
Chapter 8

The $\overline{\text{MAP}}$ Circuit

In the preceding chapter, we described the connections to the 68000 microprocessor and actually got it to the point where it ran. It is now time to add some of the circuits which make it into a working system.

8-1. Discussion

(Let us begin with a short Detour): We have by now learned that signals could be either active high or active low, and that the name of a signal was often a dead giveaway if it was “notted”; that is, if it had the “not bar” over it. A signal like $\overline{\text{DTACK}}$ was therefore active low because of being notted, whereas FC0 was active high because it did not have a not bar.

Another way to mark a signal in a diagram is by using a *bubble*, which is simply a small circle at the end of a wire. Many engineers and technicians tend to get a bit careless with bubbles so they can’t always be trusted, but when properly used they can be very helpful. Look at Fig. 8-1 for an example; this is just a repeat of part of the HALT LED circuitry discussed in Chapter 3. The signal arriving from the left is $\overline{\text{HALT}}$; it has a not bar over the name (we say it is *notted*) to indicate that this signal is low when asserted. In other words, this signal goes low when the 68000 is halted. This is also shown by the bubble at pin 13, the input side of inverter U32f. The output on pin 12, as well as the input into pin 9, has no such bubble and so

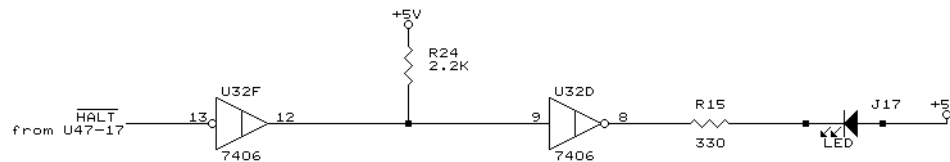


Fig. 8-1. The HALT LED circuit.

we know that this wire is high when the 68000 is halted. Pin 8, the output of U32d, on the other hand, has a bubble again, so we again see that this point is low during a halt. Put another way, pins 13 and 8 are active low, whereas pins 12 and 9 are active high. Here is a prime example where active low and active high circuits are separated by just a tenth of an inch.

It's important to understand that U32f and U32d are really the same type of device even though their symbols are different. They are both part of U32, a 7406 IC, and are both exactly the same. The fact that one has a bubble on the input whereas the other has it on the output is just a convenient way of drawing the same physical circuit. It is drawn that way to explain what is happening in *this particular circuit*.

Fig. 8-2 shows some more examples of this. The left diagram at (a) is normally called an AND gate; described in words, its job is to "make the output high if input A is high AND input B is high." Thus if both inputs are high, then the output is high. But there is another way of looking at it: if either input is low, then the output is low. Thus we could say "make the output low if A is low OR B is low." That sounds more like an OR gate which works with lows. In a way, the right diagram at (b) is supposed to tell us just that - think of each bubble as being the word 'low'.

Hence if you have two active high circuits and want to AND them to assert an active high output if both inputs are asserted (which means high), then you use the AND gate symbol at the left. But if you have two active low circuits and want to assert an active low output when either input is asserted (which, in this case, means low), then you use the strange-looking OR symbol at the right.

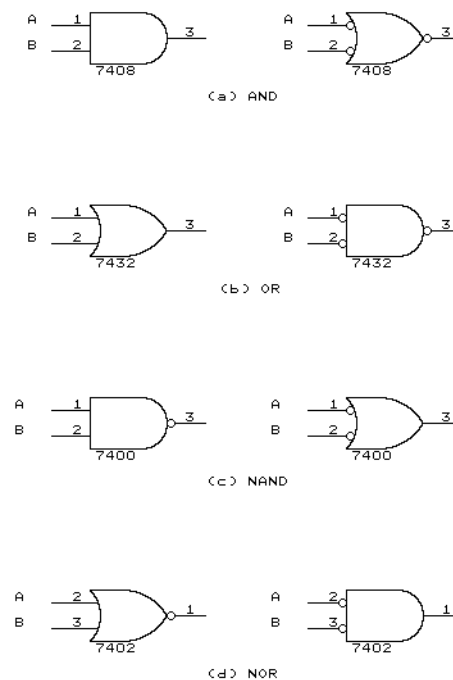


Fig. 8-2. Logic gate equivalents.

It's important to understand that both symbols in Fig. 8-2 (a) are really the same device - they are just different on paper because they stress a different function. A designer might use either symbol for the same device, depending on what *that particular circuit* is supposed to do. An IC manual, for example, describes a 7408 as a "quadruple 2-input positive-AND gate". This means the 7408 has four 2-input gates which act as "positive-AND". Remembering that 'positive logic' is just another phrase for 'active high', this says that the 7408 is used for ANDing in an active high circuit. But the 7408 could just as well be called a "negative OR" (though most IC manuals assume the reader already knows that and hence don't bother to use those words.)

In the same way, Fig. 8-2 (b) shows a "positive-OR" at the left, and a "negative AND" at the right. In reality, both of these could be used for the same 7432 IC. The left circuit says that "if A is high OR B is high, then the output is high", whereas the right circuit says that "if A is low AND B is low, then the output is low." Both of these sentences say the same thing.

If you think of the bubble as being the word "low", whereas the lack of a bubble means "high", then the left circuit in Fig. 8-2 (c) means "if A is high AND B is high, then the output is low", whereas the right circuit says "if A is low OR B is low, then the output is high." In reality, this is again saying the same thing. Both of these symbols could apply to the same 7400 NAND gate even though the right symbol is really doing an ORing function - if you assume that the inputs are active low, then if either input is asserted low, the output is asserted high.

Likewise, the left circuit in Fig. 8-2 (d) means "if A is high OR B is high, then the output is low", whereas the right circuit says "if A is low AND B is low, then the output is high." Again, this is really saying the same thing. Both of these circuits could apply to the same 7402 NOR gate.

All of this may sound strange to you, but as we go on, you will see that these new symbols greatly help to explain how some parts of the SK68K computer work.

(End of Detour, and back to the $\overline{\text{MAP}}$ circuit.)

When the 68000 first starts operating after it is turned on, it has to know (a) where to place its stack, and (b) where to find the very first instruction to perform. It looks for these two addresses in the first eight bytes of memory, starting at address 0.

In most 68000 computers, however, address 0 is the beginning of RAM; moreover, the RAM is empty when the computer is first turned on. How do these two addresses get there then? The solution is usually to doctor up the address decoder so that at the very beginning it is the ROM, not the RAM, that is at location 0. The ROM is there just long enough for the 68000 to get its two addresses, after which the address decoder removes the ROM and replaces it with RAM. Such a ROM is sometimes called a *phantom*. The MAP circuit of Fig. 8-3 tells the address decoder when to switch in RAM or ROM at address 0.

U90 is an eight-stage shift register - a group of eight flip-flops connected so an input applied to the A and B pins shifts through the register, one flip-flop at a time, as it receives clock pulses. When the computer first starts, or each time it is reset, all the flip-flops in the register are cleared by the $\overline{\text{RESET}}$ signal. The QD output, which is normally jumpered out through J25,

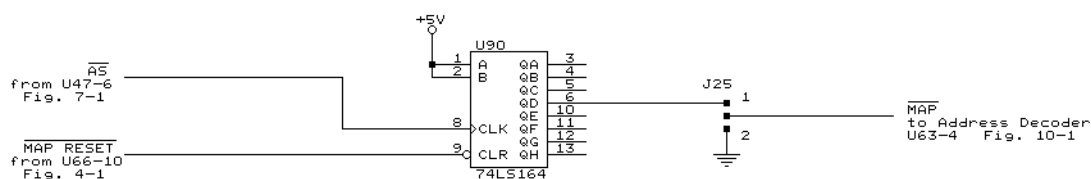


Fig. 8-3. Circuit to generate MAP signal.

then provides a low $\overline{\text{MAP}}$ signal which tells the address decoder to put the ROM at address 0. (In normal operation, there would be a jumper from the center pin of J25 to pin 1.)

When the 68000 starts, it fetches the stack address and program start address from ROM in four 16-bit reads; each one of these reads is accompanied by an $\overline{\text{AS}}$ address strobe. At the same time, the positive edge (i.e., the trailing edge) of each $\overline{\text{AS}}$ strobe clocks U90. Since the A and B shift register inputs are connected to a high (+5 volts), each clock pulse shifts a high into the register. After the first clock pulse, that high gets to QA; after the second pulse it gets to QB, and so on. After exactly four $\overline{\text{AS}}$ strobes, that high gets to QD, out the MAP lead to the address decoder, and tells the decoder to disconnect the ROM from address 0 and substitute RAM instead of ROM.

In normal operation, therefore, when the computer first starts, the $\overline{\text{MAP}}$ signal is low and therefore the 68000 sees ROM at address 0. After four $\overline{\text{AS}}$ pulses - exactly long enough for the 68000 to read eight bytes of data from the ROM - $\overline{\text{MAP}}$ goes high and address 0 becomes RAM instead of ROM.

In the SK68K, however, the RAM used at address 0 is dynamic RAM. If we want to use the computer without the DRAM, then we must disable the $\overline{\text{MAP}}$ circuit and keep ROM at address 0. This can be done by jumpering the center pin of J25 to position 2, which keeps $\overline{\text{MAP}}$ low at all times.

8-2. Construction

Reminder:

There should still be two jumpers on your board, left over from Chapter 7: one connecting U47 pins 14 and 22 to negate the 68000's $\overline{\text{BERR}}$ signal, and one connecting U66 pins 1 and 14 to assert the 68000's $\overline{\text{DTACK}}$ signal. You should also still have the two sets of Molex pins plugged into the U21 and U27 sockets to provide zeroes on the data bus. Leave these until we tell you to remove them.

Now install the following:

U90 74LS164 and its socket

J25 three-pin header. Place a shorting plug from the center pin to position 1.

U66 is already installed from a previous step.

8-3. Testing

Turn on the power and verify that the $\overline{\text{MAP}}$ signal on the center pin of J25 goes low while you short the reset pins (J23), and goes high about a second later, at the same time as the HALT LED goes off. (Unless you have a good quality oscilloscope, it is very difficult to verify that exactly four $\overline{\text{AS}}$ pulses go by before $\overline{\text{MAP}}$ becomes high, so we will have to take it on faith.)

