LAB 1H - Pre-Lab

Read before your scheduled lab time.

Our first hardware lab is designed to introduce the LD-14, a machine that is an example of simple computer architecture. This machine is a copy of Digital Equipment Corporation’s PDP-8. The second lab will examine “fundamental conditions” of logic circuits using a logic probe. The third lab will study “load signals”, and the fourth hardware lab will be an exercise in locating and replacing bad chips on the LD-14 circuit board.

The purpose of this pre-lab description is familiarize ourselves with the LD-14 control panel and some of the basic PDP-8 commands that can be used with the machine.

Familiarity with the LD-14.

Part of the purpose of these labs is to acquaint you with the actual hardware of a machine (the LD-14) which is a straight-forward implementation of the fetch-execute cycle (i.e. Von Neumann architecture). In the Verilog labs you will also see a simulation of the same machine.

The Front Panel

For those of you who took COSC 2150 here at UW, much of the terminology used in all the labs will be vaguely familiar. Using the LD-14, however, is vastly different from running the C or Pascal simulation programs of the PDP-8 you developed in 2150.

Figure 1.1 shows the layout of the LD-14. Placed horizontally across the panel is a row of toggle switches and above those switches are six rows of twelve red lights representing bits of a binary word.

The toggle switches and the red lights are labeled at the top from left to right as 0 to 11. (Big Endian Ordering). There are two common ways to label the bits of a word known as big endian and little endian (figure 1.2). Big endian progresses left to right starting at zero and little endian progresses right to left starting at zero.

![Figure 1.2 - Ordering of Bits (12 bit word)]

<table>
<thead>
<tr>
<th>Big Endian Ordering</th>
<th>Little Endian Ordering</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 10 11</td>
<td>11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

Each row of twelve lights (labeled on the right) represent the AC, IR, PC, MB, M, and MA register displays, respectively. A short description of each of these 12 bit registers follows.

pre-lab 1-1
AC (Accumulator) - reflects the binary result of an operation.
IR (Instruction Register) - reflects the current instruction being executed.
PC (Program Counter) - reflects the memory address containing the current instruction.
MB (Memory Buffer) - reflects binary values of instructions or data being transferred between memory and the AC, PC, or IR registers.
M (Memory) - reflects the binary content of the current memory address (MA).
MA (Memory Address) - reflects the current memory address.

To the right of each of these register lights is push button used to load the contents of each register. Each row contains four additional lights on the left which will be covered in more detail as the labs are implemented. One of these lights is the LINK, located to the immediate left of the AC. The LINK acts as the 13th bit of the accumulator. On the far left of each row are six more push buttons that will be explained as we use them in each lab. At the top of the control panel is a switch and knob used to control the speed at which the LD-14 runs. There is also a single instruction switch located on the left of the machine. The positive and negative posts for the logic probe are located on either side of the clock speed switch. In the first lab you will learn how to control the speed of machine.

The bottom row of twelve toggle switches underneath the lights is called the Switch Register (SR). Unlike the other six registers represented, the SR is not a display. These toggle switches used to enter binary values into the registers and into the memory of the machine.

The circuit board is mounted vertically on the back of the machine. Each row of chips are labeled horizontally from right to left (A to M) and vertically from bottom to top (I to 2I). The chips from A5 to D16 are the memory chips of the LD-14. This circuit board contains the entire functional entity of the fetch-execute cycle used by the PDP-8. The input is controlled by the switch register and the DEP push button located on the bottom of the left-hand side. The rest of the push buttons located on the left are used to control the actual functions (run-time, start, stop, and examine) of the machine. The output is represented by the lights on the front panel (just as the traceout of the PDP-8 simulator in COSC 2150 represented output).

Basic PDP-8 Commands

The commands listed below are the Memory Reference instructions and the Non-Memory Reference instructions of the PDP-8 that we will use in the labs. Those of you who took COSC 2150 here at UW should be familiar with these commands and can use this list as a review. Note that the “xxx$” in the MRI instructions indicates an arbitrary memory address. The bits referred to below are given in little endian notation.

Memory Reference Instructions

1. TAD (1xxx$) - Two's complement ADd contents of memory address xxx to the LINK and AC.

2. DCA (3xxx$) - Deposit contents of AC at memory address xxx$ and then Clear AC.
3. **AND (0xxx₈)** - Logical AND of AC with contents of memory address xxx₈.

4. **JMP (5xxx₈)** - Jump to memory address xxx₈ so that the fetch/execute cycle will process the instruction stored there instead of the next sequential instruction. The PC is simply loaded with xxx₈.

5. **ISZ (2xxx₈)** - Increment (add 0001₈) to contents of memory address xxx₈ and Skip next instruction if contents become Zero.

6. **JMS (4xxx₈)** - Jump to Subroutine located at memory address xxx₈. The JMS instruction saves the return address at memory address xxx₈ and then the PC becomes xxx₈ + 0001₈. (The return address is the value of the PC indicating which instruction would have otherwise executed next.)

The "xxx₈" in the MRI instructions indicates a memory address used by the instruction. It is important to understand the addressing modes of the PDP-8 and how they work. There are four addressing modes of the PDP-8: direct page zero, indirect page zero, direct current page, and indirect current page. There is also a variation of the indirect addressing mode known as auto-increment.

Why do we need other addressing modes? One reason lies in the number of addresses we can represent using the page addressing bits. The page addressing bits are bits 6-0 of the IR, as shown in figure 1.3.

![Figure 1.3 - Page Addressing Bits and Address Mode Bits](image_url)

Only $2^7$ or $128_{10}$ addresses (starting at address $0_{10}$) can be represented by these seven bits. To represent the other $3968_{10}$ memory locations possible with the 12 bit address bus, the PDP-8 subdivides the $4096_{10}$ memory locations into $32_{10}$ pages (starting at page zero) of $128_{10}$ memory locations ($4096_{10}$ DIV $128_{10} = 32_{10}$). To access a particular page, the PDP-8 uses two types of addressing modes: direct and indirect. Bit eight indicates either direct or indirect addressing mode and bit seven indicates either page zero or current page. Page zero is normally used for global variables and constants and the current page ($01_{8}$-$37_{8}$) is normally used for local data and corresponding code. Table 1.1 lists the combinations of bits seven and eight for each possible addressing mode.

<table>
<thead>
<tr>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Addressing Mode</th>
<th>Effective Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Direct Page Zero</td>
<td>ir[6:0]</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Direct Current Page</td>
<td>{pc[11:7],ir[6:0]}</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Indirect Page Zero</td>
<td>m[ir[6:0]]</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Indirect Current Page</td>
<td>m[pc[11:7],ir[6:0]]</td>
</tr>
</tbody>
</table>

pre-lab 1-3
Direct page zero computes EA as simply the low order seven bits of the instruction register. This is the only addressing mode used in Appendix A. Direct current page computes EA as the high order five bits of the program counter concatenated to the low order seven bits of the instruction register. This is useful for programs that do not fit in the 128 words of page zero. Indirect page zero computes EA as the contents of memory pointed to by the low order seven bits of the instruction register. Similarly, indirect current page computes EA as the contents of memory pointed to by the concatenation of the high order five bits of the program counter and low order seven bits of the instruction register. These indirect addressing modes are useful when the address of data varies during runtime, and also in conjunction with the JMP instruction to return from a subroutine (called by a JMS instruction) or from an interrupt service routine.

The indirect addressing modes are slower, but more powerful, than the direct addressing modes since the EA comes from memory. First, the machine obtains the address of the EA from the instruction register (and possibly the program counter). Next, it accesses memory to obtain the EA. Finally, it accesses memory to obtain the data.

Autoincrement occurs on the PDP-8 with indirect addressing when the address of the EA (not the EA itself) is between 0010<sub>8</sub> and 0017<sub>8</sub>. In these eight cases, the EA in memory is incremented prior to execution of the instruction. For example, the instruction 1017<sub>8</sub> increments the word at m[0017<sub>8</sub>], and then adds m[m[0017<sub>8</sub>]] to the accumulator.

Non-Memory Reference Instructions

Group 1 Microinstructions

1. **CLA (7200<sub>8</sub>) - Clear the Accumulator, bit 7 on.** This instruction sets the AC to 0000<sub>8</sub>.

2. **CLL (7100<sub>8</sub>) - Clear the Link, bit 6 on.** This instruction sets the LINK to 0.

3. **CMA (7040<sub>8</sub>) - Complement the Accumulator, bit 5 on.** This instruction complements (sets all 1's to 0's and 0's to 1's) the AC.

4. **CML (7020<sub>8</sub>) - Complement the Link, bit 4 on.** This instruction complements the LINK.

5. **RAR (7010<sub>8</sub>) - Rotate the Accumulator and LINK Right, bit 3 on.** This instruction shifts bit 11 through bit 0 one position to the right. The LINK shifts to bit 11 and bit 0 shifts to the LINK. All other bits shift one position to the right.

6. **RTR - (7012<sub>8</sub>) - Rotate the Accumulator and Link twice Right, bit 3 and 1 on.** Bit 0 shifts to bit 11, the LINK shifts to bit 10 and bit 1 shifts to the LINK. All other bits shift two positions to the right.

pre-lab 1-4
7. **RAL (70048)** - Rotate the Accumulator and Link **Left**, bit 2 on. This instruction shifts bit 10 through 0 one position to the left. The LINK shifts to bit 0 and bit 11 shifts to the LINK. All other bits shift one position to the left.

8. **RTL (70068)** - Rotate the Accumulator and Link **twice Left**, bit 2 and 1 on. Bit 11 shifts to bit 0, the LINK shifts to bit 1 and bit 10 shifts to the LINK. All other bits shift two positions to the left.

9. **IAC (70018)** - Increment the Accumulator, bit 0 on. Adds 1 to the contents of the AC. If the AC is 77778, the LINK will be complemented (as in the CML instruction). This allows the LINK and AC to act together as a 13 bit counter register.

10. **NOP (70008)** - No **OPeration**, bits 0-7 off.

The Group 1 Microinstructions can be combined together. For example CLA CLL is 73008.

**Group 2 Microinstructions**

1. **SMA (75008)** - Skip on **Minus Accumulator**, bit 6 is 1₂ and bit 3 is 0₂. Normally used with signed data. Skips the next instruction if the value in the AC is negative.

2. **SPA (75108)** - Skip on **Positive Accumulator**, bit 6 is 1₂ and bit 3 is 1₂. Normally used with signed data. Skips the next instruction if the value in the AC is positive.

3. **SZA (74408)** - Skip on **Zero Accumulator**, bit 5 is 1₂ and bit 3 is 0₂. Skips the next instruction if the value in the AC is equal to zero.

4. **SNA (74508)** - Skip on **Non-zero Accumulator**, bit 5 is 1₈ and bit 3 is 1₂. Skips the next instruction if the value in the AC is not equal to zero.

5. **SZL (74308)** - Skip on **Zero Link**, bit 4 is 1₂ and bit 3 is 0₂. Skips the next instruction if the LINK is 0₂.

6. **SNL (74208)** - Skip on **Non-zero Link**, bit 4 is 1₂ and bit 3 is 1₂. Skips the next instruction if the LINK is not equal to zero.

7. **SKP (74108)** - **Skip** unconditionally, bit 3 is 1₂. Skips the next instruction.

8. **HLT (74028)** - **Halts** the computer. Implemented by comparison of the HALT bit.

9. **OSR (74048)** - Inclusive **OR** of the Switch Register with the AC. The result is left in the AC and the original content of the AC is destroyed.
Note that all the Memory Reference Instructions begin with an octal 0 to 5, and that all the Non-Memory Reference Instructions (group 1 and group 2 microinstructions) begin with and octal 7. The I/O (Input/Output) instructions are not given here, but they all begin with an octal 6xxx.

Interrupts are external signals that cause temporary suspension of the fetch execute cycle. On the PDP-8, there are two instructions, ION (6001\textsubscript{8}) and IOF (6002\textsubscript{8}) that control whether interrupts are ignored. ION sets the interrupt enable flag, and IOF clears it. On the PDP-8, an interrupt is ignored unless the last instruction was not 6001\textsubscript{8} and interrupt enable flag is 1. If these conditions are met, the interrupt causes the same action as executing the instruction 4000\textsubscript{8} without fetching such a machine code from memory. The interrupt also causes the interrupt enable flag to become 0. At that point, the fetch execute cycle resumes. At the end of the interrupt service routine, the programmer must put an ION instruction followed by a JMP indirect instruction.
Figure 1.1: LD-14
LAB 1H - Introduction to the LD-14

Pre-lab: Read the pre-lab description

Goals: Understand the layout of the LD-14. Understand how to enter programs into the LD-14.

Turn In: Answers to questions.

Part 1a - Enter programs

Now that you have read the Prelab and are familiar with the layout of the LD-14 and the basic PDP-8 instructions that can be used with the LD-14, let’s enter a short two line program to see how the machine operates.

1. Turn on the machine (power switch location shown on Figure 1). With the power on you will probably see a continuous display of flashing lights across the control panel.

2. Locate the CLK light, which may or may not be flashing. Locate the Clock Speed Switch. This switch has three settings: Manual, Slow and Fast. If the CLK light is not flashing, make sure this switch is toggled to the S (or middle) position. Now locate the Clock Knob. This knob further slows or speeds the LD-14 clock. When turned clockwise the machine slows down (causing the CLK light to flash at a slower rate). When the clock knob is turned counterclockwise the machine speeds up. When the clock speed switch is set to S and the clock knob is turned fully clockwise the machine’s clock is running at its slowest possible speed. Play with the clock speeds until you are familiar with them. Note that the faster the clock runs the more it seems the CLK light is lit continuously.

3. You should have the Clock Speed Switch set on S and the Clock Knob turned fully counterclockwise unless the lab tells you otherwise. To reset the LD-14, push the STOP button and then push the CLEAR button. The CLEAR signal clears the internal registers not the display on the front panel. You may still have lights lit up, but the CLEAR signal should cause the lights to quit flashing. It takes time for the machine to recognize these two signals. If you have your clock set very fast, it will happen very fast. When working with very slow clock speeds make sure you hold down any buttons you press for more than one clock cycle, this could mean for more than one second.

4. Now, make sure the SR switches are all toggled to octal 0000. Remember that there are twelve toggle switches so all octal values must be entered in binary. When all the switches are in the “down” position (facing you) they are toggled to zero. Similarly, flipping the toggle switch to the “up” position toggles a binary one.

5. Set the SR register 0000s and “load” this value in the AC, PC, IR, MB, and MA. This is accomplished by pushing each button next to the right of each register (each row of lights).
Remember to be aware of how fast your clock is running. Each button must be held down long enough to allow the machine to process the signal. (Hint: You can always speed up the clock by turning the clock knob.)

6. Notice the row of lights marked M (Memory display). You should notice some of those lights are still lit (there is a 1 in 4096\textsubscript{10} chance that the memory word is 0000\textsubscript{8}). This is random data located at MA 0000\textsubscript{8}. The M display always reflects the contents of Memory[MA]. In a moment when you toggle a meaningful value into the SR and press DEP, that value is reflected in the M display and is deposited as the contents of Memory[MA].

7. Next we will enter the following short two line program into the memory of the LD-14 (Steps 8 and 9 will explain this process to you):

\begin{align*}
0000/7001 & \quad \text{IAC} \\
0001/5000 & \quad \text{JMP}
\end{align*}

The first four octal digits are the memory address (MA) and the last four are the instructions or contents (M) located at Memory[MA].

8. Toggle 7001\textsubscript{8} into the SR register (remember the Little Endian ordering). You can enter this value into memory address 0000\textsubscript{8} by pressing the DEP button located on the bottom left side. Notice the MA display is now 0001\textsubscript{8}. The M register display is again lit with a random value. The only thing we have entered is 7001\textsubscript{8} at MA 0000\textsubscript{8}, but we are now seeing the display of Memory[0001\textsubscript{1}]. In the next step we will discuss how to examine what we have entered in to various memory locations. For Now, toggle the SR switches to 5000\textsubscript{8} and press DEP again. The MA should increment to 0002\textsubscript{8}.

9. Now let’s check to see if the functional program we entered is actually in the memory. Reset the SR to 0000\textsubscript{8} and press the MA button. You should see your first instruction, 7001\textsubscript{8} in the M display. Next, push the EXAM button. The MA display has incremented to 0001\textsubscript{8} and the second instruction 5000\textsubscript{8} should be in the M display. If something went wrong when you deposited the instructions, you will need to re-enter with the MA set to 0000\textsubscript{8} (back to step 5).

10. To run the program, press STOP, CLEAR, and CONT separately and in that order. If all goes well, you should see the AC register incrementing in binary.

The program continuously increments the AC by jumping (JMP) in an infinite loop back to the memory location containing the IAC instruction. Play with the clock speed as you watch this program. Pay particular attention to the LINK light.
Part 1b - Location of Chips - Refer to Figure 1 that is part of the pre-lab description.

Look at location H8. The chip is numbered SN74LS151N (note the number is upside-down). Even though some of these chips have two numbers the actual chip number will usually have a LS in the number. The LS stands for “Low power Schottky” which is a family of TTL logic chips. Let’s try another one, look at location E11. This chip is numbered SN74LS30N. To locate J chips (the larger ones in the middle) count from the bottom up. For example, J3 is numbered SNJ54LS181 or SNJ74LS181.

If we look at location A16 we see that the chip has two numbers. Remember that the chips in A5 to D16 are memory chips. Not all the machines have the same kinds of memory chips. They are all equivalent to “2102”s, but different manufactures use slightly different part #’s for memory. Because of this, we won’t ask you to locate part numbers on any of the memory chips.

In later labs we will locate certain chips and look at there respective circuit diagrams. We will then test certain pins on a chip with the logic probe to see how they work. In the last lab you will be asked to replace good chips with bad chips and tell us why the machine fails. Practice locating a few chips by answering the last lab question.

Conclusion

This lab was designed to familiarize you with how the LD-14 is operated and the layout of the circuit board. In the next lab you will be asked to enter in a larger program that is designed to compute a mathematical function. Because of this, be sure you understand how to deposit and check what you’ve deposited in the LD-14’s memory, as we did in this lab. You will run this program and, using the logic probe, you will watch signals as the pass though the circuitry.
LAB 1H - Questions

1. At what knob and switch setting(s) is the LD-14 clock running at maximum speed?

2. How many clock cycles does it take to “load” a register (the MA for example)? (Hint: We did this in step 5. Slow the machine down and count the clock flashes until a particular register is loaded. Initially, press and hold the load button when the CLK light is off and then start counting the number of times the CLK light blinks. Remember to STOP & CLEAR the machine first.)

3. How many clock cycles does it take to “DEPosit” a word into memory? (Hint: Count how many times the clock flashes until the MA increments. Again, initially press and hold the DEP button when the CLK light is off and then start counting the number of times the CLK light blinks. STOP & CLEAR the machine first.)

4. What does the IAC program do?

5. What are the chip numbers for K8, J7, and G1?

6. Why does the LINK light come on (or off) when the AC goes from all lit up to all off?

7. Write a short paragraph describing how to deposit a program into memory and check what you have deposited.

8. Change the first instruction of the program from 7001s to 7010s. Toggle a 0001s into the AC. Press STOP, CLEAR and CONT. Turn the knob so that the CLK light flashes about every second. What does this program do?

9. Same as question 8 using 7004s.