 (\$07)	S0 T1 H0 (\$04)	S0 T2 H3 (\$0B)	ABCDEFGH IJKLMNOPQRSTUVWXYZ[\
4	S0 T4 H0 (\$10)	S2 T0 H0 (\$40)	S2 T5 H0 (\$54)	S2 T5 H3 (\$57)	Shelby_loves_the_bark_park!!

Part 2: Meta-Analysis

Answer the following questions in a plain text file called `readme.txt`. Questions 1 and 2 are the pretask.

1. After reading this entire document, but before starting on it, how long do you think this task will take?
2. Describe your intended approach in sufficient detail for me to evaluate its likelihood of success.
3. How long did it actually take? Keep track and be honest.
4. Explain any substantial discrepancy between these two times.
5. What score (in points) do you expect to get? If less than perfect, explain why and what you will do better next time.
6. What were the easiest and hardest parts to understand?
7. What were the easiest and hardest parts to solve?
8. What did you learn from this task? Do not repeat its description.

You will need to work with me on this task. Come to class prepared with questions. Stop assuming things!!

Grading

Part 1 (required) is worth 50 points. The extra credit is worth 20 points distributed as 0, 10 (reasonable, but not entirely correct), or 20 points (entirely correct). Part 2 is worth 5 points.