University of Victoria

CENG 241

Digital Design I

Lab 7: RAM System

Instructor:
Dr. Amirali Baniasadi

Teaching Assistant:
Grace Hui

Yves Sénéchal V00213837
Tyler Stephen V00812021
A01 - B03

July 27, 2015

1 Introduction

This lab will detail the construction of a RAM system with the ability to store and retrieve 4-bit datawords. A 4-bit counter will generate sequential addresses, a tri-state buffer will control the data input line and a 4-bit register will store the retrieved data from RAM (see Figure 1). The focus of this lab will be on the construction of the controller. The purpose of the controller is to ensure that each component in the RAM system is configured properly in the correct sequence needed for reading or writing to RAM.

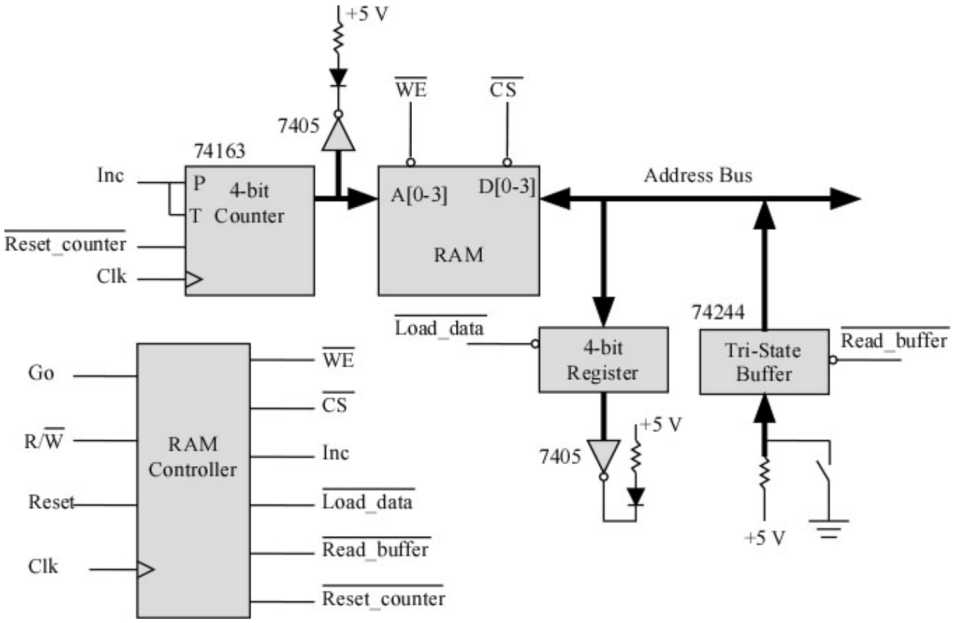

Figure 1: RAM controller and memory system

2 Discussion

The RAM, buffer and register are connected to each other via an address bus. In order for the RAM system to function properly, it is essential that there is only one correct "path" for the data to take on the bus. By changing the operating mode of the RAM, register and buffer, the controller changes the path for the data.

When the system is powered on and reset it enters an idle state where both the RAM and register are "closed" (their inputs are set to high impedance) and the buffer is passing the selected dataword to the address bus. Based on user input, the read or write cycles can be activated.

To write to RAM:

1. RAM I/O is configured as an input and is no longer high impedance
2. a clock cycle passes to prevent timing problems
3. the counter is incremented to a new address and the RAM returns to high impedance
4. the system returns to idle

To read from RAM:

1. RAM I/O is configured as an output and the buffer is set to high impedance
2. the register reads the RAM output from the bus
3. the counter is incremented, the RAM returns to high impedance, the register returns to high impedance and cycles the previous RAM output and the buffer outputs a new dataword to the bus
4. the system returns to idle

This process is represented as a state machine in Figure 4.

2.1 Timing considerations

The read and write timing diagrams, shown in Figure 2 and Figure 3 respectively, indicate that the moment in which the data is valid is offset from moment where the address is valid; the data is valid after the address is valid. The access time and hold time for the read operation and write operation respectively can be attributed to the RAM's internal delay of 85 ns maximum. However, this does not pose a problem with a manual clock, but could become a nuisance should high-frequency clock be used.

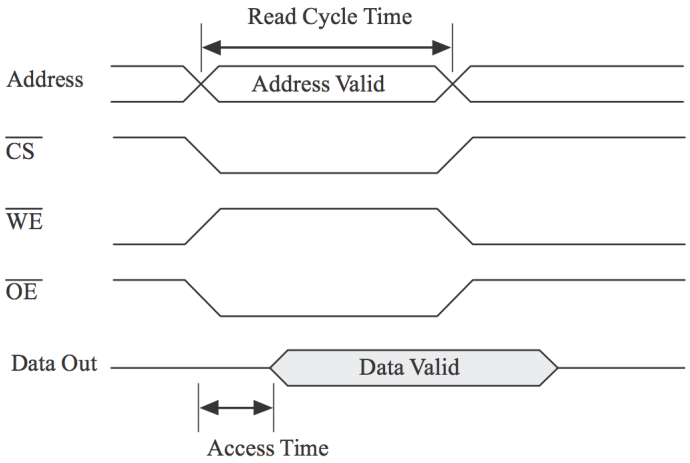

Figure 2: Timing diagram for the RAM read operation

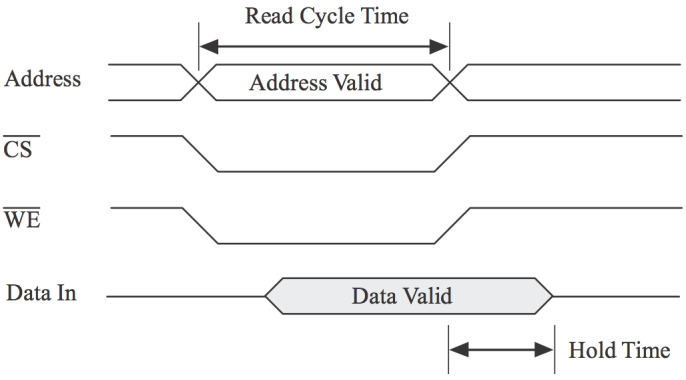

Figure 3: Timing diagram for the RAM write operation

3 RAM controller design process

3.1 State diagram

The state diagram of the RAM controller is shown in Figure 4 (note, ↑ signifies a clock rising edge). The system was designed as a Moore machine with the transition tables displayed in figure 5.

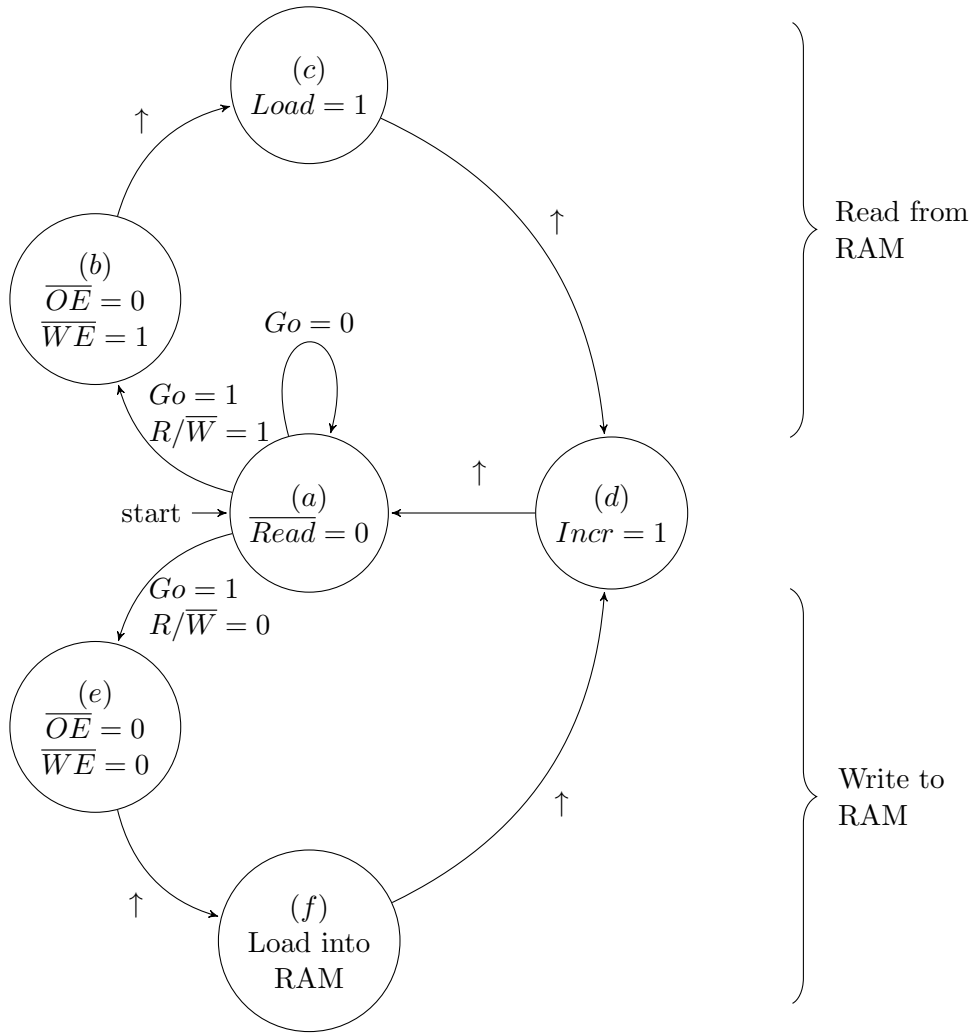

Figure 4: State diagram for the RAM controller

State	S_2	S_1	S_0
a	0	0	0
b	0	0	1
c	0	1	0
d	0	1	1
e	1	0	0
f	1	0	1
-	1	1	0
-	1	1	1

(a) State enumeration

S_2	S_1	S_0	Go	$R/\overline{W}$	S_2^+	S_1^+	S_0^+
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	1	0	1	0	0
0	0	0	1	1	0	0	1
0	0	1	0	0	0	1	0
0	0	1	0	1	0	1	0
0	0	1	1	0	0	1	0
0	0	1	1	1	0	1	0
0	1	0	0	0	0	1	1
0	1	0	0	1	0	1	1
0	1	0	1	0	0	1	1
0	1	0	1	1	0	1	1
0	1	1	0	0	0	0	0
0	1	1	0	1	0	0	0
0	1	1	1	0	0	0	0
0	1	1	1	1	0	0	0
1	0	0	0	0	1	0	1
1	0	0	0	1	1	0	1
1	0	0	1	0	1	0	1
1	0	0	1	1	1	0	1
1	0	1	0	0	0	1	1
1	0	1	0	1	0	1	1
1	0	1	1	0	0	1	1
1	0	1	1	1	0	1	1
1	1	0	0	0	-	-	-
1	1	0	0	1	-	-	-
1	1	0	1	0	-	-	-
1	1	0	1	1	-	-	-
1	1	1	0	0	-	-	-
1	1	1	0	1	-	-	-
1	1	1	1	0	-	-	-
1	1	1	1	1	-	-	-

(b) Next state

Figure 5: Transition tables for the Moore machine

From here three karnaugh maps were used to resolve the next states S_2^+, S_1^+, and S_0^+. Their state equations are shown below.

$$S_2^+ = S_2 S_0' + S_1' S_0' \; Go \; R/\overline{W}$$
$$S_1^+ = S_1 \text{ XOR } S_0$$
$$S_0^+ = S_2 + S_1 S_0' + S_0' \; Go \; R/\overline{W}$$

S_2	S_1	S_0	$\overline{OE}$	$\overline{WE}$	$Load$	$\overline{Read}$	$Incr$
0	0	0	1	1	0	0	0
0	0	1	0	1	0	1	0
0	1	0	0	1	1	1	0
0	1	1	1	1	0	X	1
1	0	0	X	0	0	0	0
1	0	1	X	0	0	0	0
1	1	0	-	-	-	-	-
1	1	1	-	-	-	-	-

Table 1: State output

We were able to reduce the complexity of the controller by eliminating two outputs from the controller in Figure 1. $\overline{CS}$ was set to LOW peramently and the $\overline{Reset}$ line was externalized from the state machine and made asynchronous. The state outputs are summarized in table 1. Five karnaugh maps were needed to resolve their state equations which are shown below.

$$\begin{aligned}
\overline{OE} &= S_1' S_0' + S_1 S_0 \\
\overline{WE} &= S_2' \\
Load &= S_1 S_0' \\
\overline{Read} &= S_1 + S_2' S_0 \\
Incr &= S_1 S_0
\end{aligned}$$

Although the lab manual made it seem necessary to include state f to accommodate timing delays from the RAM, our work in the lab found that the RAM output would appear in the registers in state e. This suggests that f and e could be condensed and the number of states reduced by one. However, we would still need three flip-flops to implement the five remaining states.

3.2 Controller logic

The logic of the RAM controller was implemented with D flip-flops. Figure 6 shows the design of the state machine and Figure 7 shows the logic for generating the output.

4 Conclusion

A functioning 4-bit RAM controller was built using D flip-flops and some external logic. The system had a 4-bit counter which determined the address of the data. Four LEDs displayed the present address, while, when in read mode, four other LEDs revealed the stored data at that address.

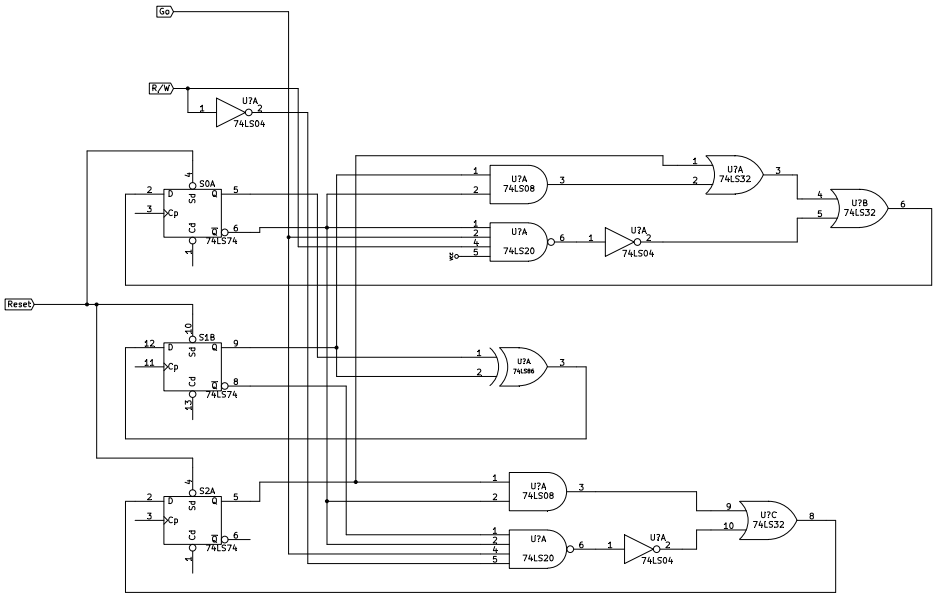

Figure 6: RAM controller state machine

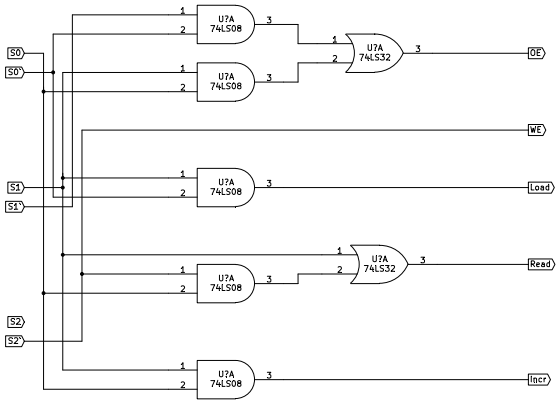

Figure 7: RAM controller output