VIC-20 Interfacing Blue Book

By V.J. Georgiou

VIC-20

INTERFACING

BLUE BOOK

BY V.J. GEORGIOU

TABLE OF CONTENTS

INTRODUCTION

0

The VIC-20 is designed to be a versatile tool. Color graphics, powerful BASIC, very low price and the 6502 microprocessor CPU give it a very wide spectrum of users, from the novice who wants to learn programming to the engineer who uses it as a super calculator.

Useful as these software-type applications may be, they do not address the vast and exciting possiblities of interfacing VIC to the real world and its uses as a controller of physical processes. Neither Commodore, it's manufacturer, nor the various people who have written articles about it, seem to apprehend the potential of the VIC-20 as controller.

It used to be that if you wanted a computer for control applications, you got a Rockwell AIM-65. Now that the VIC is available, you will find that it costs less than 40% of a comparably equipped AIM, it is much more user friendly and it is neatly packaged as opposed to the AIM that is not packaged at all.

One of the objectives of this book is to show you how to interface your VIC so that it can sense physical variables and control processes. The other objective is to show you how you can expand its capabilities to make it a more powerful and versatile system.

The wide variety of backgrounds of the VIC users makes it difficult to target the level of the book. The interfacing techniques presented here may be used by a high school student who builds a science project or a professor who needs to control an experiment. Hopefully the book will be useful to anyone who has (or is willing to acquire) a minimal background in electronics and can read schematics.

So the message is, if you dabble in things electronic, this book should contain enough information to successfully build at least some of the projects. Whether you can build something depends more on your determination to succeed and less on your qualifications when you start out.

Begin by surveying the projects to find which ones you would really like to build. Scan these desirable projects to find the easiest one for you to build and the one that you seem to understand best. To help you in your selection, the 10 first projects are the easiest ones to build. If you do encounter problems, don't give up, look for help. The local user's club is a good place to start.

An effort has been made to keep the writing style simple and uncluttered. There are many things that could have been included to make the book many times as thickto keep page counters happy. It was decided that it is much better to keep the writing to the point rather than drown the necessary and important into the mildly relevant.

Thus the reader is referred to other sources for additional information. The VIC-20 Programmers Reference Guide is a required companion to this book and in general to the serious VIC user. It is also a good idea to get from the manufacturer the specification sheets of the IC's used in the projects that interest you. The custmary (for this type of book) section on basic electronics is omitted Your local library will have at least a few good texts on introductory electronics.

Most of the projects presented here will interface to the user port. This simplifies building any one project but at the same time restricts the number of projects that will operate simultaneously because there is only one user port. This problem is not serious because it is relatively straightforward to add one or more equivalents of the user port (with only a slightly different addressing). Project 20 details how to do this.

For those who would like more advanced (and more complicated) interfacing projects, there will be a volume II of the Blue Book where all the projects will interface to the expansion bus.

Your comments and questions on this book and your suggestions for projects you would like to see in vol. II are welcome. I will not be able to answer any mail individually, but I will give credit to the readers whose input will be incorporated in future editions of this book and vol. II.

V.J. Georgiou　　　　Santa Barbara, June 1982

SOURCES OF MATERIALS

A good nearby source of materials is your neighborhood Radio Shack store. If you cannot find something there, check the yellow pages under Electronic Parts, Retail to see if there is a company in your area that can supply it. If you cannot find it locally or if you prefer mail order (usually you get better prices mail order and you don't pay state tax), you can get addresses of mail order firms from the back pages of magazines like Byte or Popular Electronics. When buying mail order keep in mind that if a business has been around more than a few months, most likely they are not dishonest. However you cannot judge the service you will get nor the quality of the materials until you try them or hear from somebody that has experience with them. Based on our own experiences with them, the following mail order companies are above average in quality of service and merchandise:

JAMECO ELECTRONICS
1355 SHOREWAY ROAD
BELMONT, CA. 94002
(415)-592-8097

DIGITAL RESEARCH:PARTS
P.O. Box 401247
GARLAND, TX 75040
(214) 271-2461

CALIFORNIA DIGITAL
P.O. Box 3097B
TORRANCE,CA. 90503
(800) 421-5041

DIGI-KEY CORP.
HIGHWAY 32 SOUTH
THIEF RIVER FALLS, MN 56701
(800) 346-5144

WALLEN ELECTRONICS
108 SAW TELL AVE.
BROCTON, MA 02402
(617) 588-6440

MOUSER ELECTRONICS
11433 WOODSIDE AVE.
SANTEE, CA 92071
(714) 449-2222

Wallen is an industial surplus distributor. Mouser is an importer of new parts from the Far East, probably the largest operation of its kind. These distibutors have a $20 minimum and they are used to deal with companies rather than individuals, so be ready to play by their rules. It also helps if you use a comany sounding name after your own for example: John Doe, The VICONER Co., 111 Any St, Ytown, RA 12345. Also a good idea is to use a purchase order number because if somegoes wrong its easier to trace things if the order has a number.

Most of the above companies have catalogs, write for your copy.

USER PORT BREADBOARD 1

DESCRIPTION

Many projects in this book employ the user port on th VIC. Here is a breadboarding system that lets you experiment easily, neatly and safely with the user port. All the important signals on the port are brought out to a solderless breadboard ('superstrip') so that you can build circuits and make connnections using only insulated wire - no soldering is necessary.

THEORY OF OPERATION

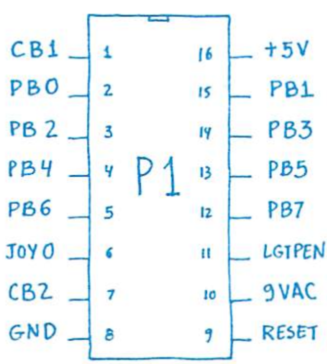
The idea behind the user port breadboard is simple. Instead of having wires flying in all directions, how about a neat arrangement consisting of a connector to the user port, a 16 conductor flat cable to bring the signals to a convenient location and a solderless breadboard to work on. Solderless breadboards like the 'superstrip' work very well and are excellent for experimenting but they are somewhat expensive for permanent circuits. When you have a circuit that works to your satisfaction you can transfer it to a wirewrap implementation or a PC board, freeing your breadboard for the next job.

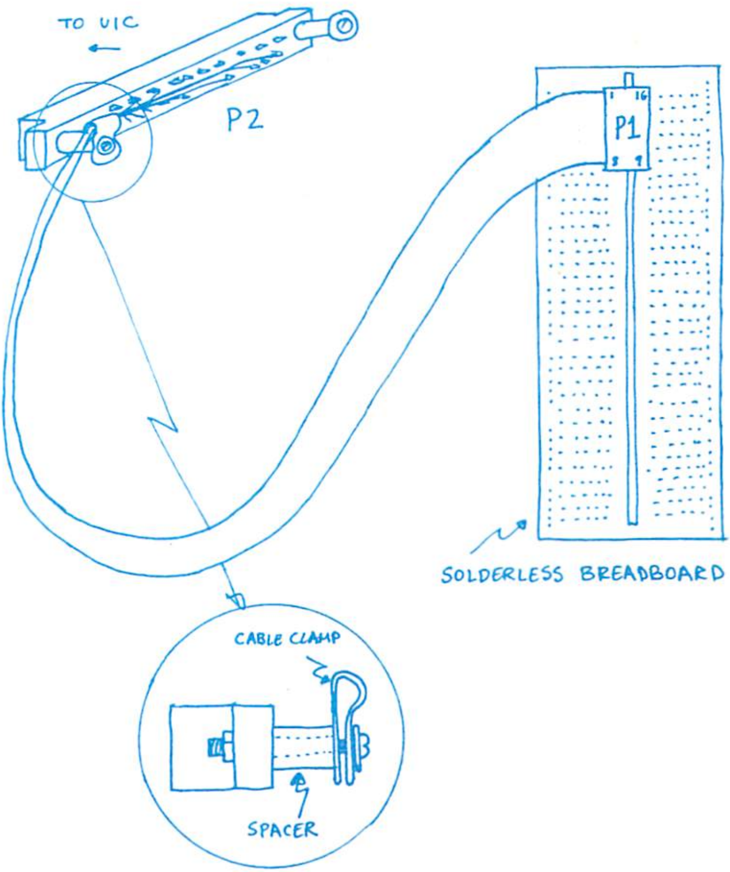
CONSTRUCTION HINTS

The parts of the breadboard go together as shown on the figure. Connector P2 has handles made out of common electronic hardware to help insertion and removal. Always push it in straight and remove by pulling straight out. Do not wiggle it during removal and do not plug or unplug it while power to the VIC is on. Pin 10 of P1 carries the +9V power from the user port. You must be careful with that voltage because most of VIC's electronics work on +5V and if you accidentally connect 9VAC to data lines you may cause damage. Instead of the 9VAC you may bring out JOY 0 (pin 4 on P2) if you don't need the 9V power.

PARTS LIST

1. 12/24 pin, 0.156'' spacing PC connector.
2. 36 inches of 16-conductor flat cable (preferrably multicolor) with DIP plug attached on one end.
3. Two each of the following: 1/2-inch stand-offs, 3/8-inch washers, 1-inch long 4-40 screws, 4-40 nuts.
4. Cable clamp for 1/4-inch cable





INTERCONNECTION GUIDE

SIGNAL	USER PORT	DIP PLUG
CB1	pin B	pin 1
PB0	pin C	pin 2
PB2	pin E	pin 3
PB4	pin H	pin 4
PB6	pin K	pin 5
JOY 0	pin 4	pin 6
CB2	pin M	pin 7
GND	pins 1 & A	pin 8
RESET	pin 3	pin 9
9VAC	pin 11	pin 10
LGTPEN	pin 7	pin 11
PB7	pin L	pin 12
PB5	pin J	pin 13
PB3	pin F	pin 14
PB1	pin D	pin 15
+5V	pin 2	pin 16

DESCRIPTION

You may find that when you tune your TV for best display the sound becomes weak or distorted or both. Or you may want to put some real power into the sound of your VIC (let's say something like 20 watts). This project tells you how to connect your VIC to any stereo for loud and clear sound.

THEORY OF OPERATION

Your VIC generates both an audio signal and a video signal available at the DIN jack at the back labeled, appropriately, AUDIO/VIDEO. These signals are normally fed into the RF modulator which converts them into a Radio Frequency signal similar to that broadcast by a TV station so that the TV set can receive them through its antenna terminals. When the TV set receives this signal it filters it, amplifies it and then decodes back the original audio and video signals available at the back of VIC. In theory, little is lost in this process and you get the audio/video signal into the TV set without having to modify the set in any way. In practice, many problems can and do crop up leaving you wishing for some form of direct connection. Fortunately this is simpler than it seems. You can feed your video and audio signals directly into a video monitor and an audio amplifier (your stereo for example) and thus avoid all problems. But you must be careful to avoid getting 110V AC (or worse) into your VIC and to do it right (i.e not disturb the connections to the RF modulator) so that your VIC can always play through the TV set. In fact, once the sound output goes through the stereo, you will be able to tune the TV for best picture and you may find that a video monitor is not needed.

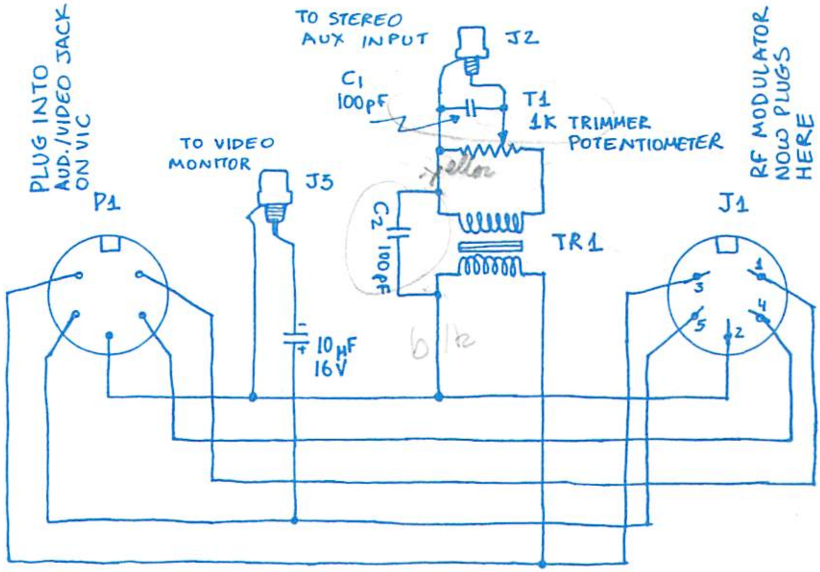
The schematic gives you the connections required. To avoid high voltages getting into the VIC, we use a transformer to isolate the stereo. C1 and C2 are used to reduce RF noise pickup. The trimmer T1 is used to adjust audio volume so that the stereo input will not overload. This is a one time adjustment. Volume normally is adjusted from the volume control of the stereo. All signals of Audio/Video jack are transferred directly (with short wires) to the Audio/Video extension jack, so operation of the RF modulator is not affected. To get good performance out of this adaptor, you must be careful with physical layout. See construction hints for more details.

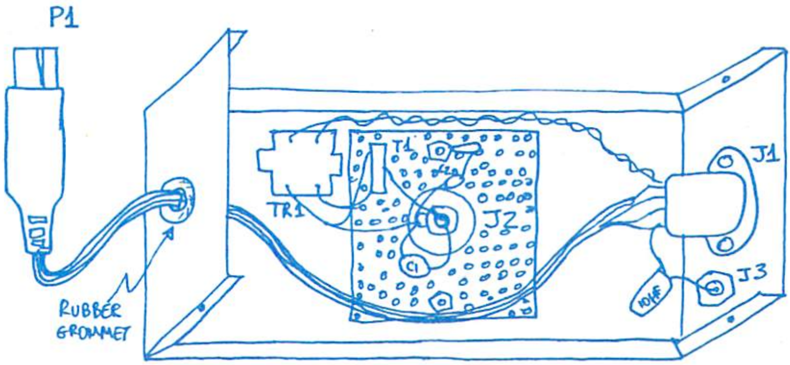
CONSTRUCTION HINTS

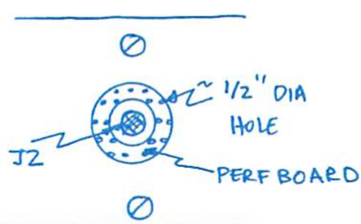
The whole unit should be housed in a small aluminum box, for example Radio Shack #270-235. If you keep wiring neat and short, a small plastic box might do as well. Use a piece of flat cable to connect P1 and J1. J2 must be isolated from the metal case, so use a piece of perf board to mount it on and drill a large (1/2") hole in the case to clear the jack. For the connection to your stereo or video monitor use shielded cable with an RCA plug in each end (a common item available at audio stores).

PARTS LIST

1. Two RCA jacks.
2. 5-pin DIN type audio plug
5. Two 100 pF ceramic disc capacitors.
6. 10 uF, 16V electrolytic capacitor.
7. Interstage transformer, 600 to 600 Ohm.
8. Wire, case, hardware.


PLUG INTO AUD./VIDEO JACK ON VIC
P1
TO VIDEO MONITOR
J3
TO STEREO AUX INPUT
J2
C1 100pF
T1 1K TRIMMER POTENTIOMETER
C2 100pF
TR1
10µF 16V
RF MODULATOR NOW PLUGS HERE
J1
1
2
3
4
5


P1
RUBBER GROMMET
TR1
T1
J2
C1
J1
J3
10µF


1/2" DIA HOLE
J2
PERFBOARD

PICKPROOF DIGITAL LOCK

DESCRIPTION

A keypad, and a few resistors and capacitors interfaced to your VIC can send master safecrackers to the unemployment line. In addition, changing the combination to this pickproof lock is as easy as changing a BASIC line.

THEORY OF OPERATION

There are three ways to open a combination lock: 1. Use the key combination (obtained by legal or illegal means). 2. Go through all combinations one at a time. 3. Inspect the lock to figure out the combination. The last two methods are called 'picking the lock' (from the use of picks on key locks). Method #3 is the one used by movie safecrackers on bank vaults where equipped with stethoscopes (or more recently, unspecified digital gizmos) they twirl the dial listening for the telltale clicks.

In our pickproof lock, there are too many combinations (one million) so method #2 will not work. In addition, every time a wrong combination is entered, the lock will not respond for 30 seconds (half a million minutes is a long time). Method #3 is also useless because the keypad and the lines to the keypad do not contain information about the combination, nor can the VIC be interrogated in any way through these lines.

In operation, the user port data lines scan the keyboard matrix to determine which key (if any) has been pressed. When the correct combination is entered, CB2 is used to activate a solenoid to unlock the door. The scanning proceeds as follows: Each of the four least significant bits of the user port is made a '0' and then the four most significant bits are checked to determine if a '0' is present, indicating that a key has been pressed. This information is translated into the value of the key pressed using a table (KTABLE). If no key is pressed, the scanning continues.

The resistors in the circuit are there to insure that the inputs are zero without signal and the capacitors help filter out noise picked up by the lines.

PROGRAMMING NOTES

The lock program is written in BASIC because execution speed is sufficient for real time operation and BASIC is much easier to use. To unlock, you enter on the keypad the combination (variable CKEY, line 240). The solenoid will be activated for 20 seconds to unlock the lock and then released to lock again. The keypad arrangement is as follows:

```
1 2 3 .
4 5 6 .
7 8 9 .
. 0 . .
```

Only 10 keys are used, arranged like the keypad on a pushbutton phone. Any other arrangement of 16 keys is possible by changing the KTABLE. The layout of the KTABLE corresponds to that of the keyboard. Unused keys are given codes to detect illegal key pushes.

The key combination in the listing is '111111' to facilitate testing. You need only connect one key to check operation of the lock.

Line 515 prevents a keypress from registering more than once if a key is held pressed down. To enter the combination, keys must be pressed and released six times.

If you would like to increase the key combination to more digits, change line 240 to reflect the new key and also change line 350 to accounts for the increased digits. For example, for an 8 digit key change N<6 to N<8.

Line 380 determines how long the solenoid remains activated. You may change the constant (now 400) to suit your application.

For master key operation, change line 240 to read:

```
240 CKEY=111111: MKEY=222222
```

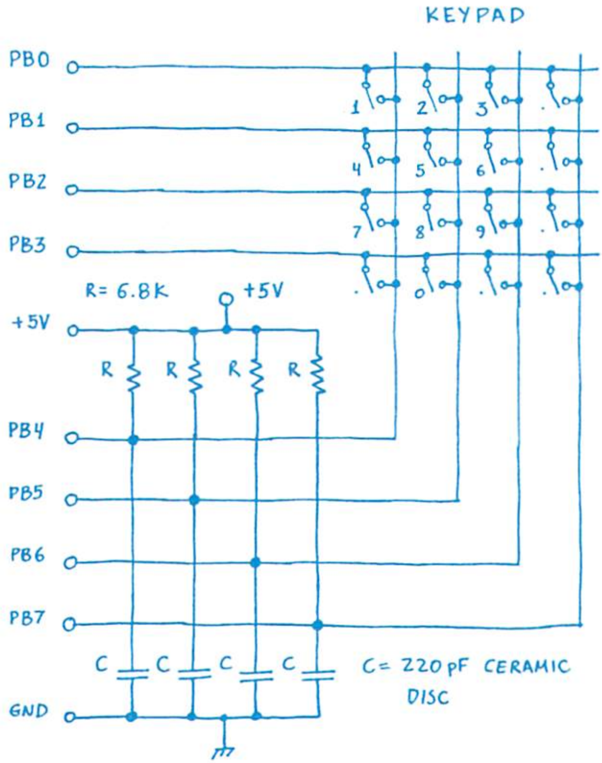
and line 360 to read:

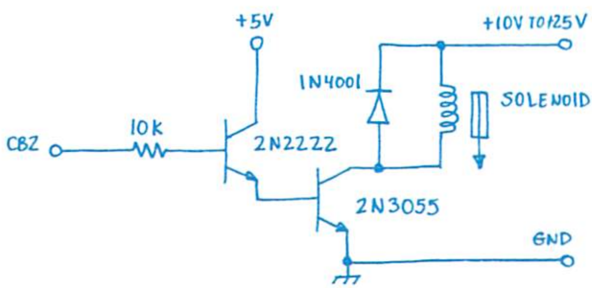
```
360 IF INT(CODE)<>CKEY AND INT(CODE)<>MKEY GOTO 700
```

Now CKEY is the regular key combination and MKEY is the master. The values assigned here are shown for easy testing, but keys can be any six digit number.

PROGRAM LISTING
===============

```
100 REM ** PICKPROOF LOCK **
105 REM
110 REM SET UP KEY TABLE
115 REM
120 DIM KT(4,4)
130 KT(0,0)=1:KT(0,1)=2:KT(0,2)=3:KT(0,3)=13
140 KT(1,0)=4:KT(1,1)=5:KT(1,2)=6:KT(1,3)=12
150 KT(2,0)=7:KT(2,1)=8:KT(2,2)=9:KT(2,3)=11
160 KT(3,0)=14:KT(3,1)=0:KT(3,2)=14:KT(3,3)=10
195 REM
200 REM SET UP PORTS
205 REM
210 PRT=37136:PCR=37148
220 POKE37138,15
230 POKEPCR,((PEEK(PCR)OR192)AND223)
240 CKEY=111111
295 REM
300 REM LOCK OPERATION
305 REM
310 CODE=0:N=0
320 GOSUB 500
330 IF KPRESS >9 GOTO 700
340 CODE=CODE+KPRESS*(10^N)
350 N=N+1: IF N<6 GOTO 320
360 IFINT(CODE)<>CKEY GOTO 700
370 POKE PCR,(PEEK(PCR))OR32
380 FORI=1TO400:A=2^2:NEXTI
390 GOTO 230
395 REM
500 REM KEYSCAN SUBROUTINE
505 REM
510 SB=1:I=0
515 IF (PEEK(PRT)AND240)<>240 GOTO 515
520 POKE PRT,15-SB
530 Q=PEEK(PRT) AND 240
540 IF Q <> 240 GOTO 570
550 SB=SB*2:I=I+1
560 IF SB < 16 GOTO 520
565 GOTO510
570 IF Q=224 THEN J=0: GOTO 620
580 IF Q=208 THEN J=1: GOTO 620
590 IF Q=176 THEN J=2: GOTO 620
600 IF Q=112 THEN J=3: GOTO 620
610 GOTO 700
620 KPRESS=KT(I,J)
650 RETURN
655 END
695 REM
700 REM ERROR DELAY
705 PRINT"ERROR"
710 FORI=1TO900:A=2^2:NEXTI
720 GOTO 300
```





CONSTRUCTION HINTS
==================

Use a flat cable to connect the user port and the keypad. The shorter cable the better. You may experience noise problems with cables longer than 40 or 50 feet. If the keypad is installed outdoors either weatherproof it or shield it from the weather. Install the resistors and capacitors close to the VIC. You should be able to mount them right on the user port connector.

ADDED FEATURES

The program presented here performs only the basic lock function. It is easy to add various nice features to the basic lock, for example you can add a master key as described above. Another feature will be time-controlled access. Using the TI variable you can make the lock respond to certain keys only certain times of the day. For example if the lock controls the power to a TV set, you could use it to determine when and for how ong your children can watch TV. Still another use is logging of usage of lock. If every employee has his own code to enter and exit the premises you will know when he is there and when he is not. Finally , if you would lke to get really advanced you can write the lock program in machine language and have a real time interrupt which will check the keypad every lets say 100 ms while the VIC is used in its normal mode.

GETTING SOPHISTICATED

The digital lock could still be picked if one could access radio emmissions from the VIC and decode its operation. A remote chance of course but paranoia is a necessary ingredient of a high security opration. In this case good shielding of the VIC is the recommended precaution. Place the VIC in a cage made of fine mesh metal screen, soldered at the seams. The case of the VIC should be at least 5'' from the wire cage for effective shielding. Also wires going to the lock should be passes through ferrite beads just otside the case.

Continuous operation is requirement for a lock, so provisions must for accidental power downs. An uninterruptible power supply based on batteries will maintain operation during power down. The lock is designed to remain locked in a power failure.

Finally, provisions should be made to break into the guarded area when the lock fails (for example the VIC malfunctions) with minimum damage and cost, while unauthorized forced entry is very difficult.

And always keep in mind that locks are almost as old as civilization and lockpickers and safecrackers have been successfully practicing throughout history and they may profitably disagree with you on what is pickproof and what itsn't.

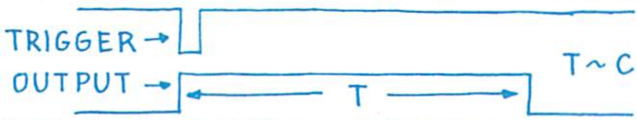
CAPACITANCE METER 4

DESCRIPTION

This meter will allow you to measure capacitances between 100pF and 50µF, yet is only uses 3 components (and your VIC which does most of the work). You will be surprised at the accuracy and stability of this little circuit, which can, in most cases, replace a standalone instrument costing as much as the VIC.

THEORY OF OPERATION

The capacitance meter operates by measuring how long it takes to charge a capacitor to a predetermined voltage, while a known current is applied. A 555 timer IC in the monostable mode is used to perform the functiion of applying the current and determining if the thresh-hold has been reached. The VIC is used to initiate a measurement and count time. The accuracy of the meter depends on the 555 which was chosen for its exceptional stabilty characteristics. The VIC initiates a measurement cycle by bringing low then high the least significant bit of the user port (PB0). This triggers the 555 and its output goes high for s long as it takes to charge the unknown capacitor to 2/3 the supply voltage. The VIC measures the duration of this output pulse using a machine language program to obtain high resolution (25us per count). Here is a timing diagram of the process:



The amount of current flowing in the unknown capacitor is determined by R. 470K will give good results for unknown capacitors from 100pF to 1µF. For values over 1µF and electrolytics (that have high leakage currents) R should be changed to 2.2K

PROGRAMMING NOTES

The program listed will drive the capacitance meter and print out on the screen the value of Cx, the unknown capacitance. The printout stops when Cx is disconnected and resumes when a Cx of 100pF or more is connected.

The program is all in BASIC form to help entering and running. The DATA statements contain the machine language driver (see next project for a listing of the machine language program).

Statements 100 to 190 set up the machine language program at a reserved location at the top of memory statements 200-240 measure and print out the capacitance.

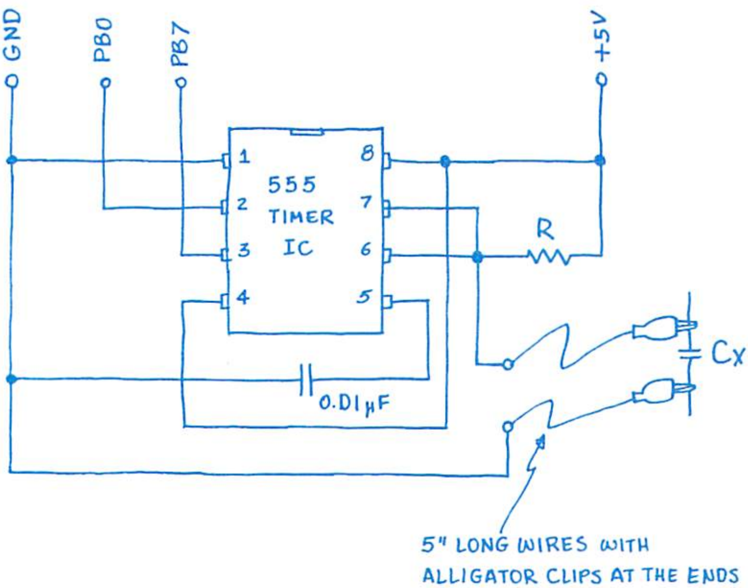
CALIBRATION

The program as given will print out capacitance in nF (1 nF = 1000 pF = 0.001 µF) for R = 470K. You should calibrate your circuit after building it. Calibration is based on a known capacitor with a value over 5 nF. Lets say you have an 10 nF (0.01 uF) capacitor with 2% tolerance. Connect it to the meter and adjust the constant in line 230 of the BASIC program (in the listing it has the value of 2364) until the printout is as close to 10uF as possible. Your capacitance meter will now be calibrated with an accuracy equal to that of your calibrating capacitor (2% in this case). To calibrate for another range (R = 2.2K) adjust the constant likewise against a known capacitor. Recalibration is required only infrequently, approximately once every six months.

PROGRAM LISTING

```
100 REM CAPACITANCE METER
104 REM
106 POKE 37138,1
110 Q1=PEEK(55)+256*PEEK(56)
120 Q2=Q1-40
130 Q3=INT(Q2/256): Q4=Q2-Q3*256
140 POKE 55,Q4: POKE 56,Q3
150 FORI=1TO35
160 READ A: POKE Q2+I,A: NEXT I
170 Q6=Q2+1: Q3=INT(Q6/256)
180 Q4=Q6-Q3*256
190 POKE 1,Q4:POKE 2,Q3
200 A=USR(0)
210 COUNT=PEEK(136)+256*PEEK(137)
220 IF COUNT<2 GOTO 200
230 PRINT (COUNT/2364)*100;"NF"
240 GOTO200
250 DATA 120,169,0,141,16,145,133
260 DATA 139,133,140,169,1,141,16
270 DATA 145,165,139,24,105,1,133
280 DATA 139,165,140,105,0,133,140
290 DATA 173,16,145,48,238,88,96
300 END
```

SCHEMATIC DIAGRAM



DESCRIPTION:
============

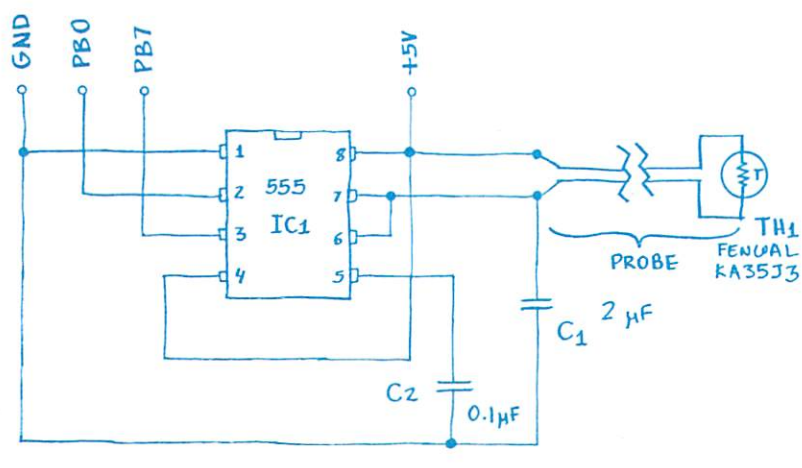
With a little imagination, the uses of a thermometer for your VIC are unlimited. Energy control of your home. Making the perfect yogurt. A fire detector. A plot of your body temperature during sleep. All it takes to add temperature sensing to your VIC is four components.

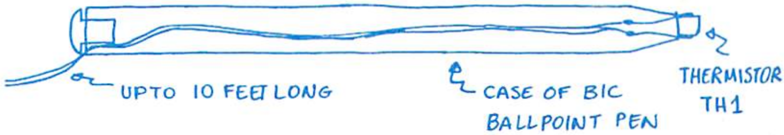
THEORY OF OPERATION
===================

A 555 timer is used as a stable and inexpensive A/D converter to read out the value of a thermistor. The circuit is essentially the same as the one used in the capacitance meter. Instead of Cx a 2 uF capacitor is used and R is replaced by a thermistor. The output pulse width of the timer varies with the temperature, and is measured by a machine language program in the VIC.
Unfortunately, the output width is not linearly related to temperature. Thus some means of linearization must be employed. The simplest way, achieves both calibration and linearization in one step. A table (array T) is set up so that for each count there is a corresponding temperature. The correspondence is found by immersing both the thermistor and a thermometer in a pot full of water whose temperature is varied by heating it or cooling it (using ice). Other linearization techniques include piecewise linear approximation and polynomial curve fitting.
To get the pulse length (count) for each temperature required to build the table use the program for the capacitance meter and change line 230 to read: 230 PRINT COUNT.

CONSTRUCTION NOTES
==================

Use a flat cable pair of flexible wire no longer than 10 feet to connect the thermistor to the circuit. Make a probe for the thermistor out of the housing of a ball point pen as shown. Apply '3M super strength household cement' or equivalent, around the thermistor to seal it to the body of the pen. Use the plug that capped the back of the pen to jam the cable and keep it from moving.





MACHINE LANGUAGE PROGRAM LISTING
================================

```
                        CTR=139
                        PORT=37136

        78              SEI
        A900    TRIGG   LDA #0
        8D1091          STA PORT
        858B            STA CTR
        858C            STA CTR+1
        A901            LDA #1
        8D1091          STA PORT
        A58B    COUNT   LDA CTR
        18              CLC
        6901            ADC #1
        858B            STA CTR
        A58C            LDA CTR+1
        6900            ADC #0
        858C            STA CTR+1
        AD1091  CHECK   LDA PORT
        30EE            BMI COUNT
        58              CLI
        60              RTS
```

PROGRAMMING NOTES
=================

The first thing the timing subroutine does is to disable interrupts. Unless this is done, the count will be erratic. Locations 139 and 140 are used to store the count. Maximum count is 65536.

DESCRIPTION

If your VIC can't tell day from night, don't be alarmed. It has no eyes but its birth defect can be corrected rather easiy. This project tells you how to connect a photocell to your VIC so that it can sense light.

THEORY OF OPERATION

The resistance of a photo conductor like CdS (Cadmium Sulfide) changes with incident light. Photocells are packaged pieces of photo-conductor material with leads attached. A typical photocell would have a resistance of 500 kiloOhms in the dark and less than a thousand Ohms under sunlight.

By using a 555 timer circuit and by replacing the timing resistor with a photocell, we can convert variations of light to variations of pulse width which the VIC can measure. The circuit is essentially that of the previous project with the thermistor replaced by a CdS photocell.

The variation of the resistance of the photocell is not a linear funtion of incident light. It is possible of course to correct this using the VIC and to calibrate the system against a known light source. In addition, using a thermometer circuit, the ambient temperature can be measured and used to correct the readings (the resistance of the photocell also varies with temperature). Thus it is possible to build a very accurate light meter using simple circuits and the VIC.

PROGRAMMING

The program is essentially that used in the capacitance meter and the thermometer. The count obtained by calling the machine language program can be directly printed on screen or used for further calculations. Since most artificial light sources are modulated by 60Hz, you may need to average 5 to 10 measurements to get a steady reading.

APPLICATIONS

There can be many innovative applications for the light sensing system. Here are a few mundane ones:

1. Turn on outside lights when darkness falls.

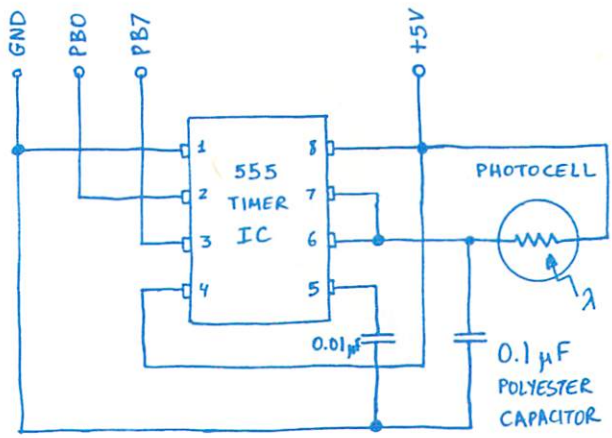
2. Detect interruptions of light beams. Using a photocell illuminated by a light source placed some distance from it you can detect people entering or leaving an area for example. Or a model railroad approaching a crossing. Or an opaque liquid rising in a tank.

3. Make a light seeking robot. Place the photocell on the shaft of a motor controlled by the VIC. Write a program so that motor turns until maximum light falls on photocell. Mount the head of a plastic doll on the shaft for an impressive effect of face looking into light source (a flashlight) and following it.

4. Use it to point solar collectors to the sun. A lens can be used to collect light from one direction so that the photocell becomes more directive. An old camera can provide both the lens and the lightproof enclosure. Mount the photocell so that its photoconductor is exactly on the film plane behind the lens.

5. Measure the light dosage of your plants. Measure intensity every lets say 5 minutes and accumulate it. The total for each 24 hour period is the daily light dosage. Keep in mind that you must linearize the readings before adding them up - otherwise result will be meaningless.

6. Make a darkroom exposure meter that computes correct exposure time and times the exposure. If you bring the VIC in the darkroom, cover bring the VIC in the darkroom, cover the screen of your monitor with the screen of your monitor with a piece of rubilith (ask a printer for one) to render its light safe.



LIQUID LEVEL SENSOR

DESCRIPTION

It's a rainy night and you are away from home. Is your basement flooded? The liquid level sensor will tell your VIC and you will be able to call it on the phone to report to you the status of your cellar. Impressive but only one of the possible applications of a liquid level sensor.

THEORY OF OPERATION

Current can flow through water (but perfectly pure water is an insulator). By placing two electrodes in a container you can tell if it contains a water based solution or not based on whether current flows. If the applied voltage is DC, electrolysis will take place which will soon corrode the electrodes. AC will have no ill effects because the products of the electrolysis will be opposite in consecutive cycles and will recombine.

The circuit shown here uses a CMOS NOR gate to act as an oscillator to generate an AC voltage. If the probe is not immersed in liquid this voltage is converted to DC by diodes (configured as a voltage doubler), amplified and used to present a logical "1" to the user port. If liquid touches the probe, current flows through the probe and the liquid to the ground and the voltage at point A drops so that the output is now a "0".

If the container is metallic,it should be connected to the ground. If it is non-metallic, a second electrode should be placed next to the probe (but not touching) and connected to ground. The length of the other electrode and its closeness to the probe will determine the sensititivity of the sensor. Experiment with various probe lengths and placements until you get satisfactory operation.

PROGRAMMING

Connect the output of the sensor circuit to any bit of the user port. Make this bit an input by placing a 0 at the corresponding bit of the data direction register.

To check the output of the sensor, assuming you have connected the sensor to PB2 use the following:

```
 90 POKE 37138,0
100 T = PEEK(37139) AND 4
110 IF T = 1 THEN PRINT  "HIGH"
120 IF T = 0 THEN PRINT  "LOW"
130 GO TO 100.
```

The '4' in statement 100, is a mask to isolate the third bit of the user port.

PWM MOTOR CONTROLLER

DESCRIPTION
===========

Operating a motor under program control is not only fun but has many applications. From automating a model railroad to opening and closing the curtains of your house to increase energy savings, motor control enables your VIC to actually do something. In this project, a PWM technique is used to give precise control even on toy motors.

THEORY OF OPERATION
===================

The small shunt DC motor is the most widely used type of electrical motor. It will run on both DC and AC and it is used in toys, cassette recorders, disc drives, drills, hair driers, mixers and so on. Its speed can be controlled by adjusting the voltage applied to it. Unfortunately, torque depends on current and current decreases with decreasing applied voltage, so a point is reached when the torque due to the applied voltage is less than the friction of the bearings and the motor stops. For small toy motors this happens at around 500 RPM.

A solution to this problem is to use Pulse Width Modulation (PWM). Full voltage is applied for some amount of time and then power is turned off. By adjusting the ratio of power ON to power OFF the total energy going into the motor(and its speed) is controlled. And by making the ON-OFF switching fast enough, the inertia of the motor smooths out the pulses and we see a continuous motion. Using PWM, a toy motor can be controlled down to 1 RPM or less without stalling. Torque remains costant regardless of speed because when ON the motor operates at full voltage (and current). An additional significant benefit is that the power transistor that drives the motor is either ON or OFF, thus dissipating only a few percent of the power it controls.

The PWM circuit consists of an oscillator that drives a counter, a comparator and a control input (the VIC user port). As long as the count is below the control input, the output of the comparator is an '1'. When it is more than the control input, it is '0. The comparator output changes states once for every full cycling of the counter, thus the period of the output wave is constant (T). When it changes depends on the control input, thus the width W varies with the control input, giving us a PWM signal.

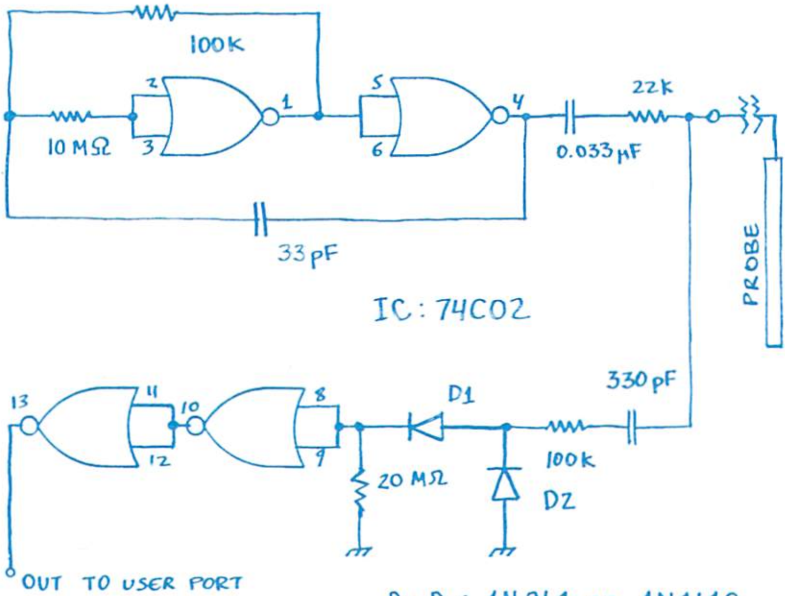
This is connected to a transistor driver that in turn controls the motor. The diode at the collector of the transistor is for protection during the times the transistor is off.

PROGRAMMING
===========

To control motor speed, set the user port to output (POKE37138,1) and output a control parameter, from 0 to 255. Assuming you would like to operate at half speed, POKE37136,128 will do it. To stop the motor POKE37136,0. To go full speed, POKE37136,255.

NOTES
=====

Try various values of RA, RB and C in the 555 until operation is achieved at desired speeds. You will find that not all motors operate smoothly at all speeds. The better the motor quality the wider the range. Good quality miniature DC motors can be obtained from old cassette recorders. Turntable motors are usually AC only and will not operate on DC. A source of inexpensive medium power DC motors are motors used to power electric windows and seats in old luxury cars. Also check with your local electronic surplus stores.



D_1, D_2: 1N914 OR 1N4148

PROBE: STAINLESS STEEL ROD 1/8" DIA. IS PREFFERED, STEEL WIRE ADEQUATE. EXPERIMENT WITH LENGTH (1" TO 12")

DESCRIPTION

Did the telephone ring when you were away? When? How about keeping a log of telephone rings with your VIC. A ring detector can also be the basis for a telephone answering machine. This project shows how build a reliable optically isolated ring detector and how to integrate it into a computerized telephone answering machine.

THEORY OF OPERATION

The telephone ringing voltage is a 20 Hz, 110 volt AC signal, superimposed with a 400Hz signal. The 20Hz signal activates the bell in your telephone set while the 400Hz signal is heard by the calling party to indicate that the phone rings.

In building anything that connects to the phone lines it is important to take extraordinary safety precautions. While you may do anything you wish at your home, you have no right to damage or make unsafe (by sending high voltages) the telephone network which is used by all.

Telephone company regulations are very strict in this matter. The device you connect to the phone lines must be either FCC approved or you must rent a coupler from the phone company. The coupler makes sure that nothing damaging gets to the phone line.

Although required by law to make arrangments for you (for a fee) to connect anything reasonable to their lines, your local telephone company may not like the idea and try to refuse helping you.

In case you want to exercise your alternate option, the ring detector is designed to be undetectable by electrical inspections of the line by the phone company.

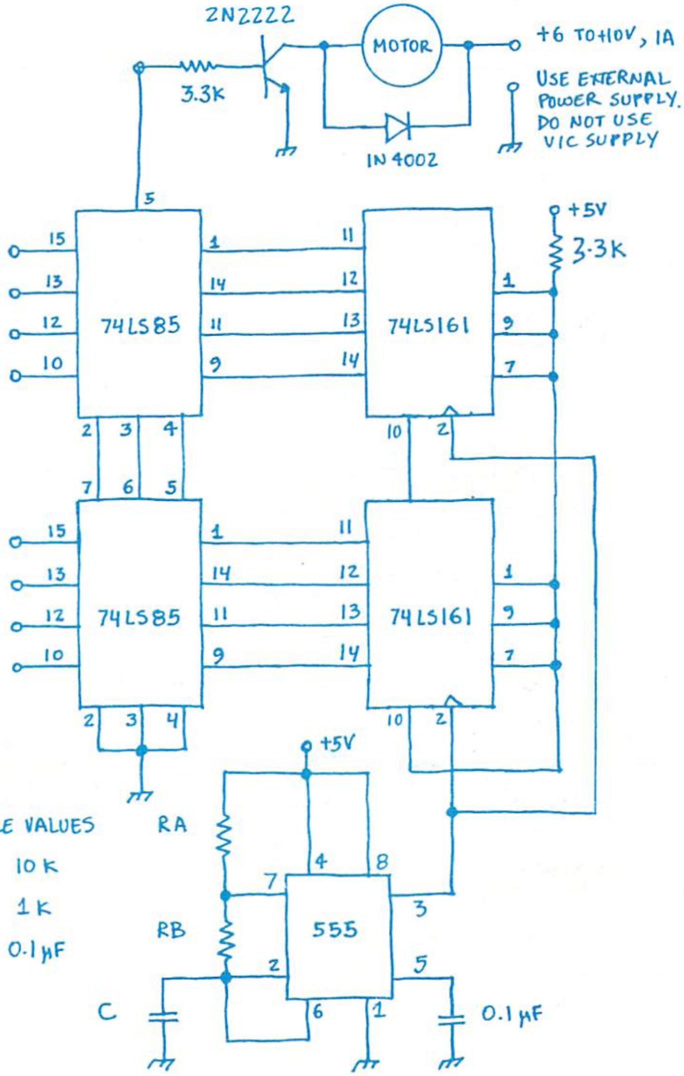
This is achieved by presenting a very high impedance, more than 100K to the telephone line. A small neon lamp is used to convert the ringing voltage to light which is picked up by a Darlington phototransistor. The resulting signal is amplified by an op-amp and presented to the user port as a logical '0' or '1'. The use of the neon lamp gives excellent noise immunity because the lamp does not fire at voltages less than 70 volts. Thus line noises, clicks, etc. do not give false readings. In addition, it offers perfect isolation of the phone line since no electrical connection is made between your circuit and the line.

Programming is identical to that of the liquid level sensor, since both devices have an 1-bit output. Use the TI variable to create a clock to log the time of telephone ringings. You could also keep a count of how many time the phone rang each time.

AN ANSWERING MACHINE

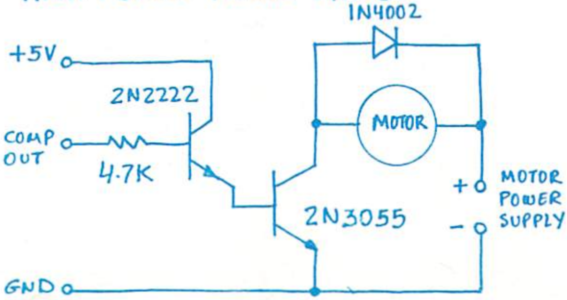
To answer the telephone when you detect a ring, you only need connect a 680 resistor across the lines. This can be done using a small 5V relay, driven by one of the bits of the user port.

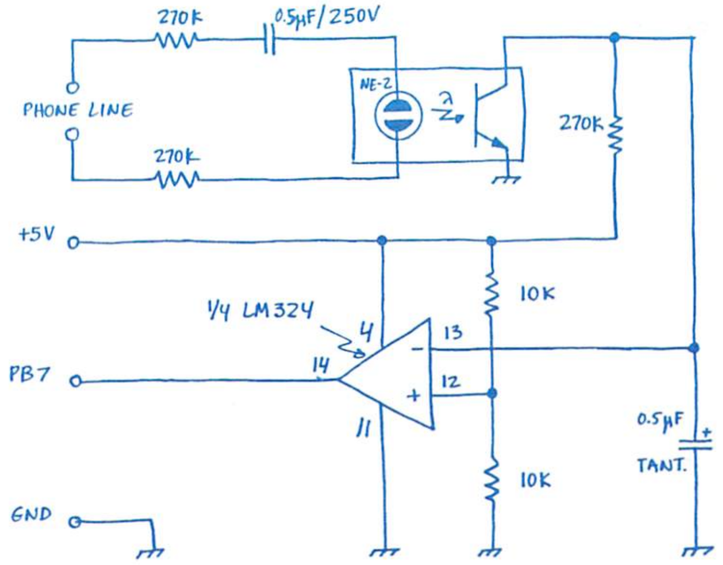
The connections for a complete answering machine are shown in the schematic. The casssette player should use an endless loop tape to deliver the message. It is turned on after the desired number of rings is detected, and the answer relay is activated. After a delay equivalent to the length of the message (usually 20 seconds) the cassette recorder is turned on for a specified duration, lets say a minute. A program that does this can be very simply written in BASIC.

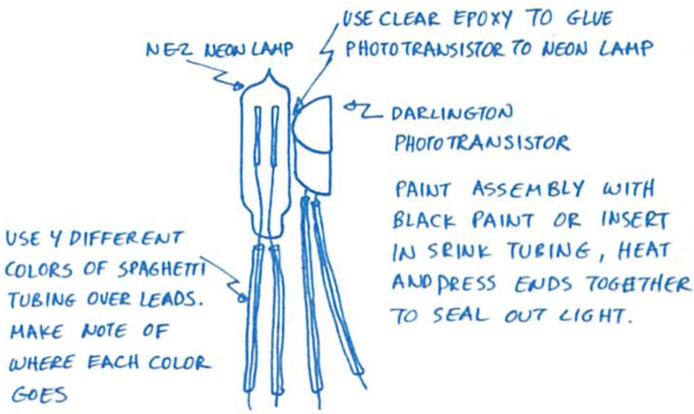


If you are using the high power circuit, build it solidly and provide some heat sinking for the 2N3055. Make a single ground connection from the digital circuit ground to the power circuit ground to avoid loops.

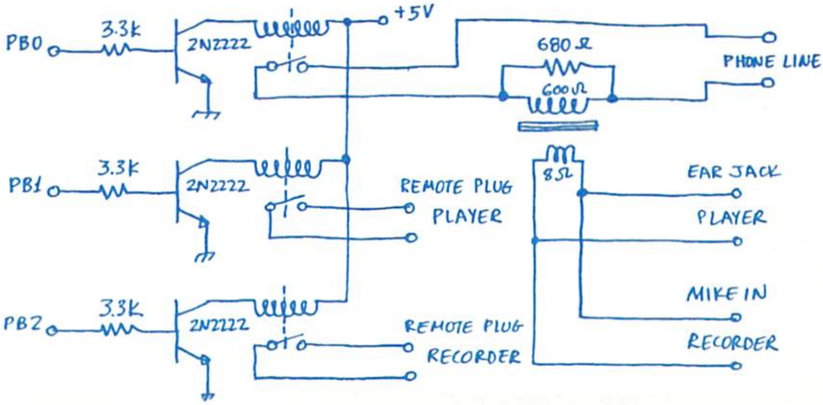
HIGH POWER MOTOR DRIVER







ANSWERING MACHINE



DESCRIPTION

This project gives your VIC the capability to dial a telephone. Pulse dialling is accepted by all exchanges in the USA (as opposed to tone dialing) so this system has universal appeal.

THEORY OF OPERATION

Pulse dialing operates by interrupting the current flow in the telephone line. To send a digit, lets say '5', the line is interrupted five times. The interruptions last 40 ms with a pause of 60 ms between interruptions. There is a minimum of 800 ms pause between digits. Pulsing with these specifications is called 10pps (pulse per second) dialing. Some telephone exchanges may require 20pps dialing where all timing values are half of these given above.

The circuit consists of a transistor driving a reed relay that interrupts the line. A BASIC program drives the relay through the user port, creating the required pulses.

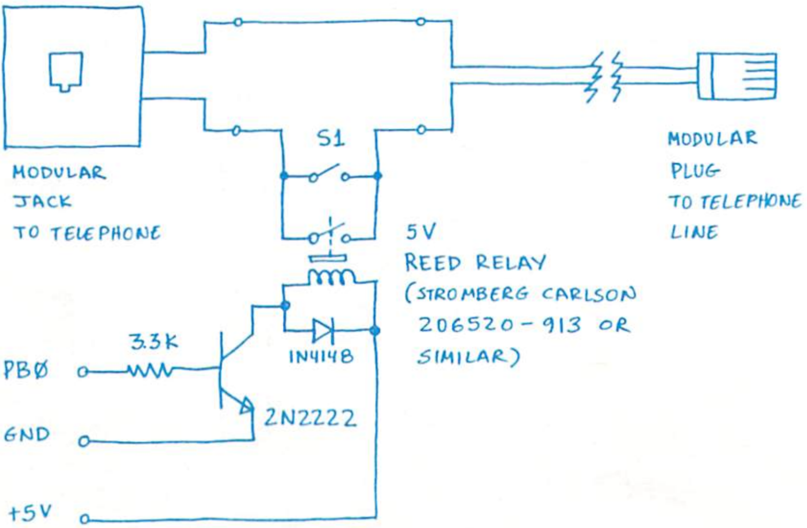
A reed relay is specified because it is fast. It has the disadvantage that reed relays when not activated are usually in the open state. Here switch S1 is used to manually restore continuity of the line when the VIC is not operating.

NOTE: The telephone headset must be picked up to establish connection before dialing can proceed.

PROGRAMMING NOTES

Lines 120 to 150 input the string of digits to be dialed. There is no limit to the length of this string. Lines 160 to 210 extract the digits from the string one at a time and convert the ASCII codes to the actual numbers (0 is replaced by 10 because 10 pulses represent a zero). Lines 300 to 410 form the subroutine that does the actual pulsing. Lines 270 to 290 add a redial feature where the same number is dialed again if, for example, the line is busy.

The program can be modified so that it dials a group of numbers sequentially (perhaps to deliver an advertising message) or dials numbers derived from a mailing list database (to play a recorded announcement to friends or customers for example).

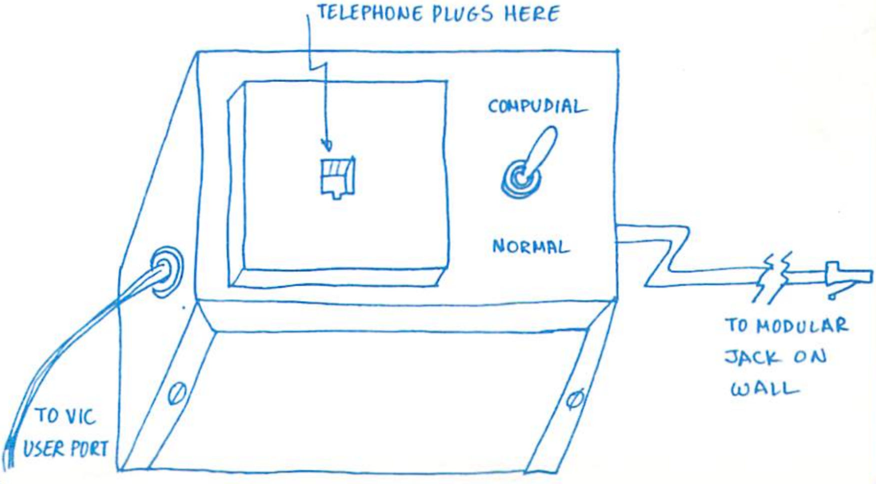


PARTS LIST
==========

1. Modular telephone jack
2. Modular telephone plug with cord attached
3. Single pole single throw miniature toggle switch
4. 5V reed relay
5. 1N4148 or equivalent diode
6. 2N2222 or equivalent transistor
7. 3.3K, 0.25w, 5% resistor
8. Case and hardware as needed.

PROGRAM LISTING
===============

```
100 REM PULSE TELEPHONE DIALER
105 REM
110 POKE 37138,255: P=37136
115 POKE P,255
120 PRINT"ENTER TELEPHONE NUMBER."
130 PRINT"DO NOT LEAVE SPACES BETWEEN DIGITS.
140 PRINT""
150 INPUTN$
160 L=LEN(N$)
170 FOR M=1 TO L
180 D$=MID$(N$,M,1)
190 IF D$="0" GOTO 230
200 D=ASC(D$)-48
210 GOSUB 300
220 GOTO250
230 D=10: GOSUB 300
250 NEXT M
260 PRINT""
270 PRINT"TO REDIAL PRESS 'Y' ELSE 'N'";K$
280 GET A$:IFA$=""GOTO280
285 IF A$="Y" GOTO170
290 GOTO120
300 POKEP,255
310 FOR I=1 TO D
320 FORK=1TO21
330 NEXTK
340 POKEP,0
350 FORK=1TO34
360 NEXTK
370 POKEP,255
380 NEXT I
390 FORK=1TO450
400 NEXTK
410 RETURN
```

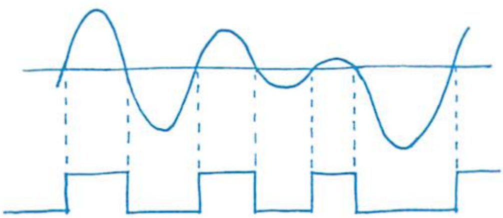


DESCRIPTION

If VIC could talk what would it say? Find out by giving your computer the gift of speech. This project lets you digitize and play back speech with low data rate by taking advantage of an unusual property of the speech signal.

THEORY OF OPERATION

Human speech is a very unusual signal in many ways. One of its startling properties is the fact that the zero crossings of the speech wave contain just about all the intelligibility of the original. In other words, if we take the speech signal coming out of a microphone and throw away all information except the times the signal crosses zero, whatever remains is easily understood (but it sounds badly distorted). We can get the zero crossing signal by hard clipping the original as follows:



The zero crossing signal is either high or low (actually the term 'zero crossing' is a sonorous misnomer. A more accurate term is the sign of the signal). This is a signal easily accomodated by the VIC. The schematic shows a circuit that takes the speech signal and generates the zero crossings by amplifying and clipping it. The program inputs these zero crossings in the memory of the VIC as follows:

The zero crossing signal is samples every 170 us . The information (either a '0' or a '1') is shifted into a temporary location. After 8 shifts, a byte is formed and stored. This goes on until the allocamemory is filled, signifying the end of the input process. To play back speech the process is reversedand the resulting wave is filtered and fed to an external amplifier for playback.

PROGRAMMING NOTES

The programs use for temporary storage the random number seed area so it is advisable not to use the RND function in BASIC if your program has voice output.

Every path throught the program must have the same execution time equal to the sampling period. To equalize paths a program segment labeled DUMMY is included.

To alter the sampling frequency, change the constant in the line labeled 'DELAY'. In the listing the constant has the value 240 (F0) making the sampling period 170 us. Smaller constants will give lower data rates and reduced quality of output. You may try different values to find the lowest setting that sounds adequate to you.

The constant in the routine ENDCK is the high order byte of the address of the last memory location where voice data is stored. The low order byte is always 0. Thus the end of speech data is allocated in 256 byte increments. The value of 60 (3C) shown in the listings indicates an end of data address of 15616 (3C00). It was chosen somewhat arbitrarily and you may wish to change it if your VIC has more 8K of additional RAM or if you would like to edit a sound.

Program to digitize voice.

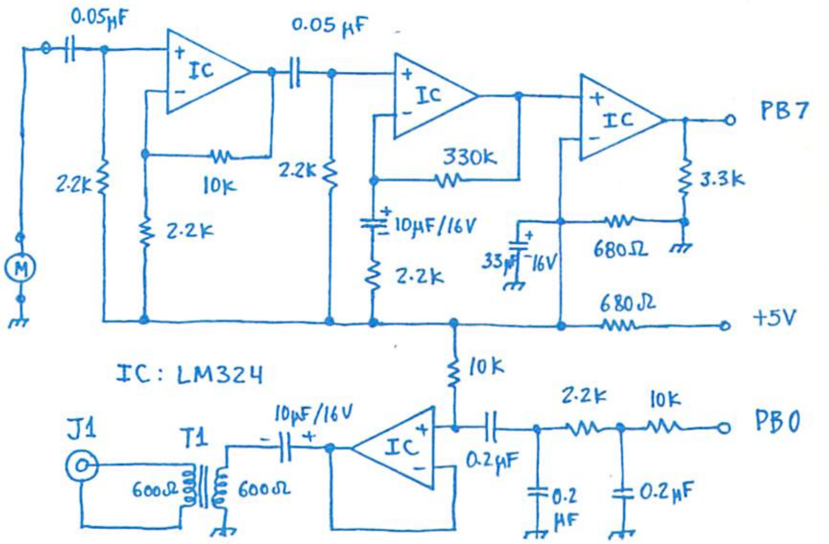
```
                         PTRL=139
                         PTRH=140
                         CTR=141
                         TEMP=142
                         DEL=143
                         PRT=37136
                         DDR=37138

A901      INITL   LDA #1
8D1291            STA DDR
A900              LDA #0
858B              STA PTRL
A91E              LDA #30
858C              STA PTRH
A9F8              LDA #248
858D              STA CTR
A000              LDY #0
A9F0      DELAY   LDA #240
858F              STA DEL
E68F      DELOP   INC DEL
D0FC              BNE DELOP
AD1091    GDATA   LDA PRT
2A                ROL A
268E              ROL TEMP
E68D              INC CTR
F009              BEQ INCRM
E68F      DUMMY   INC DEL
E68F              INC DEL
E68F              INC DEL
38                SEC
B0E5              BCS DELAY
A58E      INCRM   LDA TEMP
918B              STA (PTRL),Y
A9F8              LDA #248
858D              STA CTR
E68B              INC PTRL
D0D9              BNE DELAY
E68C      ENDCK   INC PTRH
A93C              LDA #60
C58C              CMP PTRH
D0D1              BNE DELAY
60                RTS
```

Program to output voice.

```
                         PTRL=139
                         PTRH=140
                         CTR=141
                         TEMP=142
                         DEL=143
                         PRT=37136
                         DDR=37138

A901      INITL   LDA #1
8D1291            STA DDR
A900              LDA #0
858B              STA PTRL
A91E              LDA #30
858C              STA PTRH
A9F8              LDA #248
858D              STA CTR
A000              LDY #0
A9F0      DELAY   LDA #240
858F              STA DEL
E68F      DELOP   INC DEL
D0FC              BNE DELOP
268E      GDATA   ROL TEMP
2A                ROL A
8D1091            STA PRT
E68D              INC CTR
F009              BEQ INCRM
E68F      DUMMY   INC DEL
E68F              INC DEL
E68F              INC DEL
38                SEC
B0E5              BCS DELAY
B18B      INCRM   LDA (PTRL),Y
858E              STA TEMP
A9F8              LDA #248
858D              STA CTR
E68B              INC PTRL
D0D9              BNE DELAY
E68C      ENDCK   INC PTRH
A93C              LDA #60
C58C              CMP PTRH
D0D1              BNE DELAY
60                RTS
```



ALL RESISTORS 0.25 W, 5%. M: DYNAMIC MICROPHONE, 200-600 OHM IMPED.
J1: RCA JACK, CONNECTS TO STEREO AUX INPUT. T1: 600 OHM TO 600 OHM.

DESCRIPTION

The expansion port is the large connector in the back of the VIC, allowing addition of RAM or ROM memory and I/O ports. Apparently in an effort to reduce costs, Commodore did not buffer the line coming to this port. To compound the problem, the backplanes for the VIC currently marketed ignore the issue of buffering. No buffering can mean erratic operation and lost data. Here is how to do it right, if you plan to use the expansion port.

THEORY OF OPERATION

The data and address lines coming to the expansion port on the VIC are directly driven by the 6502 MPU. The drive capability of the 6502 is limited (one TTL load), and internal circuits already use some of this capacity. Thus unless buffering is used, the danger exists that operation of the system will be marginal, giving rise to soft (noise generated and not repeatable) errrors. Of course with heavy loading the system will stop.

An address and data buffer can be made with only 4 IC's and it will guarantee correct operation (up to 20 TTL loads or 200 LSTLL loads) with the added benefit it will provide a physical barrier to external voltages in case of improper connections (If something goes wrong, chances are the buffer IC's will be destroyed leaving the VIC intact). Looking at the schematic, the LS244 octal buffers are used to buffer the address lines and the R/W signal. The data bus buffer is bidirectional, and its direction is switched by the R/W signal. It is enabled only when external access is requested (using the BLK and I/O select signals) so that there are no conflicts during accessing of locations internal to the VIC.

The RAM signals are not used to generate the buffer enable signal so the buffer cannot be used in connection with a 3K RAM expansion. This should not be a problem because VIC ignores the 3K space if there is any other expansion RAM. Should you find a use for the 3K RAM space, you must modify the enable generator (74LS11) to accomodate the RAM signals, by adding another 74LS11. The +5v supply from the VIC is fused, as a precaution. Be careful not to draw too much power from this line. Commodore does not specify current loading here but one to two amps should be the limit. If your system requires more power, use an external power supply.

All signals on the expansion port that are not buffered should be brought out for use during expansion. This is not shown in the schematic to avoid clutter.

CONSTRUCTION NOTES:

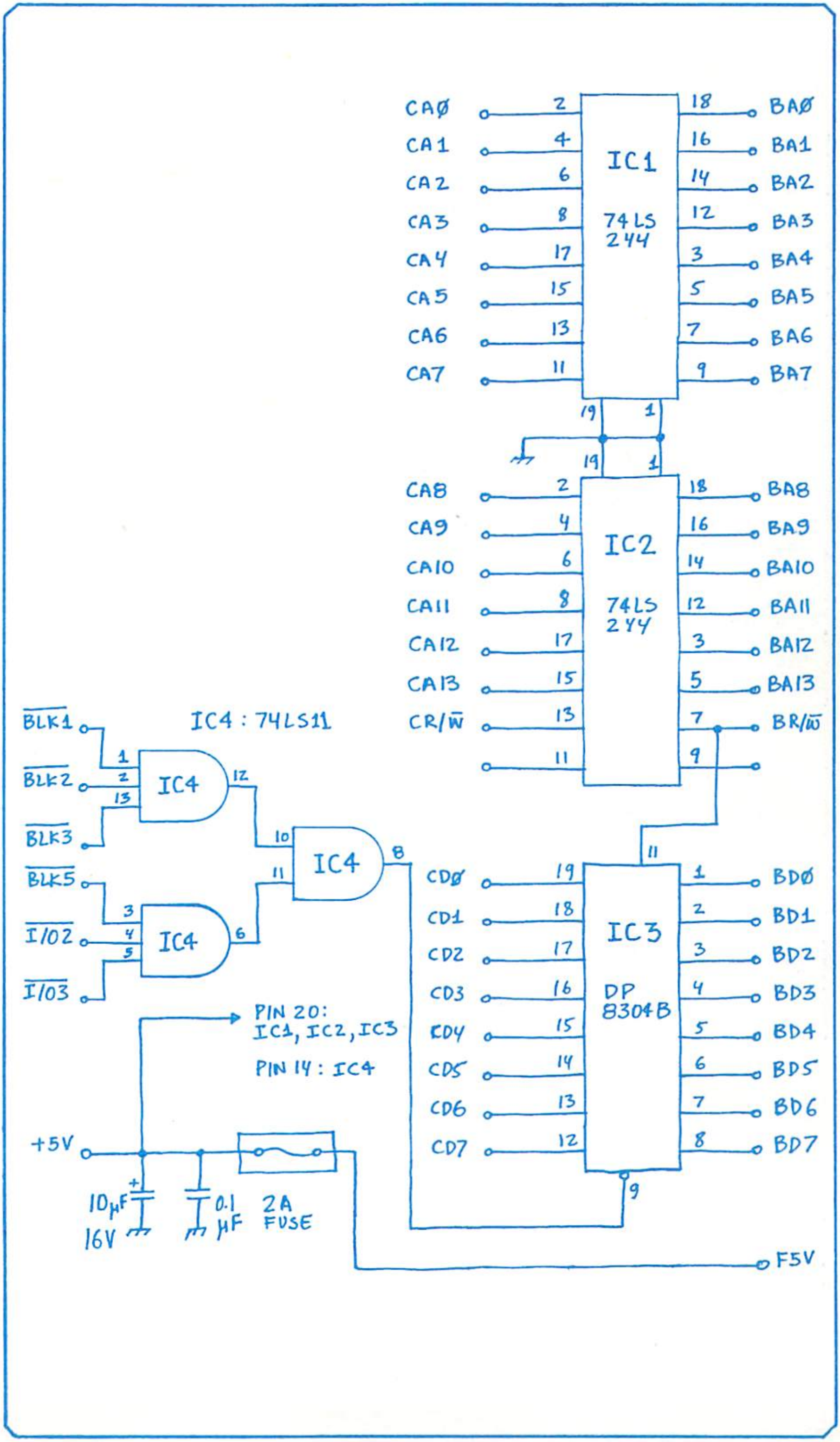
The best way to implement the buffer is by using a P.C. board. The P.C. board should also have positions for expansion boards, 4 or 5 should be sufficient.

Another approach is to use a small P.C. for the buffer and a 40 pin flat cable, (up to 36" long) with connectors in both ends, going into an expansion box that contains backplane and power supply.

The next best method to P.C. is wirewrap, which is totally adequate. Use sockets with short, two-level pins.

PARTS LIST

1. Two 74SL244N Octal Buffers
2. DP 8304B 8-bit Transceiver
3. 74LS11 triple 3-input AND gate
4. 10uF/16V electrolytic capacitor
5. 0.1uF ceramic disc capacitor
6. 2A fuse, P.C. board or perf board, sockets, connector.



CAØ
CA1
CA2
CA3
CA4
CA5
CA6
CA7
IC1
74LS 244
BAØ
BA1
BA2
BA3
BA4
BA5
BA6
BA7
CA8
CA9
CA10
CA11
CA12
CA13
CR/W
IC2
74LS 244
BA8
BA9
BA10
BA11
BA12
BA13
BR/W
IC4 : 74LS11
BLK1
BLK2
BLK3
BLK5
I/O2
I/O3
IC4
IC4
IC4
CDØ
CD1
CD2
CD3
CD4
CD5
CD6
CD7
IC3
DP 8304B
BDØ
BD1
BD2
BD3
BD4
BD5
BD6
BD7
PIN 20: IC1, IC2, IC3
PIN 14 : IC4
+5V
10µF 16V
0.1 µF
2A FUSE
F5V

8K OR 16K RAM/ROM EXPANSION

13

DESCRIPTION

VIC is endowed with only 3596 bytes of user RAM at birth. It doesn't take long to realize how nice is to have more memory. This project will take you up to '19500 bytes free' easily, thanks to the latest IC marvels, the 2KX8 static RAMs. What's more you can mix RAM and ROM on the same board without any changes.

THEORY OF OPERATION

The RAM memories ICs in your VIC are 1KX4 static RAMS meaning they contain 1024 locations each storing 4 bits. This type of memory was the industry standard static RAM when the VIC was designed, but a year later the industry standard is the 2Kx8 RAM IC, so we will use it in this project.

Static memory chips are very easy to use (the opposite is true of dynamic RAMS). What you need to do is provide addresses, data (in and out), a chip select signal and a read/write signal.

These signals are all available at the expansion port of your VIC so interfacing becomes a matter of connecting all the like IC pins together with a little decoding thrown in to point to the chip to be read or written at any given time.

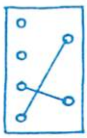
The VIC provides what is known as block decoding signals, $\overline{BLK1}$, $\overline{BLK2}$, $\overline{BLK3}$, and $\overline{BLK5}$. Each signal corresponds to 8K of memory. For example, when $\overline{BLK1}$ is active (low), the memory block, in locations $2000 to $4000 can be read or written. The exact location in the block is pointed to by the information on the address bus (CA0-CA10) Data is transported to and from the memory on the data bus(CD0-CD8). The read write signal (CR/$\overline{W}$) orders a read (from memory) when it is high and a write when it is low.

Since the memory chips contain 2K bytes each, 4 chips are needed for 8K of RAM, and it becomes necessary to use a decoder to select one of these chips.

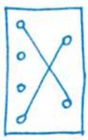
An 1/4 (read: one out of four) decoder is an IC that costs as much as an 1/8 decoder. So we'll use an 1/8 decoder and go for 16K. If you'd rather stay with 8K, no problem, just don't wire the sockets for the extra 4 RAM IC's.

Looking at the schematic, the decoder is IC9 (74LS138). It uses CA11 and CA12 to decide which one of four RAMS to select. It also uses one of the $\overline{BLK}$ signals to select the upper or lower RAM bank. IC1 is used to buffer the R/$\overline{W}$ signal and to generate the enable for the decoder (the decoder becomes active only when one of the $\overline{BLKs}$ selected by the jumper block is low). Here are two examples of connecting the jumper block:

16K
FROM $2000
TO $5FFF



8K FROM
$2000-3FFF
PLUS
8K FROM
$A000-BFFF



The BANK jumper to be used with bank switching. In normal use it is always connected to ground.

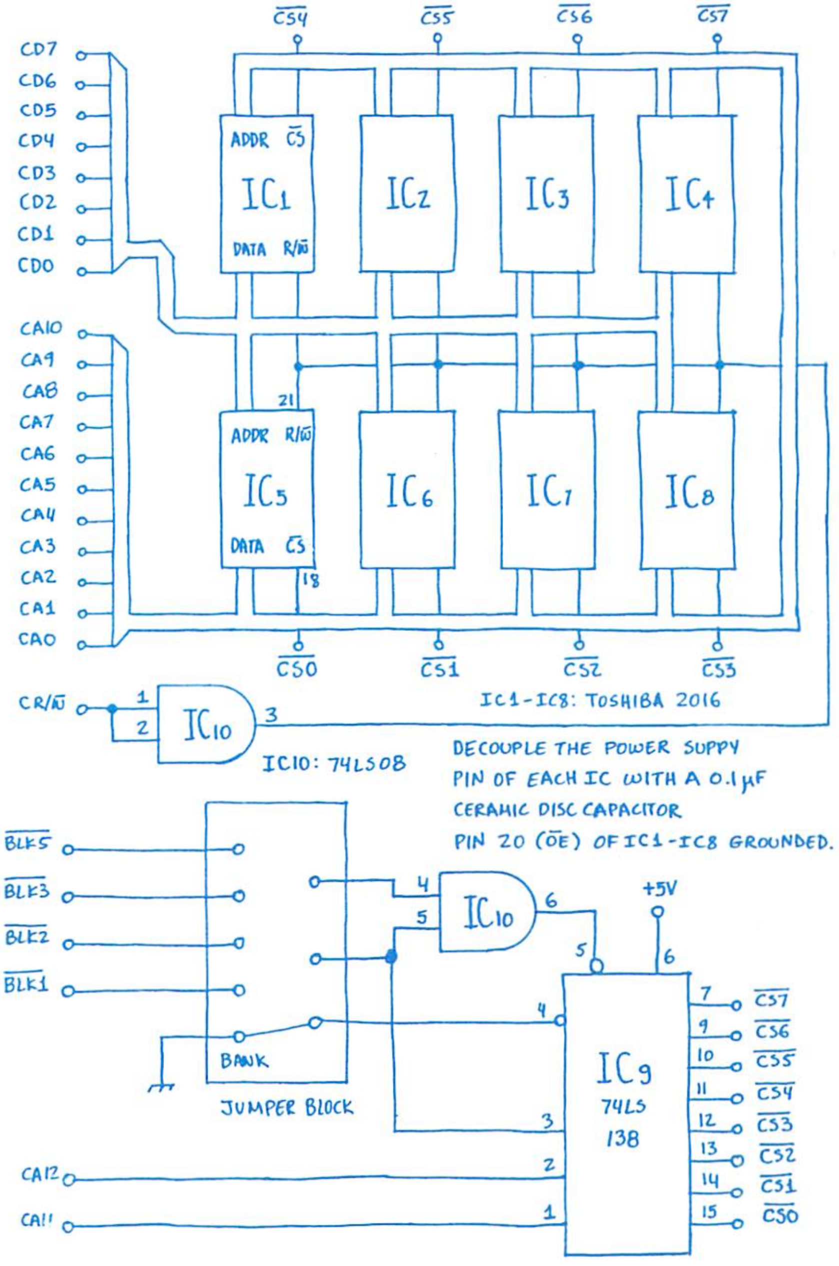
LIMITATIONS

This memory design does not offer any buffering of the address or data buses and as such only one board can be reliably used with the VIC, plugged into the expansion port. In addition, power is derived from the VIC power supply so if 16K is implemented, it should be done using CMOS RAMS (6116) instead of NMOS (2016) to avoid a large power drain.

If you need more RAM, it is recommended that you build first the expansion port buffer and then connect the RAM boards to it.

USING ROMs

You can replace any RAM chip with a 2716 EPROM without any circuit changes. However, BASIC will not recognize EPROMS and all RAM beyond the EPROM. This is because on powerup, the VIC checks for Read/Write memory and sets the top of RAM pointer to the first location it cannot read what it writes. By manipulating the beginning of BASIC pointer you could execute a program in BASIC stored in ROM. How to do this is beyond the scope of this book.



DESCRIPTION

When power fails, the contents of RAM memory dissapppear, unless of course it has a data retention feature. Here is how to build an 8K RAM with this feture, and an application for it that might raise some eyebrows.

THEORY OF OPERATION

CMOS 2KX8 memories like the HITACHI HM6116 have a standby operation mode in which they draw very little power, typically under 100uA at 3 volts.

A memory board can be built with these memories using some extra circuitry to activate the standby feature and a couple of batteries to provide standby power. Such a board would retain data for a few hundred hours after the power to the VIC is turned off. The batteries will recharge when the power is turned on again, ready for the next power down cycle.

The memory circuit is similar to the one shown for the 8K/16K RAM/ROM. It operates in identical fashion when the +5V power from the VIC is present. Resistor R is used to trickle charge the NiCad batteries.

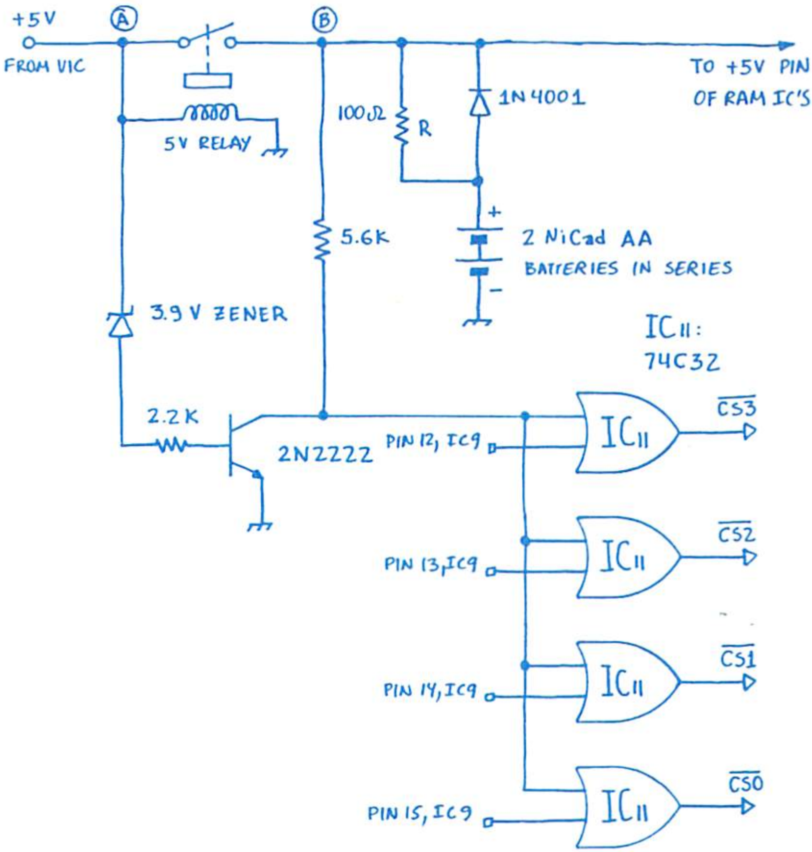
When the +5V is lost, the relay opens up, isolating the RAM array form the VIC. Now the diode conducts and powers the RAM. The transistor is also turned off, and all outputs of the NOR gate become high, satisfying the need for high chip selects during power down. The zener diode insures that the chip slects become high early during the power down cycle.

NOTE : NEVER solder directly on a battery (inless it has wires welded on by its manufacturer). Use a battery holder instead, and solder on its terminals.

AN UNORTHODOX (MIS) APPLICATION

It is possible to transfer the contents of a game cartridge to this 8K RAM and use it instead of the cartridge. Using a PEEK-POKE loop, all data from the cartridge can be transferred to the RAM which now can substitute for the cartridge. Extra RAM loaded with programs can be stored to a rack with connectors that supply +5V to the appropriate pin for trickle charging of the NiCad batteries.

Needless to say, dupliction of copyrighted programs for sale is clearly illegal. Duplicatin for personal use might not be so clearly illegal but is certainly clearly unethical.



POWER (+5V) TO IC9 AND IC10 IS DERIVED FROM POINT Ⓐ.
POWER FOR IC11 IS DERIVED FROM POINT Ⓑ

SEE PROJECT 13 FOR THE REST OF THE CIRCUIT.

DESCRIPTION
===========

The microprocessor CPU within the VIC can address up to 64K of memory. Only about 35K of this space is user accessible. The rest is used by the operating system and BASIC interpreter. This project lets you add up to 128K of RAM to the VIC using only 8K of the address space, using a trick called paging.

THEORY OF OPERATION
===================

The 6502 CPU used in the VIC has 16 address lines. Thus the maximum memory ot can directly address is 64K, or 2 to the power 16. To increase the size of the directly addressable RAM, the CPU chip must modified. This obviously cannot be done because the 6502 is cast in silicon and it is not possible to change it. However, if something other than directly accessible RAM is acceptable there are techniques that will allow us to implement very large memories.

A very simple technique to increase addressable memory is paging. A long time ago, scrolls were replaced by books and the concept of a paging became established in publishing. A page is actually a small window to the text. You can only view a portion of the text at a time (as opposed to the whole text on fully unrolled scroll). In a book, accessing the text by flipping pages does not change the space of your desk taken up by the book.

Book paging directly translates to paging done in a computer. A page can be let's say 8K. You position a stack of pages (book) at any available and convenient location on the address space (desktop). At any given time you can only access the contents of the current page. To access the contents of another page it must be made current by swapping (flipping) pages.

Seems like a wonderfully simple idea but how complicated is it to build it? Actually it is very easy to implement paging. We need a number of memory boards that have an extra enable line. The 8K/16K RAM/ROM board described earlier has a line called BANK (the form of paging described here is also known as bank switching) that would do just fine.

Now lets say we have two of these memory boards populated with 8K of RAM, and we configure the jumper blocks so that each one responds to addresses between $2000 and $3FFF. If we plug them both in a backplane, every time we try to read from any location between $2000 and $3FFF, both memory boards will be activated and the result would be noise instead of data as each board would try to force the bus to reflect its own data.

Now lets say that we connect the BANK line of the first board to PB0 of the user port and them we connect the BANK line of the other board to PB1. We have four possiblities:

PB0	PB1	
0	0	:NOT ALLOWED. WILL ENABLE BOTH BOARDS.
0	1	:BOARD #1 ENABLED
1	0	:BOARD #2 ENABLED
1	1	:BOARDS #1 AND #2 DISABLED.

So by POKEing a 1 or 2 to the user port we can switch between two banks of 8K of memory both in location $2000 to $3FFF. The page size is 8K and there are two 8K pages. We can take this to the limit by populating the boards with 16K of memory and using all 8 bits of the user port. This will give us eight 16K pages or 128 of accessible RAM using only 16K of the address space.

PROGRAMMING NOTES
=================

So far so good, but what's the catch? Well the catch is the soft ware. For one, the BASIC in the VIC has not been written to accom date paging. Not that it is impossible to do (on the contrary) but memory has become cheap only recently while MICROSOFT'S BASIC was designed 7 years ago. However it may still be possible to work things around these limitations by manipulation of the pointers to BASIC text stored in page zero.

Without any modification to BASIC we can use the extra RAM for data storage, as long as we access the data using PEEK and POKE. If we leave BASIC altogether and go to machine language, the limitations are few. You must be careful not to swap pages from a program that is on the page being swapped out (doing that is equivalent to standing on a tree branch you are chopping off). And of course you must keep track of where the data and programs are and swap pages at the ap-priate times.

BASIC programs can be stored in ROMS or RAMS pages and loaded into the problem location using a simple program in machine language. When the program is loaded, an unused RAM page is swapped in so the BASIC program sees all memory present and is not affected by paging. Much faster than a disk and much more reliable.

SOME IMPLEMENTATIONS

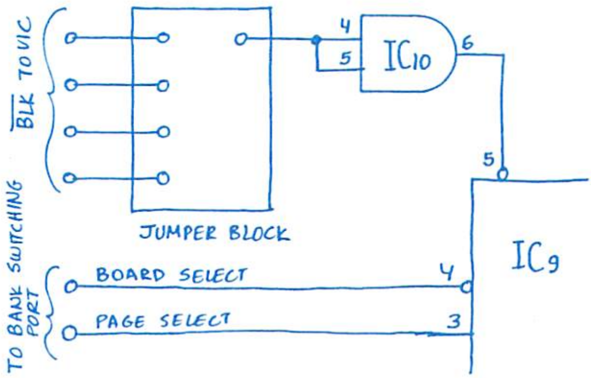
The use port offers the simplest way to implement paging at the cost of using up this versatile port. An alternate apprach is to dedicate an output port to bank switching. See the project on the dual I/O port on how to add ports to the VIC.

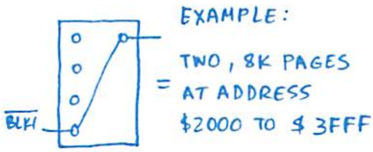
If you would like to have 8K pages, you can very easily convert a fully populated 16K board to two 8K pages as follows:

1. Connect pins 2 and 3 of IC10 together
2. Connect one jumper only from the BLK signal that represents the desired page window to pin 2 of IC10
3. Connect pin 4 of IC9 to the paging port.

The board is selected when a '0' is is present at the board select input (pin 4, IC10). It will respond to adresses within the block determined by the BLK signal used to drive IC10.

The page select input selects the lower 8K on the board when it is '0' and the upper 8K when it is '1'. Thus with one output port, eight 8K pages can be swapped. An additional port will allow 8 more pages and so on.





DESCRIPTION

A very simple bridge between the digital and analog worlds can be made using only resistors connected to the user port. Although not accurate enough for precision work, this A/D can be used to generate audio signals and to demonstrate the operation of the D/A converter.

THEORY OF OPERATION

This project uses the R-2R ladder which is the heart of all modern D/A connectors. One of its advantages is the use of only two values of resistance, R and 2R. Only the ratio of the resistances is important, not their actual value making it relatively insensitive to variations in temperature.

In operation, the output at the point marked 'LADDER OUT' is 1/2 the voltage at PB4, plus 1/4 the voltage at PB3, plus 1/8 the voltage at PB2, plus 1/16 the voltage at PB1, plus 1/32 the voltage at PB0. Thus the resulting output voltage is a properly weighted sum of the individual inputs and represents the value of the binary number at the inputs PB0 to PB4.

The output impedance of the ladder is R Ohms (regardless of how many steps it has. Electrical Engineering undergraduates usually have to prove this paradox as an exercise).

To avoid loading the ladder, an operational amplifier is used as buffer. It should be powered from a +9V supply to avoid non-linear operation around +5 volts. Resistor R1 raises the minimum output above the ground about 2 volts to avoid poor operation of the op-amp around that area too. An LM324 op-amp is shown in the schematic but a 741 can also be used as well. With an LM324, R1 could be omitted without serious effects in the linearity of the output.

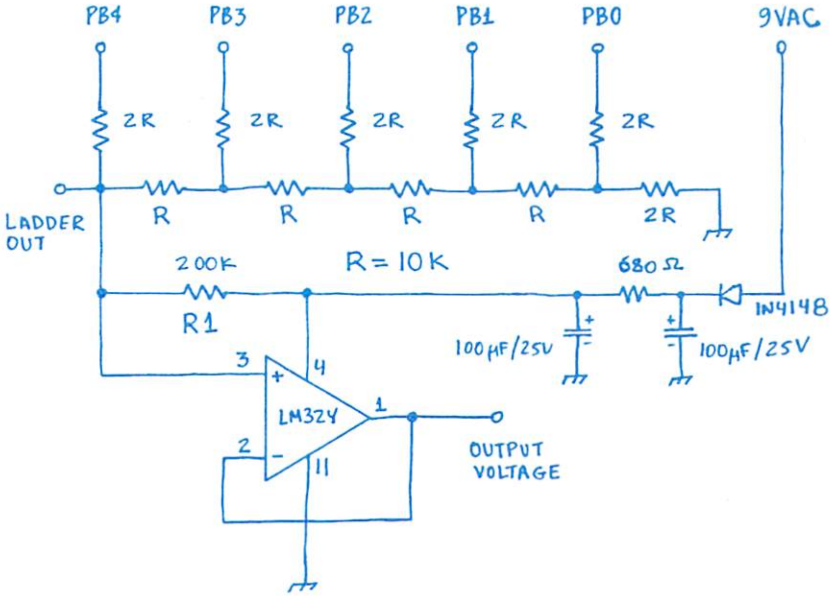
The R-2R resistor ratio should be as close to 2 as possible. 1% tolerance resistors are prefered, but 5% will work O.K. If you have 5% 10K and 20K resistors to sort through, you may use the circuit shown elsewhere for the thermometer and substitute the resistors for the thermistor, printing out the count for each resistor. Resistors with counts as close as possible should be selected.

PROGRAMMING

Machine language programming is generally necessary to achieve output at audio frequencies. You only need to store to the 5 least significant bits of the output port the value you want to convert to analog. (Do not forget to set the data direction register to output first).

The following program in BASIC will generate a triangular wave made up of 32 steps in each side. It can be observed by using an oscilloscope at the output.

```
100 POKE 37138,255
110 P=37136
120 FOR I=0 TO 31
130 POKE P,I
140 NEXT I
150 FOR I=31 TO 0
160 POKE P,I
170 NEXT I
180 GOTO 120
```


PB4
PB3
PB2
PB1
PB0
9VAC
2R
2R
2R
2R
2R
LADDER OUT
R
R
R
R
2R
200K
R = 10K
680 Ω
R1
1N4148
100µF/25V
100µF/25V
3
4
LM324
1
2
11
OUTPUT VOLTAGE

DESCRIPTION

This project describes a full 8-bit precision D/A converter with built in reference and voltage output. An 11-bit D/A converter chip is used giving excellent linearity,so this D/A approaches textbook performance.

THEORY OF OPERATION

An ideal D/A converter is impossible to make. However, if we want a very good 8-bit D/A converter, we can take a converter specified for higher resolution and configure it for the lower resolution we need. Its cost is somewhat higher but we get outstanding performance. In this project we use an 11-bit D/A converter IC configured for 8-bits plus sign.

The data lines of the user port drive the data lines of the converter and CB2 drives the sign, controlling the polarity of the output.

The converter has a built-in voltage reference and voltage output so no extra components are necessary. However it needs a ±15V power supply which must be supplied externally. Modular power supplies are availablbe for these voltages or one can be built as shown in the schematic.

Full scale output can be adjusted to 10.24 volts exactly (for 40 mV steps) by using an external potentiometer as shown.

Maximum output drive of the converter is 10 mA, which should be sufficient for most applications. If more output is desired a buffer should be used.

PROGRAMMING NOTES

Set the user port to output by a POKE 37138,255. For positive output set CB2 to high using the manual mode as follows:

```
POKE 37148, (PEEK(37148)AND 31) OR 192
```

For negative output set CB2 to low as follows:

```
POKE 37148, (PEEK(37148)AND 31) OR 224
```

Poke the value you want converted to analog in the user port data register

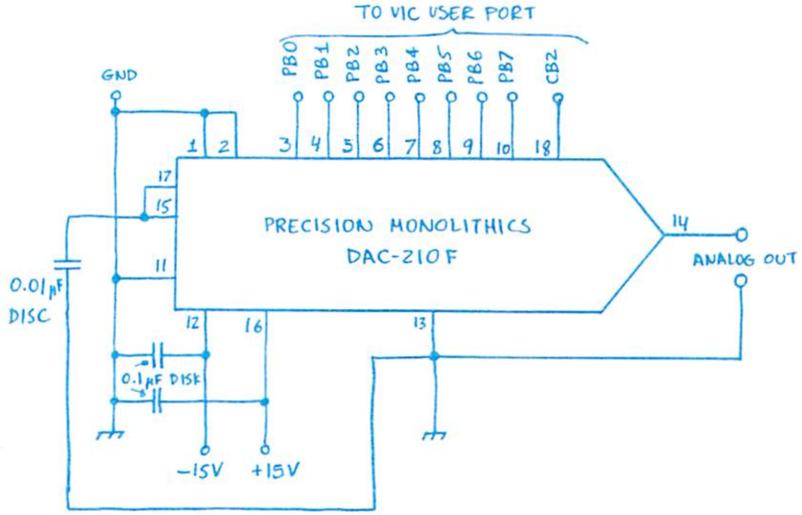
```
POKE 37136, VALUE
```

The DAC-210 settles within 1.5µs so it can take data as fast as you can output it, even if your program is in machine language.

AN UNUSUAL APPLICATION

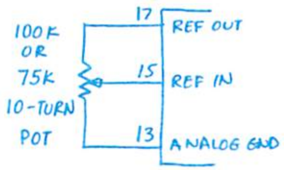
The D/A converter can be used as a multiplier if the reference input is driven by an external signal. The requirements are that the external signal varies between 3 and 10 volts. If the signal fed into the reference input is audio, the D/A converter can act as a digital attenuator to control volume or to create sound effects by modulating under digital control the input signal.

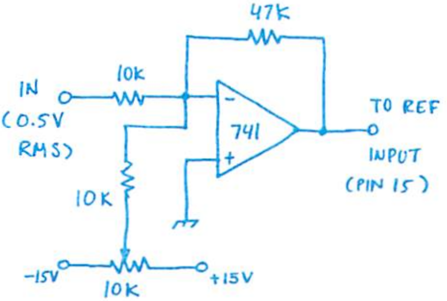
The schematic shows a level shifter required to feed audio in the reference input.



MULTIPLIER OPERATION
LEVEL SHIFTING AMPLIFIER

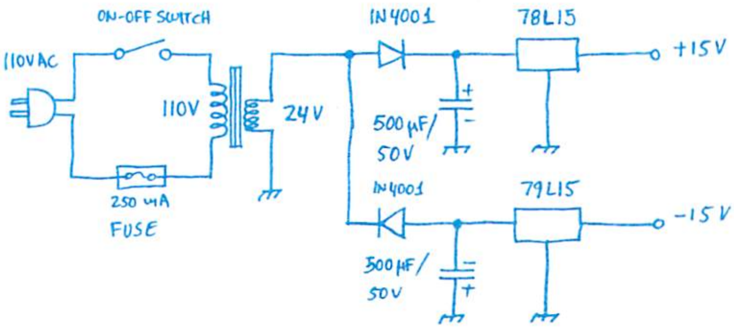
FULL SCALE ADJUSTMENT





10-TURN POT. ADJUST FOR 6.5 V OUT WITHOUT INPUT SIGNAL

± 15 V POWER SUPPLY



8-BIT A/D CONVERTER

18

DESCRIPTION

The 555 timer forms an effective converter from an analog quantity such as resistance or capacitance to a number. An A/D converter converts a voltage to a corresponding number. In this project we interface an IC A/D converter to the user port, allowing the VIC to measure voltage directly, and indirectly any quantity that can be converted to a voltage.

THEORY OF OPERATION

The National ADC0809 is an 8-bit A/D converter with an 8-channel input multiplexer. In this project we will not use the multiplexer because the user port does not have the extra lines required to address it. The ADC0809 requires an external clock which is provided by the 555 timer connected as an astable multivibrator. CB2 on the user port is used to initiate the conversion process, by generating the start-of-conversion pulse. CB1 senses the end-of-conversion signal that signifies that the data lines contain valid data.

The input signal to the A/D can range from 0 to 5 volts. The 100 Ohm resistor and 0.01 uF capacitor remove noise that might be picked up by external connections. They should be located physically close to the A/D IC itself. The driving impedence of the source must be less than 1000 Ohms.

It is good practice to locate a buffer amplifier close to the source of the signal and use a shielded cable to connect the output of the buffer to the A/D input.

Two buffer designs are shown, one for DC signals and the other for AC. (AC must be level shifted and scaled so that it ranges between 0 and 5 volts before it can be input into the A/D converter).

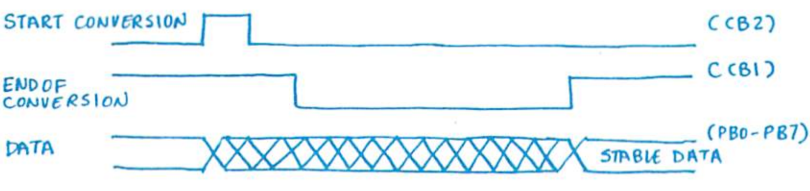
PROGRAMMING NOTES

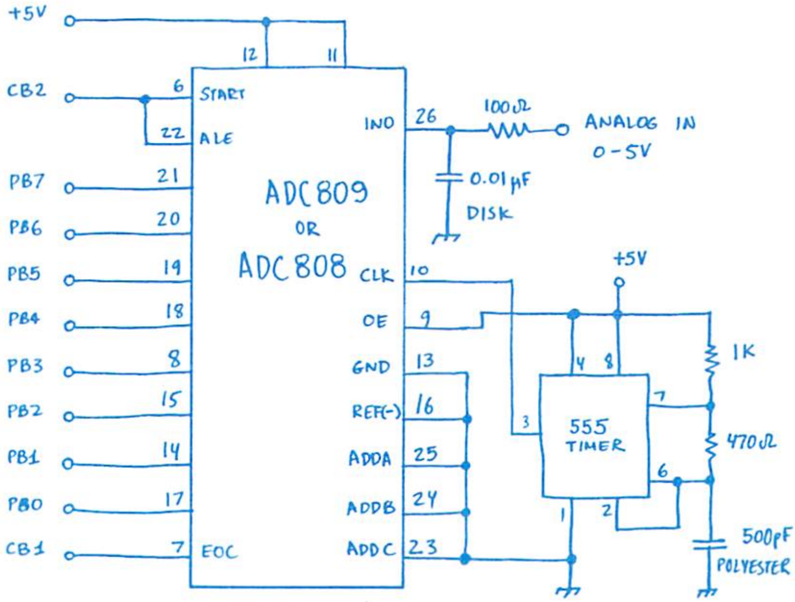
Start by making all bits of the user port inputs <POKE 37138,0>. Then put CB2 in the manual output mode low and CB1 to trigger on a positive transition <POKE 37148,(PEEK(37148) AND 240) OR 208>. (208= 11010000).

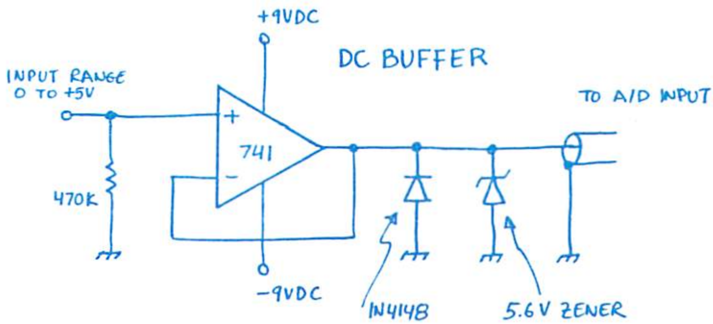
To start a conversion, bring CB2 high <POKE 37148,(PEEK (37148) OR 240>, and them low <POKE 37148, (PEEK (37148) AND 240) OR 208>.

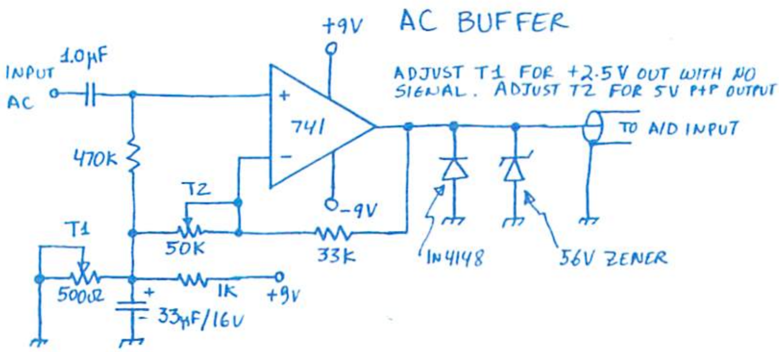
The conversion is complete after about 120 µs, so its much faster than BASIC statement execution. Thus you can directly read the results by <PRINT PEEK (37136)>.

If your program is in machine language it will operate much faster so you will have to check the interrupt flag corresponding to CB1 to detect the end of conversion before proceeding.




+5V
CB2
PB7
PB6
PB5
PB4
PB3
PB2
PB1
PB0
CB1
12
11
6 START
22 ALE
21
20
19
18
8
15
14
17
7 EOC
ADC809
OR
ADC808
IN0 26
CLK 10
OE 9
GND 13
REF(-) 16
ADDA 25
ADDB 24
ADDC 23
100Ω
ANALOG IN
0 - 5V
0.01μF
DISK
+5V
555
TIMER
1K
470Ω
500pF
POLYESTER


+9VDC
DC BUFFER
INPUT RANGE
0 TO +5V
TO A/D INPUT
741
470K
-9VDC
IN4148
5.6V ZENER


AC BUFFER
+9V
INPUT
AC
1.0μF
741
ADJUST T1 FOR +2.5V OUT WITH NO SIGNAL. ADJUST T2 FOR 5V P+P OUTPUT
TO A/D INPUT
470K
T2
50K
33K
-9V
T1
500Ω
1K
+9V
33μF/16V
IN4148
5.6V ZENER

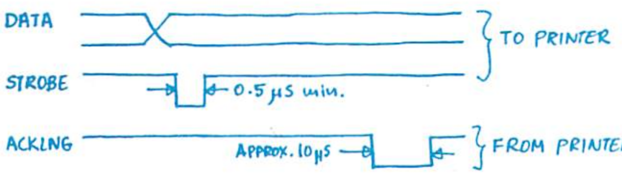
DESCRIPTION

The EPSON MX-80 printer gives very good printout quality and excellent dot graphics. In addition, it is very reliable, faster than the Commodore 1515, uses standard width paper and can be bought mail order at about what you will pay for the 1515 at a Commodore dealer. This project details a very simple interface to the MX-80 via the user port. While limited in applications (BASIC LIST and PRINT# won't work) it does give you access to the MX-80 dot graphics and allows word processing using a word processor like the WORD WHIZ/80.

THEORY OF OPERATION

The EPSON MX-80 comes standard with what is known as a 'Centronics Parallel Interface', named after a company the MX-80 almost put out of business. Centronics, a U.S. company, was the first to introduce and make popular the low cost dot matrix printer and their interface became the de facto industry standard (the NEC 8020A and the C. Itoh PROWRITER use the same interface and this project applies to them as well).

The Centronics interface is a handshaking 8-bit parallel interface that works as follows: The computer puts the data (ASCII code of character to be printed) out on the port and generates a low pulse on a line called 'STROBE' to indicate to the printer that valid data is available. The printer accepts the data and replies by pulsing low a line called ACKNLG (ACKNowLedGe). If the printer is busy no data is lost because it does not generate the acknowledge until it has accepted the data, and the computer waits for the ACKNLG pulse before proceeding. The timing diagram of the process is shown below.



This type of output with handshake suits the user port perfectly. PB0-PB7 can carry the data, CB2 generates the strobe and CB1 senses the ACKNLG pulse. All that is required in terms of hardware for the interface is a cable with connectors to plug to the user port on one end and to the printer on the other. The wiring of the cable follows the list shown.

PROGRAMMING

First the port is configured for output by POKE 37138,255. Then CB2 is configured for pulse output when data is written to the user port and CB1 is set up to detect the negative transition of ACKLNG. This is done by writing 1010 to the high order bits of the PCR.

```
POKE 37148, (PEEK (37148) AND 15) OR 160
```

Data is output to the printer by POKEing the ASCII codes to the user port. For example, to print an 'A', POKE 37136,65. The printer operation is sometimes (during carriage returns for example) slower than the execution of BASIC so it is necessary to check for the acknowledge (bit 4 of the interrupt flag register (loc. 37149).

```
200 IF (PEEK(37149) AND 16)=0 GO TO 200
```

For example, here is a program that will print out 10 A's.

```
100  REM TEST
110  POKE 37138,255
120  POKE 37148,(PEEK (37148) AND 15) OR 160
130  FOR I=1 TO 10
140  POKE 37136,65
150  IF (PEEK (37149) AND 16) = 0 GO TO 150
160  NEXT I
170  END
```

WIRING LIST

COMPUTER		PRINTER	
SIGNAL	PIN	SIGNAL	PIN
CB2	M	STROBE	1
PB7	L	DATA 8	9
PB6	K	DATA 7	8
PB5	J	DATA 6	7
PB3	F	DATA 4	5
PB2	E	DATA 3	4
PB1	D	DATA 2	3
PB0	C	DATA 1	2
CB1	B	ACKLNG	10
GND	A	GND	19

DUAL I/O PORT — 20

DESCRIPTION

The user port on the VIC is a very convenient feature that simplifies interfacing significantly. Since more of a good thing is a better thing, here is a way to add more I/O ports to your VIC. This project describes a dual I/O port using the expansion bus, and the 6522 VIA for compatibility with the user port.

THEORY OF OPERATION

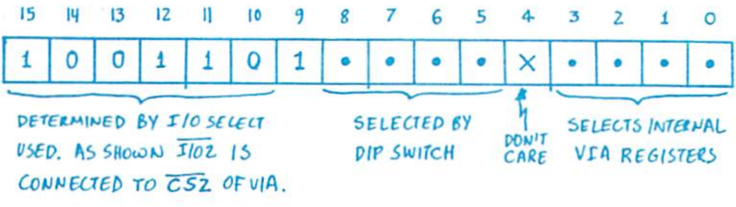
The VIC designers have made provisions for I/O interfacing by setting aside two memory blocks ($9800 to $9BFF and $9C00 to $9FFF). There are many ways to implement input or output ports, but nothing rivals the versatility of the 6522 VIA IC which was specifically designed to provide I/O ports for the 6502. An additional reason to use the VIA is software compatibility with the user port format.

Interfacing the VIA is straightforward. To the outside world it looks like two ports, A and B with handshake lines (CA1,CA2 and CB1,CB2). To the 6502 the VIA looks like 16 registers, addressed by lines RS0-RS3 on the VIA. Two chip selects are provided, one active high (CS1) and one active low (CS2). A bidirectional data bus connects to the 6502 bus and the control signals RESET, R/W, IRQ and o2 are directly compatible with the equivalent signals on the 6502.

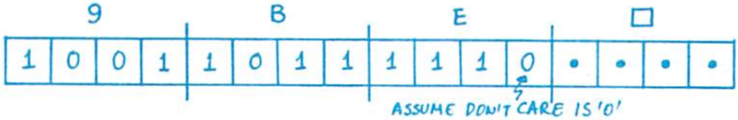
In our design, CS2 is driven by either I/O2 or I/O3, the I/O area select signals available at the expansion bus. CS1 is driven by the output of a comparator that places the 6522 in the upper 16 positions of a total of 32 address blocks. Each block has 32 locations but the VIA uses only 16. A 4 position DIP switch configures the comparator to respond to the addresses within a selected block.

PROGRAMMING

The VIA will respond to addresses in the range shown in the diagram below:



As an example, assume that I/O2 is used to drive CS2 and that all switches in the DIP switch are open ('1'). The addressing will be:



The □ represents on of the registers of the VIA as described in page 218 of the VIC-20 Programmers Reference Guide.

If I/O3 was used instead of I/O2, the address would have been 9FE□ If in this second case switch S0 was closed ('0') the address would be 9FC□.

CONSTRUCTION HINTS

Use either PC or wirewrap construction. If you would like plug compatibility with the user port, put on the top of the board a 12/24 pin, .156'' spacing PC connector for PB0-PB7 and CB1,CB2. Port A can go to a 16 pin socket.