LAB NOTES

IN PURSUIT OF THE WILD QuASH

by John S. Simonton, Jr.

Now that we have a way to interface our synthesizers to computers - the 8780 D/A - we can begin thinking of ways to independently control large numbers of musical elements simultaneously. Lots of VCOs, lots of VCFs.

The first time that you think of this your reaction may be something like:

WOW! - ALL THOSE D/As.

Multiple D/As (one for each control "channel") would be a possible way to go. An expensive way - at $35.00 each, controlling just 4 VCOs means almost $150 worth of just D/As.

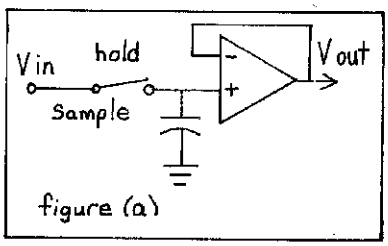
There's a much cheaper way.

You may find this a little circuitous, after working so hard at our digital interface, but we're going back to analog Sample and Hold circuits.

Now wait, don't panic. These S/H's are nothing like the ones that we're accustomed to. They don't have to hold a voltage steady for a long period of time - only a few milli-seconds. Long before even that short time has passed we will have used the computer to come back and re-write the correct voltage into the circuit. Computer re-freshed S/Hs.

Magic!

When you're designing a S/H to be good for only a fractional part of a second it gets really easy. Like this:



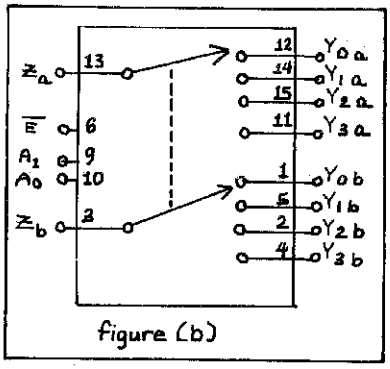
I'm sure that we've all seen this kind of thing before. It's an op-amp used as a unity gain voltage follower.

When it comes time to take a sample, the switch closes causing the capacitor to charge up to the input voltage. The output of the voltage follower "follows" this voltage (what else?), and when the switch opens again, the capacitor "remembers" the voltage.

One of the characteristics of this circuit is that the "+" input represents a very high input impedance to any load that it sees. A relatively small capacitor can accurately hold a voltage for almost a second.

Now, we're not going to use a mechanical switch here. Last time, we looked at the 4051 multiplexer and decided that we would be using it a lot. And we are, just not this time.

This time, we're going to use a very close relative of the 4051 - the 4052 (I defy you to get any closer than that). The 4052 looks like this:



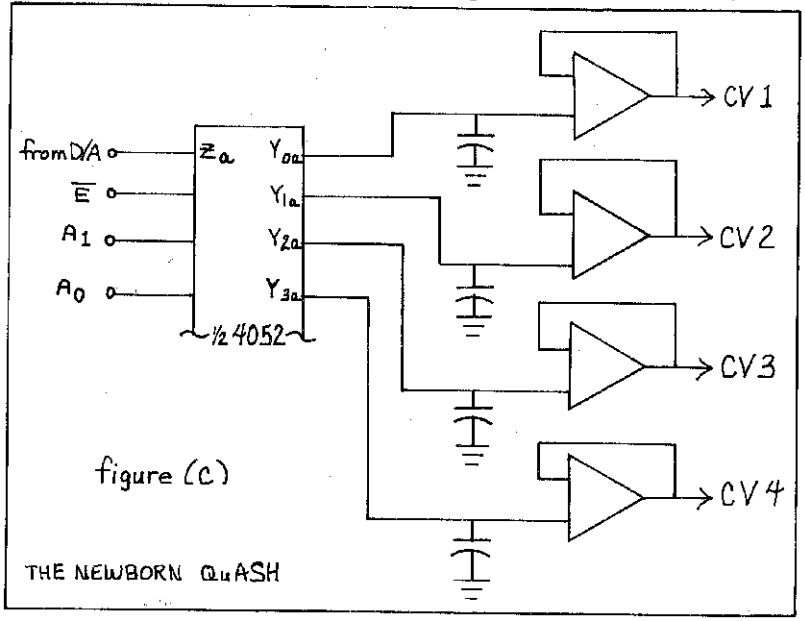
and whereas the 4051 was an electronic equivalent of a Single Pole Eight Throw switch, the 4052 is like a Double Pole Four Throw one.

Which pairs of switches are to close is specified by the two address lines (A_0 & A_1). The switches actually close when the $\overline{E}$ pin goes to ground.

Using 1/2 of one of these devices we can come up with a Quad Addressable Sample and Hold (QuASH?) that looks like figure C, and it works about the way that it looks. An address applied to the A_0 and A_1 pins sets up one of the four switches and when the $\overline{E}$ pin is taken to ground that switch closes connecting the output of the D/A to the selected S/H. Simple.

That takes care of our control voltage output - but there are still other things to think about. For instance, we need a trigger flag (gate signal) to go along with each of the control voltages to take care of things like triggering envelope generators. *(1)

An easy way to handle this is to use the other 1/2 of the 4052 to route one of the two trigger flags available from the D/A to an output corresponding to the control voltage output. And since we're time sharing the D/A we also need some way to hold the status of that flag during the times that other control



channels are being addressed. Do latches come to mind? Forget them - in this application they're going to be far too expensive and complex by the time we get them to act the way we want.

Instead, we'll use a small capacitor and a CMOS inverter like this:

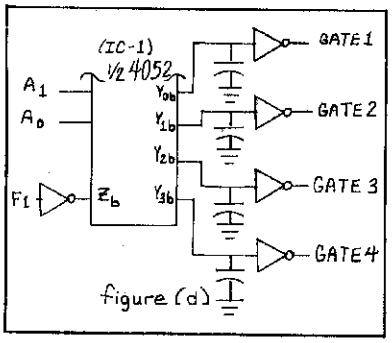


figure (d)

This is a little S/H in its own right - but it doesn't hold an analog voltage, only a "1" (output high) or "0" (output low).

Oh, yes - since we are buffering the condition of the capacitor with an inverter, we need to also invert the trigger line going into the 4052 so that everything comes out right. That's why that other inverter goes between the trigger flag line from the D/A (F_1) and the Z pin of the 4052.

But, there are two trigger flags available from the D/A - and here we a are only using one of them. Waste, ugh.

Let's do something neat with the left over flag, something really sexy. Let's use it to:

SELECT GLIDE

(tah-dah)

You may think that because we're time sharing the D/A we've eliminated the possibility of doing things like this, but we haven't. In a functioning system the S/H's are being up-dated so fast that we can in fact generate glide the same way that we did in our old pure analog system, simply by placing a variable resistor in front of the holding capacitor. We'll use a regular 4066 Quad Bilateral Switch to turn the glide off by shorting out the resistor (so that the glide is on when the switch is off), and to latch the status of this glide bit we'll use the same capacitor/inverter trick that we used on the other flag. One section of this circuitry looks like figure e.

For programming reasons, it will be handy to have the glide select bit (which is now flag 2) be a "1" when the glide is enabled and that requires a second inversion - between the trigger output of the D/A and the Z pin of this new 4052.

And now here we are with 1/2 of a 4052 left over.

Don't you believe it.

Since we will frequently have more than a single synthesizer module controlled from one of our control voltage outputs (two VCO's or a VCO and VCF would be two typical cases), it will be handy to have a modulation input associated with each control channel so that all modules driven from that channel will experience the modulation at the same time.

Another thing that ties into this is that our D/a is an exponential converter of sorts and so for the first time gives us the opportunity to do equally tempered vibrato (for example) with our linear oscillators.

We'll use the left over section of 4052 to multiplex a modulation voltage back into the D/A in the same way that we multiplexed the control voltage out. Like this:

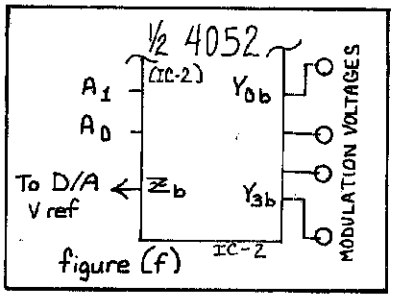


figure (f)

Because the modulation voltage corresponding to a given control channel is applied to the D/A only when that channel is re-freshed, you may think you will be able to hear the modulating influence as a series of steps. But you don't for the same reason that the glide doesn't appear to be a series of steps. Everything is just happening too fast.

One last detail and we're done with the design of this circuit.

Addressing (selecting) one S/H out of the four on this card is of course handled by the address pins of the 4052's. But, many systems will not stop at just 4 outputs; some folks, I'm sure, will want to take the system to the limit (in practical terms about 32 outputs) - which implies that more than one of these cards may (and probably will) be used in a system. We need a way to be able to select not only one of the four outputs on this card, but also a means of selecting one card from many.

Here's the address decoding scheme we'll be using:

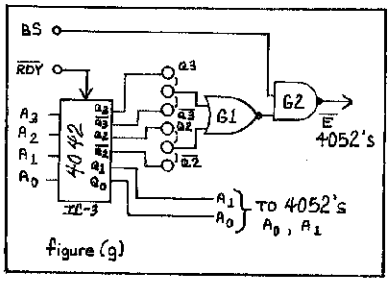


figure (g)

The 4042 Quad Latch is an old friend - here we' re using it to latch the computer's 4 least significant address bits at the same time that data is put out to the D/A (the RDY line on this card is connected the same as the corresponding line on the D/A).

We want to latch these address lines because the WRITE cycle of any computer we come up with is going to be much shorter than the time required for settling of the D/A and S/Hs. Latching the address lines allows us to output

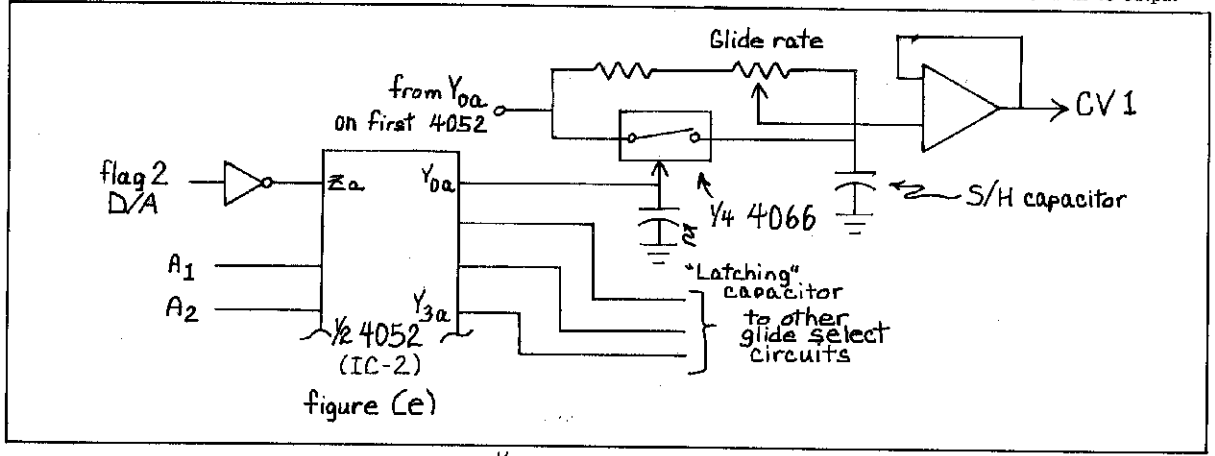


figure (e)

Y = a3m.Hance

data and then wait (or do something else) while these analog circuits get to where they're supposed to be. *(2)

Notice that the Q_0 and Q_1 outputs of the latch - corresponding to the two least significant address bits - go directly to the 4052's where they serve to select one of the four outputs.

Notice also that Q_2 and its complement $\overline{Q_2}$ as well as Q_3 and $\overline{Q_3}$ from the 4042 come out to pads on the circuit board. By jumpering these outputs to the inputs of the NOR gate G1 we can determine which group of addresses the card we're working with represents.

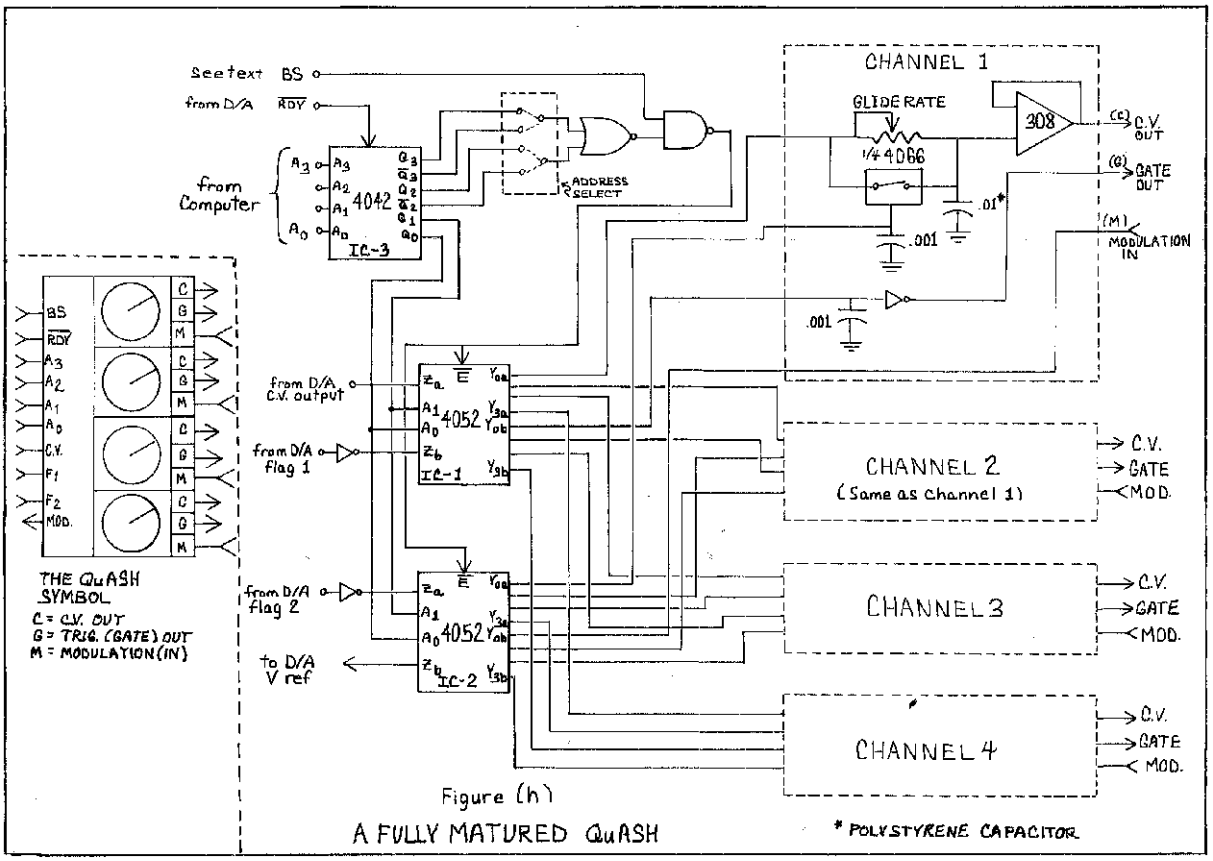
For example, if we connect the inputs of G1 to $\overline{Q_2}$ and $\overline{Q_3}$ then this block of four S/H's occupies the addresses 00XX in binary where XX represents the bits that select one of the four S/H. Address 0000 corresponds to the first S/H, 0001 to the second, and so on. By connecting the inputs of G1 to Q_2 and $\overline{Q_3}$, the S/H's occupy the address 01XX. The first S/H is 0100, the second 0101, and like that. This scheme allows us to easily use up to four of these expanders (16 outputs) in a system without needing to do anything but set the jumpers properly.

You will notice that there is another line coming out of this decoding circuit which is labeled "BS". This is not my opinion of this whole mess, it's a means by which we may expand the system beyond even four expander modules - BS stands for "Bank Select" and as long as this line is held at a logical "1" level the system operates as described to this point.

But, when the BS line is pulled low one input of the NAND gate G2 is not fulfilled resulting in its output being high which in turn holds the 4052's enabling input $\overline{(E)}$ high - which means that none of the switches in the multiplexer will close (even if addressed otherwise) and none of the S/H's will be selected.

External decoding circuitry is required to drive the BS input, naturally, but we would begin to need external circuitry at about this point anyway to buffer address lines. The decoding required here will be covered in the instruction manual for this kit.

When we tie all of these bits and pieces together, we come up with a thing that looks like figure H, our complete QuASH. And in the interest of saving space and time, we will from this point forward represent it with the symbol shown (at least until we can come up with something more abbreviated). The knobs in the output "boxes", by the way, represent the glide rate controls associated with each output channel.



NOTES:

*(1) Those of you who have been thinking about this stuff for a while will, of course, recognize the imminent demise of the ADSR. Providing Attack, Decay, Sustain and Release parameters is one of the easier tasks to turn over to the computer entirely. On the other hand, I've played with this some and can testify that varying the position of a knob is handier - in this case - than changing parameters in the memory of the machine. Some Hardware ADSRs mixed with some Software ADSRs seems a good compromise.

*(2) This off-hand statement is not meant to imply a wait in human terms (major fractions of a second), but rather a wait in machine terms - micro-seconds. You don't have to wait for a GLIDE to finish (for instance) before doing something else.

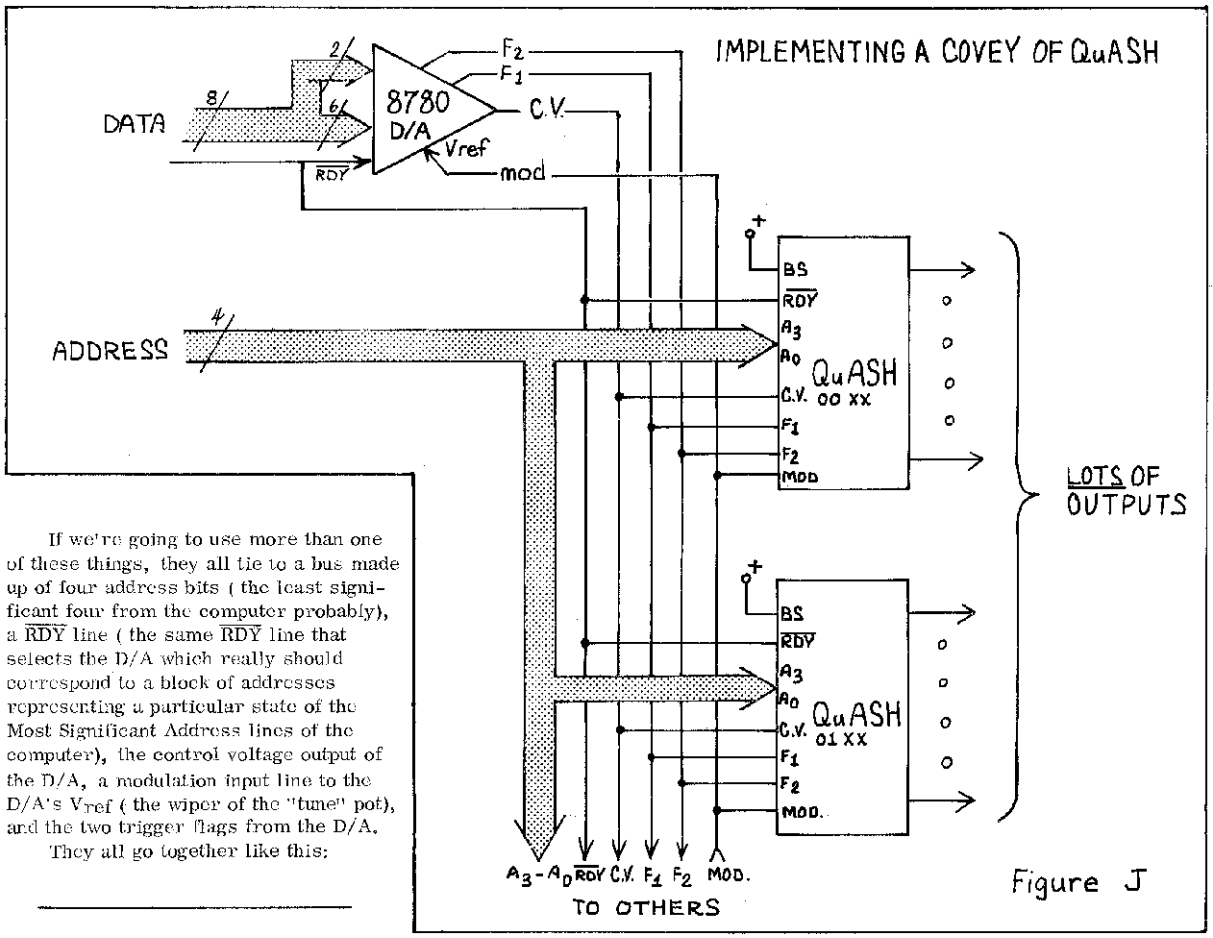


Figure J

If we're going to use more than one of these things, they all tie to a bus made up of four address bits (the least significant four from the computer probably), a $\overline{RDY}$ line (the same $\overline{RDY}$ line that selects the D/A which really should correspond to a block of addresses representing a particular state of the Most Significant Address lines of the computer), the control voltage output of the D/A, a modulation input line to the D/A's V_{ref} (the wiper of the "tune" pot), and the two trigger flags from the D/A.

They all go together like this: