https://www.ski.org/sktf-winter-1985#national

THE NATIONAL DIGITALKER
By Tom Fowle

Abstract

Use of the "National Digitalker," a chip set which produces high-quality, limited-vocabulary digitized speech output, is described. Full details on pin connections, control, and vocabulary are included.

The "National Digitalker" is a set of from two to five chips, which when properly controlled and addressed, can produce a very high quality reproduction of "digitized" human speech.

The 40-pin controller chip (MM54104) is powered from a 7 to 11 VDC supply, and controls from one to four ROM's (read-only memories) containing the digital representation of the vocabulary. The audio output pin of this controller is taken through a voltage follower, a simple R.C. filter, and an audio amplifier to drive a speaker.

Using the basic set of two ROM's, MM52164SSR1 and MM52164SSR2, a vocabulary of 143 words is available, consisting of the entire alphabet, the numbers "one" to "ten," "twenty" to "hundred," "thousand," and "million." The rest of the vocabulary is made up of a set of words having, more or less, industrial applications -- including the common electrical terms, such as "volt," "ampere," etc. Using the full set of four ROM's (which requires additional circuitry), you get 250 words total. Unfortunately for designers of talking computers, this does not easily make up the ASCII character set, but who wants a computer that talks mostly in spelled speech, anyhow?

The full numbers-vocabulary ROM, and both sets of two additional ROM's, is given later, but due to space considerations and the tendency to feel snowed with too much information, the circuitry for connecting all four ROM's is not included. This needs an extra control line and four more chips, and editorial opinion is that most applications we might run into wouldn't justify the extra complexity and expense.

The Digitalker is controlled by eight address and two control lines. A binary representation of the address associated with the desired word is impressed upon the eight address lines, and the "Write/Not" line is brought to logic low. This loads the desired address into the controller. When this line is again brought to logic high, speech begins, and the "Interrupt" line is taken low by the controller.

When speech is finished, the "Interrupt" line is brought high again, alerting the system driving the Digitalker that it is ready for another word address.

This "Interrupt" is only a signal; nothing will stop you from loading a new address and starting a new word while the previous word is still being spoken, except that the speech will be interrupted.

If the Digitalker is being driven by a computer, and if you are into the gory details of programming "exact timing loops," you might be able to build new words using parts of words already in the vocabulary. This might be cheaper than National's price of $400 per new word (this is if you are a really hot programmer and you can justify the time). (I haven't become this desperate yet, and I get paid for my programming time.) The vocabulary includes such extras as "s," "re," and several silences, making it possible to mix and match words to form phrases and arrange the timing of words for clarity. The resultant speech is of very high quality, good enough so that the naive listener usually has little trouble understanding it.

Unfortunately, the speed of the speech cannot be changed, except by varying the frequency of the controller's external crystal clock; this also changes the pitch of the speech, leading to the "Donald Duck" effect. However, since the Digitalker starts out at a male baritone, there is some leeway for this, and some users have chosen a clock frequency a bit higher than the prescribed 4mHz with good results. Since the pulse width of the signal from an external clock, which can be connected to pin 1 of the controller chip, is specified as no less than 100 nanoseconds, you are limited to a maximum frequency of 5mHz, assuming you've got a really "squarewave" clock.

Circuit Considerations

On the controller chip, pin 40 is VCC, while pin 20 is ground. VCC for this chip is between 7 and 11 VDC, and pin 40 is bypassed to pin 20 by 0.1uF. Maximum current is listed at 45mA.

Pin 3 is called "Chip Select Not," and can be taken high to "open" the input address and control lines. This is used in cases where the Digitalker is connected to a computer bus, and the address lines need to be floated while the bus is doing something else. In other words, taking pin 3 high makes the Digitalker turn a deaf ear to all of its inputs.

Pin 4 is "Write Not," and, as mentioned before, is brought low to load an address into the controller, then brought high again to start speech. In other words, this is the pin by which you "trigger" the Digitalker.

Pin 5 is "Not ROM-Power Enable," an output which can be used to control the power to the ROM's. This is used in cases of battery supply where current drain is important; the ROM's will have their power controlled by the controller.

Pin 6 is the "Interrupt Output," (equivalent to the "Busy Line" of the old TSI Speech Board); this line goes low when an address is loaded into the chip, then goes high again when speech is finished. This signal can be used to control the driver circuitry (or other controlling device), in which case it tells the driver to "Hold the phone!" while the speech is running.

Pin 7 is called "CMS," and its state controls the action of the "Write Not" line. With pin 7 low, the operation of pin 4 is as described. If pin 7 is brought high, raising pin 4 high after loading an address serves only to reset the interrupt and does not start speech. This facility is probably intended for use where the interrupt line really controls the hardware interrupt of a computer, and where the program taking care of the interrupt may not have another word to say every time the Digitalker is finished. I have found no particular use for pin 7, and I simply ground it for normal operation.

Pins 8 through 15 are the eight input address lines, with pin 8 being the most significant BIT and pin 15 being least significant. These address lines are "active high." They should never be left open. They are TTL-compatible; this means that logic low is ground and logic high is plus 5VDC. (Actually, being MOS inputs, you can take them as high as the VCC on the controller, but a 5V supply is required for the ROM's anyhow -- it's there if you want to use 5V.)

If you want to build up a Digitalker -- not to be controlled by other logic circuitry, but by switches -- you will need to run each of the eight address lines to ground through a "pull-down" resistor of 47K or so; then run each line through a normally-open switch to plus 5 volts. Closing a switch brings that line to logic 1. I did this using eight DIP switches, but toggles might be more fun, though more expensive. Then again, you could be fancy and use "hex" switches, two being needed, which have a common swinger and four digit lines. These little items have 16 positions, and give the binary representations of the numbers 0 through 15 on the four lines, with the common (or swinger) being taken to plus 5VDC in this case.

Pins 16 through 24 are the eight data lines which bring data from the ROM's to the controller, with pin 16 being called "ROM Data 1," and pin 24 being "ROM Data 8."

Pins 25 through 38 are the thirteen address lines which select location in the ROM's to be read by the controller. Pin 25 is "Address 0," pin 37 is "Address 12." Pin 38 is called "Address 13," and its use is interesting, as you shall see.

These lines which interconnect the controller with the ROM's are taken to the same pin on each of however many ROM's you use, from one to four. The proper interconnection of these pins is given in the "wire-wrap table" that appears later. At first it was hard to see how the two ROM's could be differentiated until we discovered that on each set of two ROM's, pin 38 of the controller goes to a "Chip Select" on one, and a "Chip Select Not" on the other. (As can be seen from the table, pin 38 of the controller goes to pins 20 on all the ROM's.) Thus when pin 38 is high, one of each set of two chips is selected, and when 38 is low, the other is active. As said before, using four ROM's is harder, and won't be covered here for space-and-fear considerations.

As mentioned, there will be a pin table which gives the interconnection of these pins by numbers only; this is a common way of describing digital circuitry, where there are only chips involved, and all you need to know is which pins go to which.

After the pin table, you will find the vocabularies for the three sets of ROM's. The binary code for each 16th word is given, so if you can count from 0000 to 1111 in binary, and count words between the stated addresses, just replace the four trailing 0's with your count; set that 8-BIT address into the eight input lines, and with an audio system and power in place, bring "Write Not" low and high again. No, someone didn't sneak up and sit on your bench, that's Digitalker.

Pin 39 is the audio output. National provides two filtering circuits, of which I have tried only the simpler, as it works fine. I have decided not to include the more complex filter here; it seems unnecessary, and uses a quad op-amp and many sections of R.C. filtering.

Audio Circuit (simpler, but quite adequate, filter amplifier)

The pin 39 output is buffered by a follower, made up of just about any old op-amp, say a 741. Thus pin 39 of the Digitalker goes to pin 3 of the op-amp; pins 2 and 6 of the op-amp are tied together. Pin 4 of the op-amp is grounded, while pin 7 goes to VCC (along with pin 40 of the controller). Pin 6 of the op-amp, the output of this follower, goes through 9.1K, then through 0.1uF to ground. The junction of this resistor and cap goes through 10K to the top of a 50K volume-control pot whose bottom is at ground.

The wiper of this volume-control pot is taken through 0.1uF to pin 3 of the LM386 audio amplifier. The 386 has pins 2 and 4 grounded, with pin 6 going to VCC. Pin 6 is bypassed to ground by 100uF (plus at pin 6). Pin 7 goes to ground through 25uF (plus at pin 7). Pin 8 goes through 1K to the negative side of a 10uF cap whose plus lead goes to pin 1. Pin 5 goes through 0.22uF to pin 4 (close to chip). Pin 5 also goes to the plus of a 220uF cap whose negative end goes to one side of the speaker. The other side of the speaker is grounded.

Pins 1 and 2 are the connections to the crystal oscillator whose circuit will be given presently. Pin 2 is the drain of an FET, and pin 1 is the gate. Without the crystal oscillator described, an external clock signal can be put into pin 1. They specify a required voltage swing from below 1.2V (logic low) to above 5.5V (logic high). The minimum pulse width (for both the high and low halves of the cycle) is 100 nanoseconds.

Crystal Oscillator Circuit

Pin 1 of the controller chip goes through 20pF to ground, and pin 1 also goes through 1 megohm in series with 1.5K to pin 2. The junction of the 1meg and the 1.5K goes through 50pF to ground. The 1meg resistor is shunted by the 4mHz crystal. (The specified crystal is available from Jameco as part No. CY4A.)

The ROM's are powered from plus 5V, which can be obtained from somewhere else in the circuit if it is available. However, it is common practice to put a 5V regulator on the same board as the Digitalker, powering the regulator, the controller and audio system, from the same 7 to 11 volt source.

The "Common" terminal of the regulator is grounded. The "Input" to the regulator, a uA7805, goes to VCC, and it should probably be bypassed by 100uF or so; its output should of course have 0.1uF going to ground. This 0.1 might very well be paralleled by 10uF or so just to keep ROM noise off the system.

On all the various ROM's (which are 24-pin chips) pin 24 is VCC (plus 5V), pin 12 is ground, and (you guessed it,) pin 24 is bypassed to pin 12 by 0.1uF, close to the chip.

Wire-Wrap Table

As mentioned before, all the ROM's are in parallel; that is, in sets of two. If you really want to use all four ROM's at once, I guess you'll have to complain to the author or editor and see if you can get the necessary additional circuitry out of me. This would almost certainly mean you are going to run the beast with a computer, as the hard-wired logic to control 250 words otherwise would make our old talking meter circuits look small.

Probably the single "numbers" ROM will be enough for most applications, since it contains the numbers and a "point." This should do nicely for meters and so forth. The only lack here is a minus sign, and I suggest you try a few illegal addresses; they make strange noises, and you can probably find one which makes an appropriate bleep or buzz for a minus sign. Try them anyhow; no matter what ROM's you use, they are a real kick, and include one of the greatest "spring" sounds I've ever heard.

This is a good wire wrap project, and if you have a wrapping tool which can "chain," or make more than two connections without breaking the wire, the hookup of this beast with two ROM's will be a lot easier. Don't worry about what these pins do as you wire them up, just count pins. Do so carefully! Count in one direction, then recount pins from the other end of the socket. Then, check your work against the wire-wrap table with a continuity tester. Retesting can rarely be overdone, especially when wire wrapping. If you think it's hard the first time, just hope you don't have to correct many mistakes. Try it -- it's a great way to kill time till the next good book comes along, and you will really like the finished project.

In using a pin table, sighted folks mark completed connections on the chart with different-colored ink so they don't forget what is done. Try moving a paper clip or a small paperweight around the page; it's probably a mess, but otherwise you'll have to retest everything every time you leave the job for a cup of coffee -- or just to go scream.

In this table, the first number of each set of two numbers is the pin on the 40-pin controller, and the second is the pin on one and/or two ROM's to which it is to be hooked. For example, you first see 16-9. This means that pin 16 on the controller is connected to pin 9 on one or both ROM's, depending on how many you have. In other words, when two ROM's -- of a set of two -- are used, they are wired in parallel.

16-9; 17-10; 18-11; 19-13; 21-14;

22-15; 23-16; 24-17; 25-8; 26-7;

27-6; 28-5; 29-4; 30-3; 31-2;

32-1; 33-23; 34-22; 35-19; 36-18;

37-21; 38-20.

ROM Vocabularies

Numbers ROM

This is the simplest ROM, and perhaps the most useful, especially if you want just numbers for your talking digital windmill -- or whatever. It is a single ROM containing: the numbers from "0" through "9," a "point," as well as three measured silences to be used for spacing numbers out over time. For projects containing hard-wired logic, this is definitely the ROM set of choice, since the others do not say "zero" for an address of 00000000; instead, they waste this address on advertising.

Address 00000000 (remember, this is in binary) is the word "zero." Address 00001001 is "nine," and 00001010 is "point." The next three are the silences (short to longer, respectively), ending with the address of 00001101. The rest of the possible addresses are not used. You won't hurt anything by trying them, and they often make interesting noises. The National's part number for this "numbers ROM" is MM52116SHRL, and Jameco's catalog number is DT1055.

The first set of multiple ROM's has all the letters, numbers, and many commonly used words. National's part numbers for this set are MM52164SSR1 and SSR2. Jameco has them as DT1053. A 1kit including these ROM's and the controller is DT1050. This set costs about $35.

One annoying feature of this set is that "0" is not at an address of 0; a little advertisement is spoken instead; namely, "This is Digitalker" is spoken in a female voice, whereas the rest is in a male baritone. (Chauvinists.)

The vocabulary for this set, beginning with a binary address 1, is given below. To save space, the table is broken up into groupings; the binary number heading each group is that which gives you the first word in the group. For example, the first group (speaking just numbers) starts with an address of 00000001, which sets this "chip set" up for speaking "1." In the third group, for example, an "A" is spoken for the address of 00010000; by counting in binary, you can see that the letter "J" would be gotten with an address of 00101010.

- 00000001:
  - 1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13; 14; 15.
- 00010000:
  - 16; 17; 18; 19; 20; 30; 40; 50; 60; 70; 80; 90; hundred; thousand; million; zero.
- 00100000:
  - A; B; C; D; E; F; G; H; I; J; K; L; M; N; O; P.
- 00110000:
  - Q; R; S; T; U; V; W; X; Y; Z; again; ampere; and; at; cancel; case.
- 01000000:
  - cent; parenthesis; 400Hz tone; 80Hz tone.
- 01000100:
  - silences (in milliseconds) of 20; 40; 80; 160; 320.
- 01001001:
  - centi; check; comma; control; danger; degree; dollar; down.
- 01010000:
  - equal; error; feet; fuel; flow; gallon; go; gram; great; greater; have; high; higher; hour; in; inches.
- 01100000:
  - is; it; kilo; left; less; lesser; limit; low; lower; mark; meter; mile; milli; minus; minute; near.
- 01110000:
  - number; of; off; on; out; over; parenthesis; percent; please; plus; point; pound; pulse; rate; re; reader.
- 10000000:
  - right; ss; second; set; space; speed; star; stop; than; the; time; try; up; volt; weight.

The second set of multiple ROM's available -- and probably the least useful, unless you have some complex industrial process to monitor -- is the MM52164SSR5 and SSR6. The Jameco number for this set is DT1057. Its vocabulary follows:

- 00000000:

    - abort; add; adjust; alarm; alert; all; ask; assistance; attention; break; button; but; call; caution; change; circuit.
- 00010000:
    - clear; close; complete; connect; continue; copy; correct; date; day; decrease; deposit; dial; divide; door; east; ed.
- 00100000:
    - ed *1; ed *2; ed *3; emergency; end; enter; entry; er; evacuate; exit; fail; failure; farad; fast; faster; fifth.
- 00110000:
    - fire; first; floor; forward; from; gas; get; going; (oh really?) half; hello; help; hertz; hold; incorrect; increase; intruder.
- 01000000:
    - just; key; level; load; lock; meg; mega; micro; more; move; nano; need; next; no; normal; north.
- 01010000:
    - not; notice; ohms; onward; open; operator; or; pass; per; pico; place; press; pressure; quarter; range; reach.
- 01100000:
    - receive; record; replace; reverse; room; save; secure; select; send; service; side; slow; slower; smoke; south; station.
- 01110000:
    - switch; system; test; th; thank; third; this; total; turn; use; uth; waiting; warning; water; west; which.
- 10000000:
    - window; yes; zone.

*1 This is the "ed" used with words ending in T or D.

*3 This "ed" is used with words ending in soft sounds such as "k" in the word "ask."

*2 This "ed" is used with all other words.

As mentioned before, the addresses beyond those listed in the vocabularies are not used, and produce strange and entertaining noises. These are especially useful for causing laughter when you have made a wirewrapping error.

As you can see, this system can be controlled with just nine switches, or from a computer -- or anything in between.

If you build up a Digitalker in which the "Write Not" line is controlled by a switch, notice that the switch bounce may cause repeated starting of the word. This is no big deal, and it will not happen when a proper pulse is delivered.

In the future, we hope to present some sort of talking meter interface using a Digitalker, but it doesn't look as though it will ever be very easy to make your digital "whatever" talk. In a lot of cases, it may be easier and cheaper to go from scratch rather than trying to modify existing digital-output gear.

You might find it interesting to note that many of the so-called speech synthesizer systems being sold for the simpler home computers use this chip set. They are not really speech synthesizers, as they do not *create* speech, just play it back. A future article will discuss this issue: what is a synthesizer, and which things are just solid-state, random-access recorder/player systems. They both have their uses, and we must learn which fits where.

Review of Digitalker Hookup

Just to have it all in one spot, here is an example circuit, all-inclusive:

No matter how many ROM's you have, their pin 12 is grounded and 24 goes to plus 5V. On the controller, pin 20 is grounded, while pin 40 goes to a VCC supply of 7 to 11 volts. In both cases, located close to the chips, the supply lines are bypassed to ground by 0.1uF.

The 5V is gotten from a uA7805 regulator. Its "Common" terminal is grounded. Its "Input" goes to VCC (and to pin 40 of the controller). This "Input" is bypassed by 100uF (negative at ground). The 5V "Output" is bypassed by the parallel combination of 0.1uF and 10uF (negative of the electrolytic at ground).

The controller and its ROM's are interconnected as follows:

16-9; 17-10; 18-11; 19-13;
21-14; 22-15; 23-16; 24-17;

25-8; 26-7; 27-6; 28-5; 29-4;
30-3; 31-2; 32-1; 33-23; 34-22;
35-19; 36-18; 37-21; 38-20.

Pin 39 goes into an audio system as follows:

Audio Circuit (simpler, but quite adequate, filter amplifier)

The pin 39 output is buffered by a follower, made up of just about any old op-amp, say a 741. Thus pin 39 of the Digitalker goes to pin 3 of the op-amp; pins 2 and 6 of the op-amp are tied together. Pin 4 of the op-amp is grounded, while pin 7 goes to VCC (along with pin 40 of the controller). Pin 6 of the op-amp, the output of this follower, goes through 9.1K, then through 0.1uF to ground. The junction of this resistor and cap goes through 10K to the top of a 50K volume-control pot whose bottom is at ground.

The wiper of this volume-control pot is taken through 0.1uF to pin 3 of the LM386 audio amplifier. The 386 has pins 2 and 4 grounded, with pin 6 going to VCC. Pin 6 is bypassed to ground by 100uF (plus at pin 6). Pin 7 goes to ground through 25uF (plus at pin 7). Pin 8 goes through 1K to the negative side of a 10uF cap whose plus lead goes to pin 1. Pin 5 goes through 0.22uF to pin 4 (close to chip). Pin 5 also goes to the plus of a 220uF cap whose negative end goes to one side of the speaker. The other side of the speaker is grounded.

An external 4mHz clock can be fed into pin 1 (having a peak-to-peak voltage of 6V or so). On the other hand, a crystal oscillator can be built around pins 1 and 2 as follows:

Crystal Oscillator Circuit

Pin 1 of the controller chip goes through 20pF to ground, and pin 1 also goes through 1 megohm in series with 1.5K to pin 2. The junction of the 1meg and the 1.5K goes through 50pF to ground. The 1meg resistor is shunted by the 4mHz crystal. (The specified crystal is available from Jameco as part No. CY4A.)

Pin 3, the operation of which deafens the inputs, is grounded for normal use. Pin 7, which changes the function of pin 4 (see text) is grounded for normal operation.

Pins 8 through 15 are the "Address Inputs"; a binary number is to be impressed on them. Pin 8 is the most significant BIT, and pin 15 is the least significant.

Now, bringing pin 4 low *loads the address*; bringing it high again makes it talk. During this "busy" process, pin 6 (the "Output Interrupt" or "busy line") can be observed with a voltmeter or a logic probe.

Suppose we want to test the chip set, or make a good party gimmick. Run pins 8 through 15 all through 47K resistors to ground; put toggle switches between them and 5V (or VCC, at your convenience). Next, run pin 4 through 47K to plus 5V (or VCC), and connect a normally-open pushbutton "start" switch between pin 4 and ground.

"This is Digitalker."

Supplier Address:
Jameco Electronics, l355 Shoreway Rd.,
Belmont, CA 94002 (4l5) 592-8097