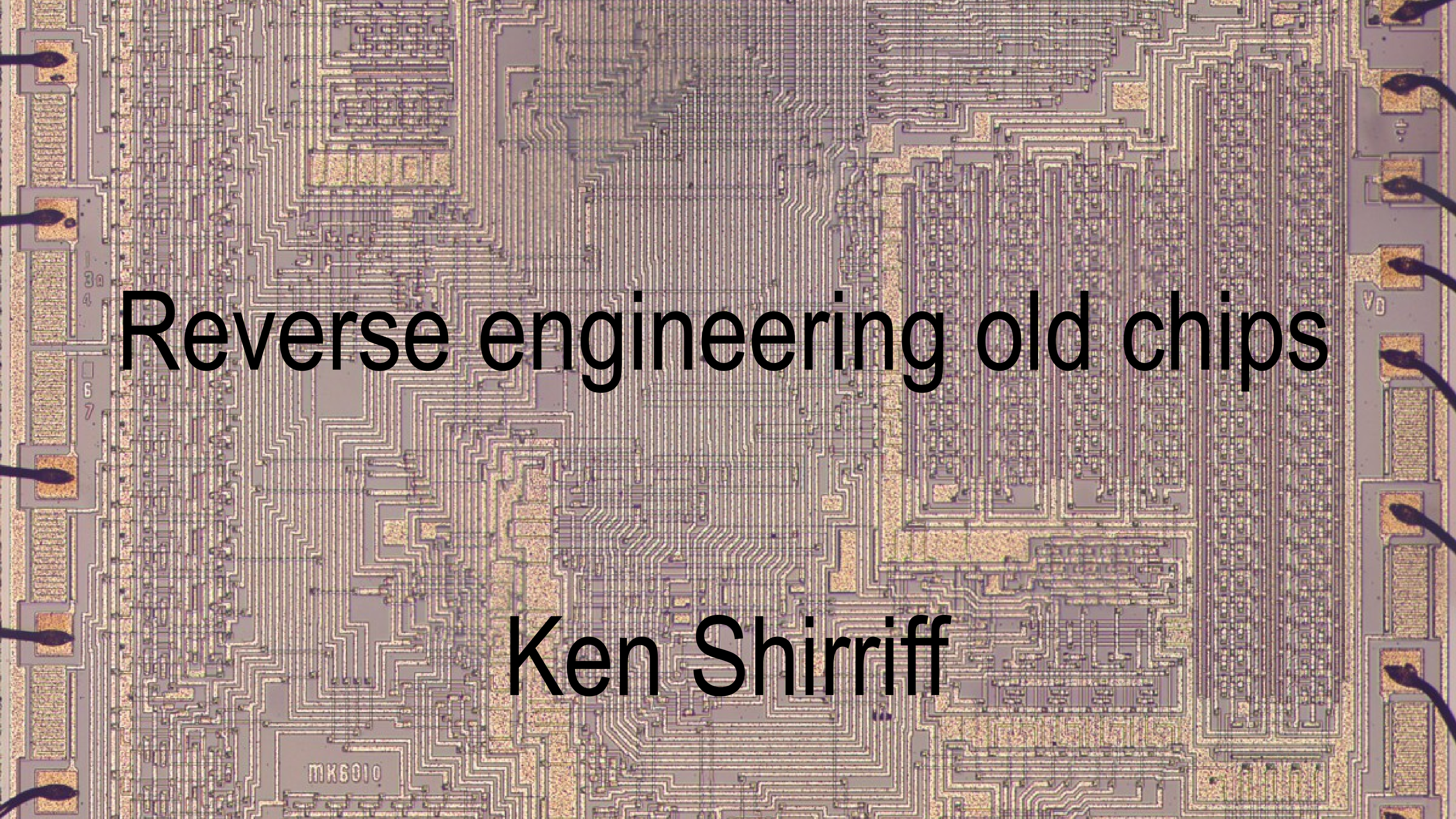
Reverse engineering old chips

Ken Shirriff



Z80
(1976)

8-bit CPU

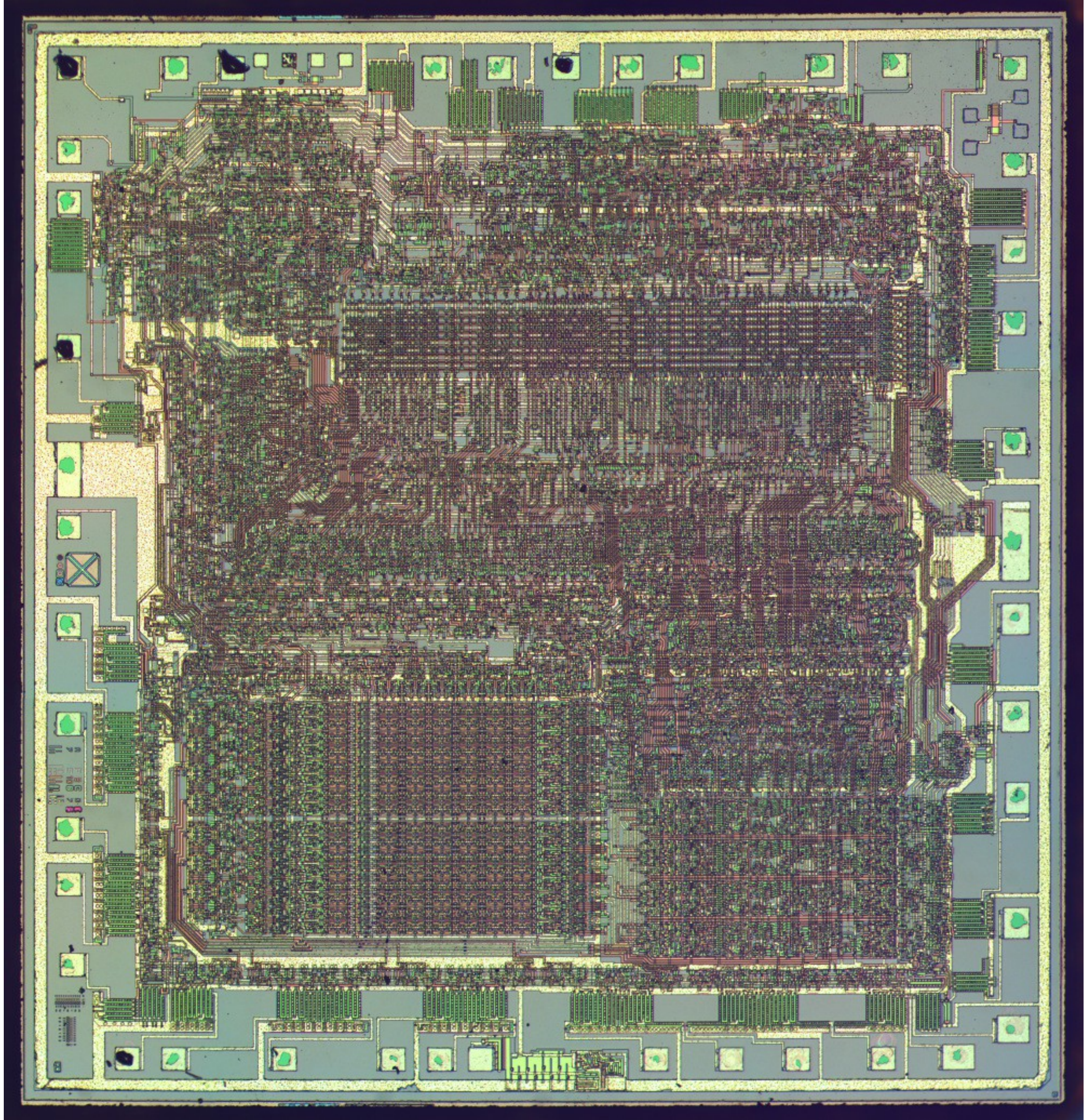


Photo: zeptobars.com

Z80

(1976)

8-bit CPU

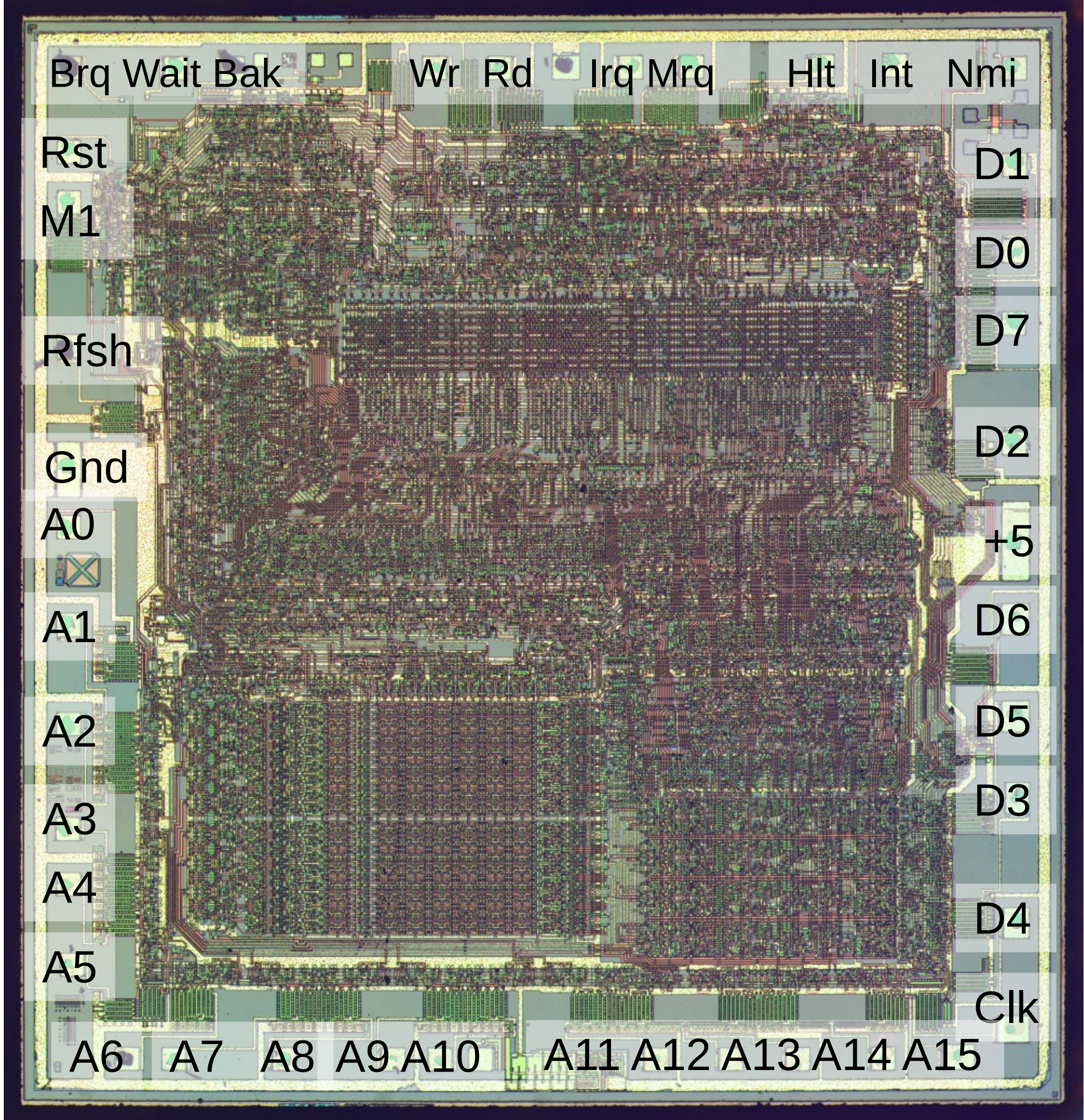
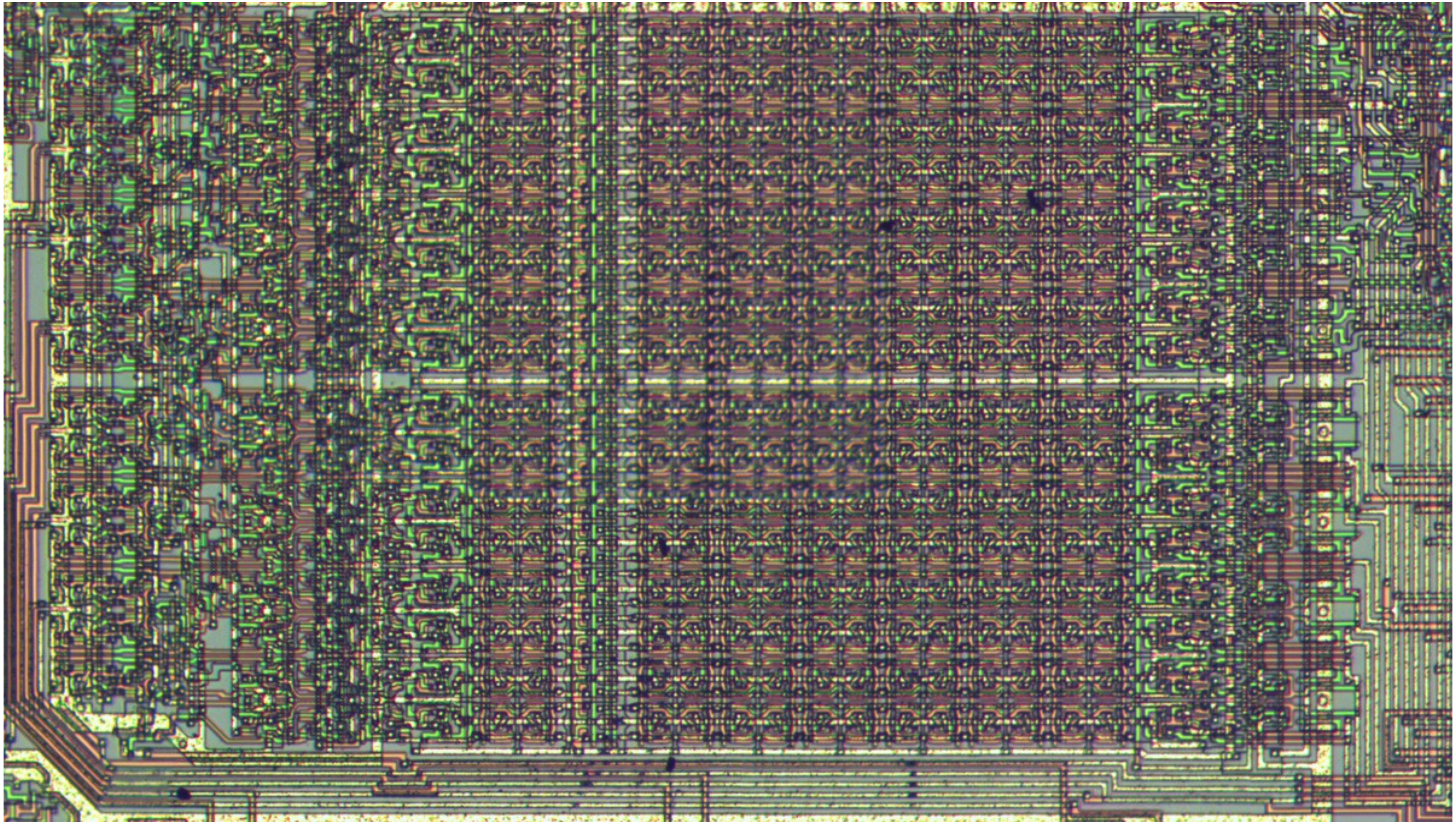
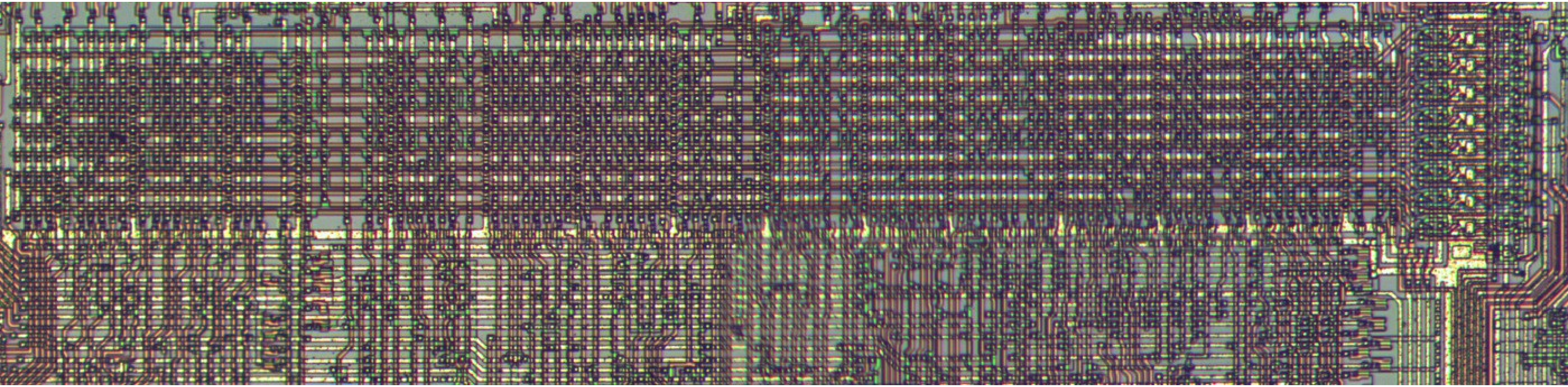


Photo: zeptobars.com

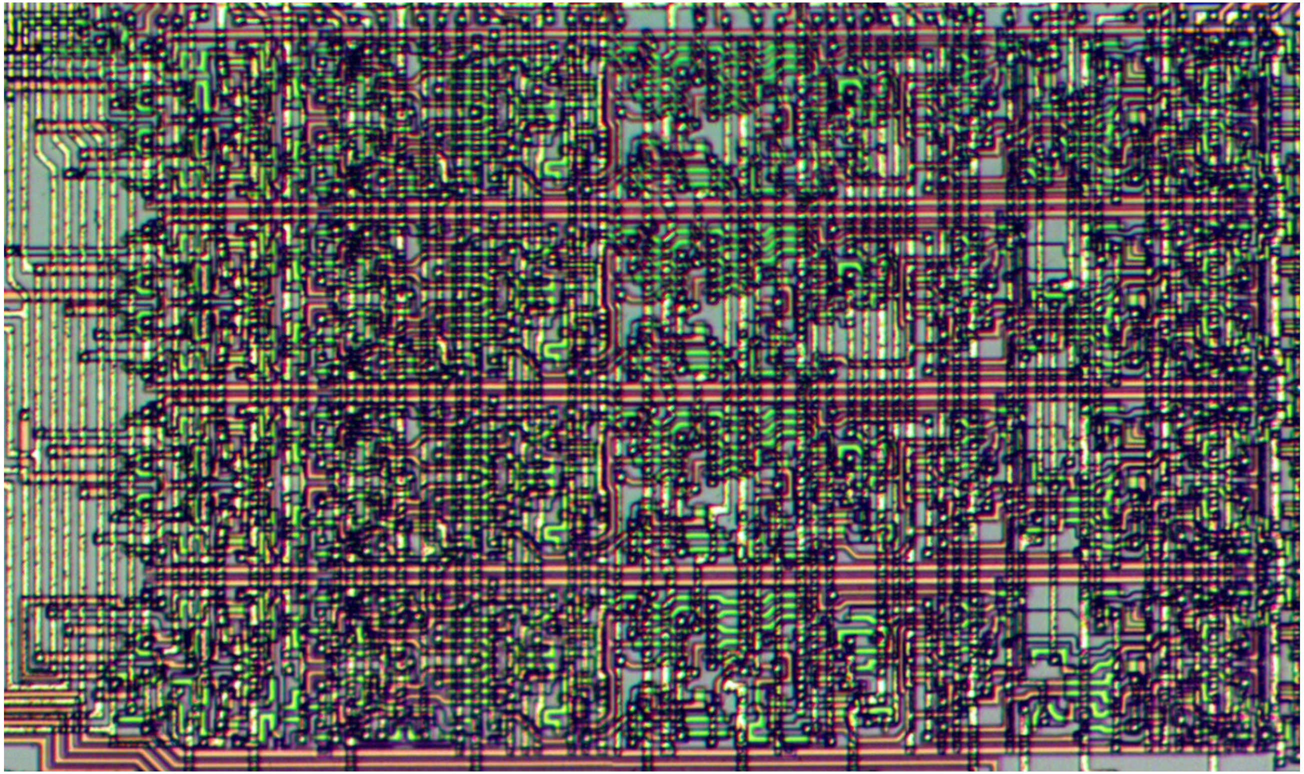
Register File



Instruction decoding

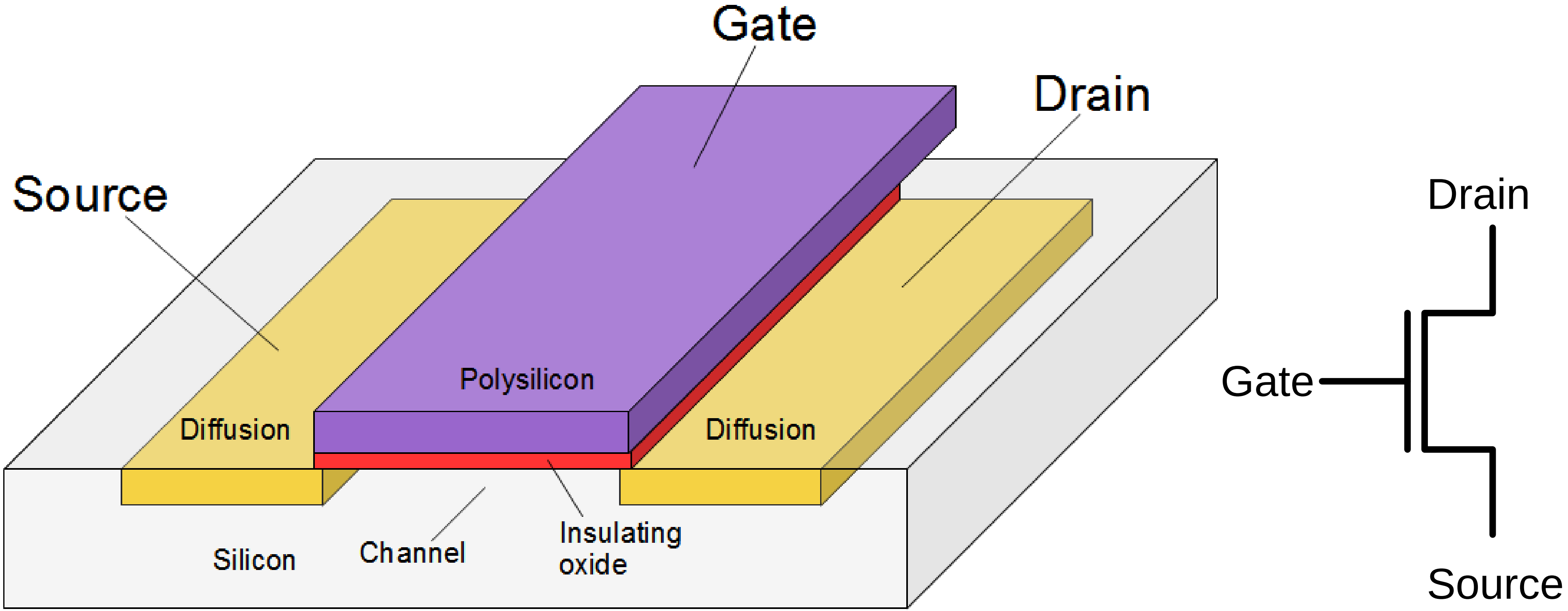


ALU (Arithmetic-Logic Unit)

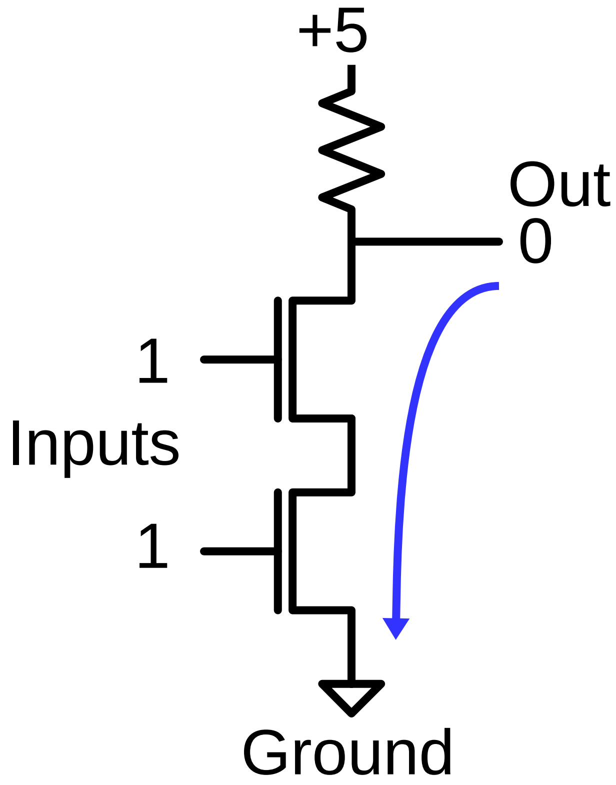


Only 4 bits!

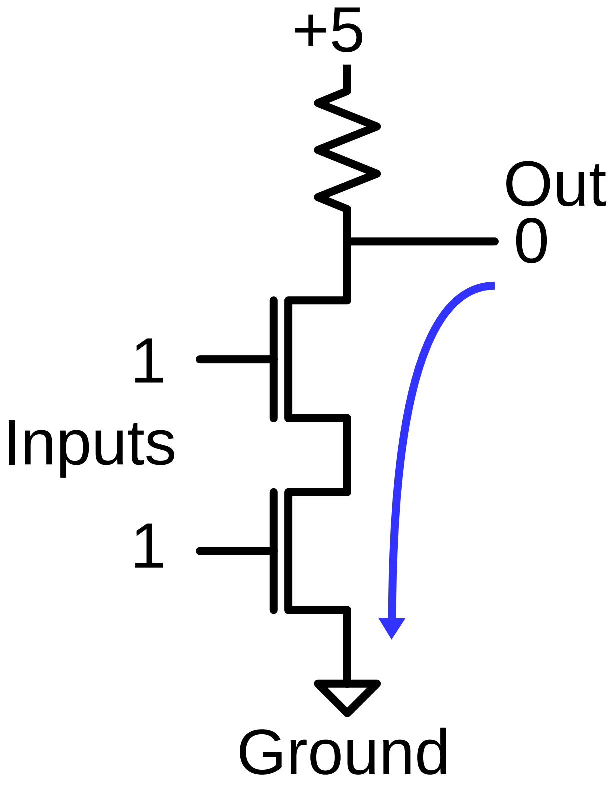
MOS transistors

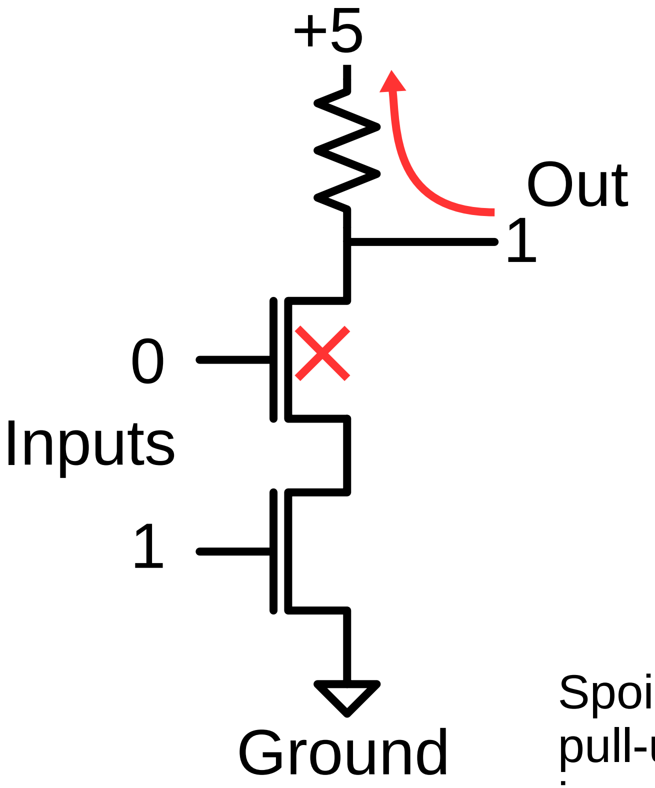


NAND gate



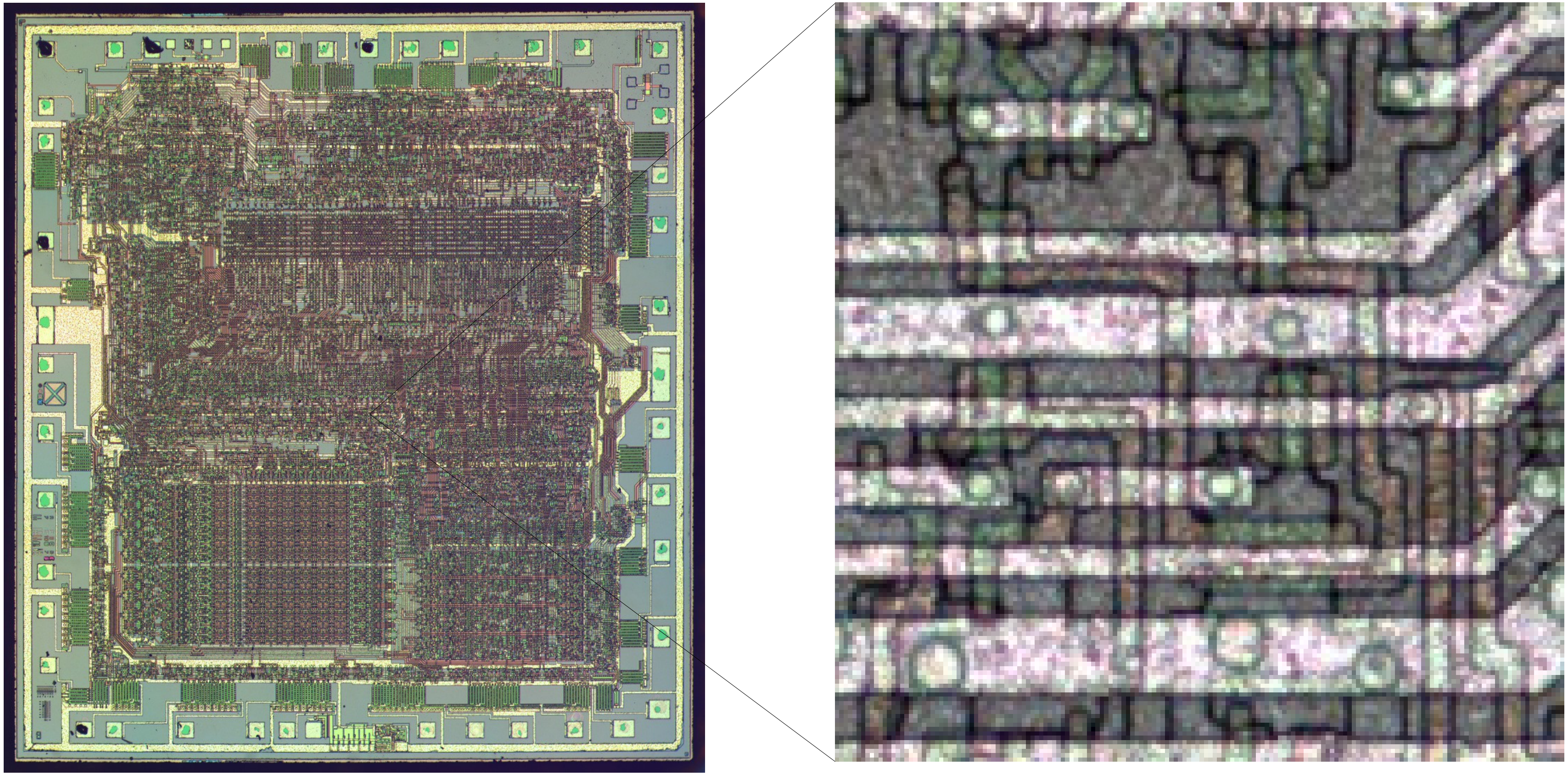
NAND gate



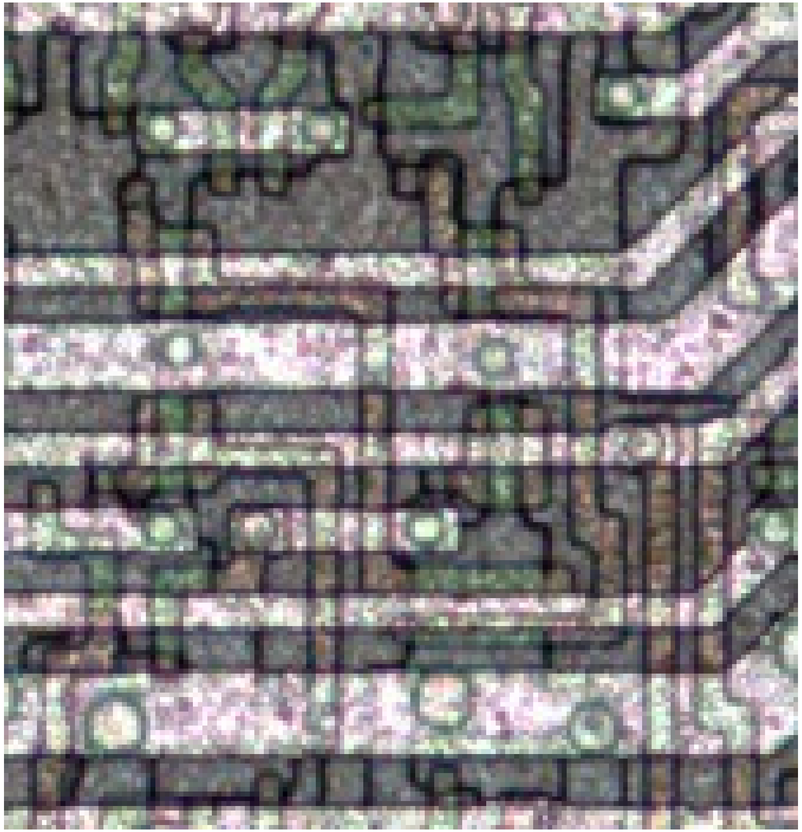


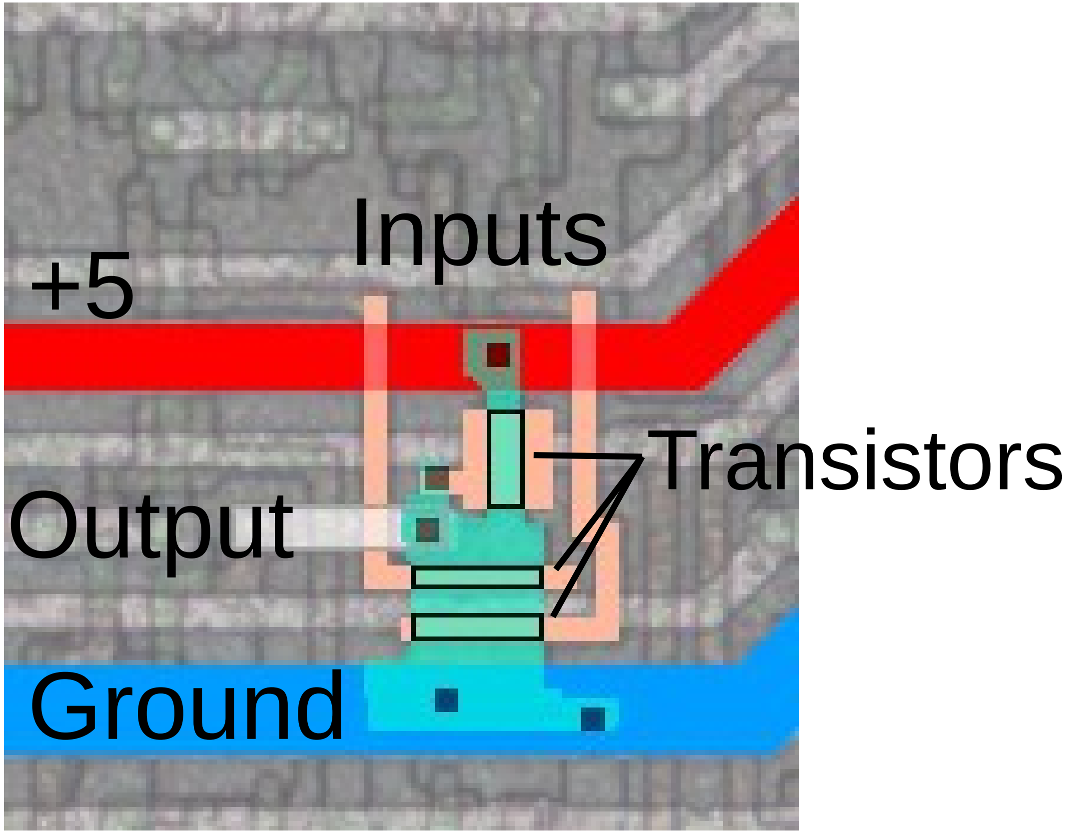
Spoiler:
pull-up resistor
is really a transistor

What do gates really look like?

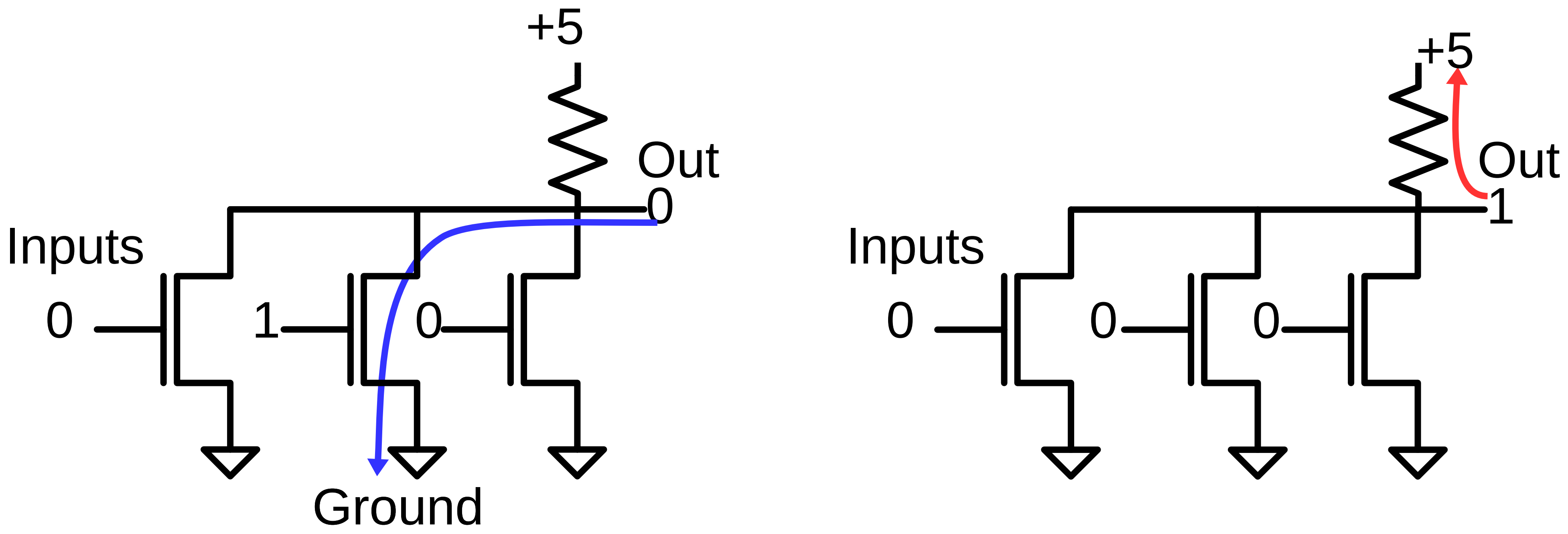


NAND gate

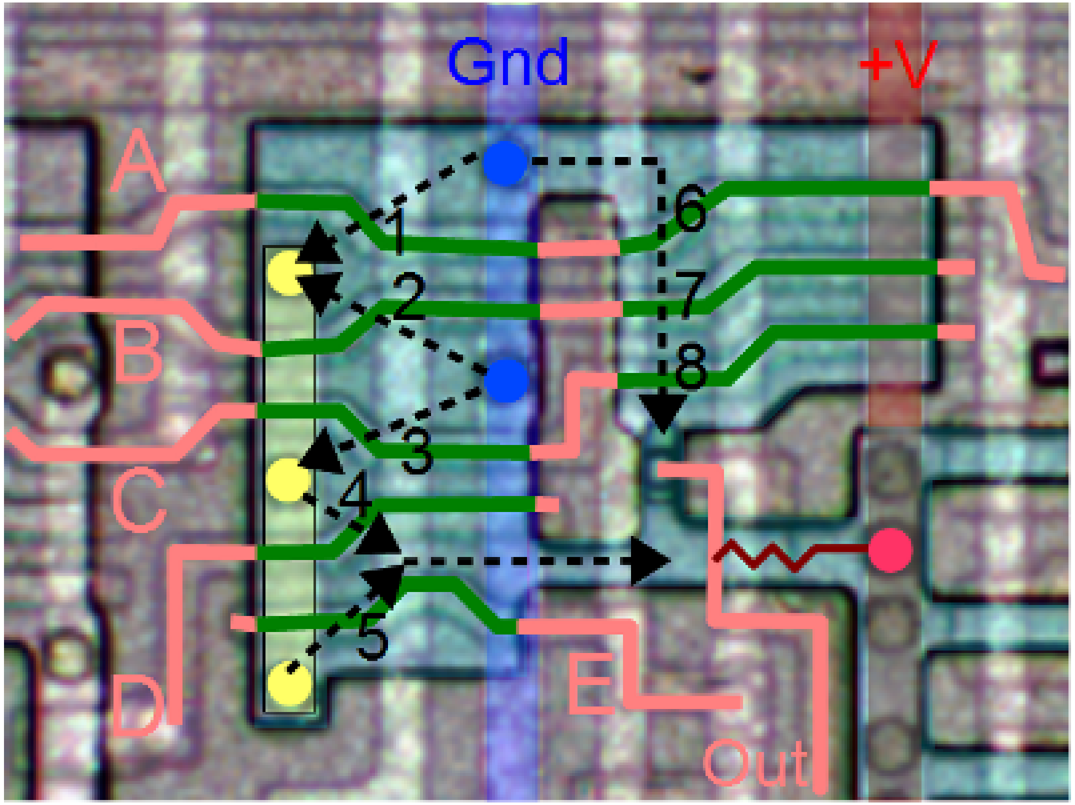


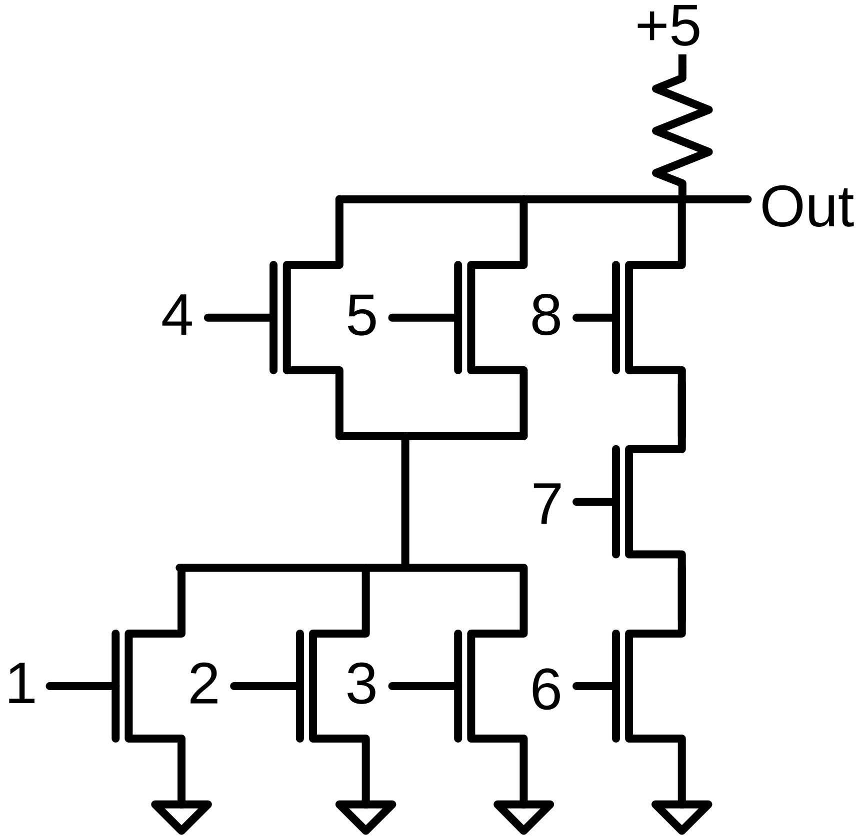


NOR gate

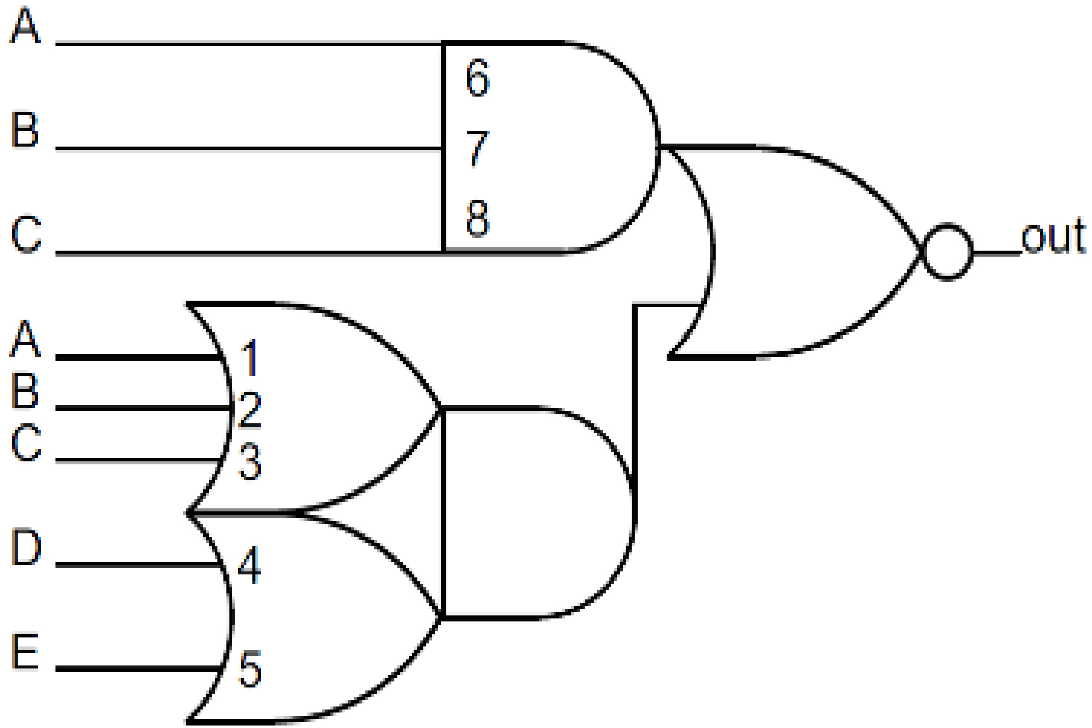


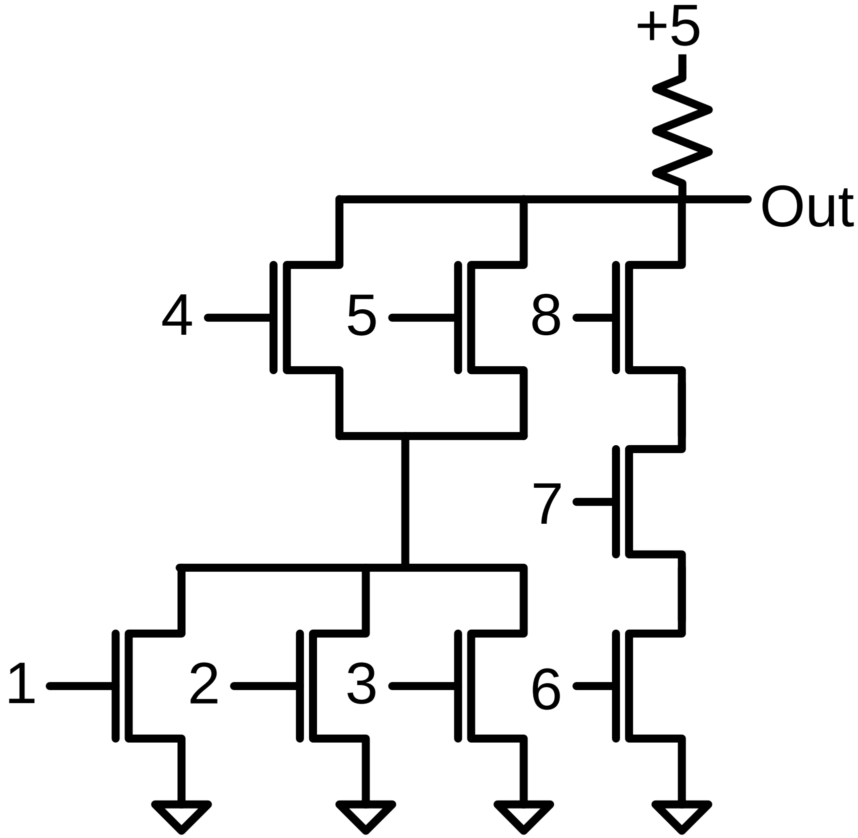
Gates get weird in the ALU





Computes sum, AND, OR, XOR



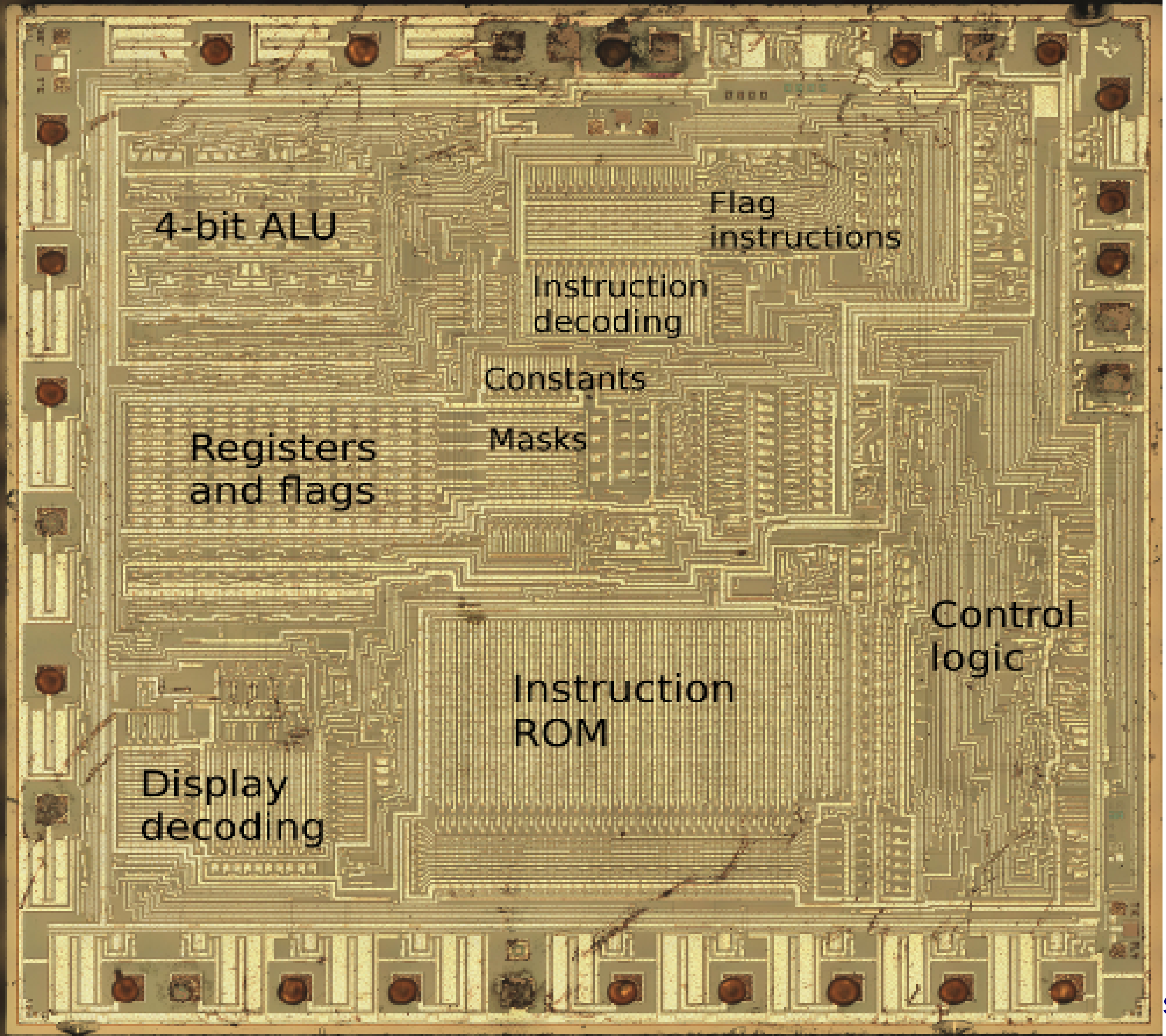


More info: righto.com/Z80

Sinclair Scientific Calculator (1974)

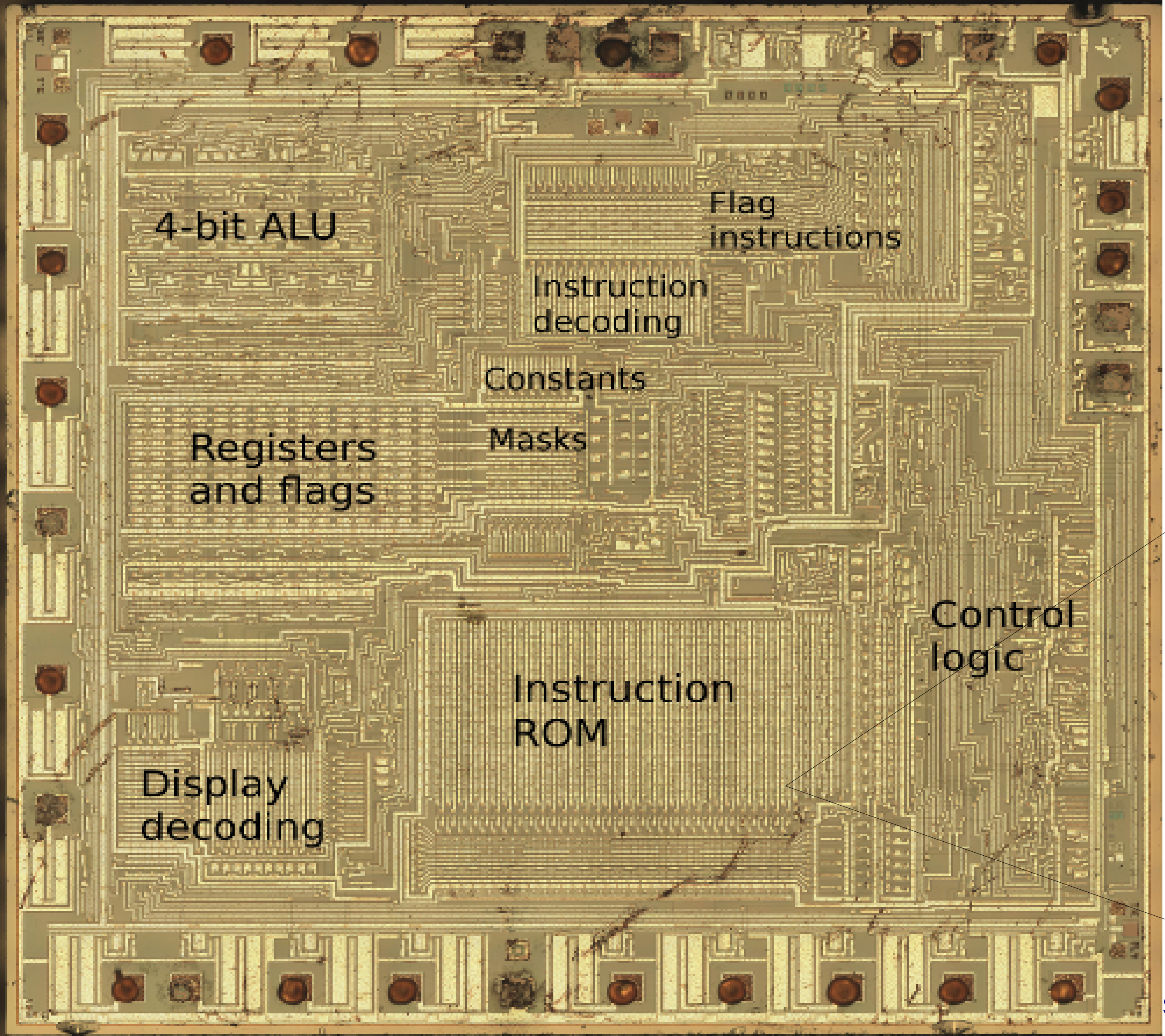
Reprogrammed TI 0800 4-function calculator chip to support trig, log. How?

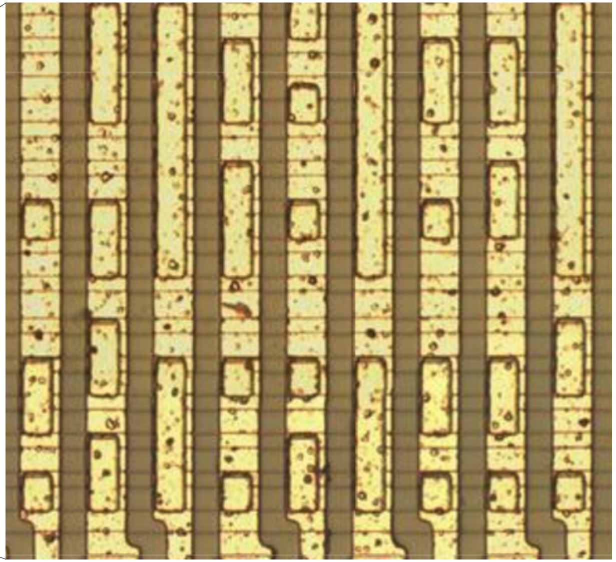
TMS 0805 calculator chip



siliconpr0n.org

Can see bits in the 320-word instruction ROM





siliconpr0n.org

Built instruction-level simulator


```
         AKC    ALL
For display, A's MANT starts in digit 5. For computation, I
C holds the previous value, with MANT starting in digit 6.
MAINLOOP SLLA   MANT     Shift mantissa for display
         AKB    ALL      clear B
WAITSCAN SYNC            loop until no key pressed
         SCAN
         BINE   WAITSCAN
WAITKEY  WAITNO WAITED   wait for key
WAITED2  SYNC            debounce: still pressed?
         SCAN
         BIE    WAITKEY  loop if no key
         SYNC
         SRLA   MANT     MANT is shifted right during calcu
         BKO    LOWERKEY sequentially scan key columns
         BKO    PLUSKEY
         BKO    MINUSKEY
         BKO    DIVKEY
         BKO    MULTKEY
         BKO    UPPERKEY
         BKO    EKEY
         BKO    ZEROKEY
         EXAB   ALL      save A in B, A=0
         AKCN   DIGIT1   get digit by incrementing until co
         EXAB   ALL      restore A, B holds count
         BINE   MAINLOOP start over if nothing pressed
ZEROKEY  TFB    EMODE    B holds key 0-9
         BINE   EDIGIT
If OPDONE, a digit starts a new number in A, leaving the pr
         TFB    OPDONE   if OPDONE...
         BIE    LABEL33
         AKA    ALL      then clear A and OPDONE
         ZFB    OPDONE
LABEL33  ACKA   DIGIT    C holds digit position
BSHIFT   SRLB   ALL      shift B right C times.
         SAKA   DIGIT1   decrement A
         BIE    BSHIFT   (no borrow)
```

Autospeed

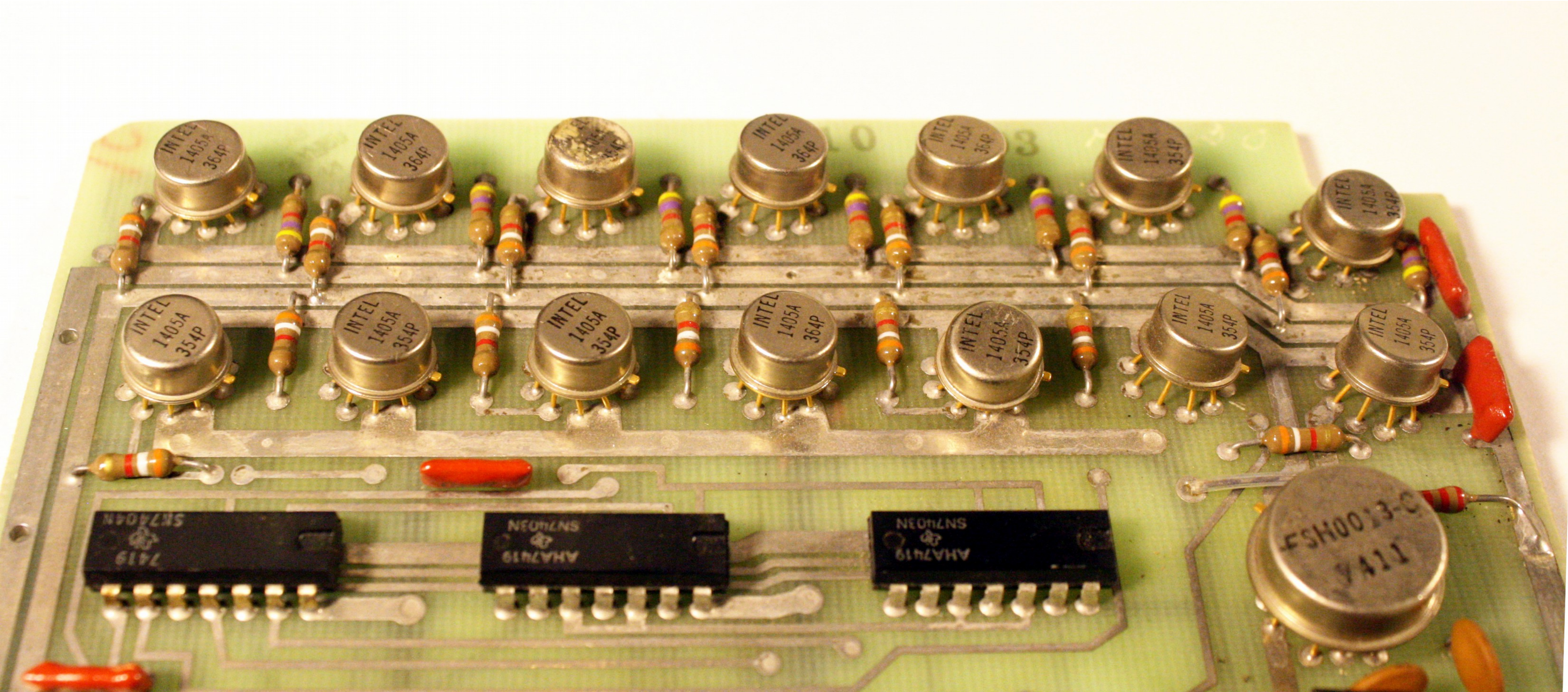
Decimal algorithms

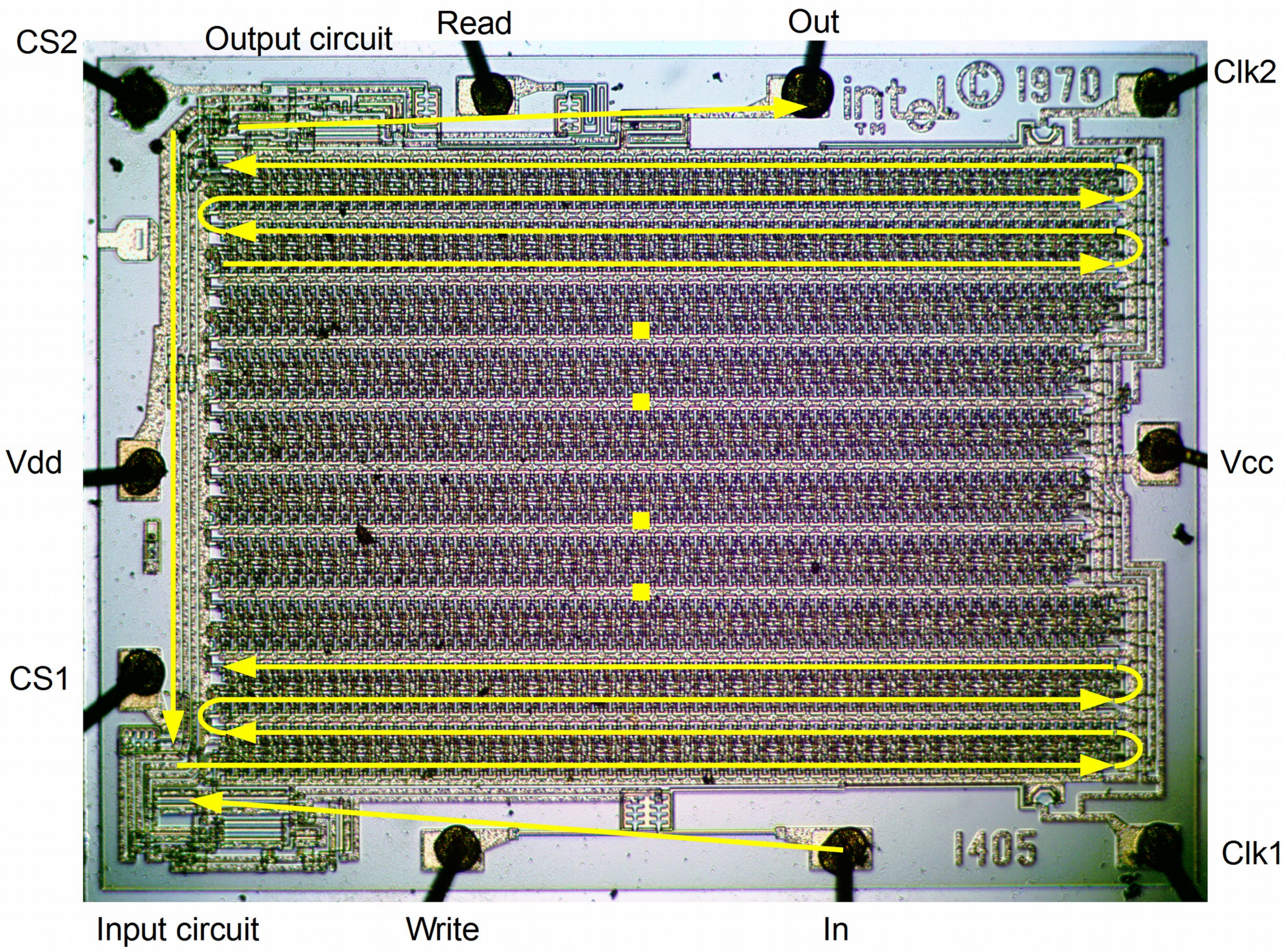
Trig: repeated rotates by .001 rad

Log: powers of 0.99

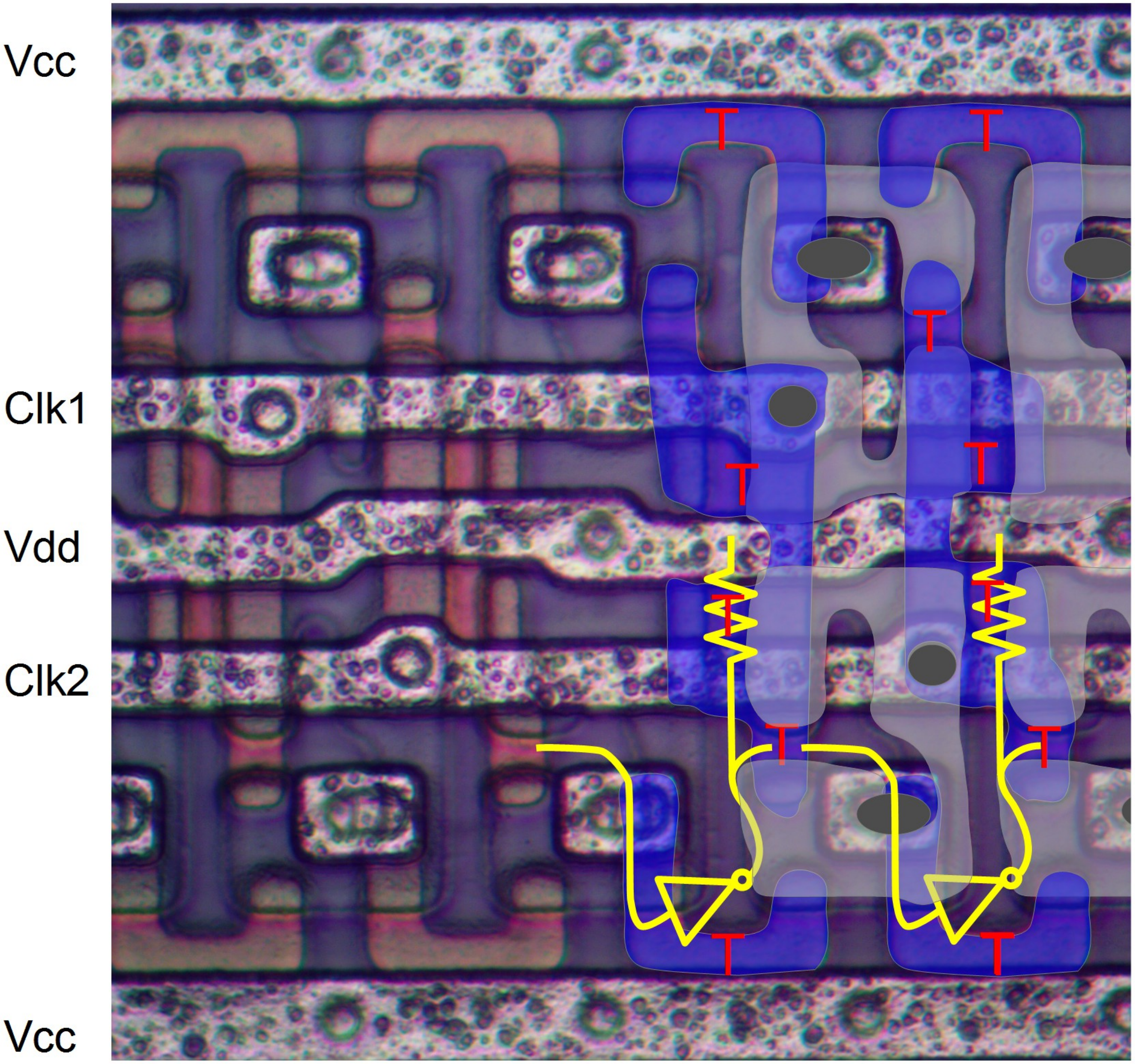
More info: righto.com/sinclair

Intel shift-register memory (1970)



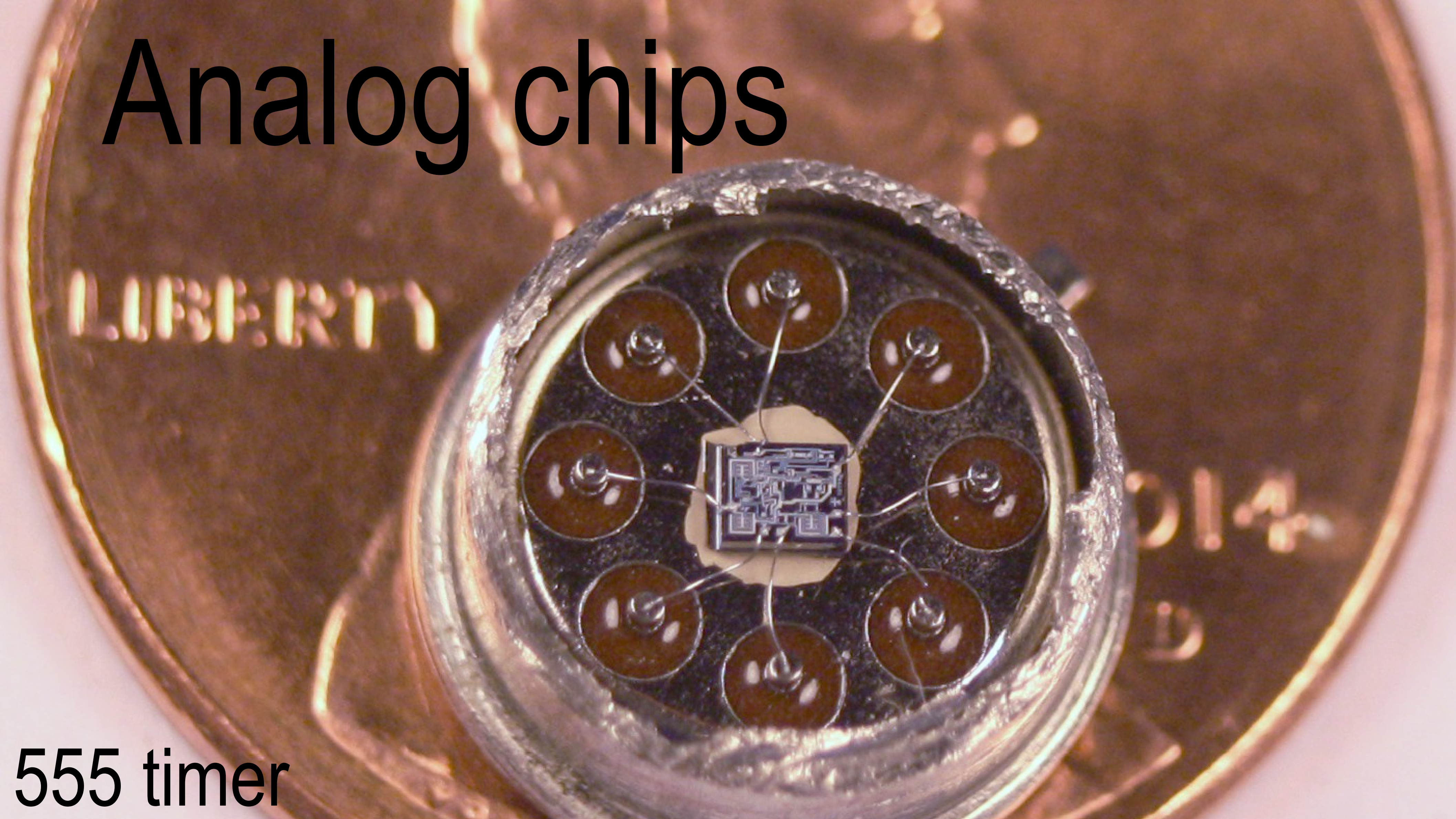


512 bits circulate.
Up to .5 ms wait

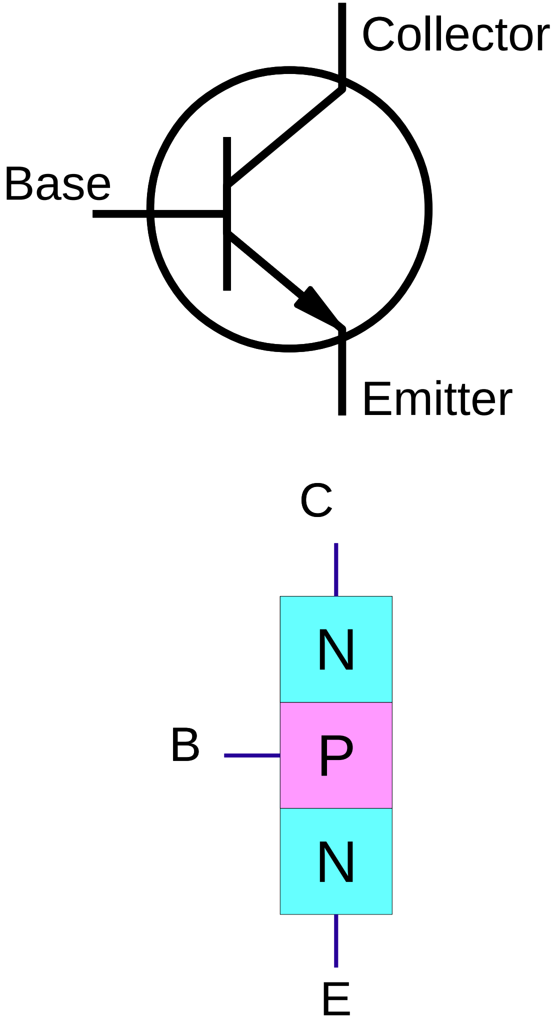


More info: righto.com/shift

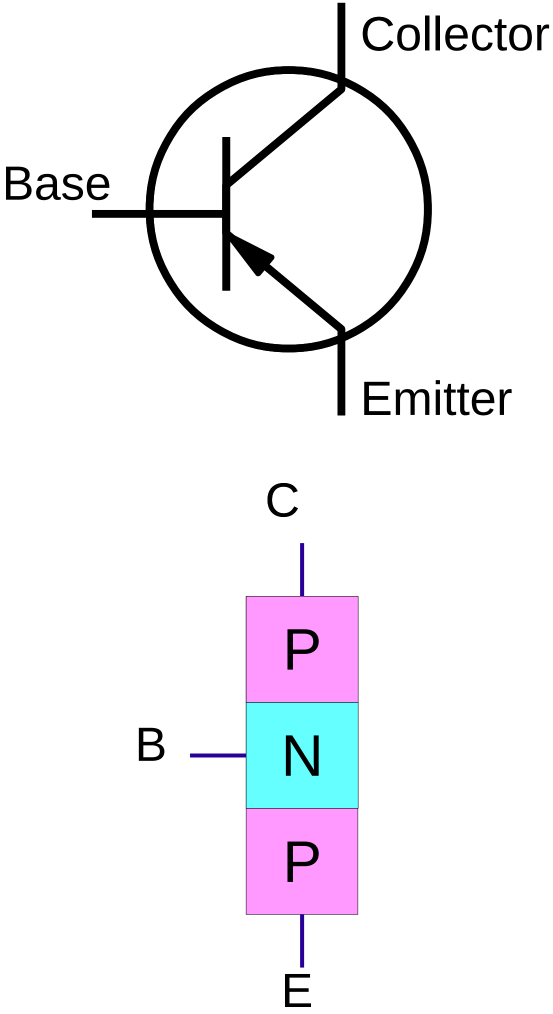
Analog chips



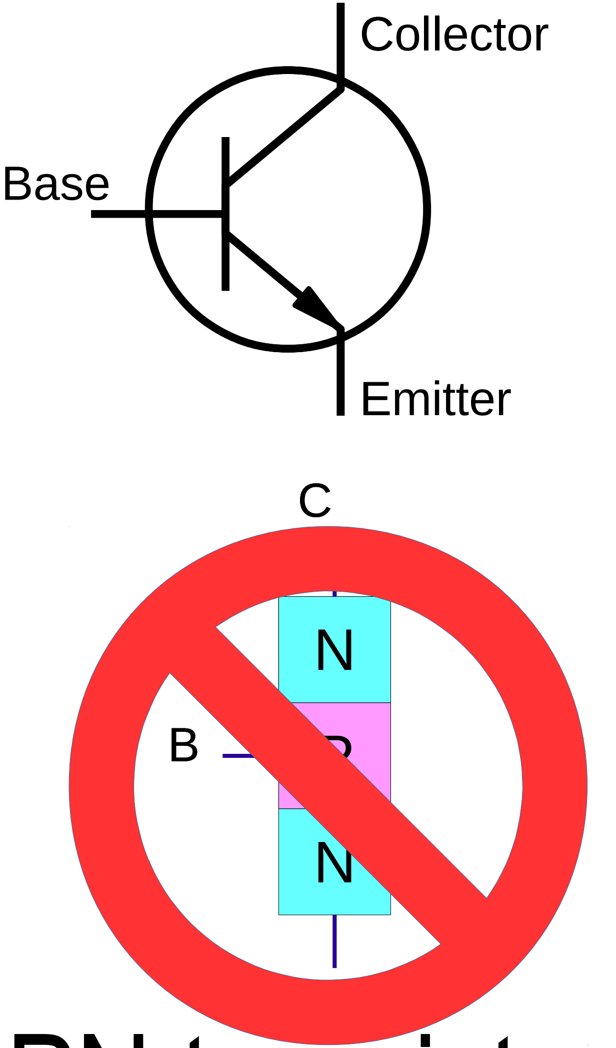
555 timer



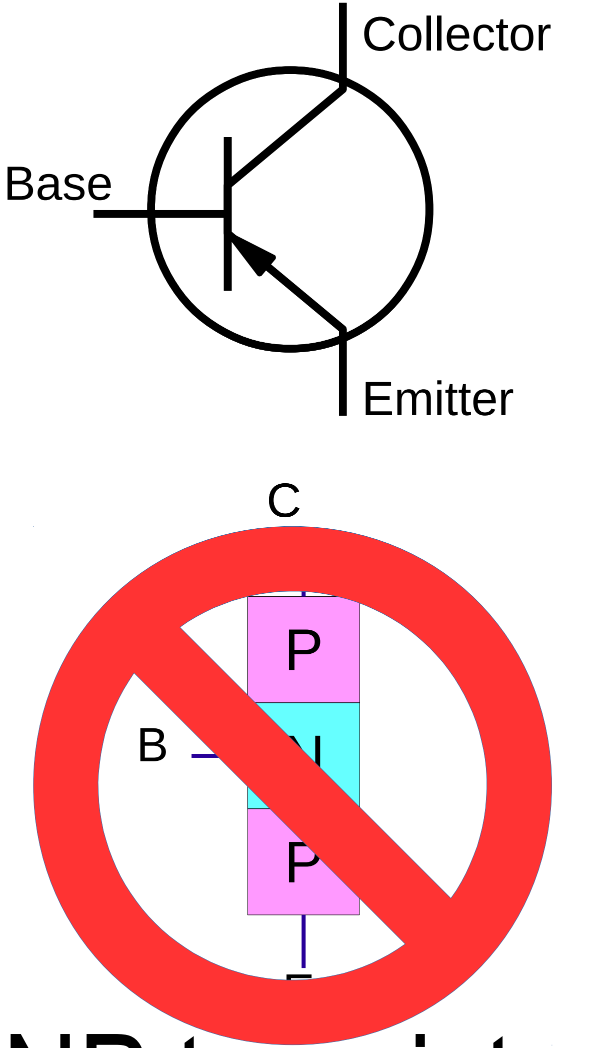
NPN transistor



PNP transistor

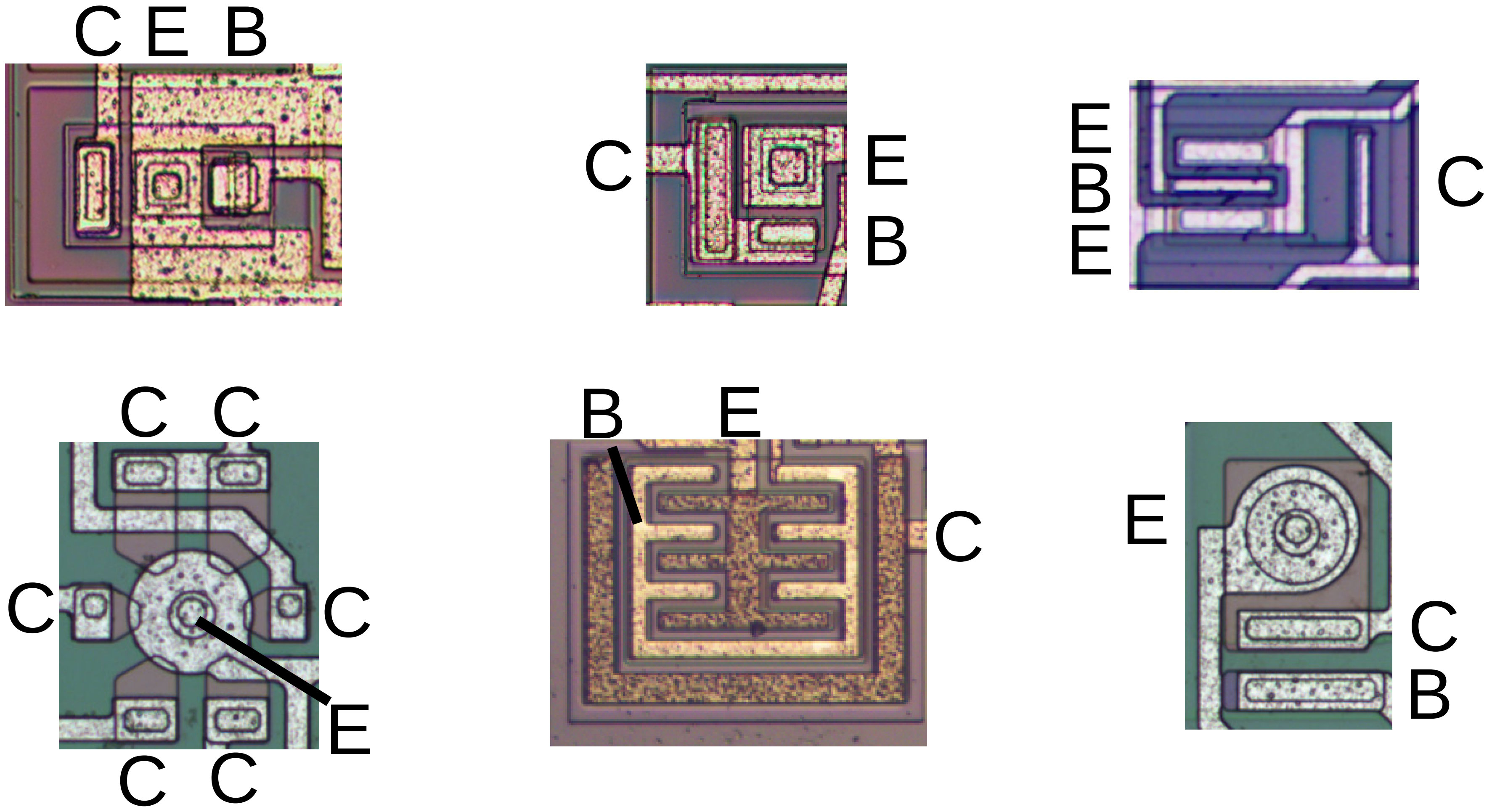


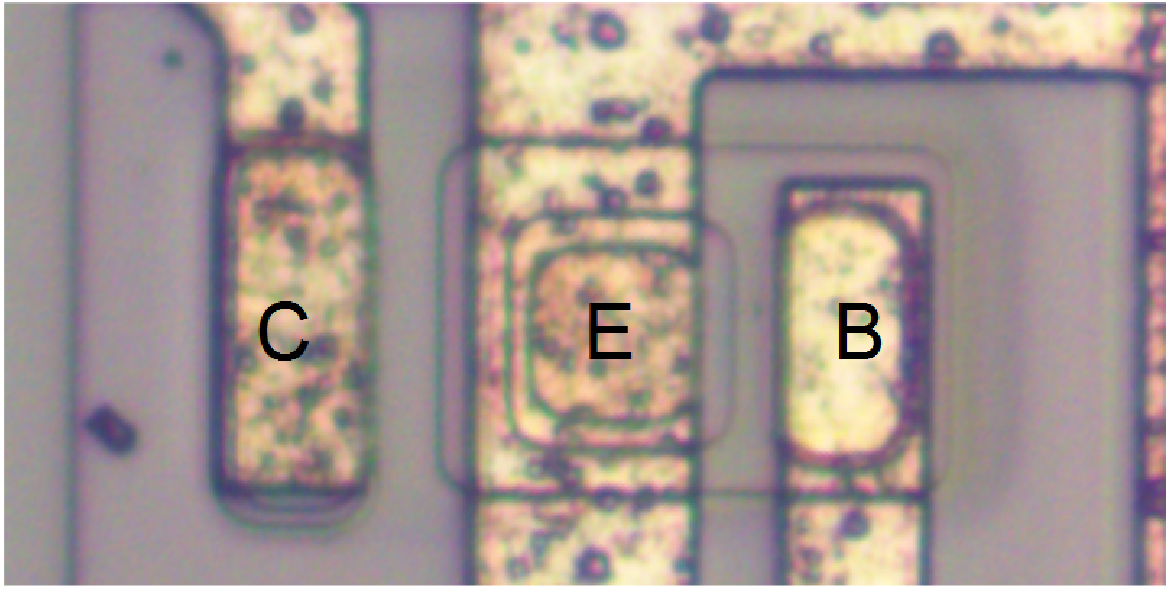
NPN transistor

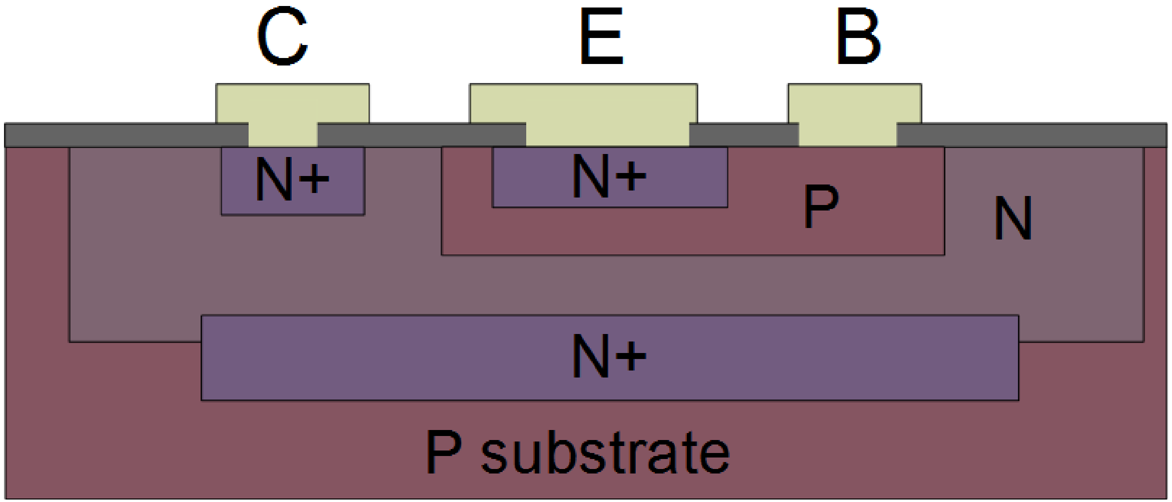


PNP transistor

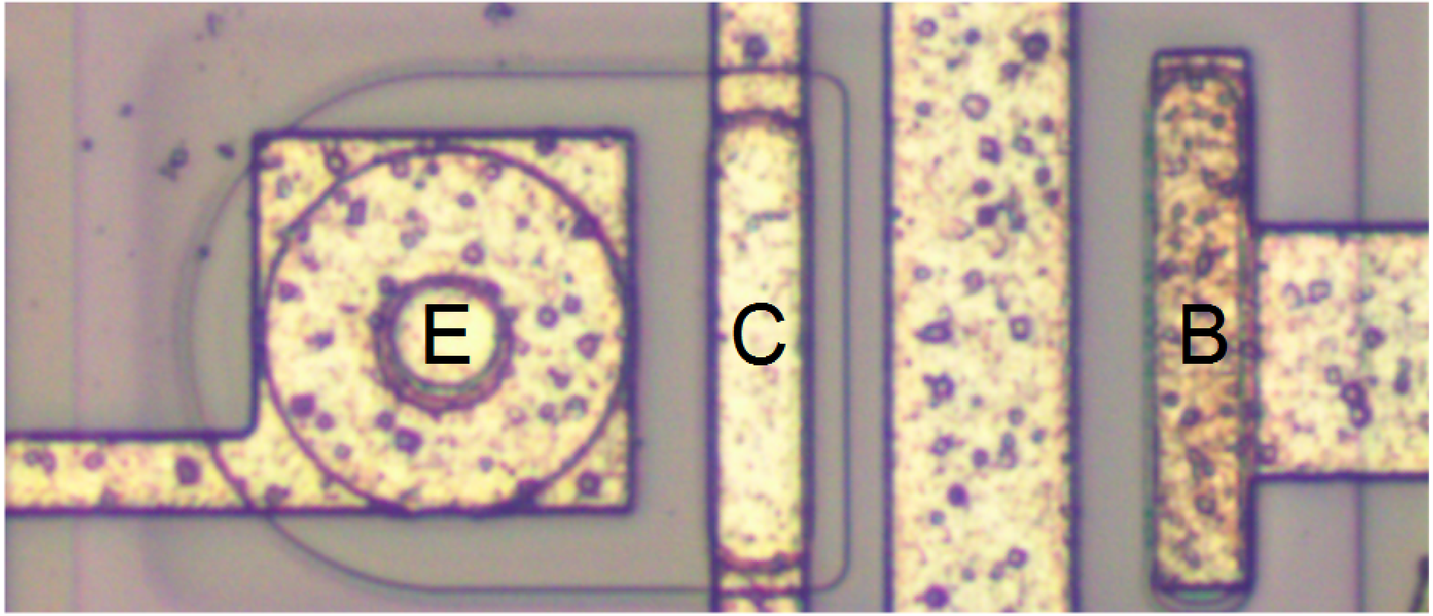
What bipolar transistors really look like

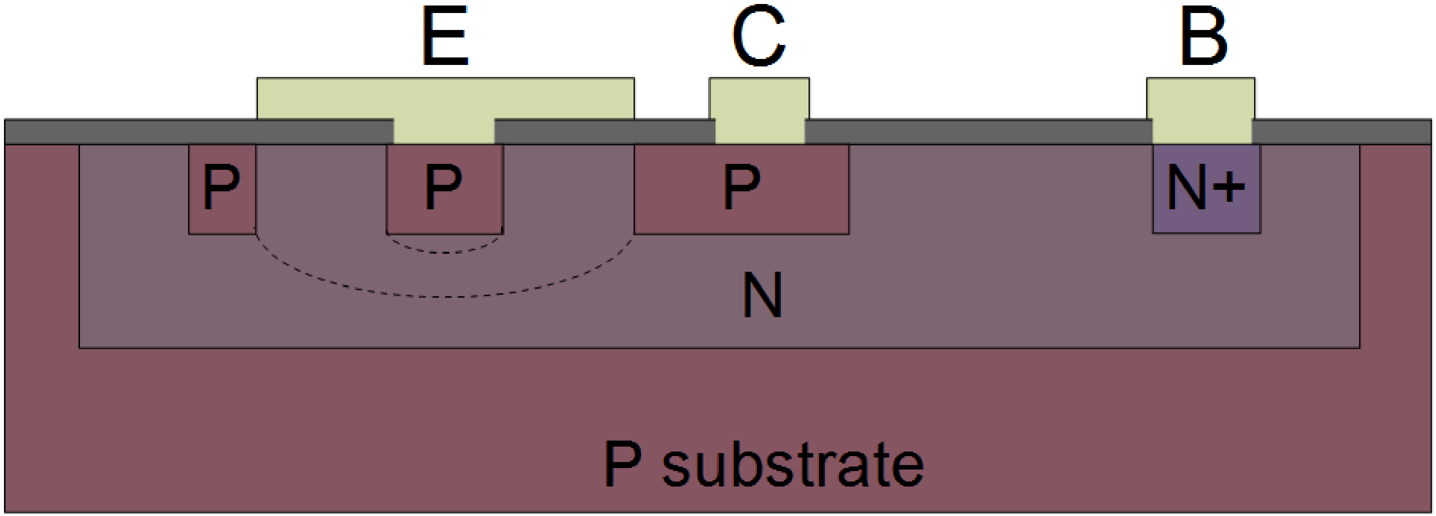




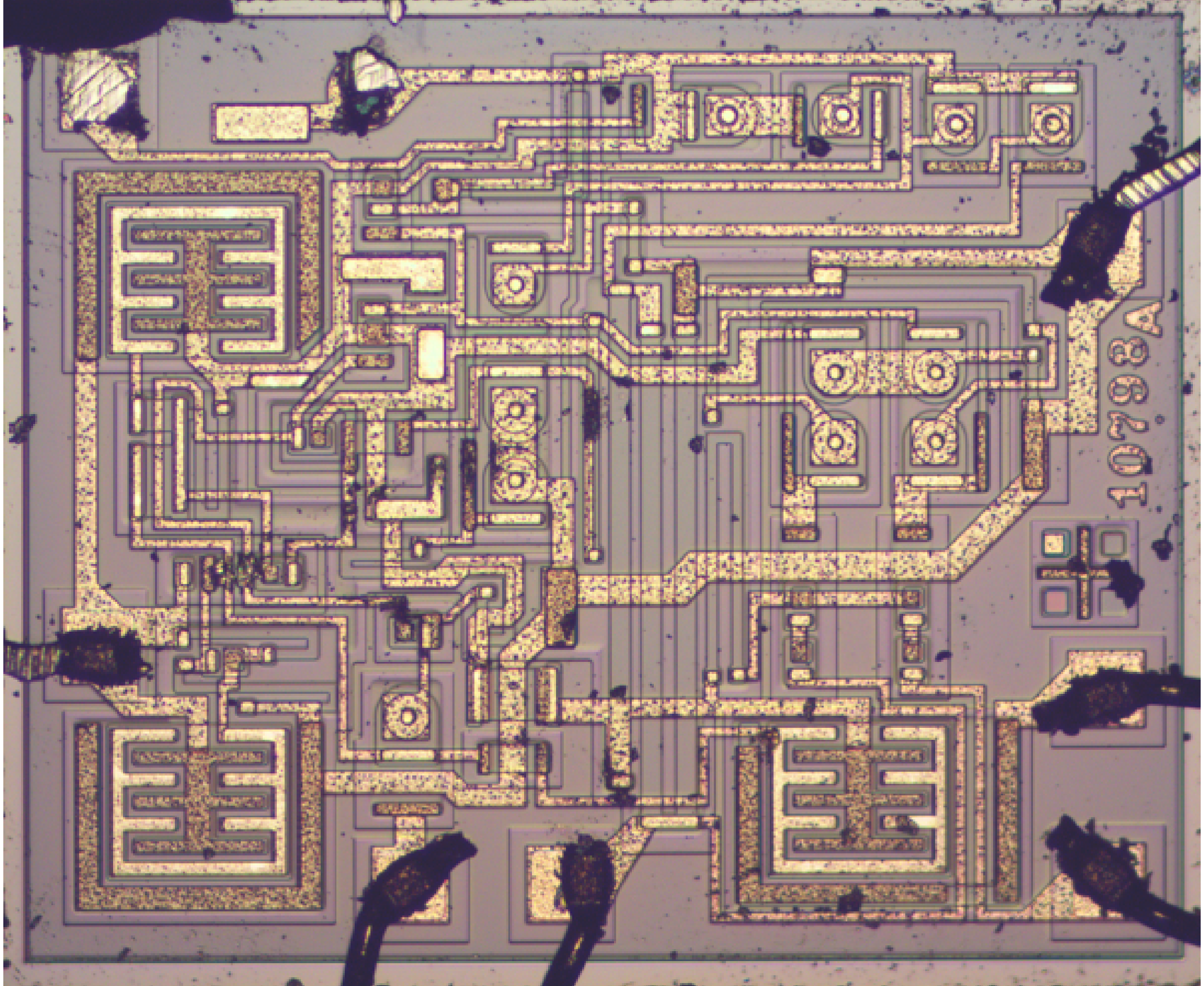


NPN transistor



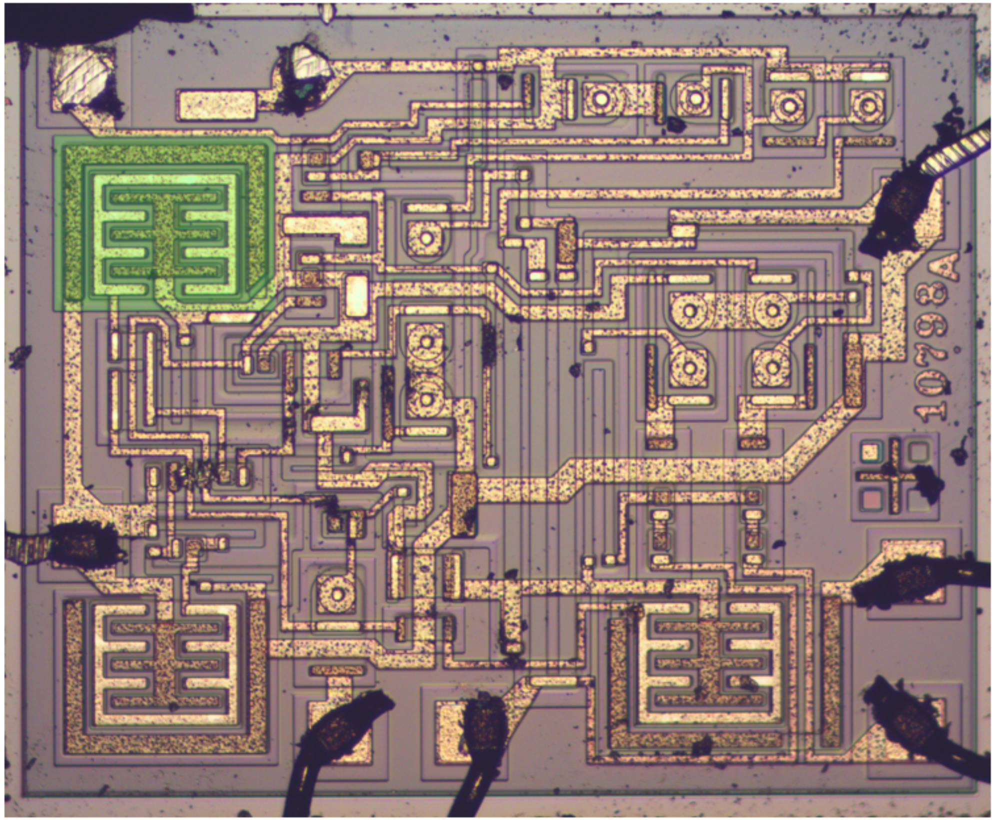


PNP transistor

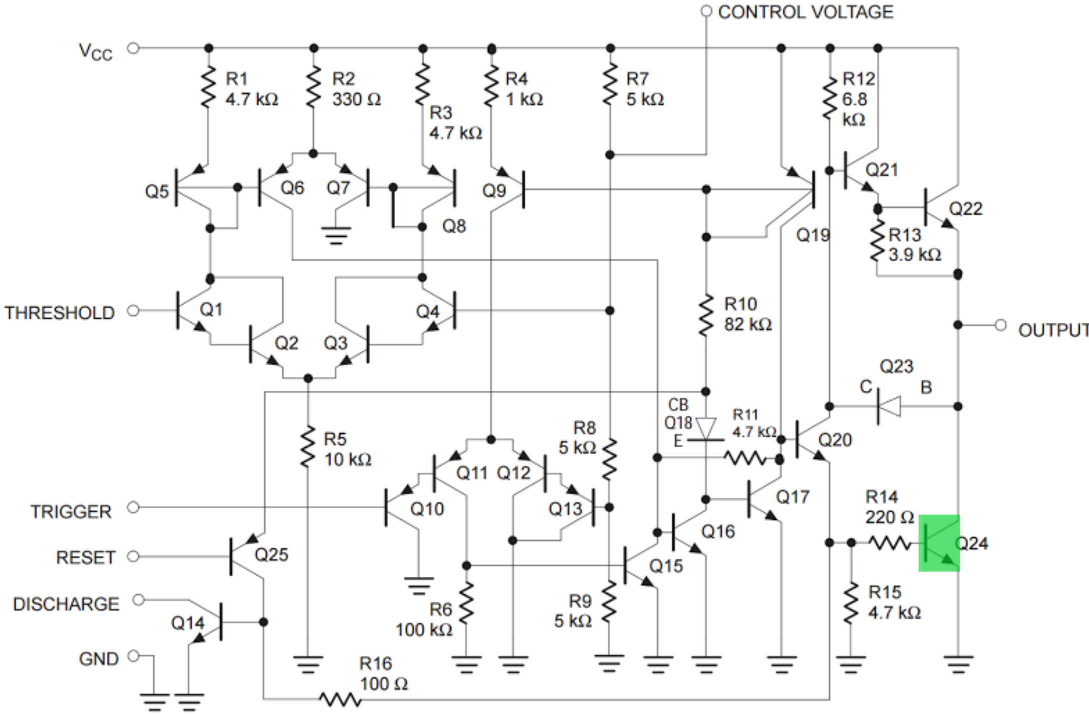


555
timer

Interactive chip viewer



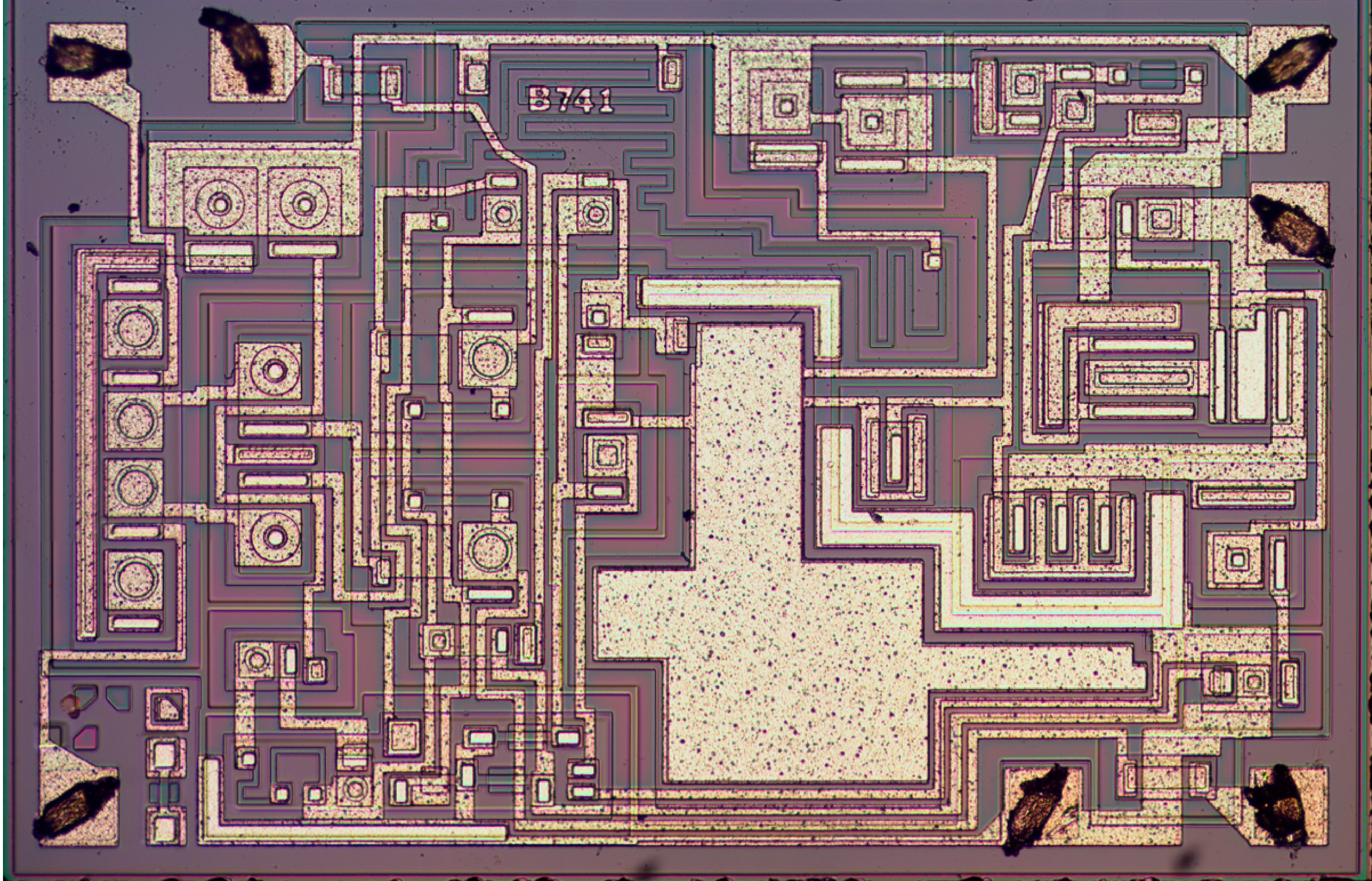
Q24 is a high-current transistor to pull the output low.

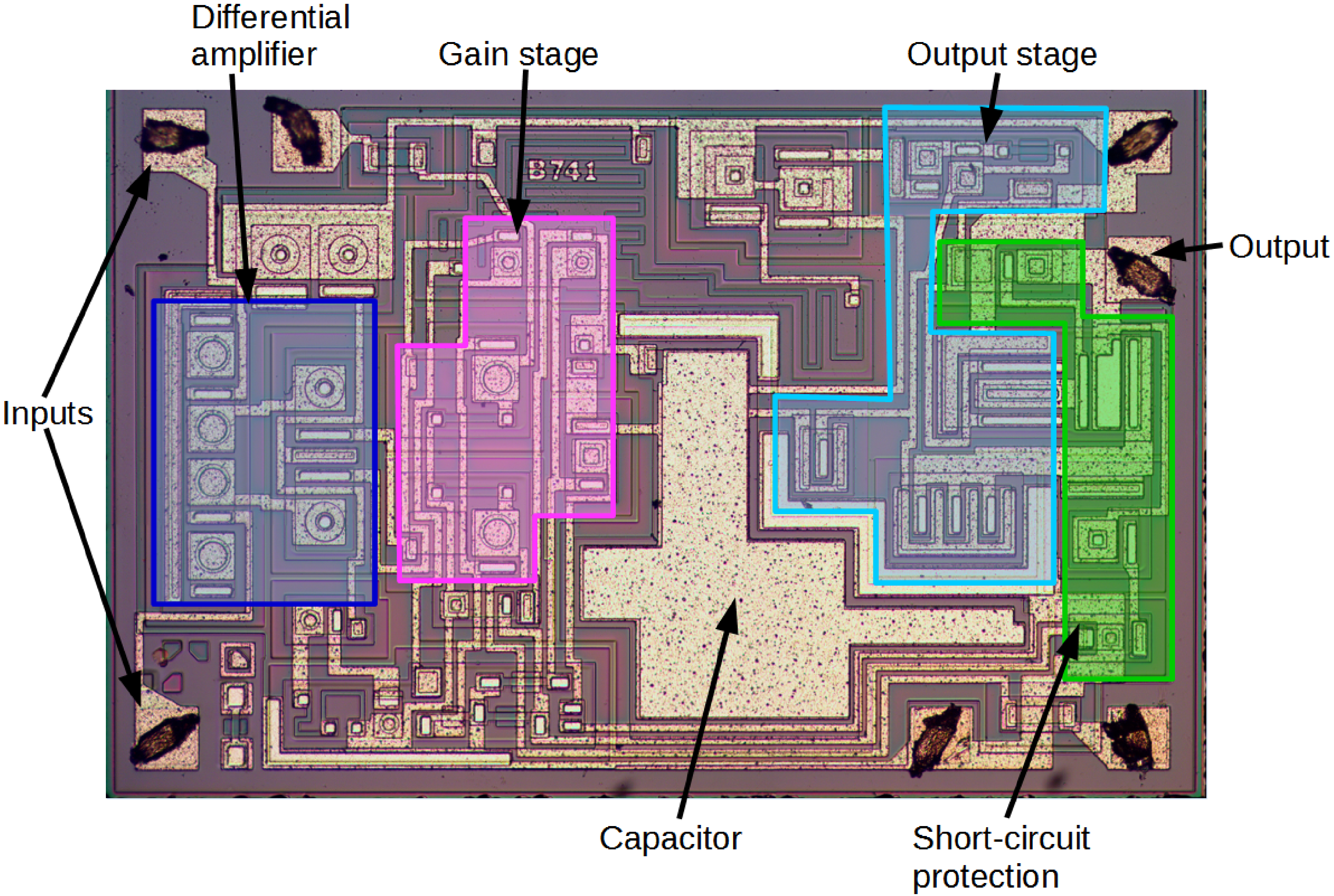


More info: righto.com/555

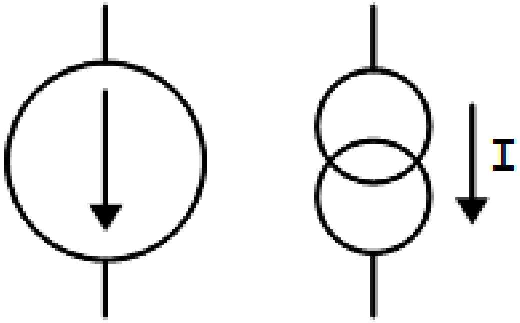
741 op amp

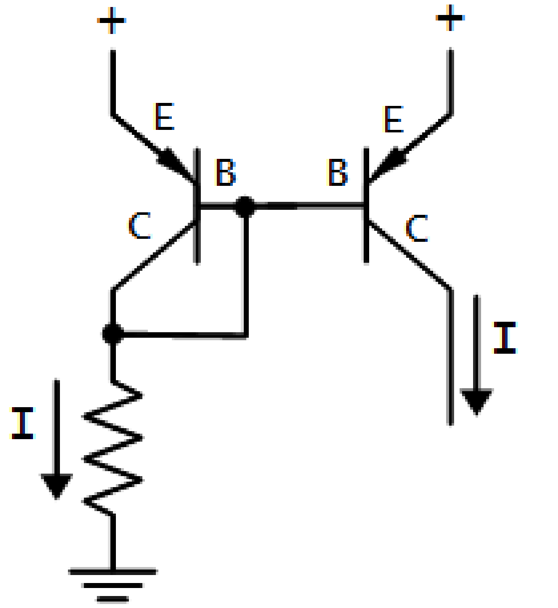
741
op
amp




Differential amplifier
Gain stage
Output stage
Output
Inputs
B741
Capacitor
Short-circuit protection

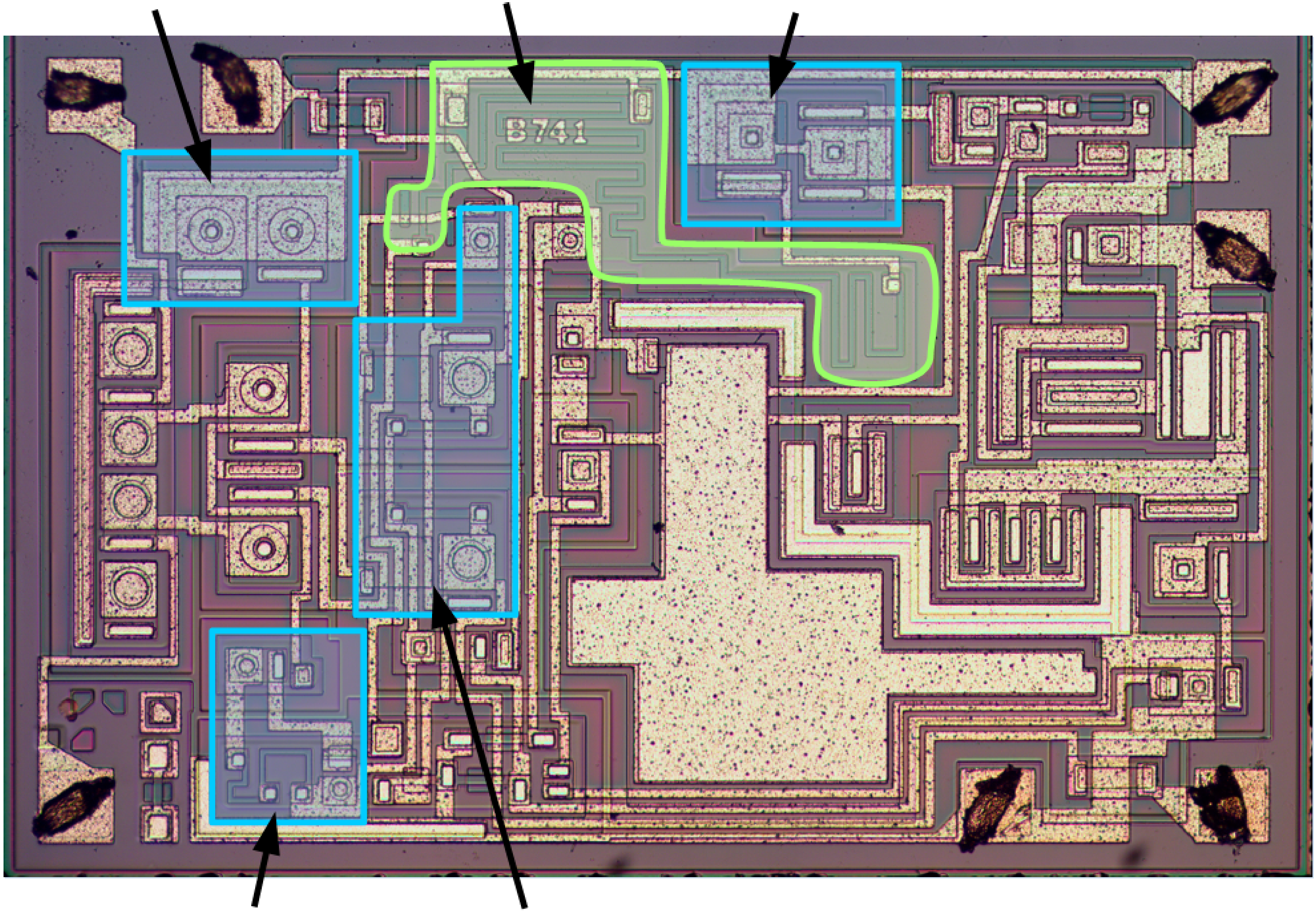
Current mirror





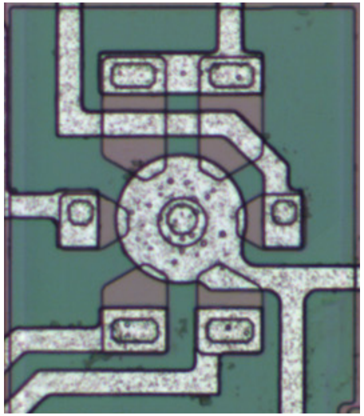
“Clone” a current.

More compact, accurate than resistors.

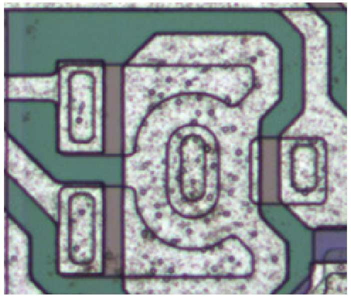


More info: righto.com/741

Unusual current mirror transistors



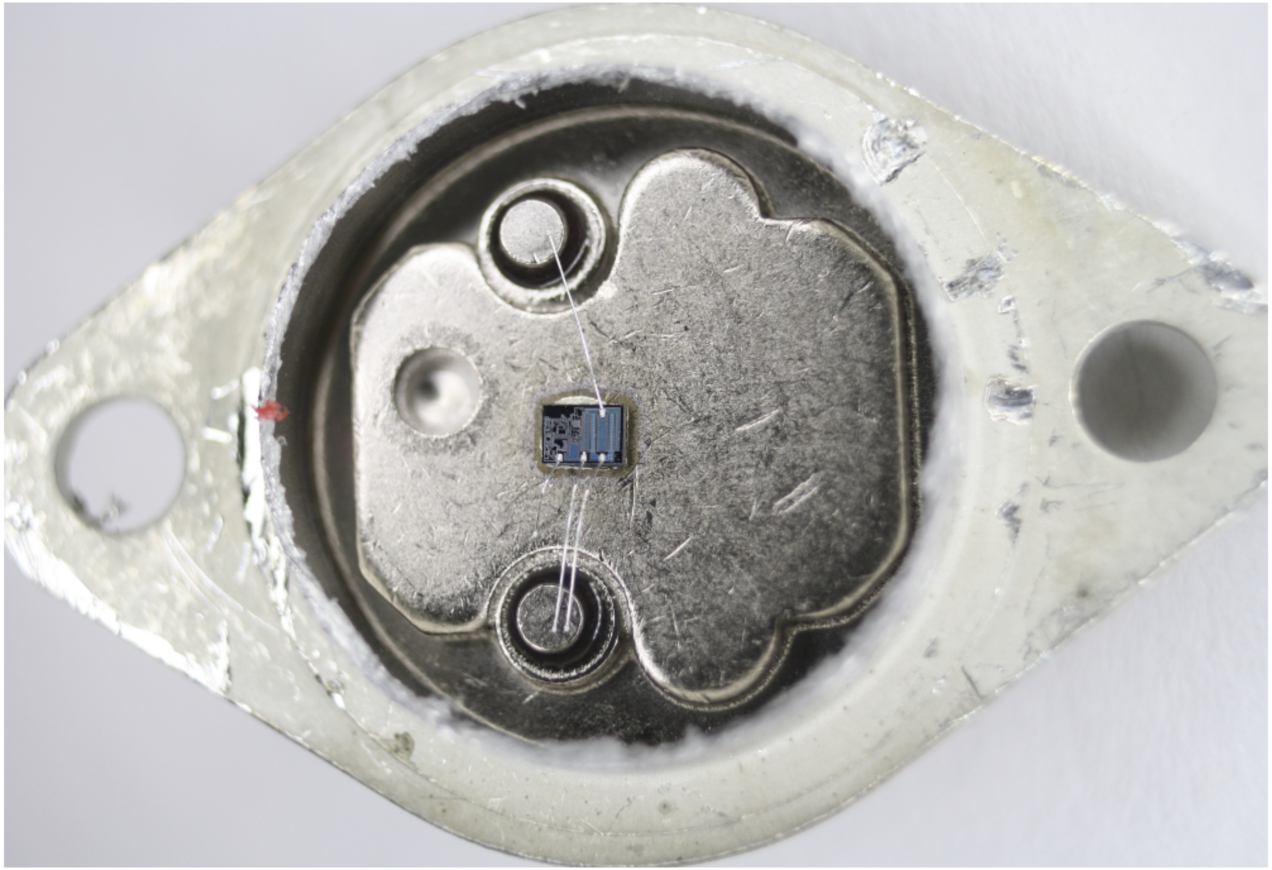
6 collectors:
6 mirrored outputs



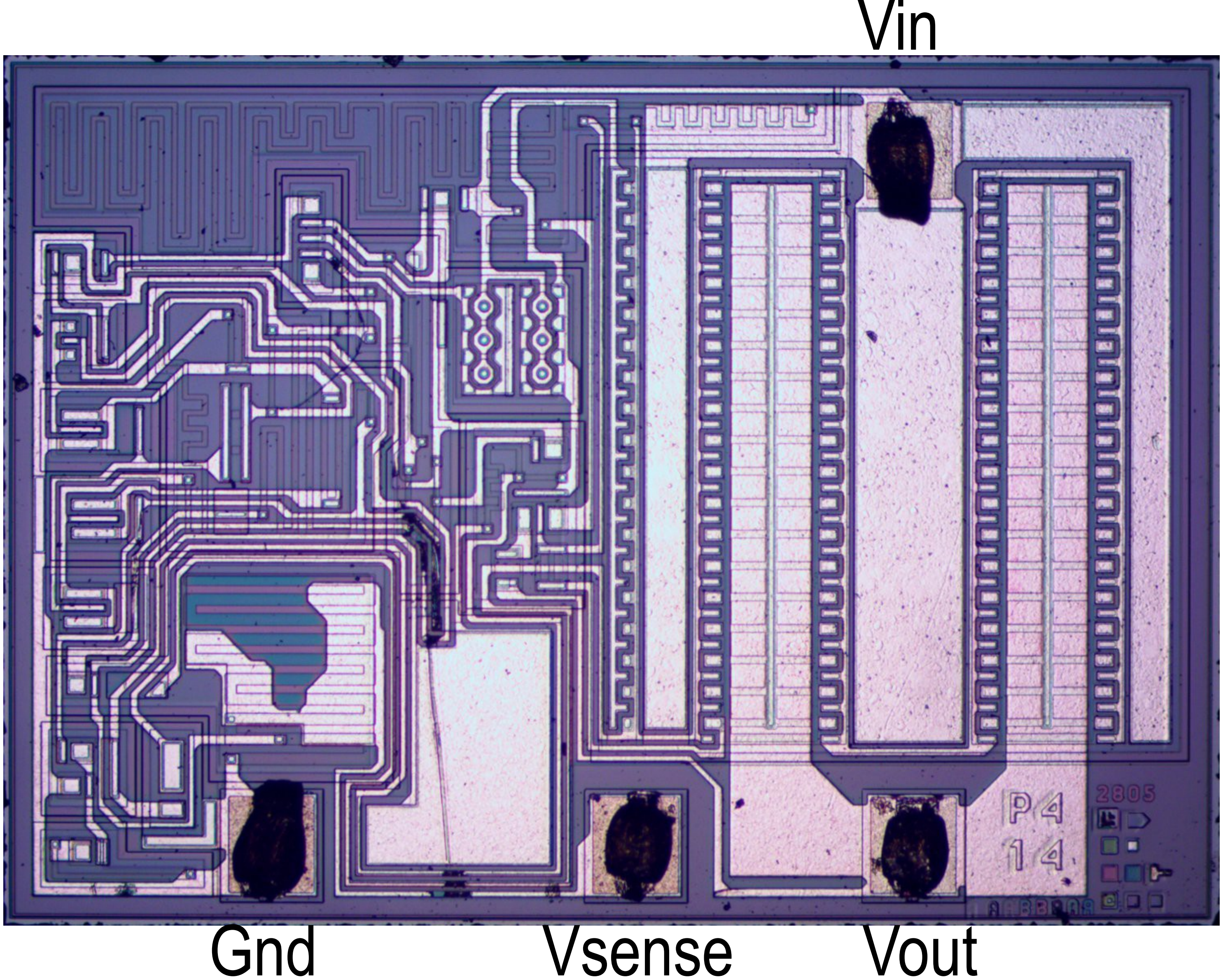
2 big collectors, 1 small:
Scaled output currents

Photos: visual6502

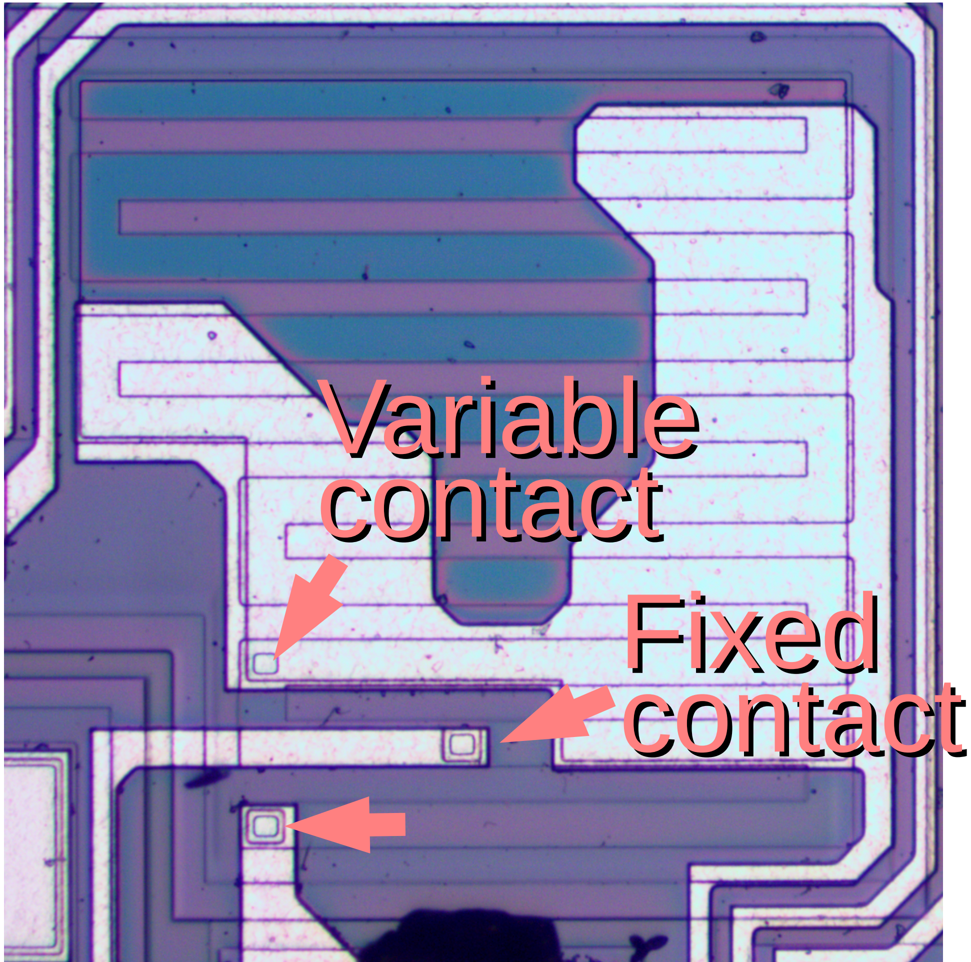
7805 voltage regulator



7-25V in
5V out


Vin
Gnd
Vsense
Vout
2805
P4
14

A family of regulators from one chip



5, 6, 8, 10, 12, 15, 18, 24 volts

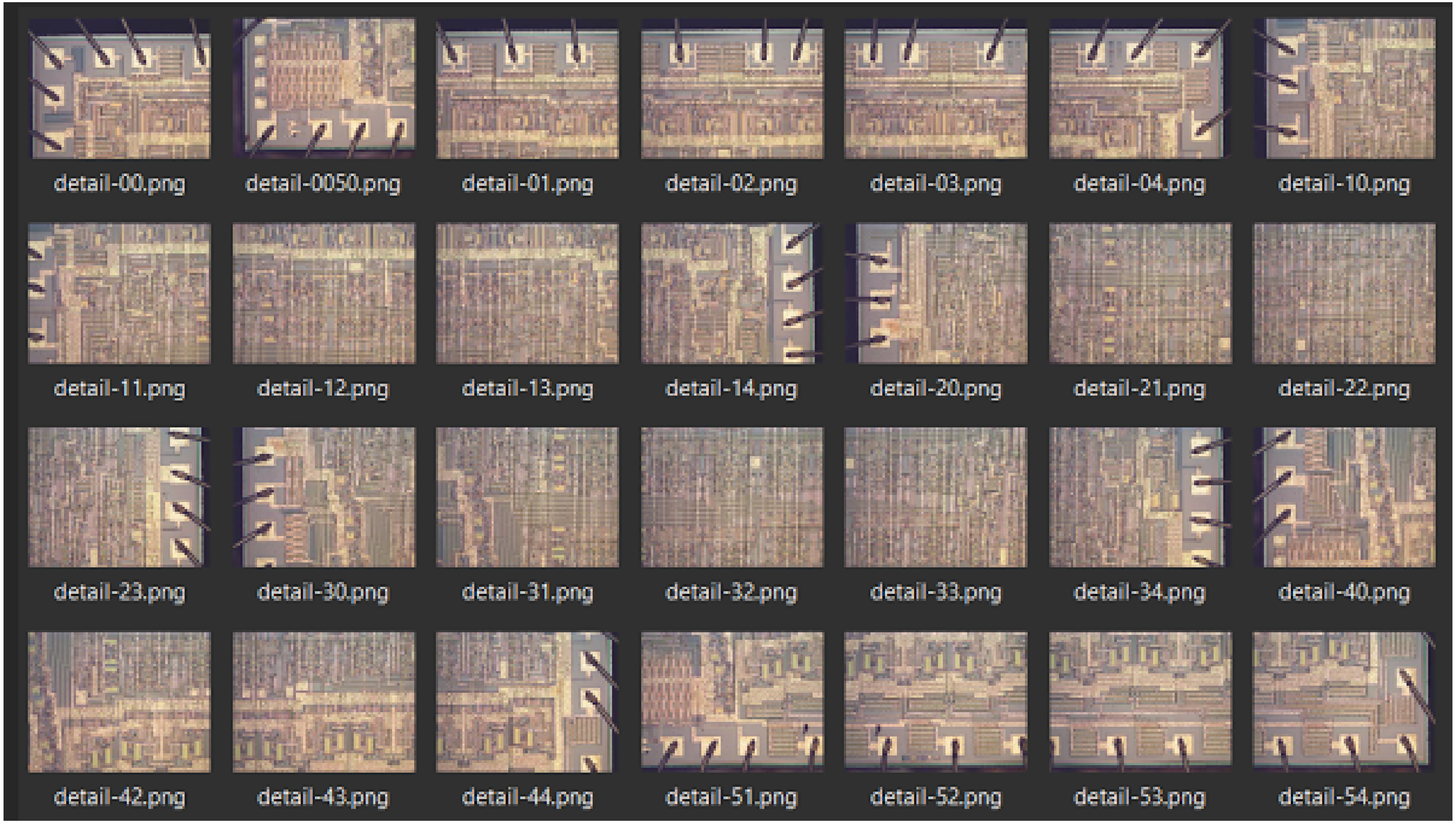
Move contact to change voltage divider

More info: righto.com/05

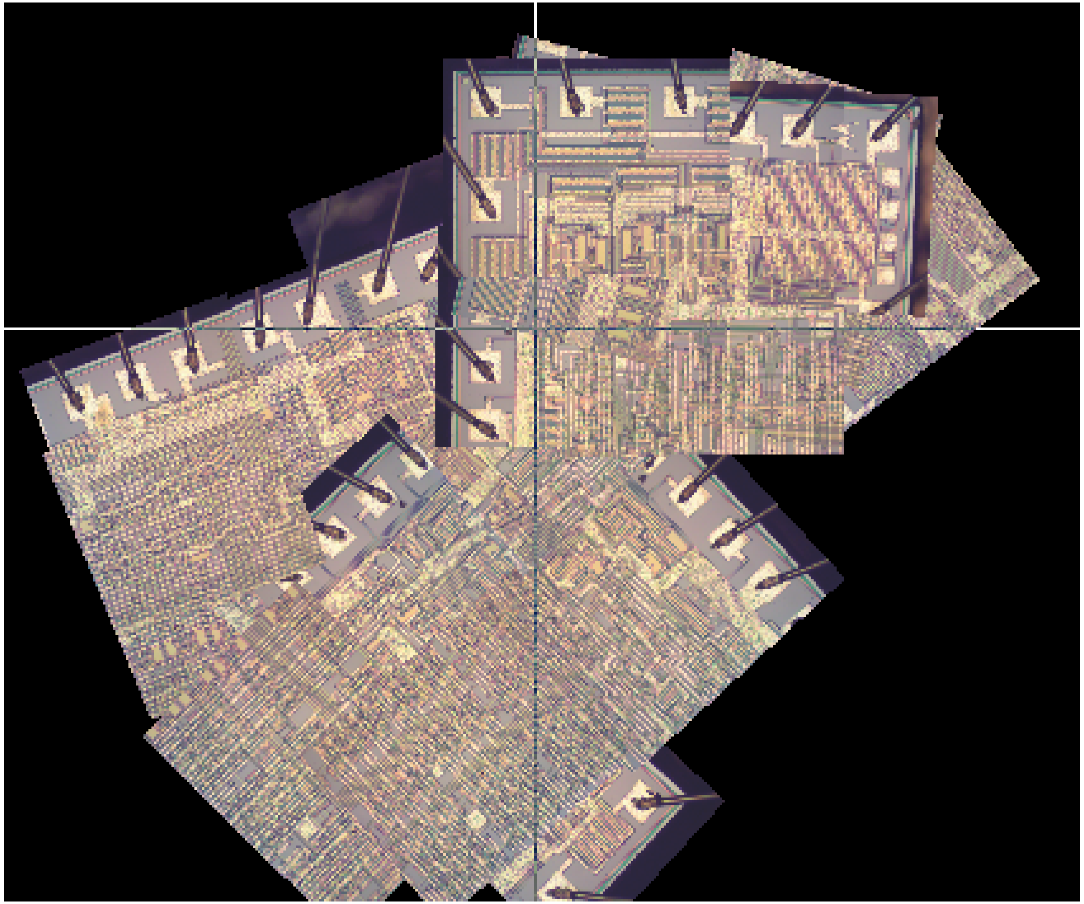
Die photos: Metallurgical microscope

Shines light from above through lens

Stitch photos together for high-resolution



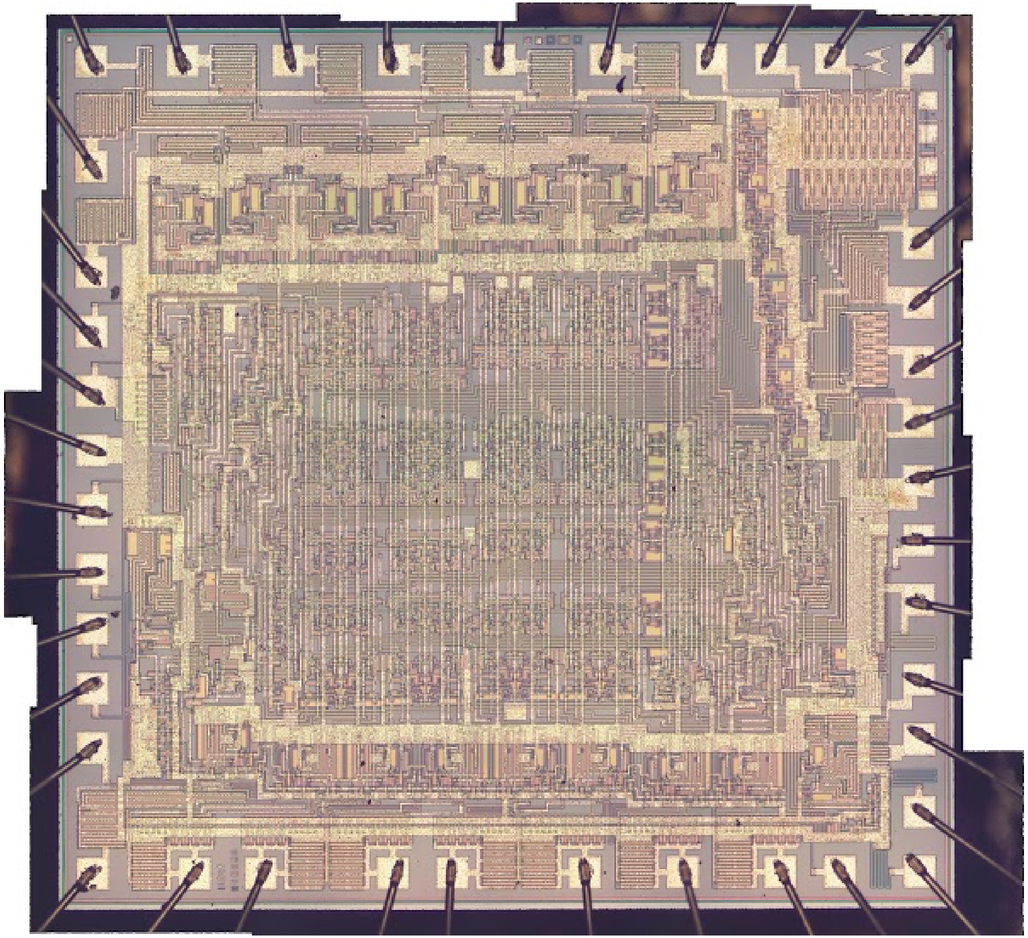
Hugin takes some practice



Tip: have lots of overlap between images

More info: righto.com/hugin

Motorola 6820 PIA chip

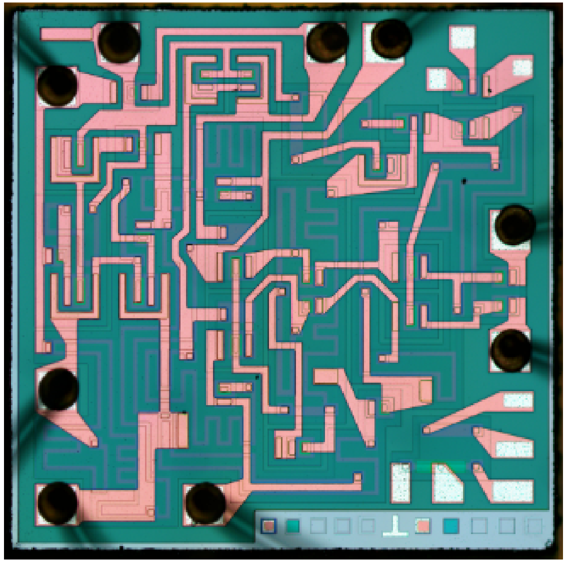


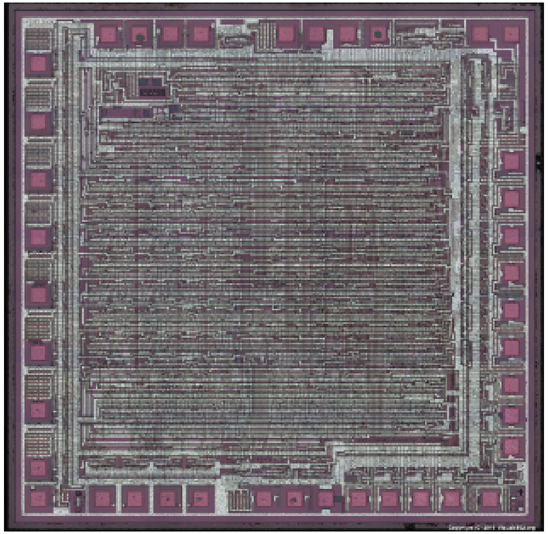
How to get to the die?

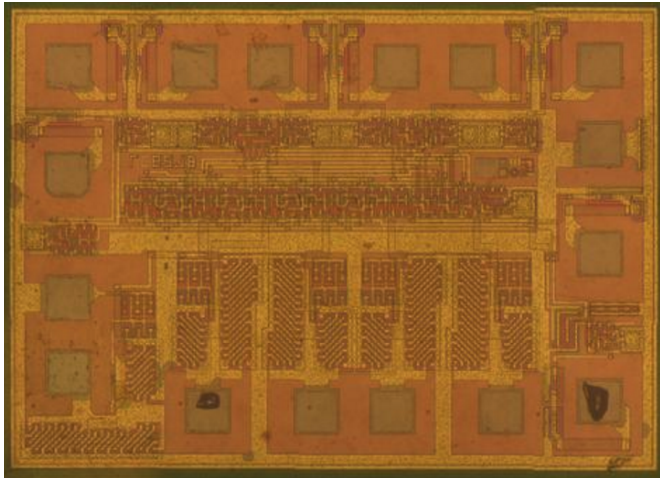
Photo: zeptobars

Hard way: boil chips in sulfuric / nitric acid

Easy way: download die photos

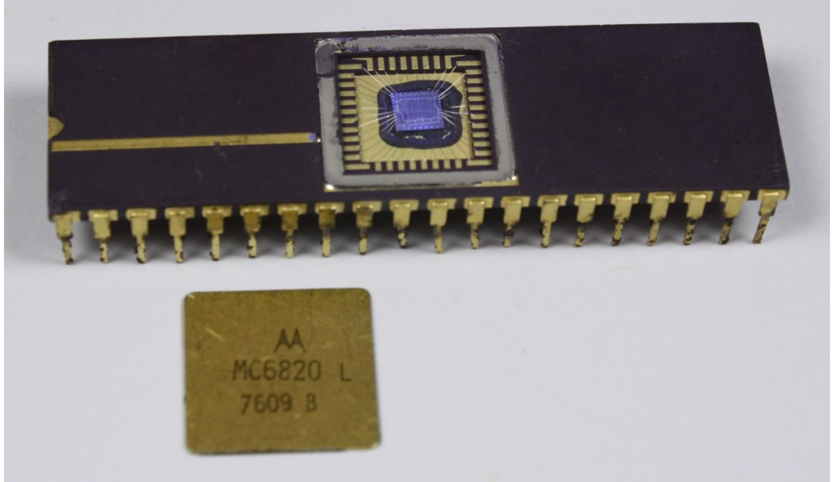




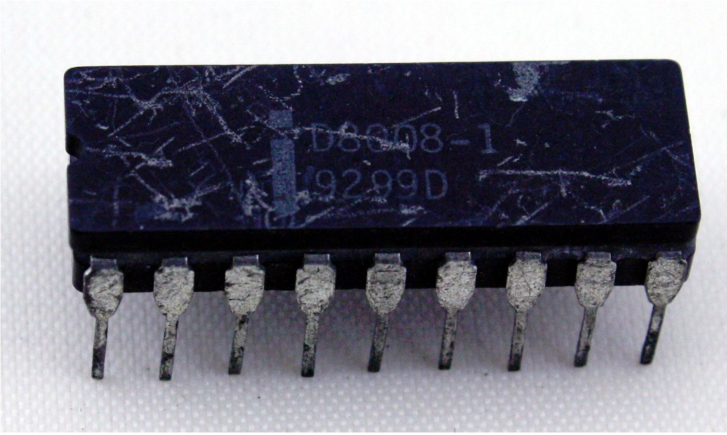


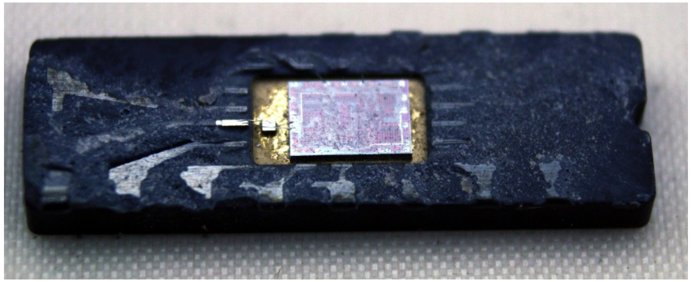
zeptobars.com, visual6502.org, siliconpr0n.org

Acid-free way: chips without epoxy

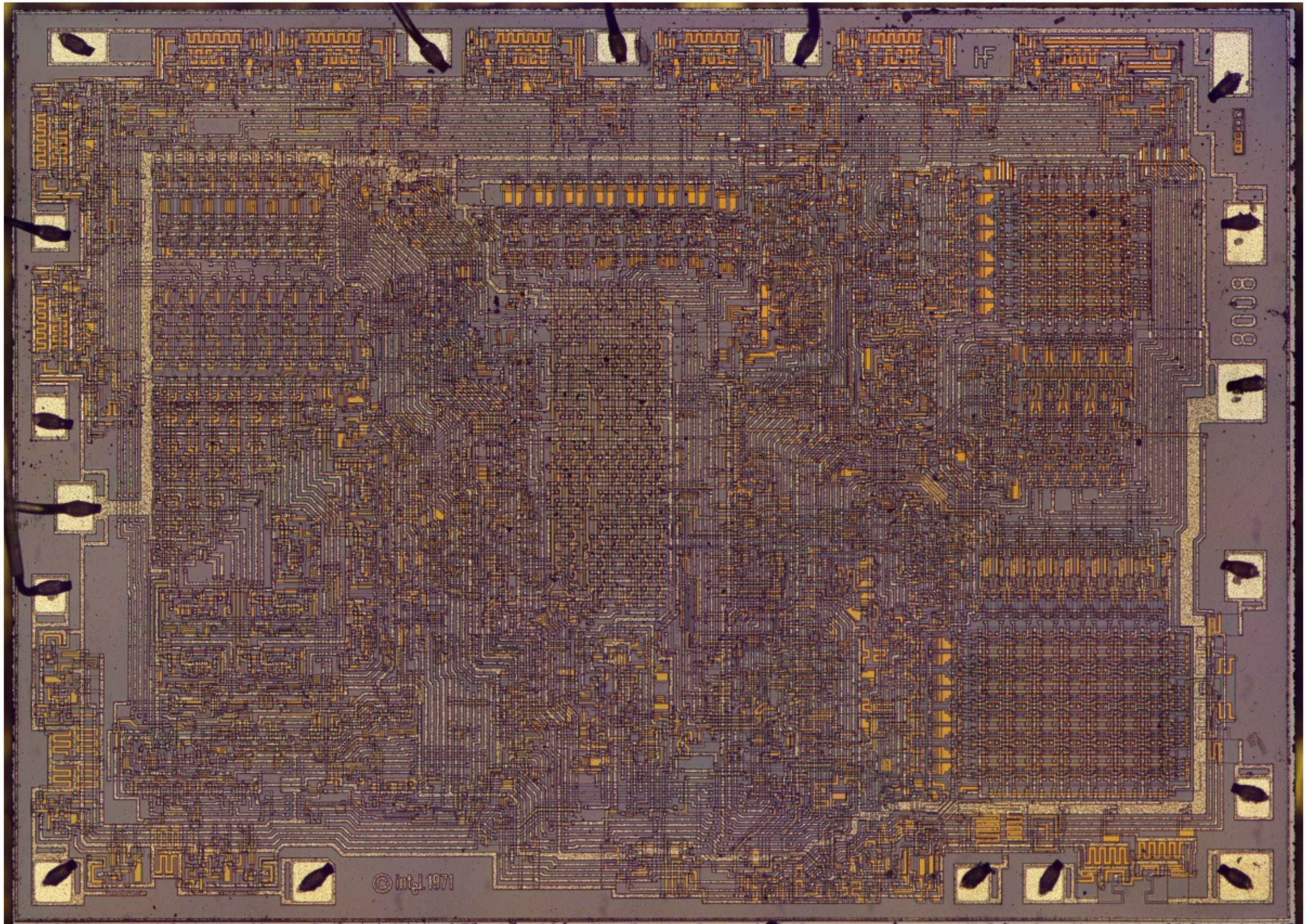


Hacksaw
(jeweler's saw)
or chisel

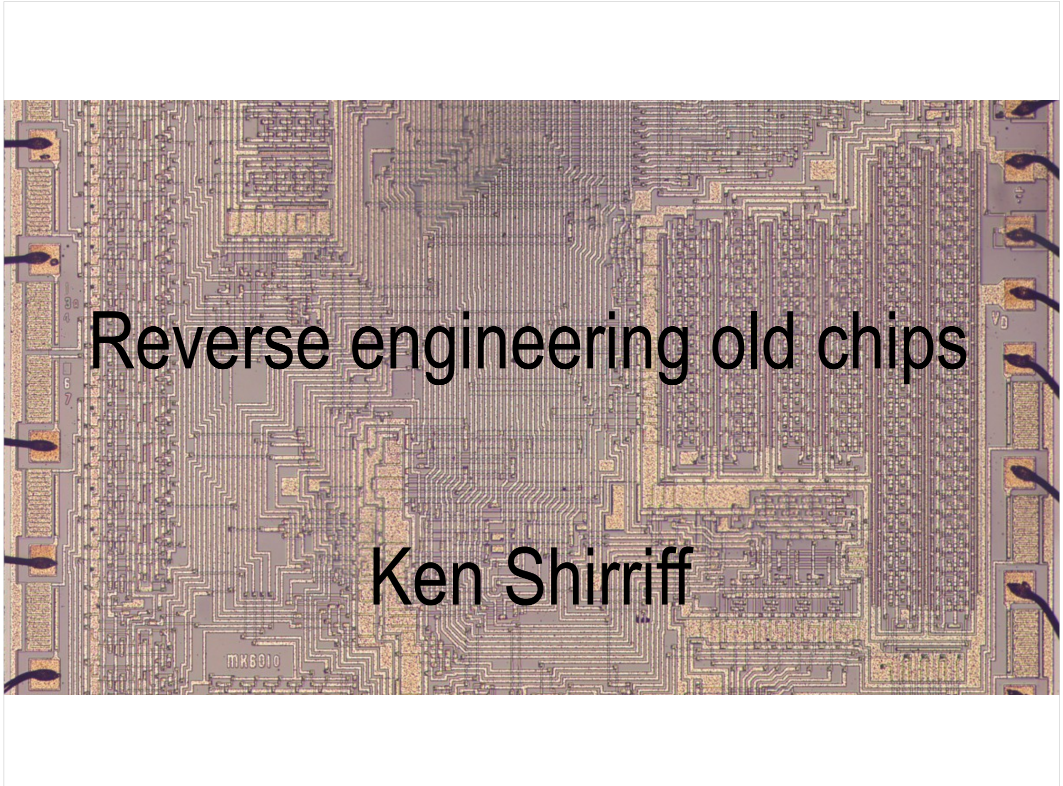




Current project: 8008 analysis




Blog: righto.com
ken.shirriff@gmail.com
Twitter: @kenshirriff



You've probably seen die photos of chips.
My reaction was:
wow, that's cool. But what is all that stuff?
In this talk, I explain what's going on in these chips
and how you can get involved in the obscure hobby of reverse engineering old chips.

Z80

(1976)

8-bit CPU

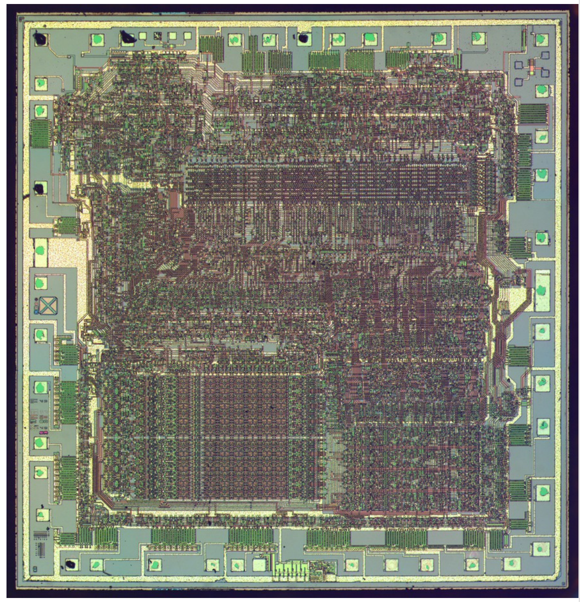


Photo: zeptobars.com

Here's the Z80, a popular microprocessor from the 1970s – maybe you've used it.

Looking at the die photo, it's a jumble, but you can pick out some features:

Pins

Big driver transistors

Power, ground

Now you can look at the datasheet and match up the pins. This gives you a lot of information.

Z80
(1976)

8-bit CPU

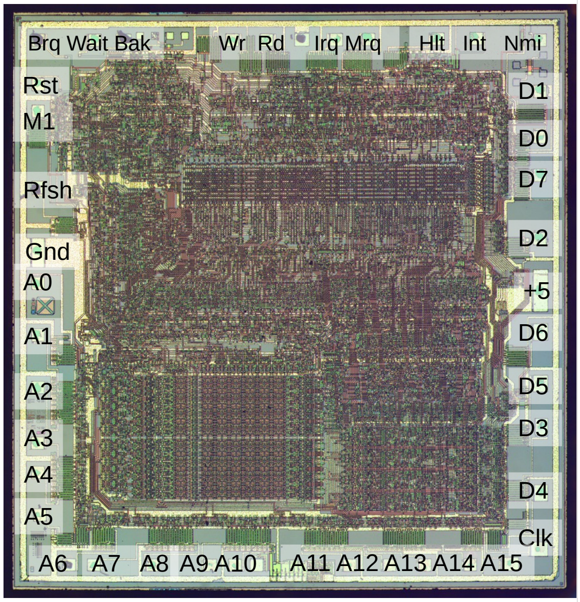
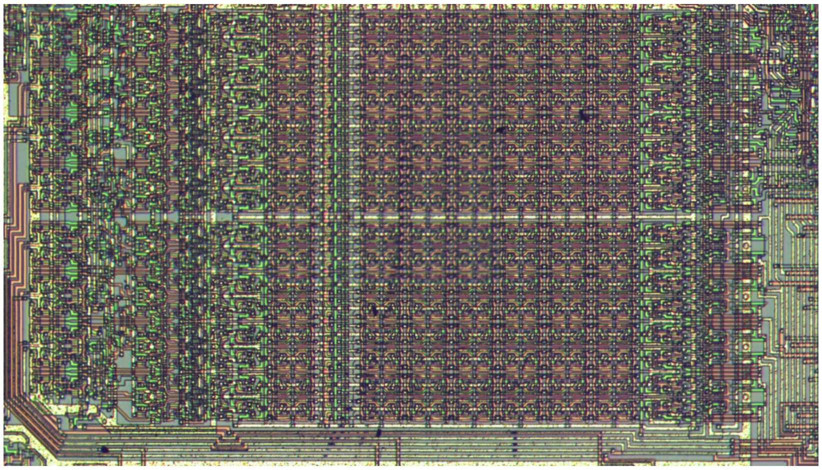


Photo: zeptobars.com

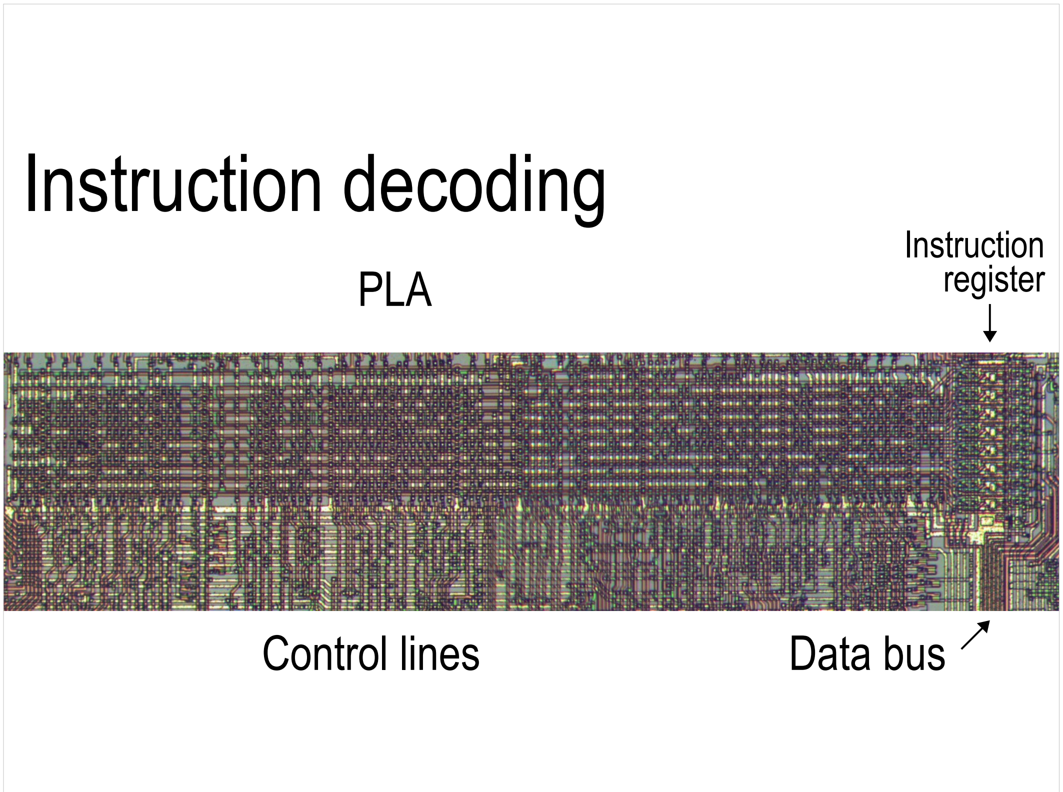
Control pins at the top, so this is the control section

Buses
Functional blocks

Register File

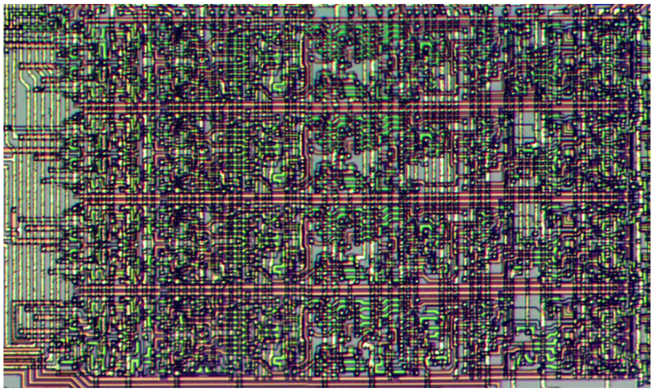


Matrix of memory cells
8 on top, 8 on bottom.
Data bus on right
Address bus on left: PC, incrementer
Secret registers



Decode instruction, generate control signals.
PLA: regular array of gates, very common in 1970s.
Now microcode.
Instruction stored in instruction register.
Each PLA column selects a bit pattern from instructions.
Design instruction set correctly.

ALU (Arithmetic-Logic Unit)

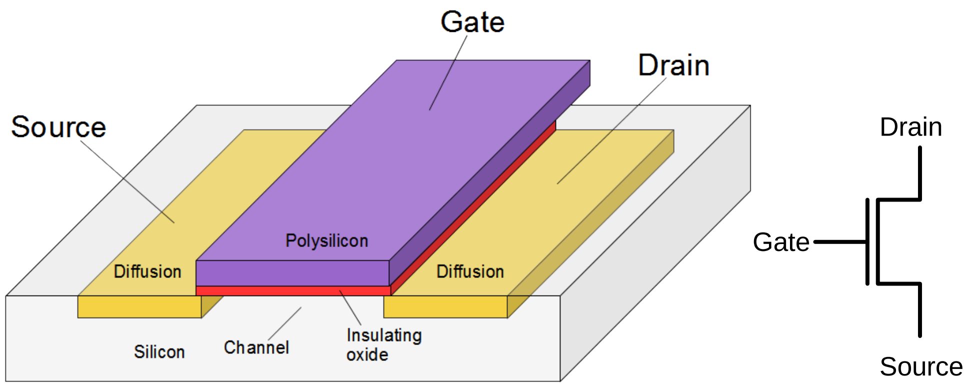


Only 4 bits!

ALU: add, subtract, boolean operations, shifts
Bitslices of complex circuitry, one for each bit.

Z-80 surprise: 8-bit processor, 4-bit ALU.
Everything processed twice.

MOS transistors



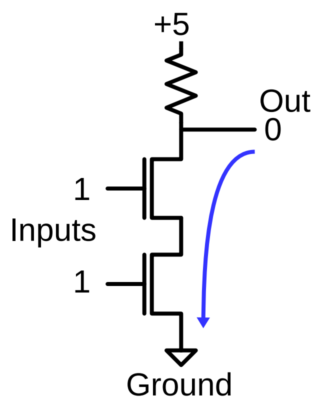
To understand circuits in more detail, need to look at transistors.
Simplified, MOS transistor is a switch. When gate is 1, switch is closed, when gate is 0 switch is open.

Starts with the silicon, which is insulator.
Dope silicon to make it semiconductor.
Charge on gate makes channel between source and drain conduct.
Thin insulating oxide layer under gate, static sensitive.

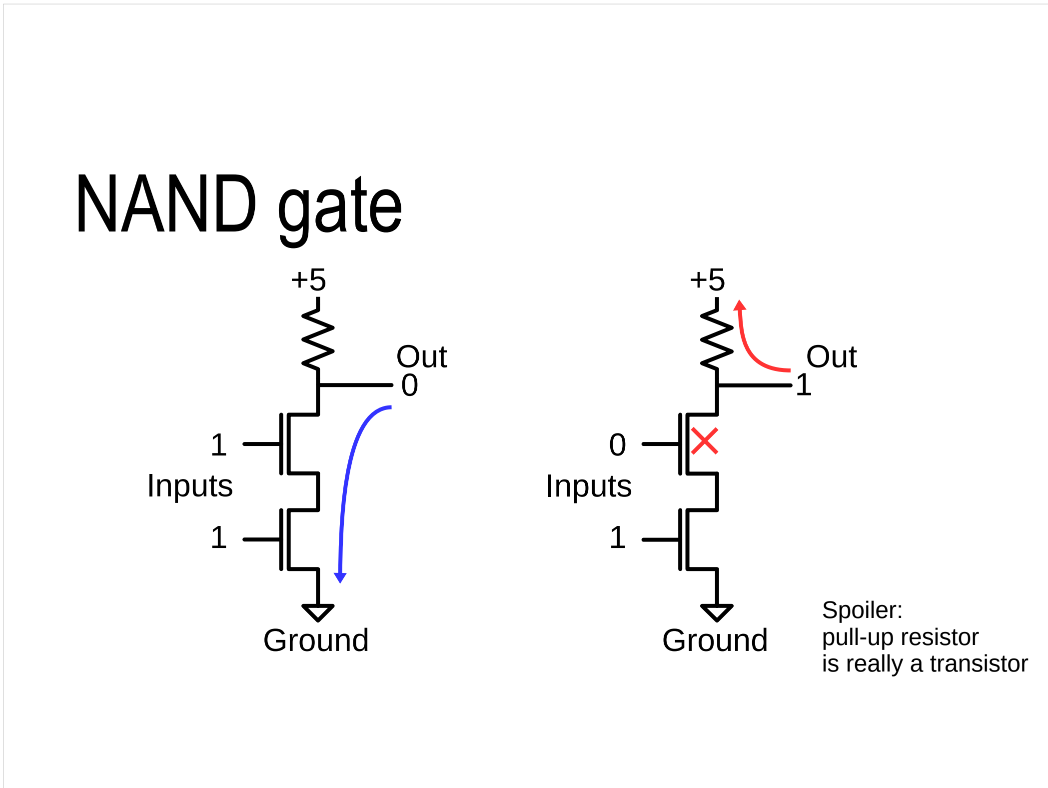
Originally: gate was metal: Metal, Oxide, Semiconductor: MOS
Polysilicon introduced in 1970: should be POS

Chip has 3 layers: silicon, polysilicon, metal wiring.

NAND gate



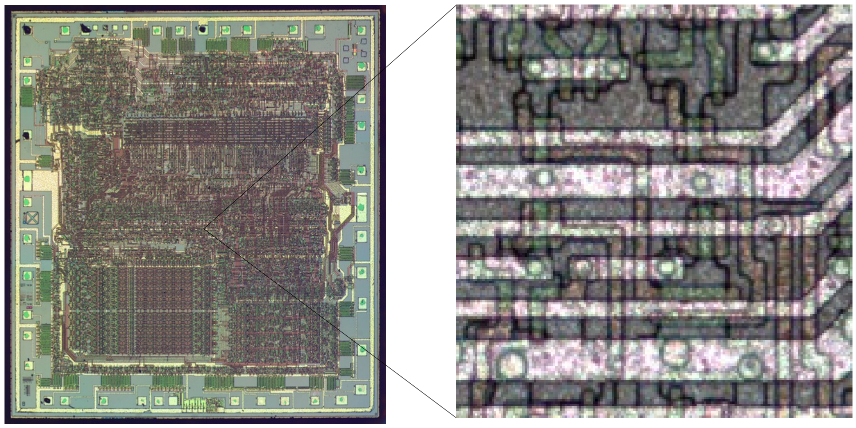
Build a NAND gate from two transistors and a resistor.
If both inputs are 1, both transistors conduct,
connecting output to ground: 0



If an input is 0, transistor is open. Pull-up resistor pulls output high.
Thus, NAND logic.

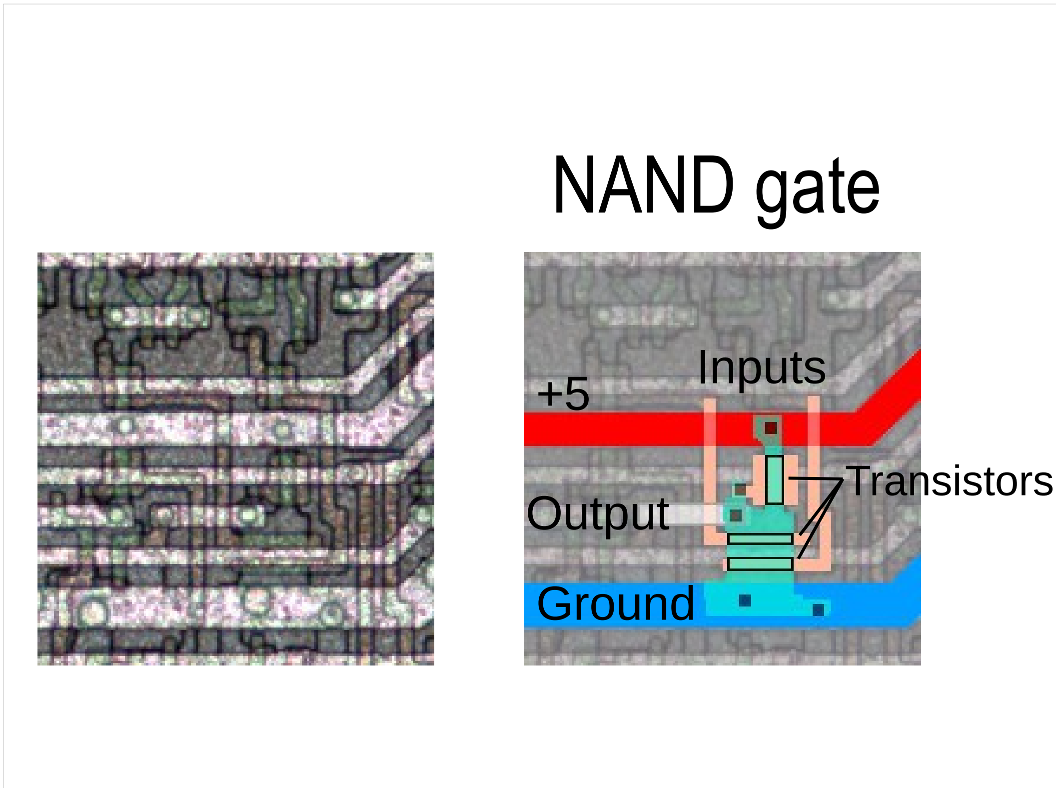
Spoiler: a transistor acts as the resistor.

What do gates really look like?



If we zoom way in on the Z80, we see the gates, which are a bit of a mess.

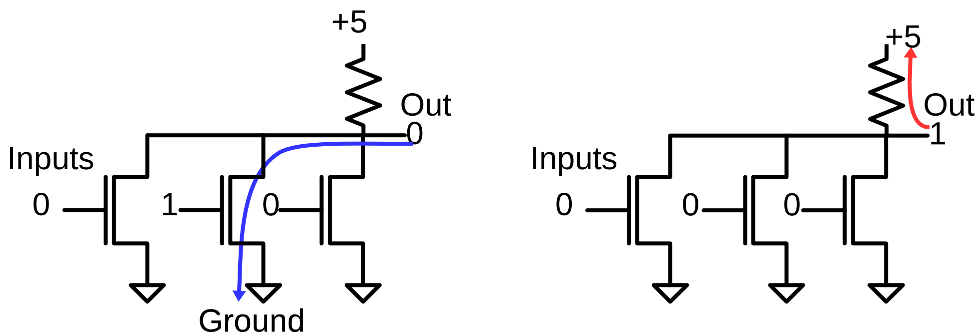
The metallic-looking strips are the metal, on top.
Underneath, you can see the polysilicon wires.
Black lines indicate the doped silicon.



If you stare at this closely, you can pick out the features of a NAND gate.
Metal lines provide power and ground for the gate.
The doped silicon is greenish.
Polysilicon inputs are pink, and form transistors where they cross the silicon.
Note the two transistors between ground and output.
If both inputs are high, the output will be pulled low.
Another transistor forms the pull-up resistor.

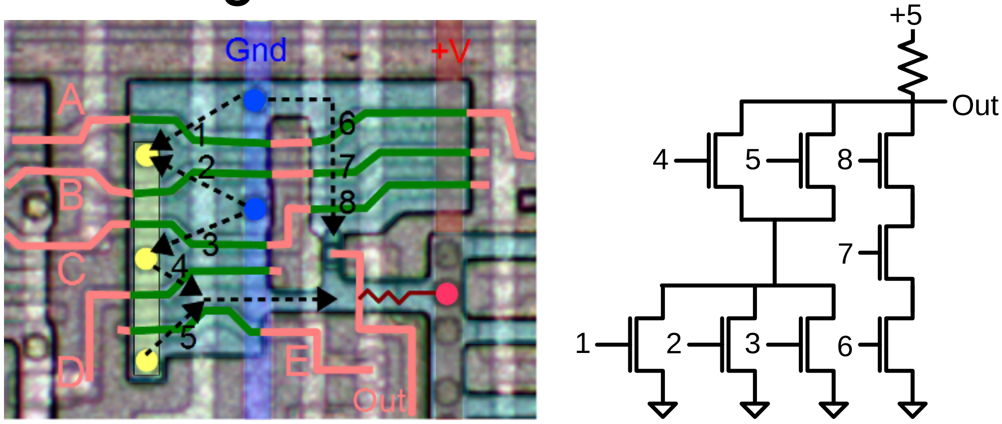
It’s tricky to see what’s happening, but people on the visual 6502 team digitized all the polygons for the circuitry for multiple chips.

NOR gate

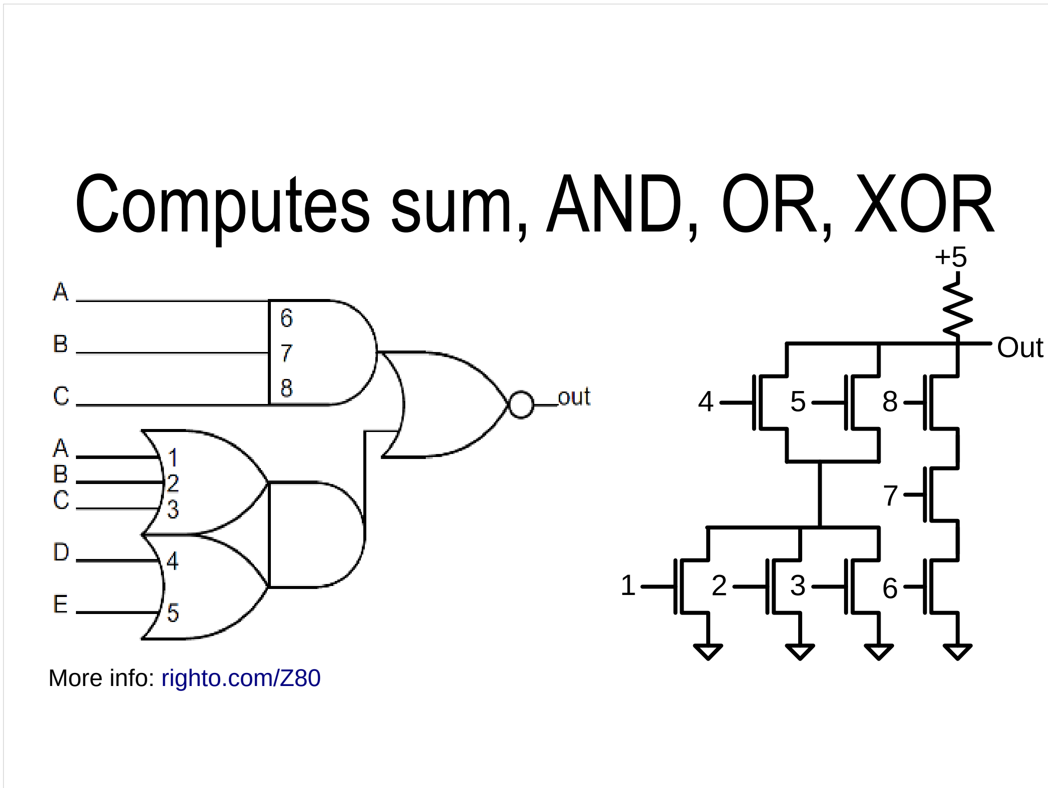


It’s not all NAND gates. By connecting transistors in parallel, you can build a NOR gate. If any input is high, the input is pulled low.

Gates get weird in the ALU



The Z80 has a lot of very complex gates. Here’s one from the ALU with 9 transistors.
If 1, 2, or 3 is on, the metal here is grounded.
4 or 5 connect that to the output.
6, 7, and 8 will also pull the output low.



Although logically this is 5 gates, it's built as one gate. Note that the AND and NOR are for free, just wires.

What does this do?
It computes B AND C or B or C.
With a partial sum and carry in, it computes B plus C or B XOR C.

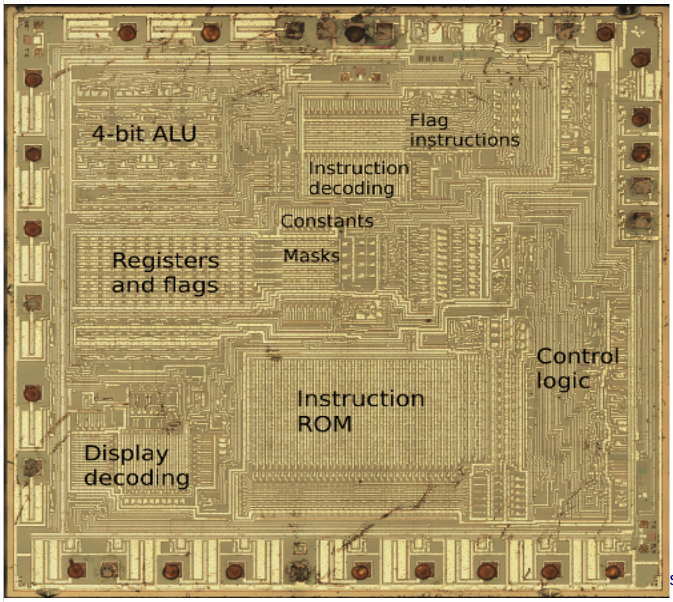
ALUs are very diverse. You'd expect a standard adder, but circuitry is highly optimized. 6502 has a totally different approach.

More ALU details on my blog.

Sinclair Scientific Calculator (1974)

Reprogrammed TI 0800 4-function calculator chip to support trig, log. How?

Now for something different.
TI built a simple four-function calculator chip. The code for the calculator was crammed into 320 words.
Sinclair reprogrammed this chip to make a cheap scientific calculator.
How is it possible to fit trig and logs into a chip that barely fits basic math?
I reverse engineered the chip to find out.

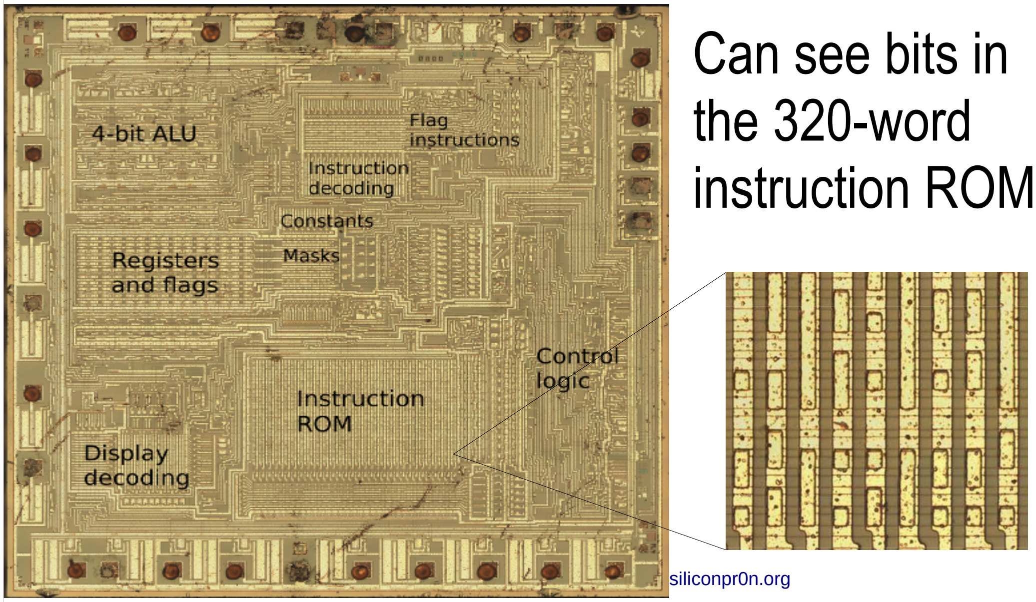


TMS 0805 calculator chip

siliconpr0n.org

John McMaster took a die photo of the chip.
You can see some of the features we’ve discussed: the pins, registers, ALU, instruction decoding.
This chip is built for calculators: so decimal arithmetic, 11-digit registers, 7-segment display driver.

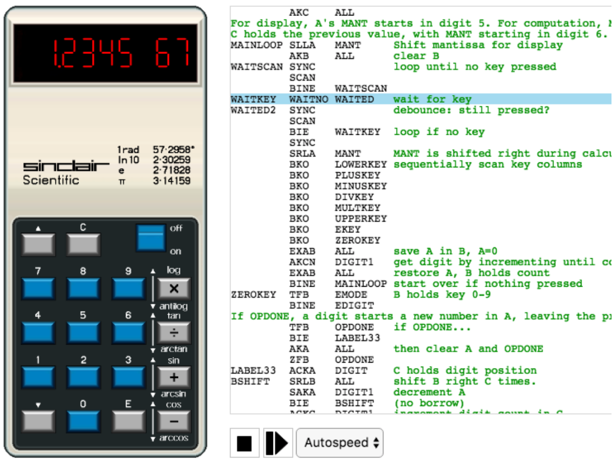
Can see bits in the 320-word instruction ROM



siliconpr0n.org

The code is stored in the instruction ROM.
Looking closely, you can see the bits, formed from transistors.

Built instruction-level simulator



Decimal algorithms

Trig: repeated rotates by .001 rad

Log: powers of 0.99

More info: righto.com/sinclair

I read out the code, reverse-engineered the instruction set and architecture, and built a simulator to run the code.

How do the algorithms work? They are slow and inaccurate, but compact.

Trig uses repeated rotations by .001 radians. The bigger the angle, the slower the operation.

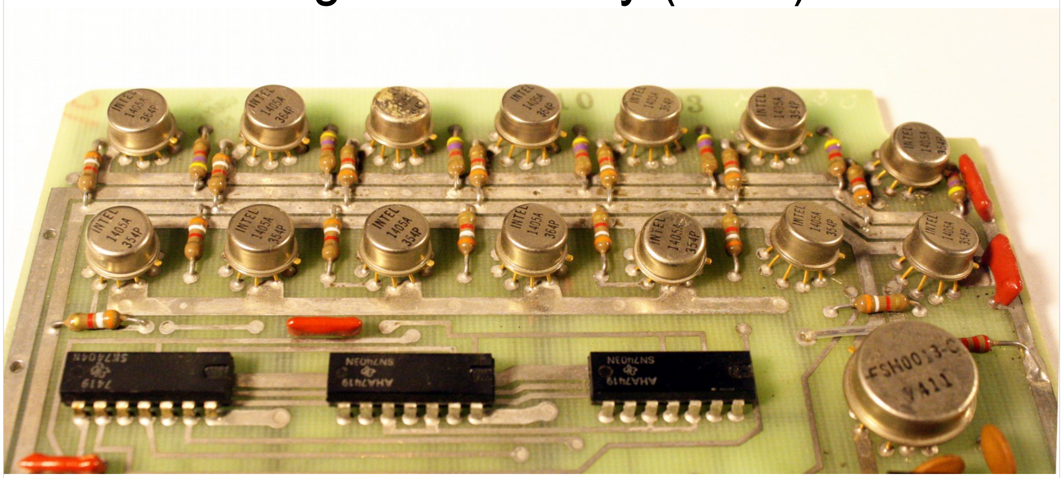
This rotation is basically divide by 1000 (just a shift) and an add.

Log is based on repeatedly multiplying by .99.

Multiplying by .99 is just divide by 100 and subtract. With decimal arithmetic, that's just a fast shift.

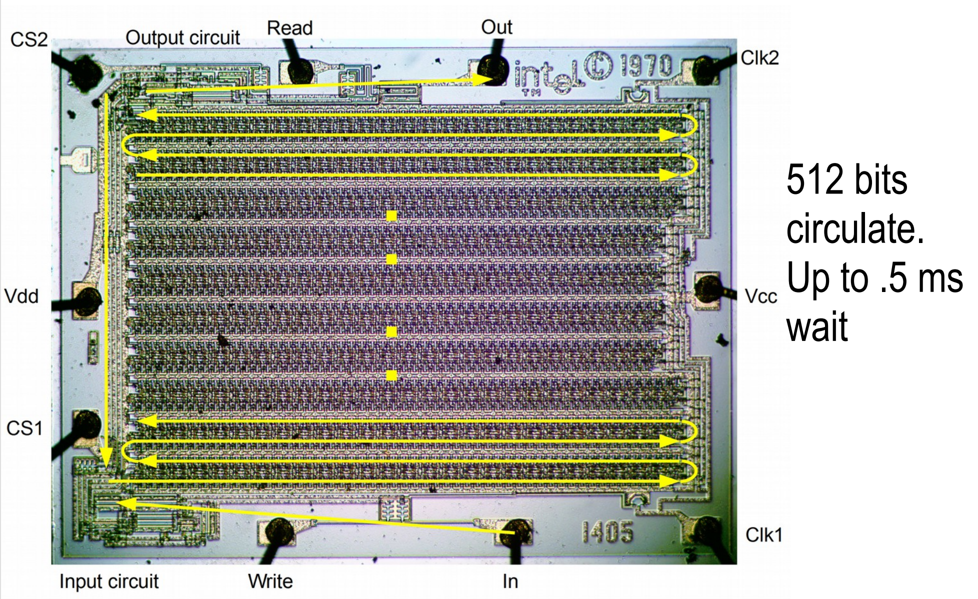
Simulator and more details on my blog.

Intel shift-register memory (1970)



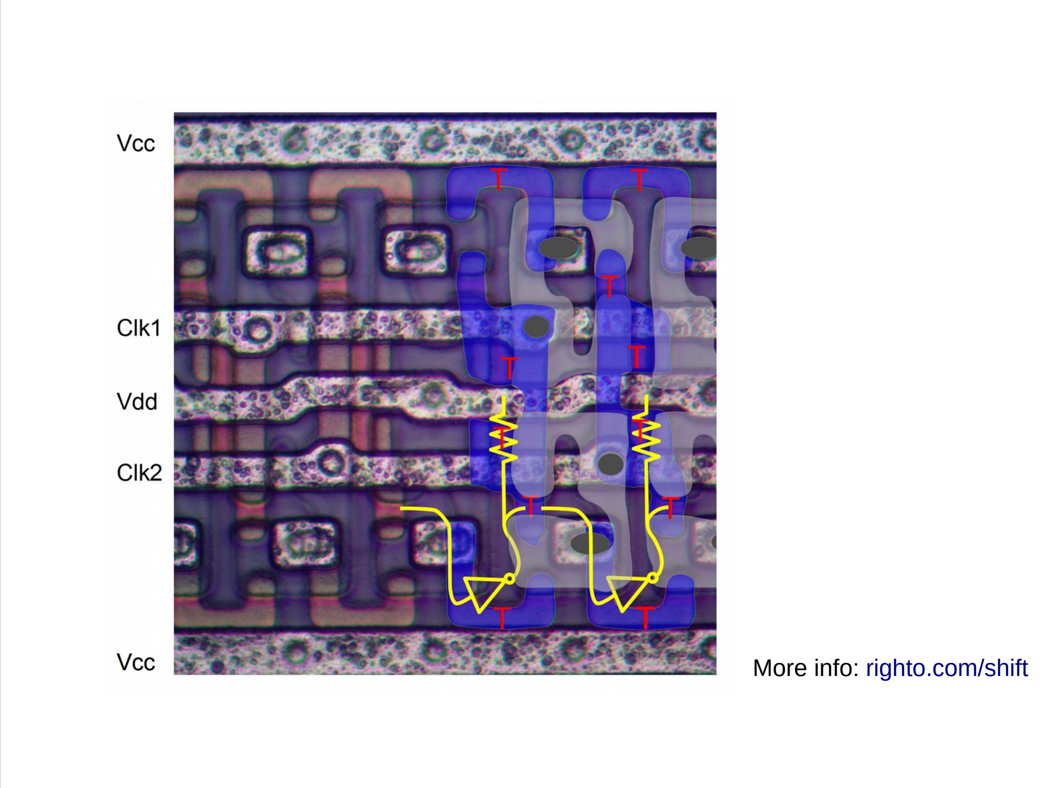
Now to jump to something else.
Before DRAM, Intel had shift-register memory.
Each chip stored 512 bits.

This board is from the Datapoint 2200, an interesting system that some people call the first PC.

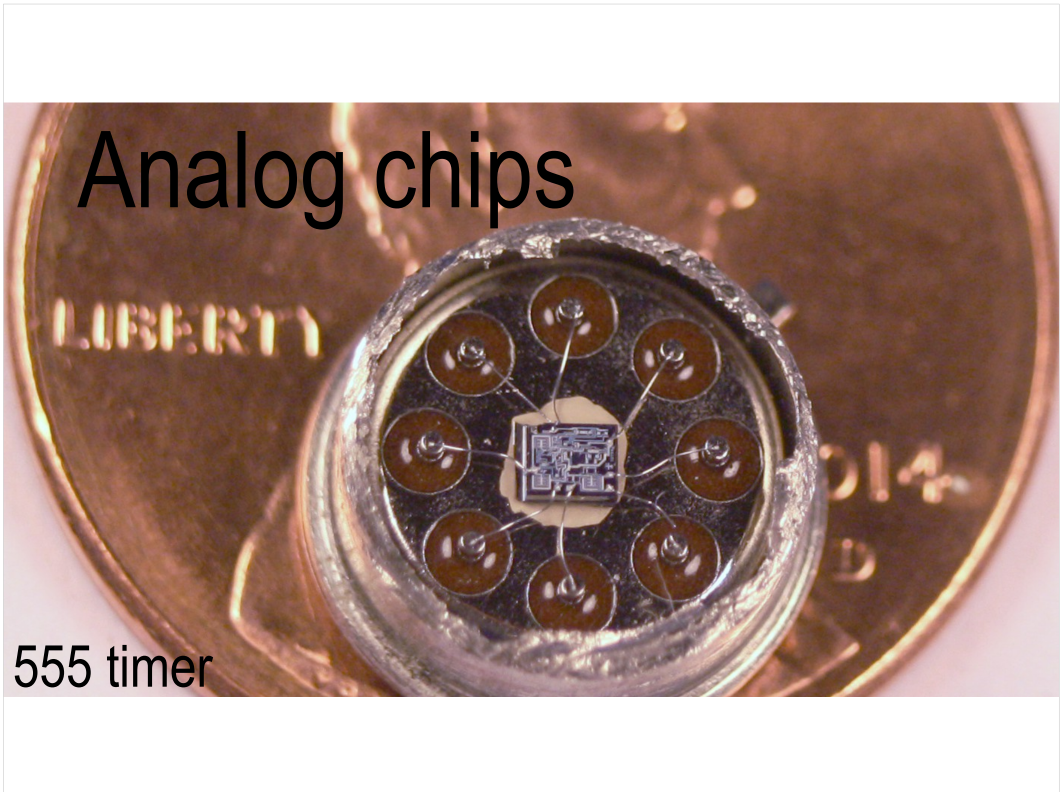


I took die photos of the chip.
Bits enter at the bottom, shift back and forth and come out the top. You can either recirculate the bit or write a new bit.

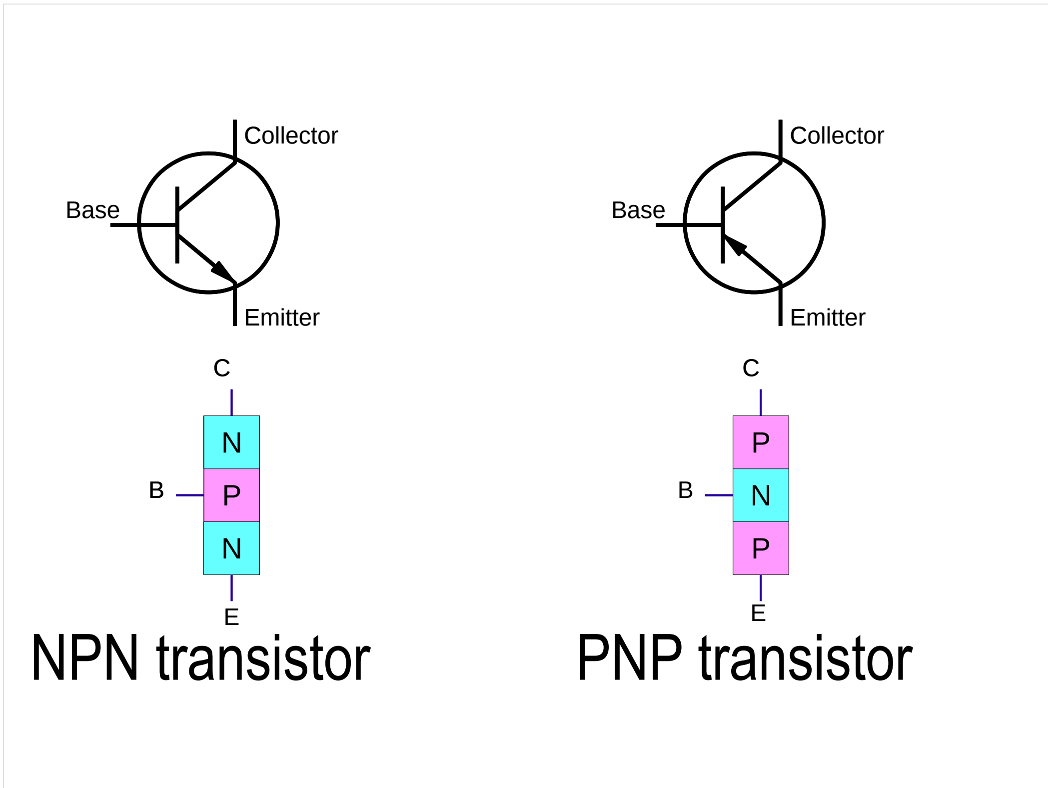
This works well for sequential access, but if you want something out of order, you need to wait until it comes around again. Like baggage claim.



Here’s a closeup of the gates.
You can see the metal and polysilicon (blue), with T marking transistors.
Each stage has two inverters. On clock 1, each bit passes from the first inverter to the second inverter. On clock 2, each bit goes from the second inverter to the next first inverter.
Thus, bits are passed from stage to stage.

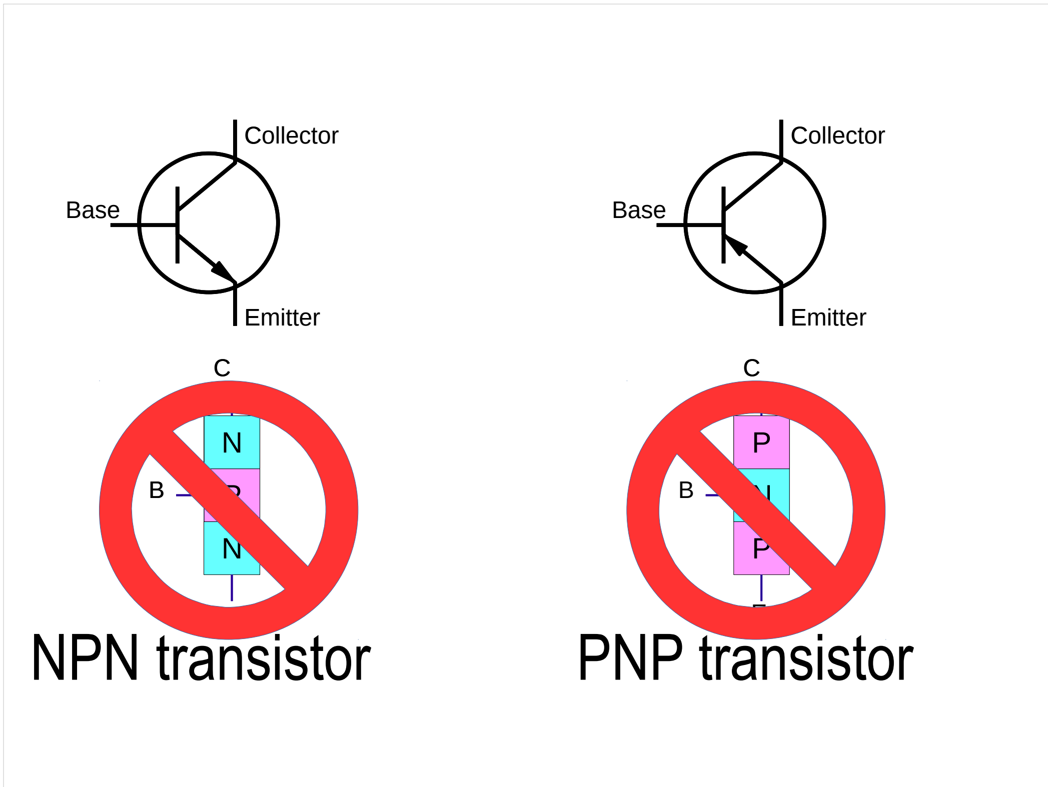


Now let’s look at some analog chips.
Has anyone used a 555 timer?



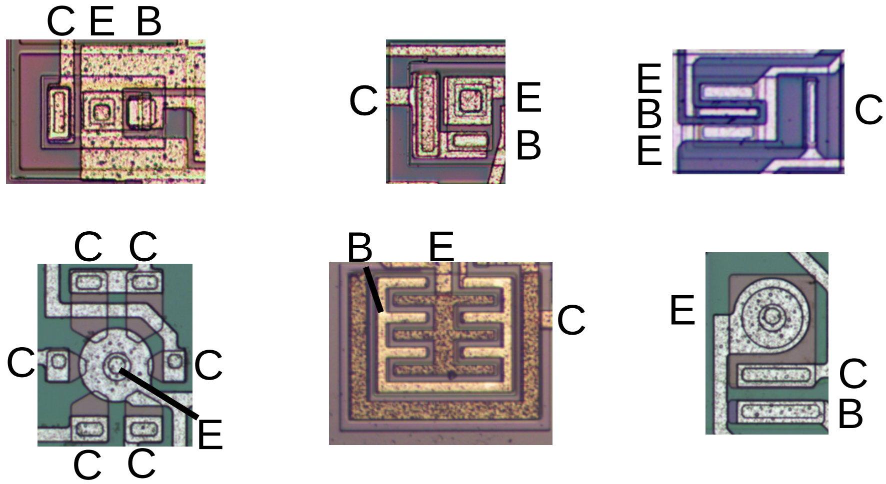
I’m looking at chips with bipolar transistors, NPN and PNP transistors.

You probably know what these look like. A NPN transistor is layers of N silicon, P silicon, and N silicon.



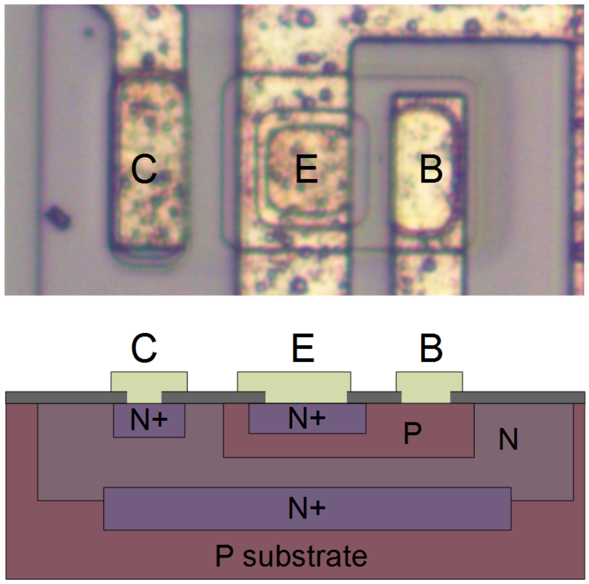
But on a chip, transistors look nothing like this. Usually the base isn't even in the middle.

What bipolar transistors really look like



Here are some real transistors.

The emitter may be in the middle, there may be two emitters, there may be 6 collectors, and you may not even see thebase.



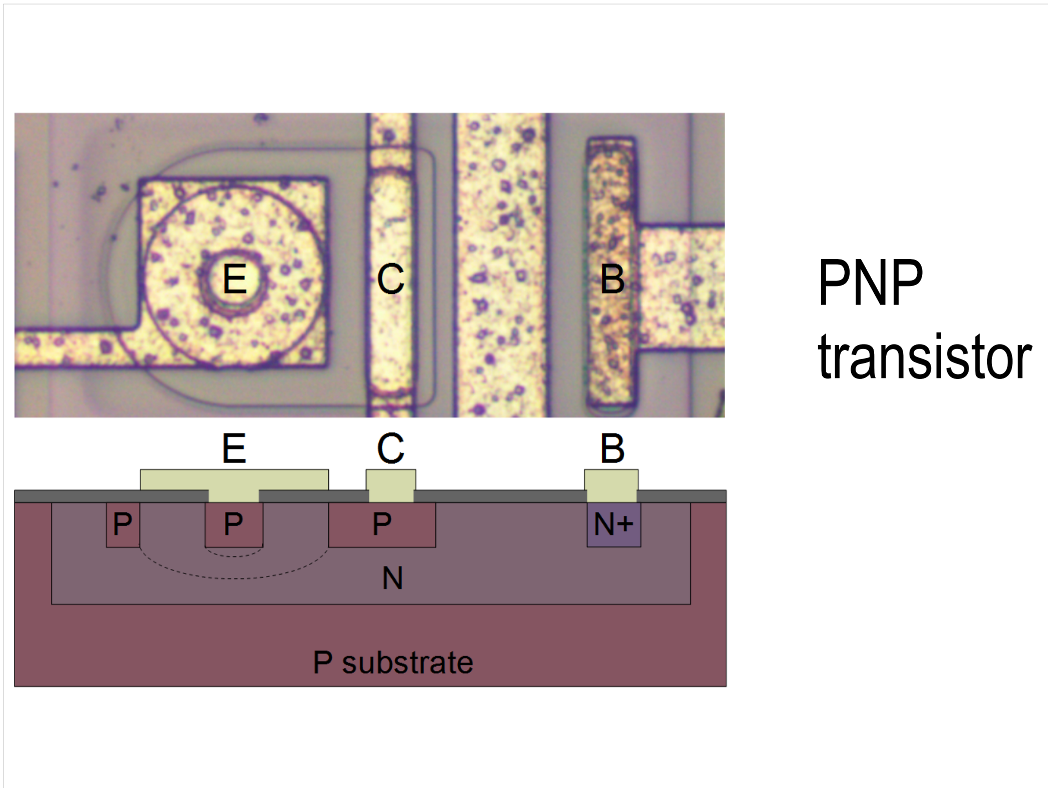
NPN transistor

To understand what's going on, let's look at a cross section.
Bipolar transistors are a lot more complicated than MOS transistors (which is one reason why computers use MOS).

You can see that there's a N region, with a P region on top for the base, and then a N region for the emitter.

Under the emitter, you can see the N-P-N stack.

When you're looking at a die photo, the emitter has multiple rings. The base region surrounds the emitter. The collector is kind of off on its own.

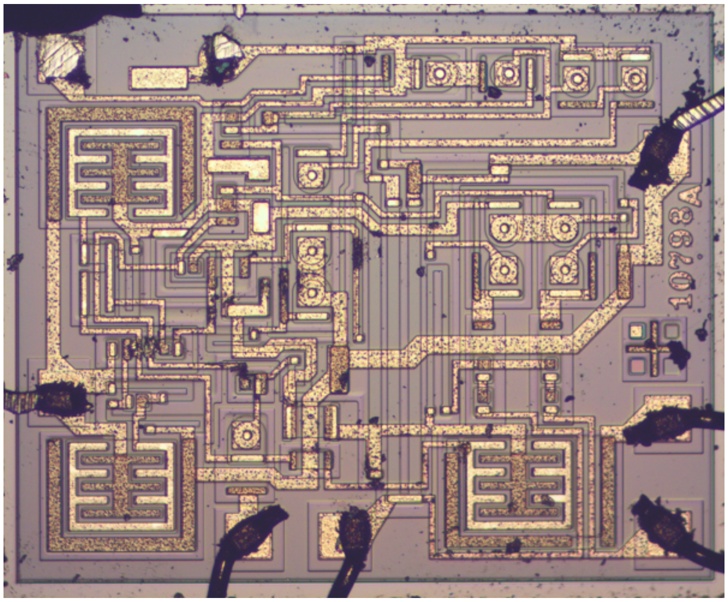


PNP transistors are totally different.

They usually have a circle structure, where the base forms a ring around the emitter, and then the collector surrounds that.

You can see the PNP layers laterally.

The base connection can be distant; there's actually a wire running across the transistor.

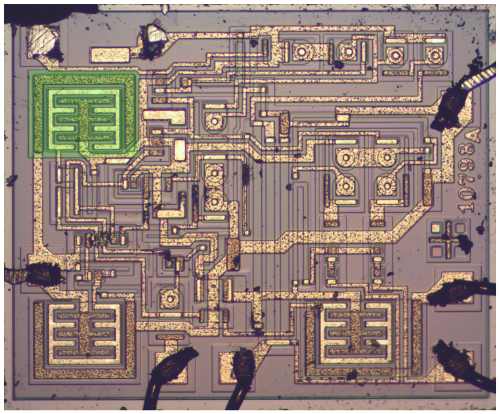


555
timer

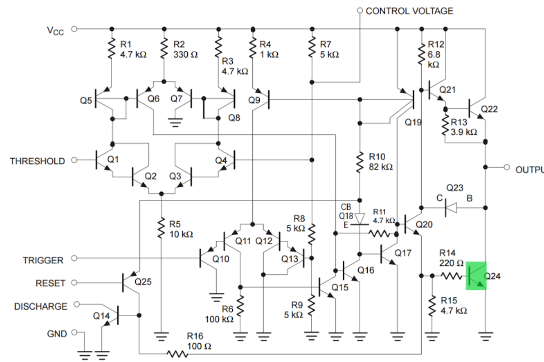
Now we can look at my 555 die photo.
You can see where the wires are attached.
Circular PNP transistors.
Rectangular NPN transistors.
Three big output transistors.

Three resistors in the middle for the voltage divider.
Resistors are inconveniently large on ICs.
Two comparators.
A flip flop to keep track of charging and discharging.

Interactive chip viewer



Q24 is a high-current transistor to pull the output low.



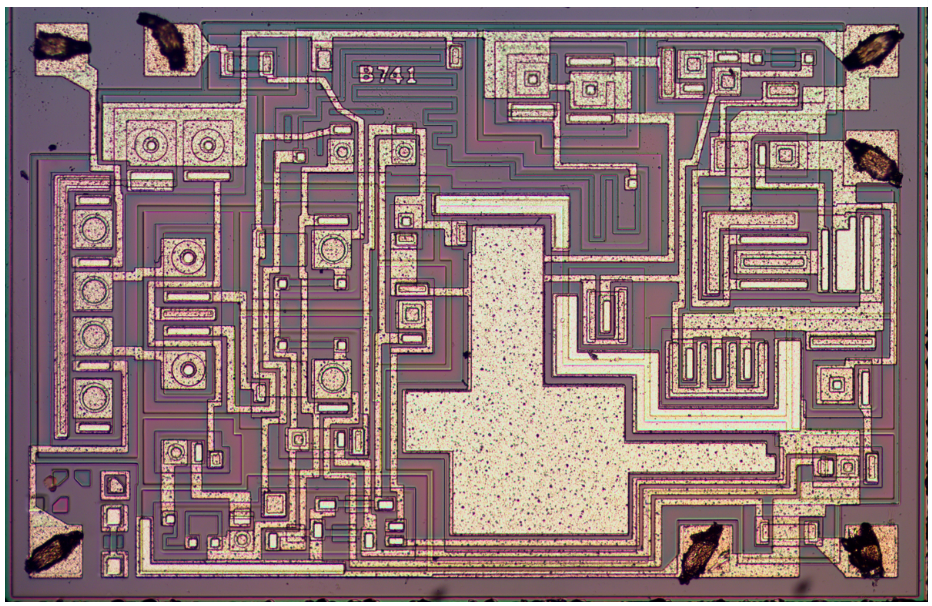
More info: righto.com/555

I made a chip viewer. Click the IC and it explains what that component is, and shows it on the schematic.

741
op
amp

Now let’s move to the famous 741 op amp.
It came out in 1968 with hundreds of millions sold.

741 op amp

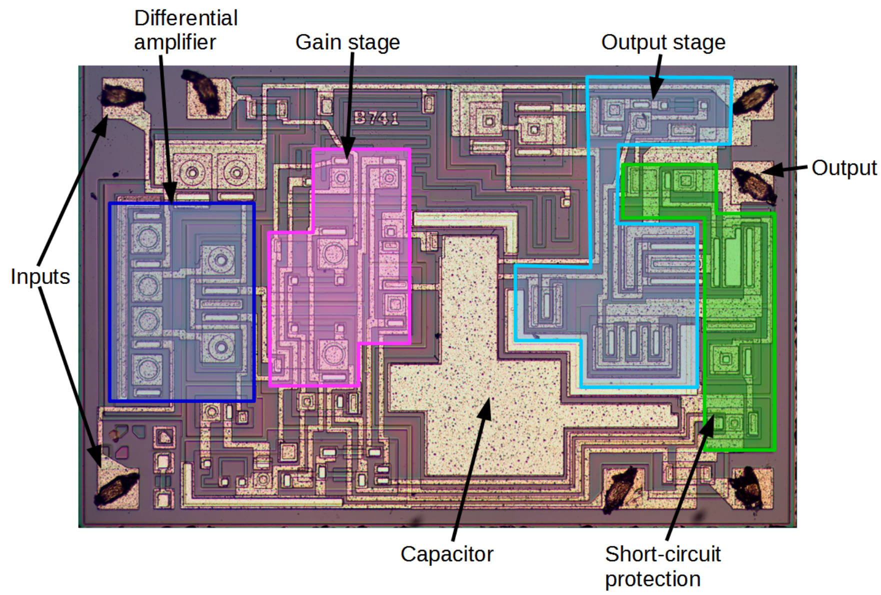


On the die you can see the transistors.

There's a big resistor snaking around the top.

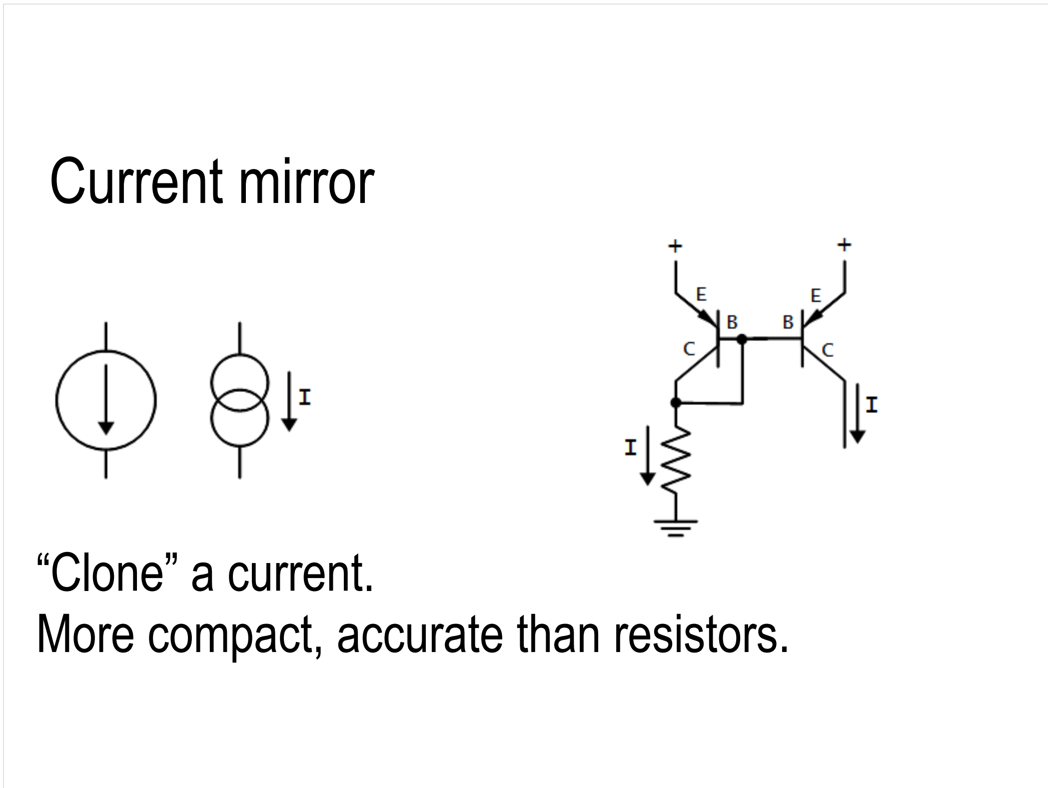
The huge thing in the middle is a capacitor.

Previous op amps required an external capacitor, but the 741 designer said engineers are lazy, so he put a capacitor inside the chip. The 741's popularity proves he's right, engineers are lazy.



Here are the main components. The differential amplifier finds the difference of the inputs. The gain stage amplifies this. The output stage has big capacitors to drive the output.

Another feature that made the 741 popular is short-circuit protection. If you short the output, these transistors shut down the chip before it burns up.



I want to talk a bit about current mirrors since they are very common on analog chips.

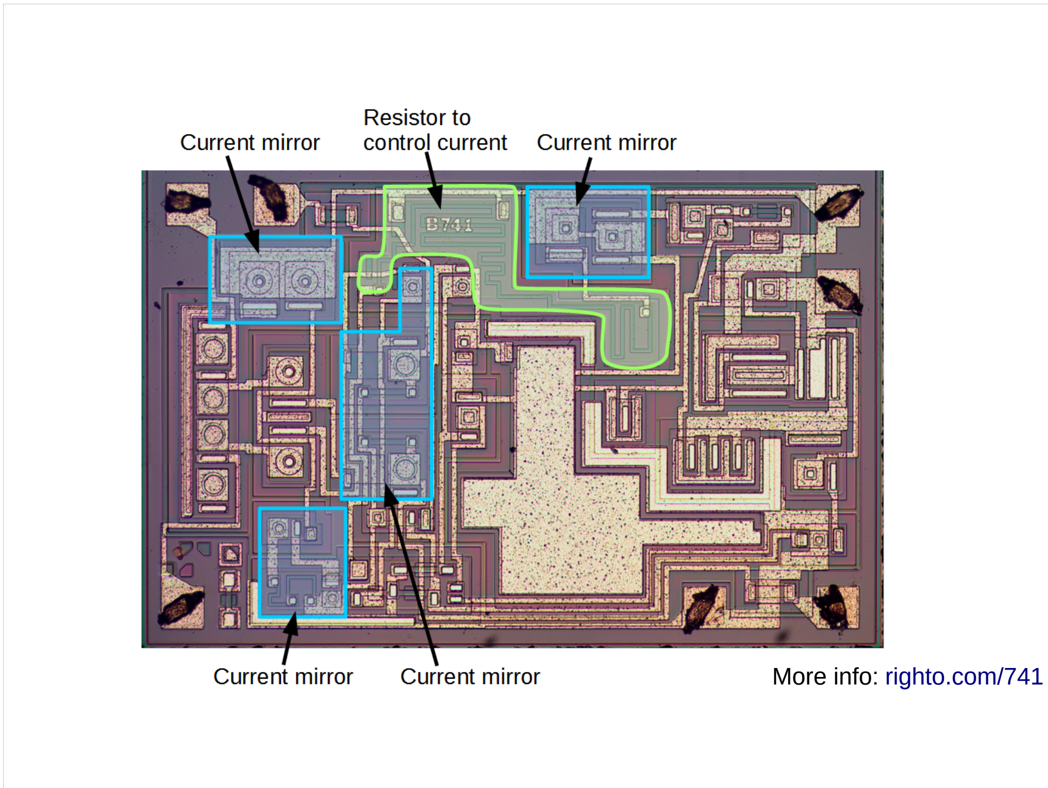
You may have seen the current source on datasheets and wondered what a current source is.

Suppose you need a fixed current, for a bias, for a pull-up, or for other circuits. You can resistors to control these currents.

When you’re building a circuit, resistors are cheap and transistors are expensive. But on an IC, it’s the other way around.

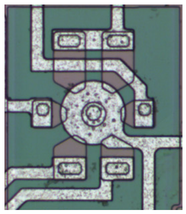
So instead of using a bunch of resistors, you use current mirrors.

You set one current with a resistor. Then you can use transistors configured like this to mirror the current. So the current on the right is the same as the current on the left.

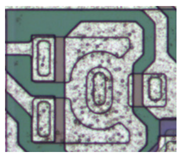


If we look at the 741, there are lots of current mirrors. There's one big resistor up here that sets the current. And then current mirrors to copy that current precisely.

Unusual current mirror transistors



6 collectors:
6 mirrored outputs



2 big collectors, 1 small:
Scaled output currents

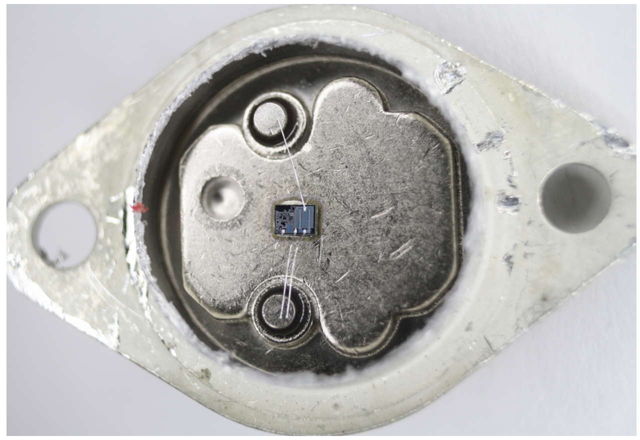
Photos: visual6502

Some chips do crazy things with current mirrors. You can make a transistor with 6 collectors, so you have 6 current mirror outputs.

Or you can make a transistor with two big collectors and 1 small collector, so you have two big currents and one small current.

One interesting thing about looking at ICs is finding these things that don’t exist as discrete components. You’re not going to find a 6-collector transistor at Frys.

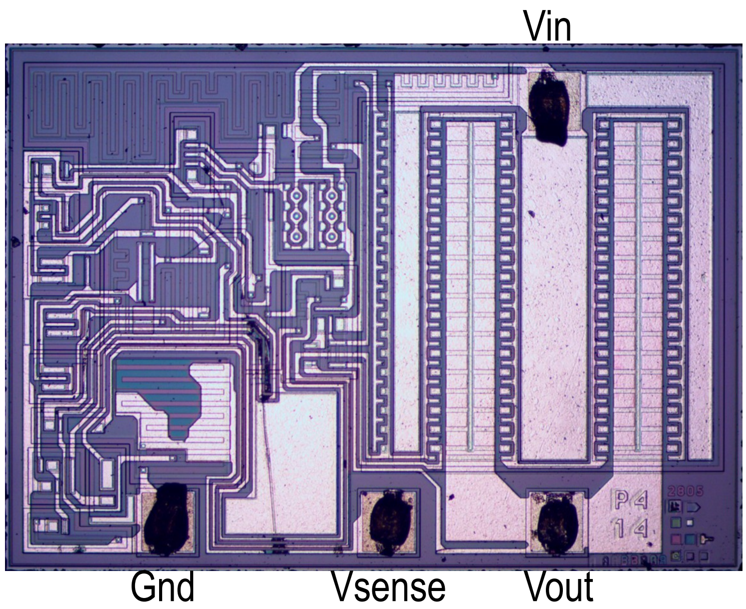
7805 voltage regulator



7-25V in
5V out

Anyone here ever used a 7805 voltage regulator to get 5 volts? Probably a lot of you.

Here's one I cut open. There's a tiny die in this huge package.
See the tiny wires connecting the die to the package.
Note there are two wires from the output pin to the die; I'll get to that.



Here's my die photo of the 7805.

You can see where the wires attach for the voltage in, the voltage out, and ground.

That second wire I mentioned is the voltage sense. Since there can be a voltage drop between the die and the pin, the second wire lets you measure the voltage at the pin, for more accuracy.

Some features of the chip:

You can see the transistors.

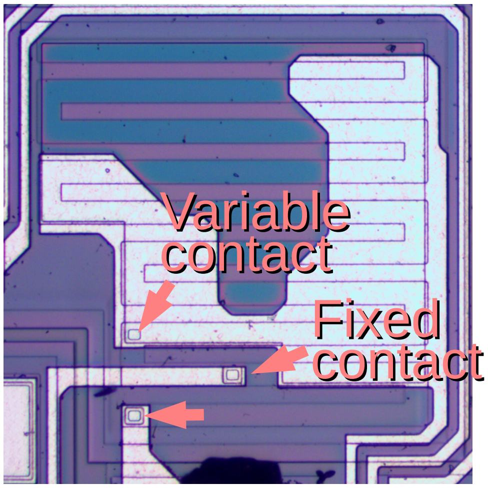
Half the chip is taken up with the 1 amp output transistor.

Here's a current mirror.

Here's a capacitor to keep the chip from oscillating.

Over here is the bandgap regulator, which is just a temperature compensated voltage regulator.

A family of regulators from one chip



5, 6, 8, 10, 12, 15, 18, 24 volts

Move contact to change voltage divider

More info: righto.com/05

This resistor looks a bit strange, with metal over it. And with the contact here, most of the resistor is wasted.

The idea is this one design can generate from 5 to 24 volts, by changing the variable contact and thus changing the voltage divider. The divider output is regulated to 3.75 volts, so the final output voltage depends on the resistor. A cute trick.

Die photos: Metallurgical microscope

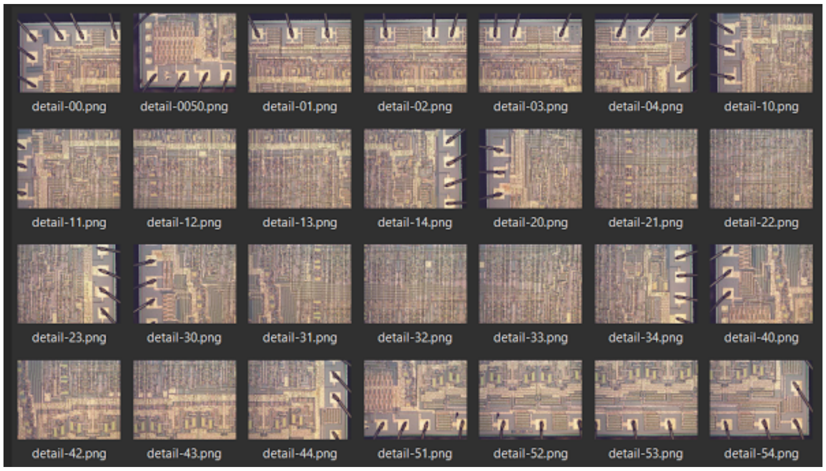
Shines light from above through lens

Now I want to step back from specific chips and explain how I get these photos.

The secret is a metallurgical microscope.
Normal microscopes shine light from below, which works okay for cells, but not for opaque ICs.
The metallurgical microscope has this goofy light unit that shines light from above, so the chip is illuminated.

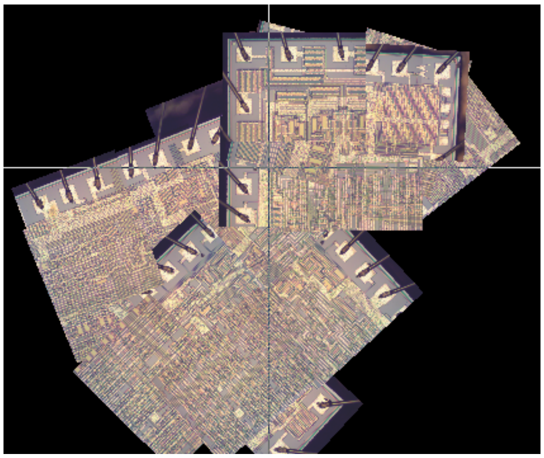
You can try a regular microscope and a flashlight but this works a lot better. You can spend a whole lot on microscopes but I got this one on eBay for a couple hundred dollars.

Stitch photos together for high-resolution



Then I take a bunch of photos and stitch them together into a high-resolution image with a program called Hugin.

Hugin takes some practice

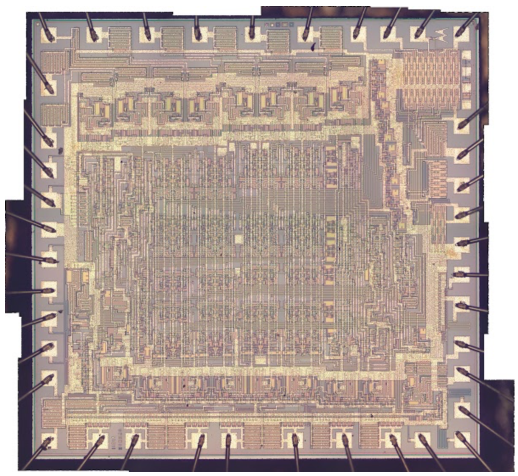


Tip: have lots of overlap between images

More info: righto.com/hugin

It took me a while to get the hang of the process. At first I ended up with cubist Picasso-style images. One tip is to overlap successive images much more than you'd expect, so the software can match them up.

Motorola 6820 PIA chip



Here’s the previous image once I got it working. This is a Motorola interface chip, used in the Apple I.

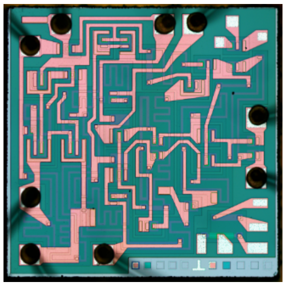
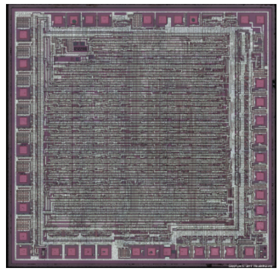
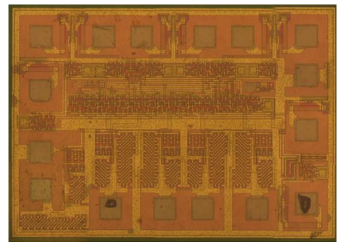
How to get to the die?

Photo: zeptobars

Hard way: boil chips in sulfuric / nitric acid

How do you get inside the chip? The experts like visual 6502 dissolve the epoxy in sulfuric and nitric acid. That’s a bit too intense for me, since I don’t want to dissolve my lungs or end up with a superfund site.

Easy way: download die photos

zeptobars.com, visual6502.org, siliconpr0n.org

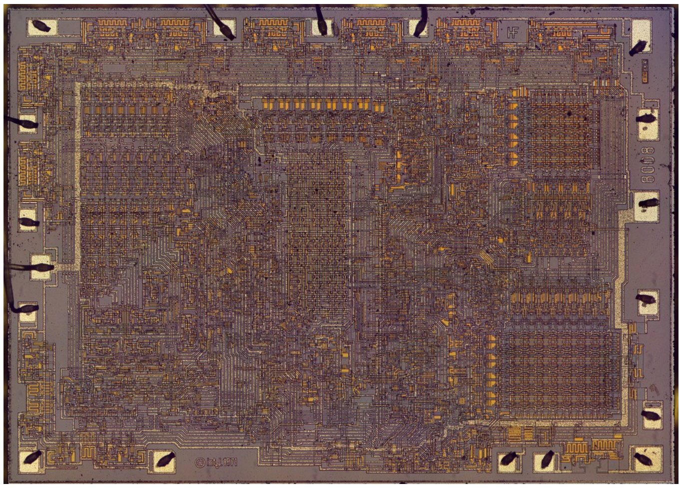
It’s much easier to let someone else mess around with acid and I just download the photos. If you want to try reverse-engineering some chips, there are lots of die photos on the internet. Here are some sources.

But there are a lot of chips you can open without acid, so that's what I do. Chips in metal cans are on eBay for a few dollars and you can open them with a hacksaw. Or get a jeweler's saw, which works better.

Other chips have a metal lid that you can pop off with a chisel.

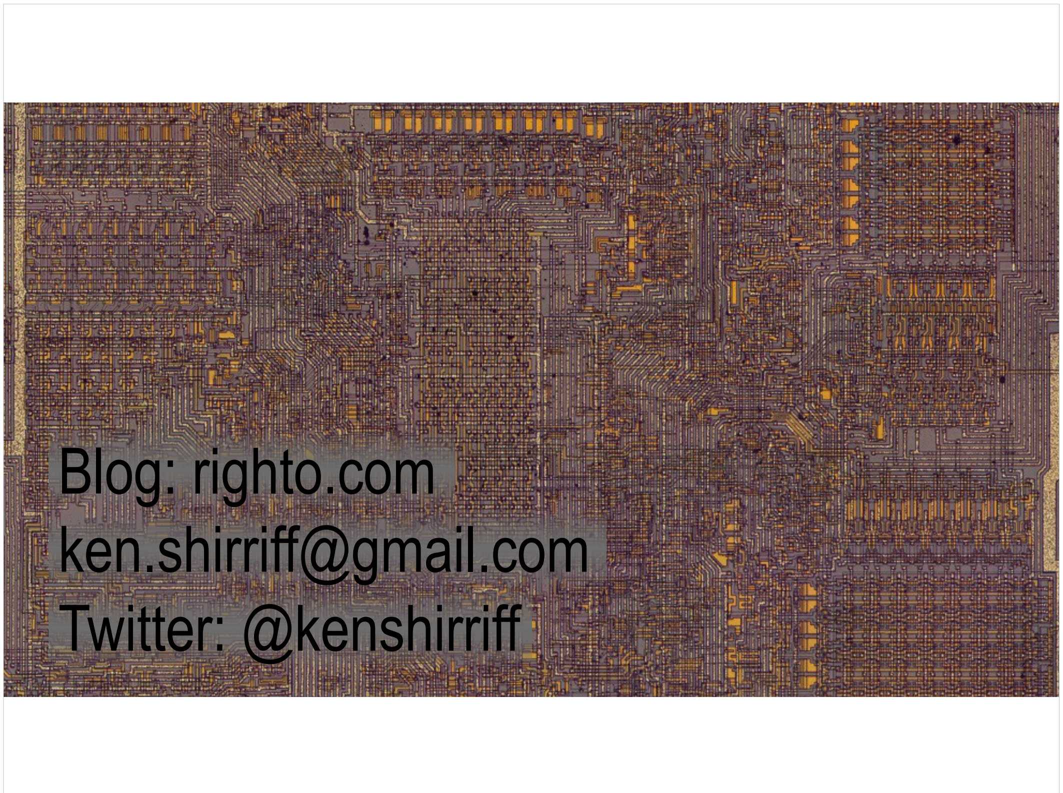
And ceramic chips like this 8008 I got off eBay also pop apart with a chisel. I couldn't find good die photos of the 8008 online. Since it's historically interesting, I figured I'd take photos myself.

Current project: 8008 analysis



Last week I took this die photo, and I’m currently analyzing it.
You may be able to recognize some of the features now: the pins, driver transistors.
Over here these regular blocks are the registers.
The data bus runs across the top.
In the middle is the instruction decode PLA, with the instruction register connected to the data bus.
On the left the data bus connects to the ALU.

I plan to write about the 8008 in the next few weeks, so stay tuned.



Thank you!
Now go out and reverse engineer some chips.