Lab 16: Data Busses, Tri-State Outputs and Memory

UC Davis Physics 116B
Rev. 0.9, Feb. 2006

1 Introduction

1.1 Data busses

Data busses are ubiquitous in systems which must communicate digital data. Examples are the RS-232 bus, originated for teletype interconnection, the GPIB bus for interconnecting digital measurement instruments and the PCI bus for connecting digital devices to computers. Additional busses allow internal parts of a computer to communicate with each other.

The data bus is named after the regular commuter bus on the streets. The commuter bus will transport many people at once and the same vehicle and roads are used for many origins and destinations. Similarly, a data bus transports many bits of data simultaneously from one digital circuit to another. To be considered a bus, several circuits must be able to send data along the same wires (only one at a time, though), and several circuits must be able to receive data from those wires. The data bus idea is also rather analogous to a group of people talking in a room. Only one person at a time should talk, her voice (the data) fills the entire room (the bus), and many people at a time can listen. For devices to communicate on a data bus, they must be able to ”shut up” and let another device send data over the bus. Thus a bus will generally have three components:

1. data lines to carry the data
2. address lines to indicate the source or destination of the data
3. control lines to organize the transmission of the data so only one device can send data at a time and the appropriate devices receive it.

The bus may contain additional control lines and some functions may be combined on a single line. The bus may also provide power supply voltages.

1.2 Tri-state Outputs

Typical logic gates such as the 74LS00 quad NAND are not very polite listeners; they are always outputting either a 1 or a 0. We need to give them a third state, a “shut up” state, if you will, to allow others to have the floor. This third state is called the high impedance (high Z) state and outputs with this capability are called tri-state outputs. They are often designated on a circuit diagram by the symbol $\bigtriangledown$. The 74LS241 chip is an octal tristate buffer partitioned in two groups of 4 lines each, as shown in Fig. 1. It allows normal logic outputs to access a shared line on a data bus. The $1\overline{\mathrm{G}}$ and $2\mathrm{G}$ inputs determine whether or not the corresponding group is in the high

Z state. Each buffer element is a non-inverting Schmitt trigger with TTL logic levels. They have high fan-out capability and can drive terminated transmission lines down to 133 Ω impedance. In the first (short) part of this experiment, you will look at the output current of the 74LS241 tristate buffer in each of its three states, 0, 1, and high Z.

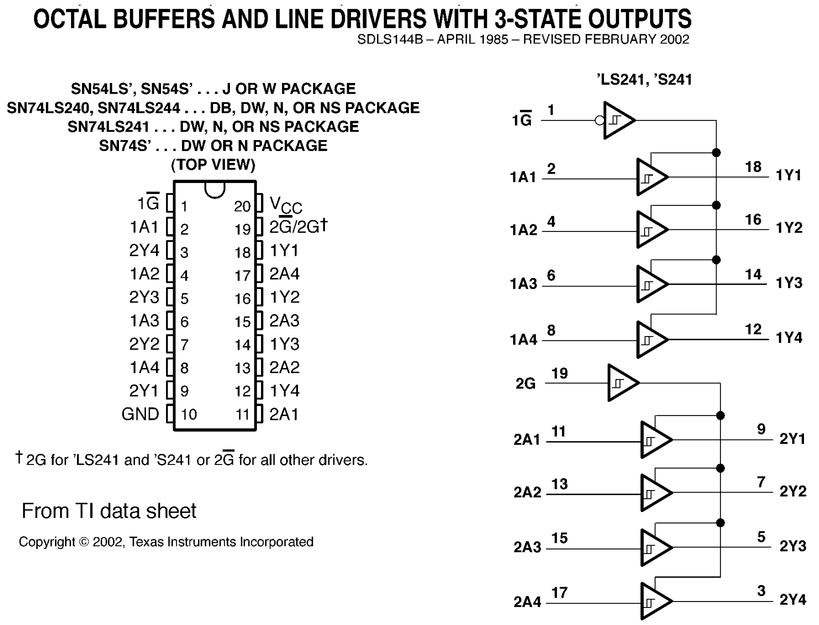

OCTAL BUFFERS AND LINE DRIVERS WITH 3-STATE OUTPUTS
SDLS144B – APRIL 1985 – REVISED FEBRUARY 2002
SN54LS', SN54S' . . . J OR W PACKAGE
SN74LS240, SN74LS244 . . . DB, DW, N, OR NS PACKAGE
SN74LS241 . . . DW, N, OR NS PACKAGE
SN74S' . . . DW OR N PACKAGE
(TOP VIEW)
1G 1 20 VCC
1A1 2 19 2G/2G†
2Y4 3 18 1Y1
1A2 4 17 2A4
2Y3 5 16 1Y2
1A3 6 15 2A3
2Y2 7 14 1Y3
1A4 8 13 2A2
2Y1 9 12 1Y4
GND 10 11 2A1
† 2G for 'LS241 and 'S241 or 2G for all other drivers.
From TI data sheet
Copyright © 2002, Texas Instruments Incorporated
'LS241, 'S241
1G 1
1A1 2 18 1Y1
1A2 4 16 1Y2
1A3 6 14 1Y3
1A4 8 12 1Y4
2G 19
2A1 11 9 2Y1
2A2 13 7 2Y2
2A3 15 5 2Y3
2A4 17 3 2Y4

Figure 1: Specifications for 74LS241 buffers and line drivers from Texas Instruments data sheets.

1.3 Memory

Recall that any sequential logic circuit must have some sort of memory. A circuit designed to do nothing but store and recall data is called a memory circuit. In this experiment, you will use 4 bits each of 16 8-bit data registers ("bytes") contained in the 6264 memory chip, an 8192-word x 8-bit High Speed CMOS Static RAM (SRAM) chip. Although RAM stands for "read only memory," it is understood to mean "read-and-write" memory, as opposed to ROM, or read-only memory. Both types of memory have random access (as opposed to serial or sequential access, such as with magnetic tape or a disk drive). An SRAM circuit contains very many basic memory cells, each equivalent to a D flip-flop. In CMOS technology, each cell is made from 6 MOSFETs, 4 as two cross-coupled inverters (the bistable memory element) and one each to control access to the read and write lines.

As you might imagine, even 16 4-bit data registers would take very many pins on a chip if they were all available at once. So, a memory chip allows you to access only one register at a time. Each is assigned a number, called its address. To select a particular data register, its address is applied to the address inputs of the memory chip. Then, data can be written to or read from that register. Once a given address on a memory chip has been selected, either read or write operations can be done on it using the same data pins.

The basic organization of the 8192-byte SRAM is shown in Fig. 2. The memory cells are arranged in a 256×256 array. Groups of 8 cells (a data "byte") are selected using the 12 address lines A0-A11 (12 address lines are needed to access $8192 = 2^{12}$ bytes). The specific cells for a given address are accessed for reading or writing by a system of multiplexers and address decoders and are connected to the lines I/O1-I/O8 under the control of the lines $\overline{\mathrm{CS1}}$, CS2, $\overline{\mathrm{WE}}$ and $\overline{\mathrm{OE}}$. It may not be obvious how the same pins can be used as both input (writing) and output (reading), but this can be done with suitable use of tristate outputs, as shown in the truth table shown in Fig. 3. *Question: if the memory array is 256 x 256 as shown, are all array elements occupied by addressible memory cells?*

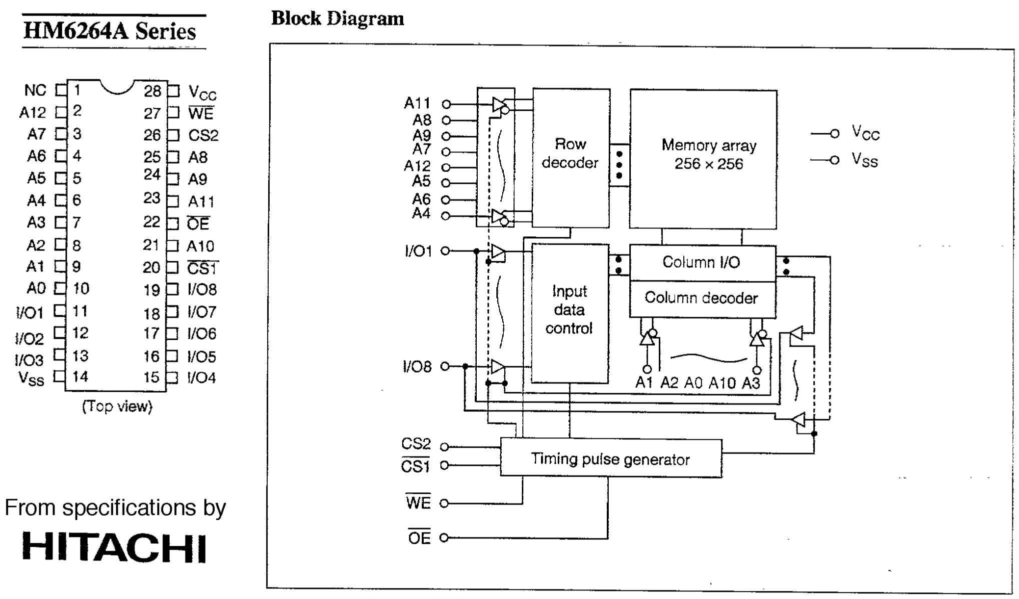


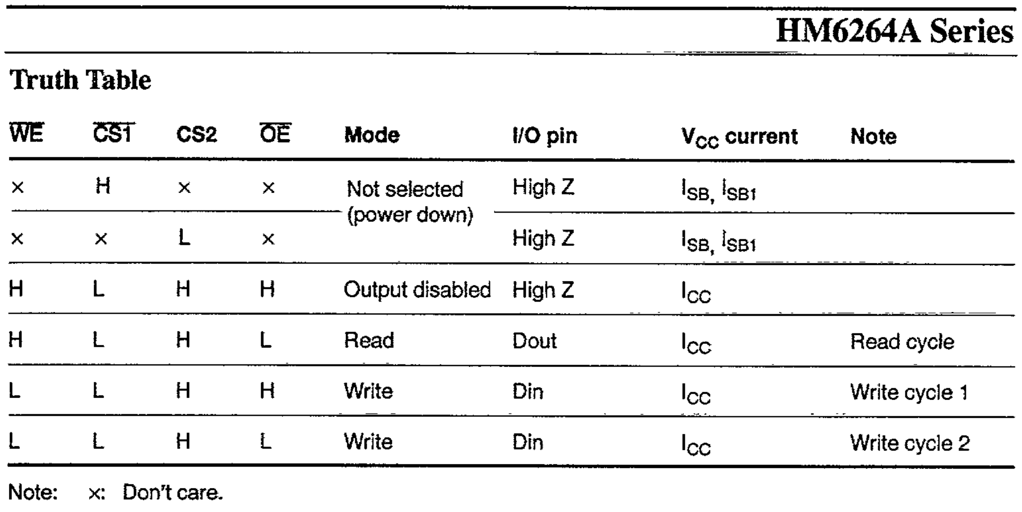
Figure 2: Pinouts and block diagram of 8192-word x 8-bit High Speed CMOS Static RAM chip from Hitachi HM6264A Series data sheets.

The symbol WE stands for "write enable," CS for "chip select" and OE for "output enable." CS controls selection of the chip for reading or writing. $\overline{\mathrm{CS1}}$ must be low and CS2 high if a byte of data is to written or read. $\overline{\mathrm{WE}}$ must be low for writing and high for reading. $\overline{\mathrm{OE}}$ provides an independent control of the chip tri-state output. It must be low to read data from the chip onto the data bus. Data may be written to the chip with $\overline{\mathrm{OE}}$ either high or low.

1.3.1 Read Cycle

Fig. 4 shows the timing diagram for reading data from the memory chip. A key to the symbols used in the timing waveforms is given in Fig. 5. *Note: in this experiment, we will wire* $\overline{\mathrm{OE}}$ *low,* CS2 *high and use* $\overline{\mathrm{CS1}} \equiv \overline{\mathrm{CS}}$ *and* $\overline{\mathrm{WE}}$ *to control access to the chip for reading and writing (Write cycle 2 in the truth table).* In this way, we can ignore $\overline{\mathrm{OE}}$ and CS2 in the timing diagrams.

To read from the chip, $\overline{\mathrm{WE}}$ must be high and the address must be established on the address lines. Then setting $\overline{\mathrm{CS1}} \equiv \overline{\mathrm{CS}}$ low will cause output data to be asserted on the I/O lines. After the data


HM6264A Series

Truth Table

$\overline{\text{WE}}$	$\overline{\text{CS1}}$	CS2	$\overline{\text{OE}}$	Mode	I/O pin	V_{CC} current	Note
×	H	×	×	Not selected (power down)	High Z	I_{SB}, I_{SB1}	
×	×	L	×		High Z	I_{SB}, I_{SB1}	
H	L	H	H	Output disabled	High Z	I_{CC}	
H	L	H	L	Read	Dout	I_{CC}	Read cycle
L	L	H	H	Write	Din	I_{CC}	Write cycle 1
L	L	H	L	Write	Din	I_{CC}	Write cycle 2

Note: ×: Don't care.

Figure 3: Truth table for control lines for 8192-word x 8-bit High Speed CMOS Static RAM chip from Hitachi HM6264A Series data sheets. *Note: in this experiment, we will wire* $\overline{\text{OE}}$ *low,* CS2 *high and use* $\overline{\text{CS1}} \equiv \overline{\text{CS}}$ *and* $\overline{\text{WE}}$ *to control access to the chip for reading and writing (Write cycle 2 in the truth table).* In this way, we can ignore $\overline{\text{OE}}$ and CS2 in the timing diagrams. We will be limited to Row 1 (High Z), Row 4 (Read cycle) and Row 6 (Write cycle 2).

are entered in the receiving device(s), $\overline{CS}$ is set high which returns the I/O lines to their high Z state and ends the cycle, after which a new address may be asserted on the address lines.

1.3.2 Write Cycle

The timing of waveforms for the write cycle is given in Fig. 6. Recall that $\overline{\text{OE}}$ is wired low, CS2 is wired high and that we are using $\overline{\text{CS1}} \equiv \overline{\text{CS}}$ and $\overline{\text{WE}}$ to control access to the chip for reading and writing (Write cycle 2). The desired write address must first be established on the address lines. Then, as specified in Note 1 of Fig. 6, a write occurs during the overlap of a low $\overline{\text{CS1}}$ and a low $\overline{\text{WE}}$. The write cycle ends at the earliest transition of either $\overline{\text{CS1}}$ or $\overline{\text{WE}}$ going high. Note that write mode is "dangerous" in that data will be written to memory in write mode ($\overline{\text{WE}} = \overline{\text{CS1}}$ = low), even if it is not the intended data or the intended address. One must end the cycle before allowing the address or input data to change.

HM6264A Series

Read Timing Waveform

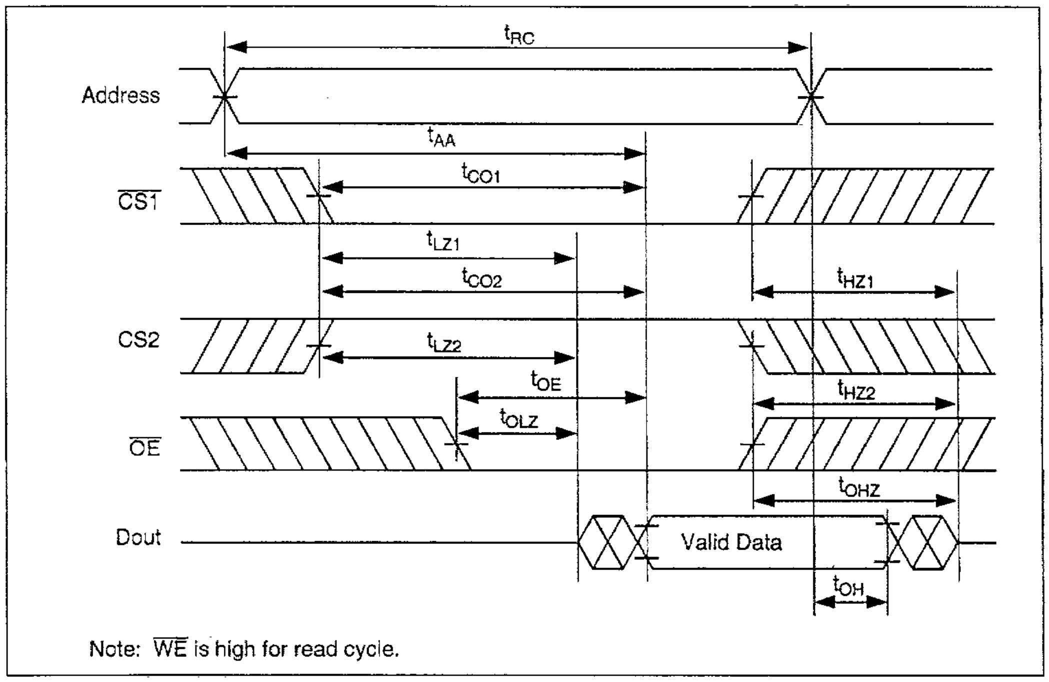
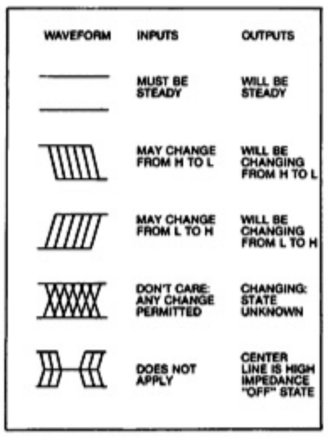


Figure 4: Read cycle timing waveforms for 8192-word x 8-bit High Speed CMOS Static RAM chip from Hitachi HM6264A Series data sheets. Details of the timing specifications are given in the complete Hitachi data sheets posted on the Physics 116 web site. In this experiment, we will wire $\overline{\text{OE}}$ low, CS2 high and use $\overline{\text{CS1}} \equiv \overline{\text{CS}}$ and $\overline{\text{WE}}$ to control access to the chip.

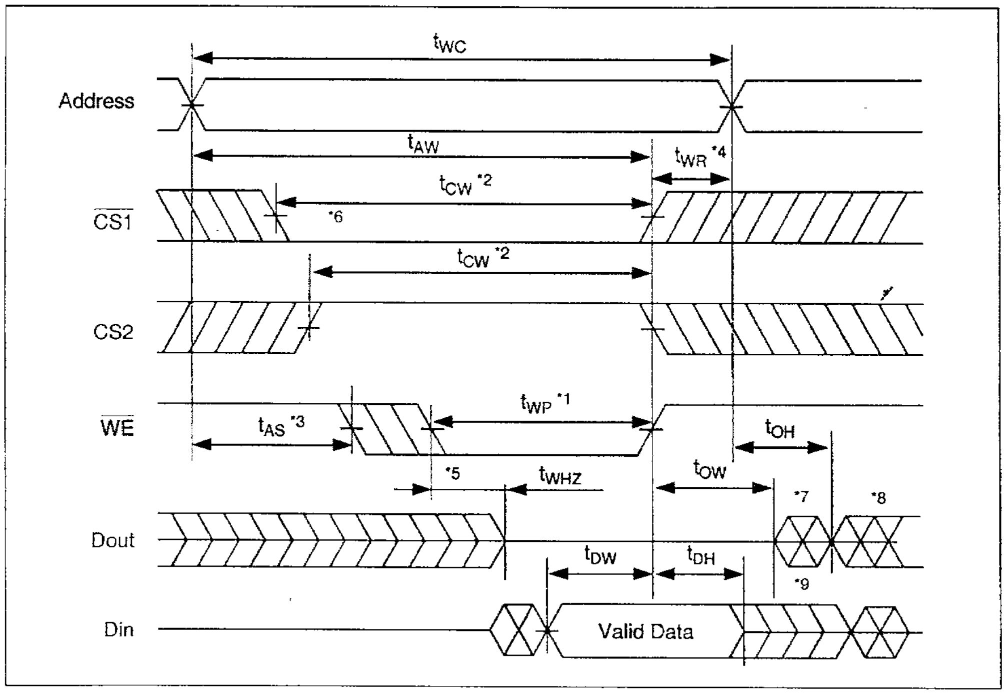
KEY TO SWITCHING WAVEFORMS



WAVEFORM	INPUTS	OUTPUTS
═	MUST BE STEADY	WILL BE STEADY
⧹⧹⧹	MAY CHANGE FROM H TO L	WILL BE CHANGING FROM H TO L
///	MAY CHANGE FROM L TO H	WILL BE CHANGING FROM L TO H
XXX	DON'T CARE: ANY CHANGE PERMITTED	CHANGING: STATE UNKNOWN
⟫—⟪	DOES NOT APPLY	CENTER LINE IS HIGH IMPEDANCE "OFF" STATE

Figure 5: Explanation of symbols used in the timing waveforms (from MOSEL MS6264 data sheets).

Write Timing Waveform (2) ($\overline{\text{OE}}$ Low Fix)



Notes: 1. A write occurs during the overlap of a low $\overline{\text{CS1}}$, a high CS2, and a low $\overline{\text{WE}}$. A write begins at the latest transition among $\overline{\text{CS1}}$ going low, CS2 going high and $\overline{\text{WE}}$ going low. A write ends at the earliest transition among $\overline{\text{CS1}}$ going high, CS2 going low and $\overline{\text{WE}}$ going high. Time t_{WP} is measured from the beginning of write to the end of write.

2. t_{CW} is measured from the later of $\overline{\text{CS1}}$ going low or CS2 going high to the end of write.
3. t_{AS} is measured from the address valid to the beginning of write.
4. t_{WR} is measured from the earliest of $\overline{\text{CS1}}$ or $\overline{\text{WE}}$ going high or CS2 going low to the end of the write cycle.
5. During this period, I/O pins are in the output state, therefore the input signals of opposite phase to the outputs must not be applied.
6. If $\overline{\text{CS1}}$ goes low simultaneously with $\overline{\text{WE}}$ going low or after $\overline{\text{WE}}$ goes low, the outputs remain in high impedance state.
7. Dout is the same phase of the latest written data in this write cycle.
8. Dout is the read data of the next address.
9. If $\overline{\text{CS1}}$ is low and CS2 is high during this period, I/O pins are in the output state. Input signals of opposite phase to the outputs must not be applied to I/O pins

Figure 6: Write cycle timing waveforms for 8192-word x 8-bit High Speed CMOS Static RAM chip from Hitachi HM6264A Series data sheets. Details of the timing specifications are given in the complete Hitachi data sheets posted on the Physics 116 web site. In this experiment, we will wire $\overline{\text{OE}}$ low, CS2 high and use $\overline{\text{CS1}} \equiv \overline{\text{CS}}$ and $\overline{\text{WE}}$ to control access to the chip for reading and writing (Write cycle 2).

2 Experiment

In this lab, you will build a fairly large circuit that will allow the user to write data to any address in a RAM and read that data out sequentially or using a DIP switch to specify the address. Once again, this is a large circuit. Give some thought to how you want to lay out the components on the breadboard and be sure to use good circuit-building technique. That is, build and debug each section of the circuit separately and then connect them all together.

2.1 Tri-state output current

First, we will measure the output current of a tristate device, one buffer of a 74LS241 circuit. Use the circuit in Fig. 7 to do this. For your lab report, make a table of the current readings for all combinations of switch settings. Also, answer these two questions: What property of a tristate output makes it useful for a bus? Which switch causes zero output current (the high Z state)?

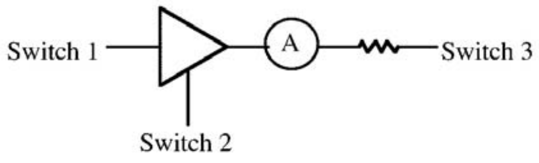


Figure 7: Circuit to measure tristate buffer output current. Use a 1 kΩ resistor.

2.2 The data bus

The goal of this section is to be able to write data to some addresses in RAM and to read it back. The block diagram in Fig. 8 shows in general how to do this. For the RAM chip, recall that we will use only the 4 low order address lines and 4 of the 8 I/O lines. Also, we will wire $\overline{\mathrm{OE}}$ low, CS2 high and use $\overline{\mathrm{CS1}} \equiv \overline{\mathrm{CS}}$ and $\overline{\mathrm{WE}}$ to control access to the chip for reading and writing (Write cycle 2). Wire the unused address inputs to ground. Connect the 4 unused I/O lines to 10 kΩ resistors connected to ground. This way they will not "float" on write cycles and have reasonable loads on read cycles.

Use logic switches on the breadboard for the address and data inputs. In some cases the switch data was enabled too soon relative to $\overline{\mathrm{WE}}$ by the 'LS241 buffer during the write cycle. To remedy this, two inverters were inserted in the path between the $\overline{\mathrm{WE}}$ input of the RAM chip and the $\overline{\mathrm{G}}$ input of the 'LS241 in Fig. 8. The "display" in the diagram is just an array of 4 LED's.

First (and for your lab report), draw a detailed circuit diagram for this circuit and write a description of how you will test it by writing data to several memory addresses and then by reading it back. Then, build the circuit and test it. In your lab report, note whether your circuit passed all your tests or whether any of the tests had to be altered.

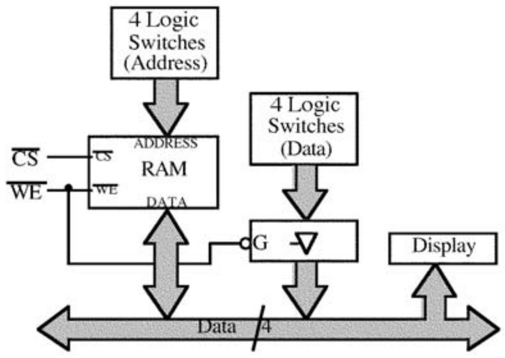


Figure 8: Block diagram for the circuit to read and write to RAM.

2.3 The address bus

Next, you will alter the circuit so that you can either give the address by hand or allow a counter to sequence through all of the 16 low order memory addresses. The block diagram for this modified circuit is shown in Fig. 9. As before, draw a detailed circuit diagram before you build the circuit. Write a test procedure which includes writing several 4-bit words to RAM and then reading all 16 using the counter. Then build and test the circuit, noting any difficulties during testing.

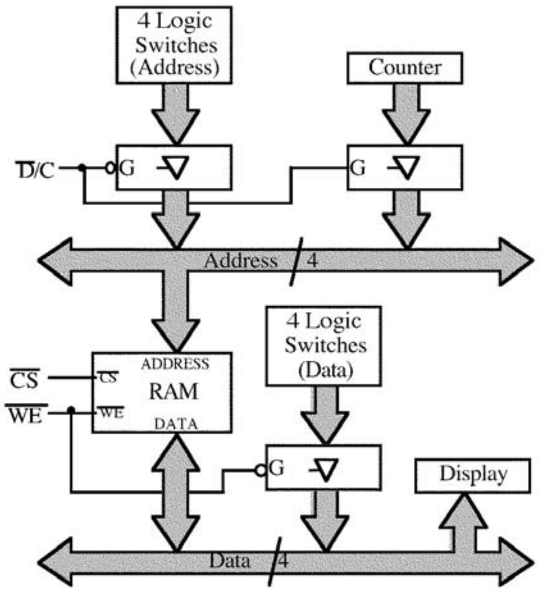


Figure 9: Block diagram for the complete memory circuit.

2.4 Digital waveform synthesizer

Now use your memory circuit to generate a voltage pattern. Add the DAC08 to the data bus as shown in Fig. 10. (That is, connect the 4 bit data bus to the DAC's 4 more significant input bits and the other input bits to ground.) The DAC08 circuit from Lab 15 is shown in Fig. 11. Using the procedure you wrote above, enter 16 numbers into memory then read them out sequentially into the DAC at a clock speed of about 10 kHz. Look at the output on the oscilloscope. If you are ambitious, try hooking this up to a speaker. Try to enter numbers that make the output look like a sine wave, a triangle wave, and any other shape you'd like to see. If you used the speaker, try to get a particular sound (a violin, trumpet, human voice, etc.). For your lab report, describe what is happening in your own words and include your favorite wave shape and the 16 numbers that made it.

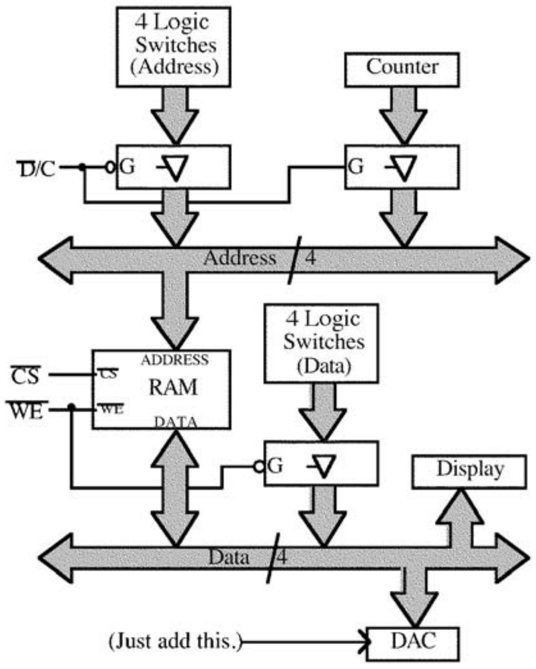


Figure 10: Block diagram for the digital waveform synthesizer.

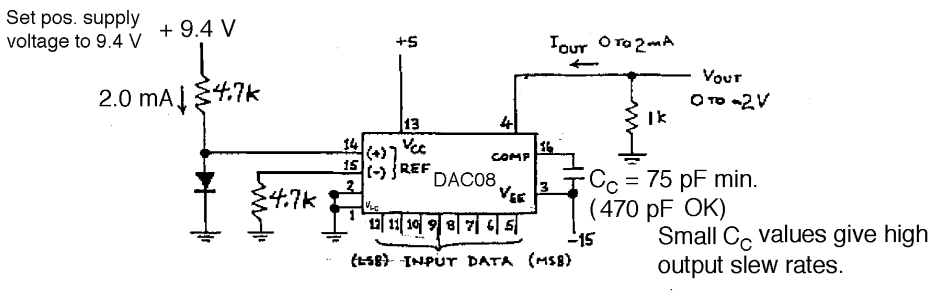


Figure 11: DAC08 wiring diagram from Lab 15.