Reverse engineering a VIC-20 expansion cartridge

Ben Dooks
ben.dooks@codethink.co.uk
ben@fluff.org

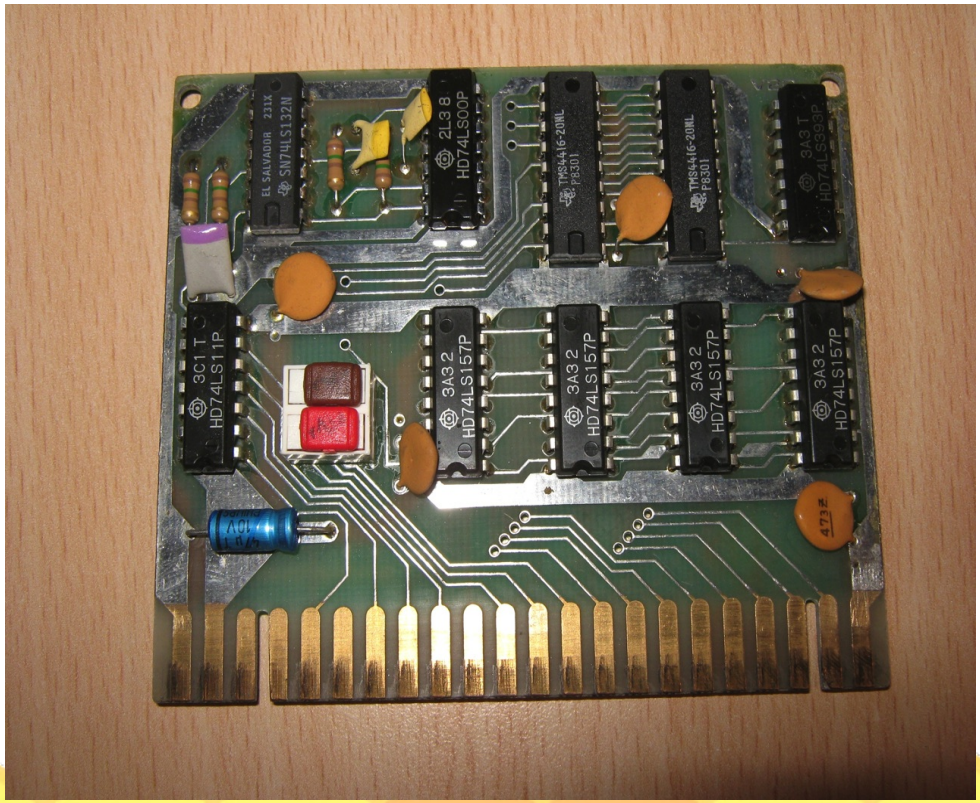
The VIC-20

- Released 1980
- 20KiB ROM
- 5KiB RAM
- $299 price

Ben Dooks <ben@fluff.org>

The Vixen RAM expansion

- 16KiB RAM expansion (8bit x 16K)



But why?

- My second ever computer.
- Browsing peripherals and found the Vixen RAM
  - And photos of the internals
  - But how did it work?
  - Is there sufficient information in the pictures?

RAM

- Two main types
  - Dynamic (DRAM) and Static (SRAM)
- Most other expansions used static RAM
  - But the vixen uses dynamic RAM
  - And the circuit looks far too simple

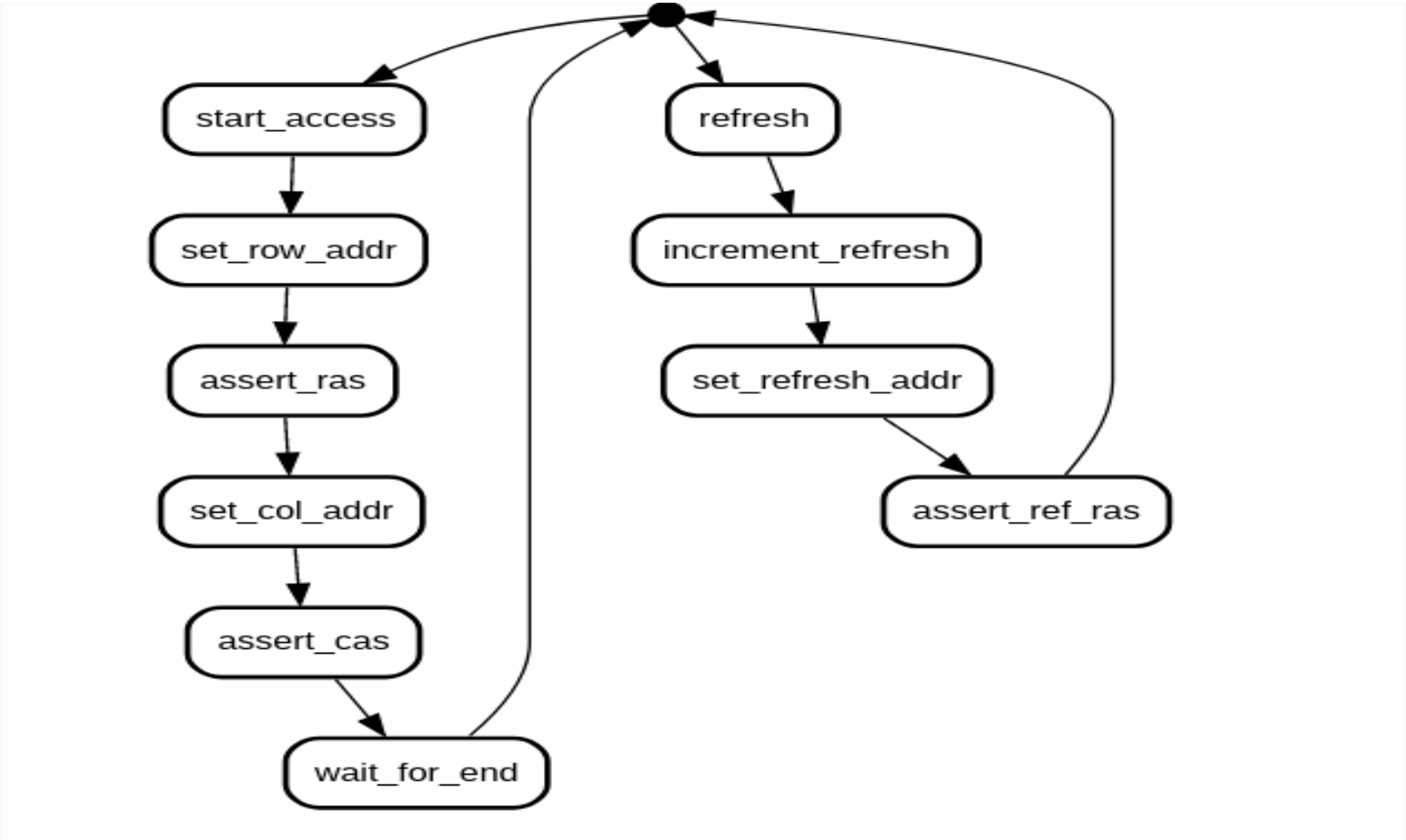
Ben Dooks <ben@fluff.org>

Dynamic vs Static RAM

Dynamic RAM	Static RAM
Fast access time	Slower access time (250ns)
Multiplexed address bus	Simple address bus
External refresh cycle	No refresh cycle
Cheap	Expensive

Ben Dooks <ben@fluff.org>

DRAM access states

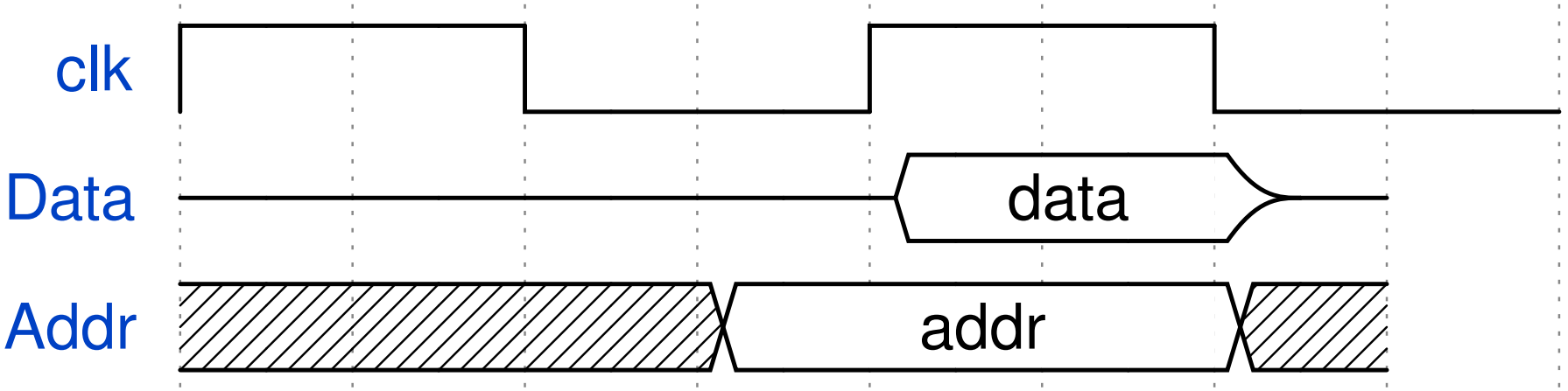


VIC-20 expansion bus

- Simple extension of 6502 main bus
  - 8 data and 13 address bits
  - Read and write
  - 9 pre-decoded chip selects
  - CPU clock
  - Soft and hard interrupt inputs

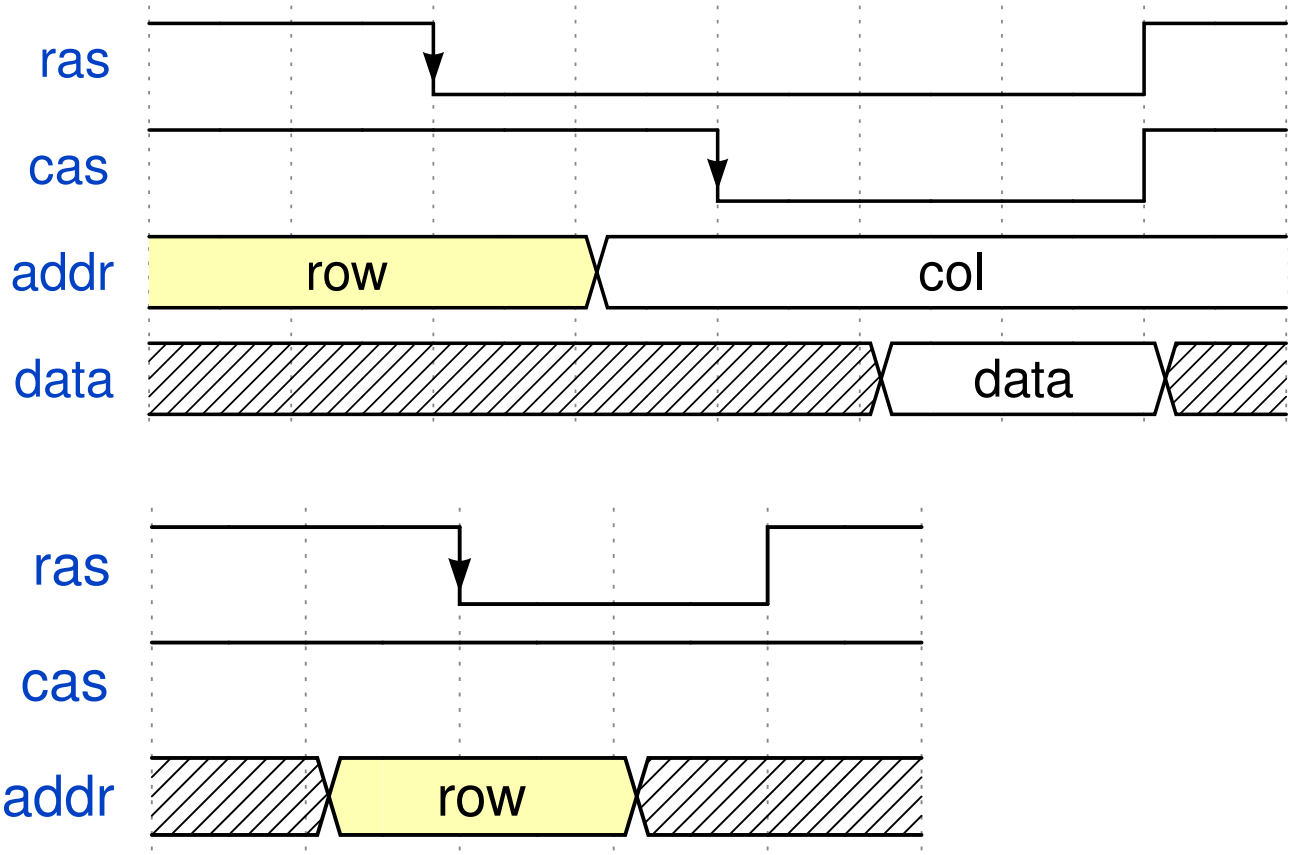
Ben Dooks <ben@fluff.org>

6502 Bus



Ben Dooks <ben@fluff.org>

DRAM timing



First thoughts

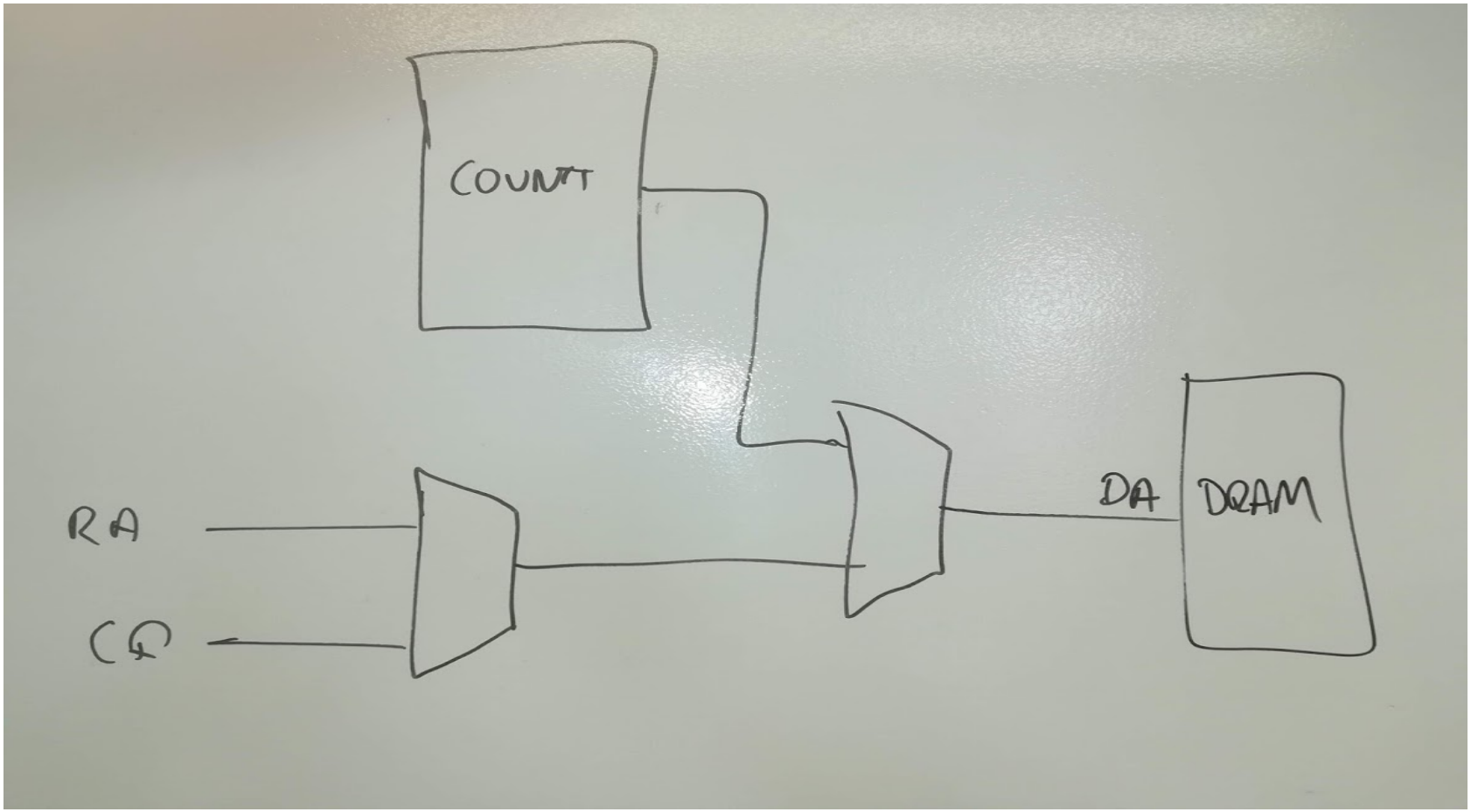
- The picture tells a lot
  - Components used
  - Basic track info
- All standard logic gates
  - Datasheets available
- Simple components
  - Resistors, capacitors
  - So no room for PLL/state machine

Ben Dooks <ben@fluff.org>

Logic ICs

- 2x TMS4416 16K - 4bit DRAMS
- 1x 74LS393 – 2 x 4 bit binary counter
- 4x 74LS157 – 2 to 1 data mux (4 channels)
- 1x 74LS11 – 3 in AND gate (x3)
- 1x 74LS00 – 2 in NAND gate (x4)
- 1x 74LS132 – 2 schmitt NAND gate (x4)

Simple start (address mux)

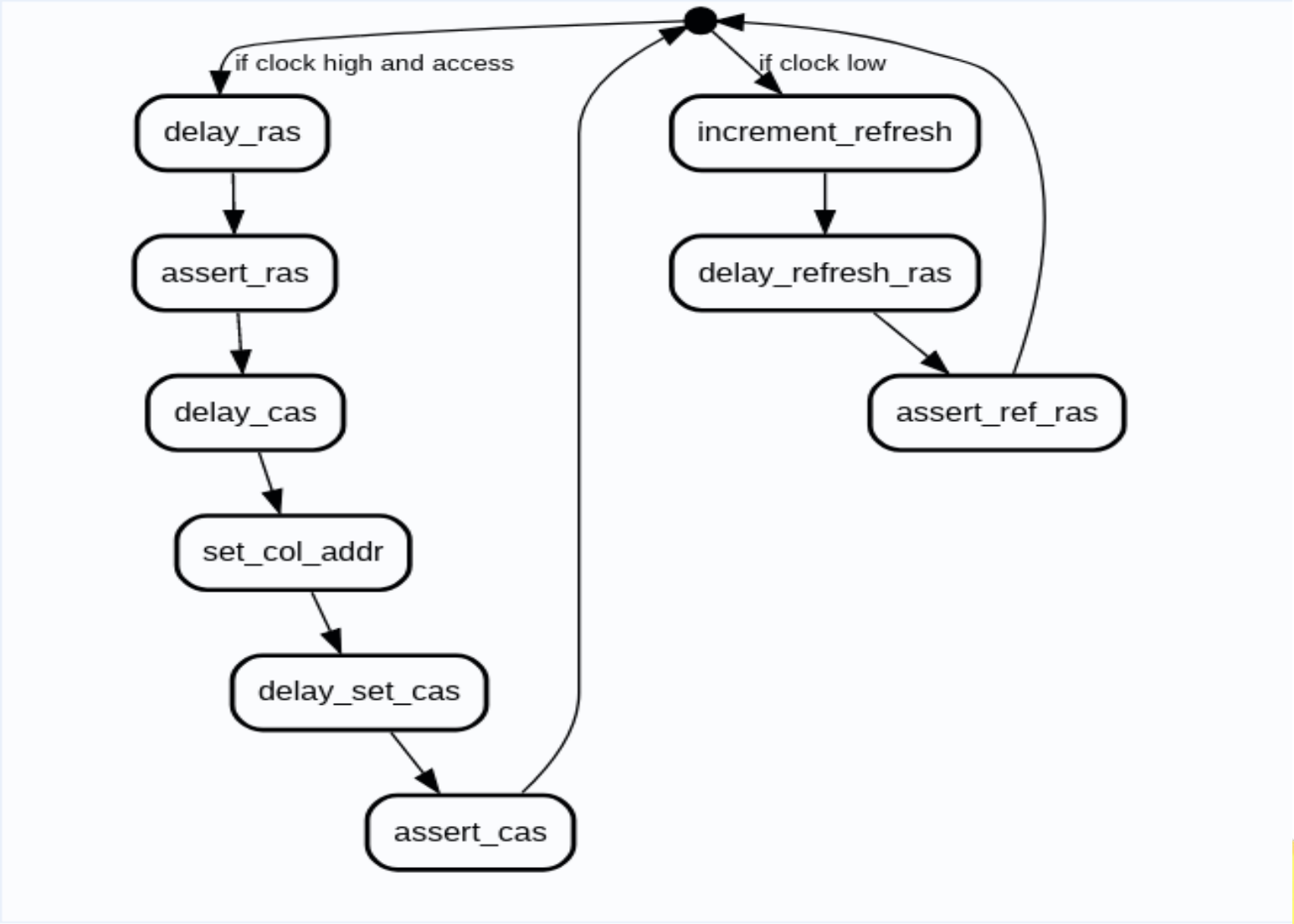


So, refresh

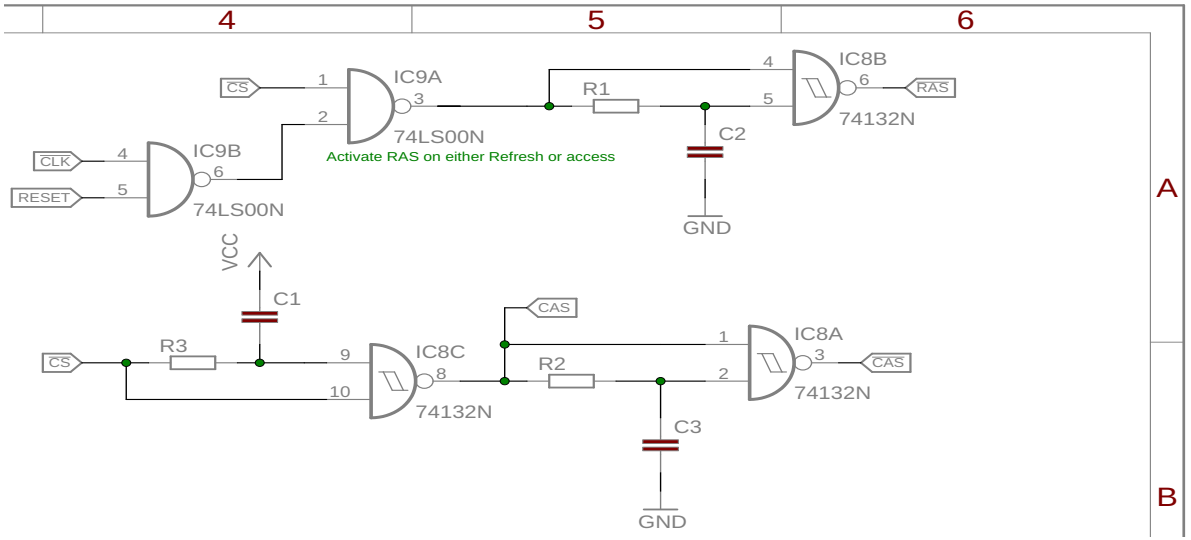
- DRAM refresh cycle is around 400ns
  - Use inactive half of cycle to refresh
  - A bit more power
  - Normal VIC20 draws 16W anyway

Ben Dooks <ben@fluff.org>

New state machine



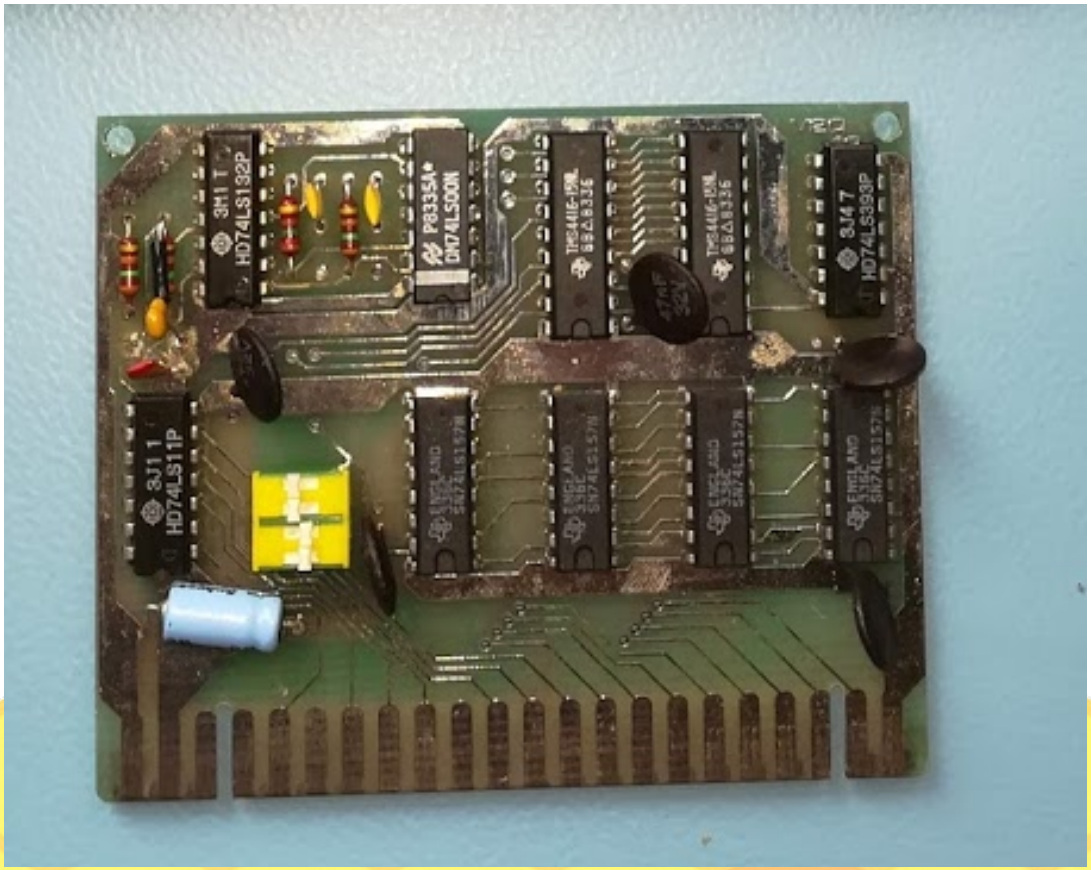
First schematic attempt



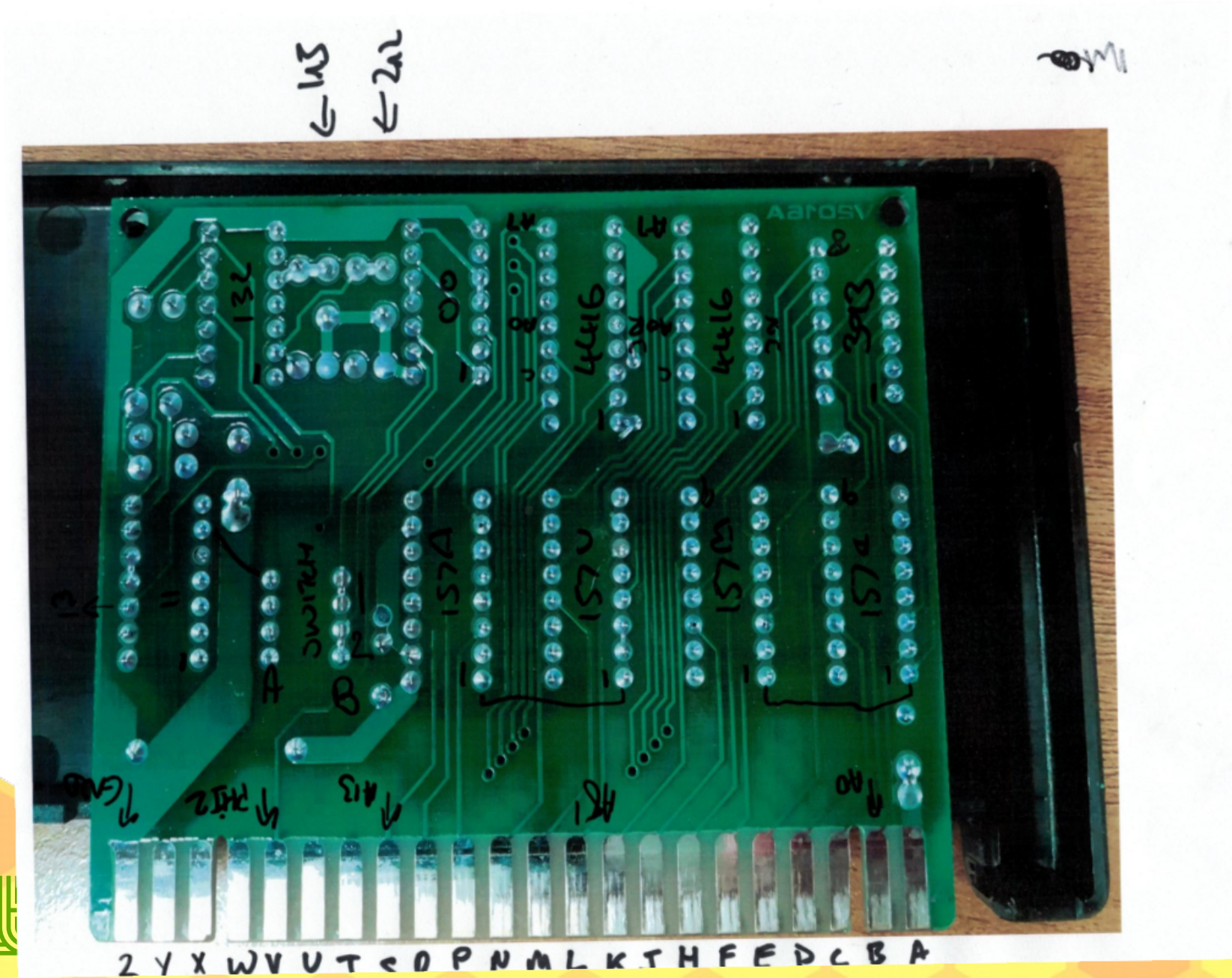
02/05/2019 23:30 f=2.00 /home/ben/eagle/vixen/vixen2/vixen1.sch (Sheet: 1/1)

The real hardware

- Track tracing
- Photograph and draw
- Multimeter
- And sneaky cap!



Tracing tracks



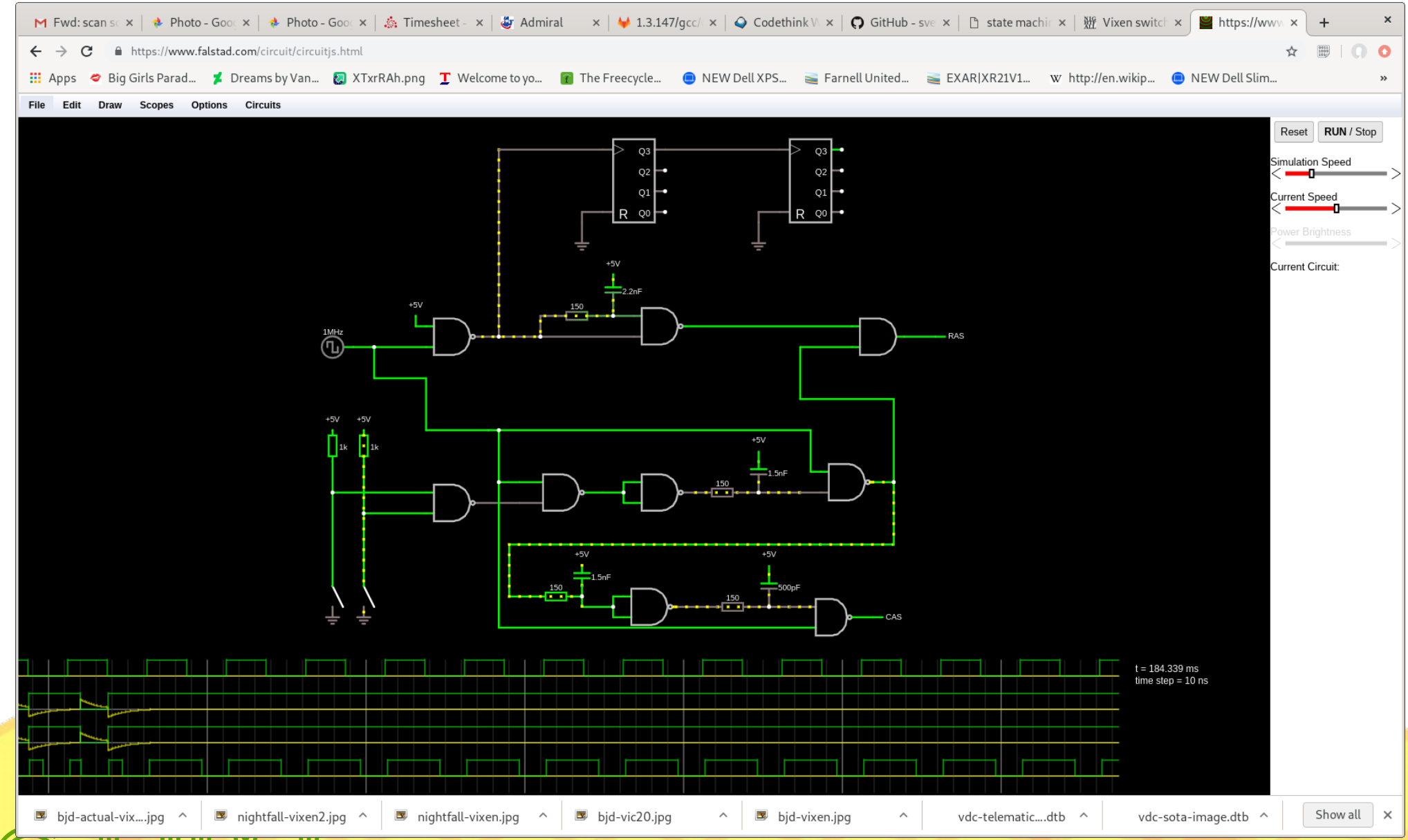
My clone board

Ben Dooks <ben@fluff.org>

How close was I

- Almost
- Refresh circuit wrong

Simulation



Ben Dooks <ben@fluff.org>

Future

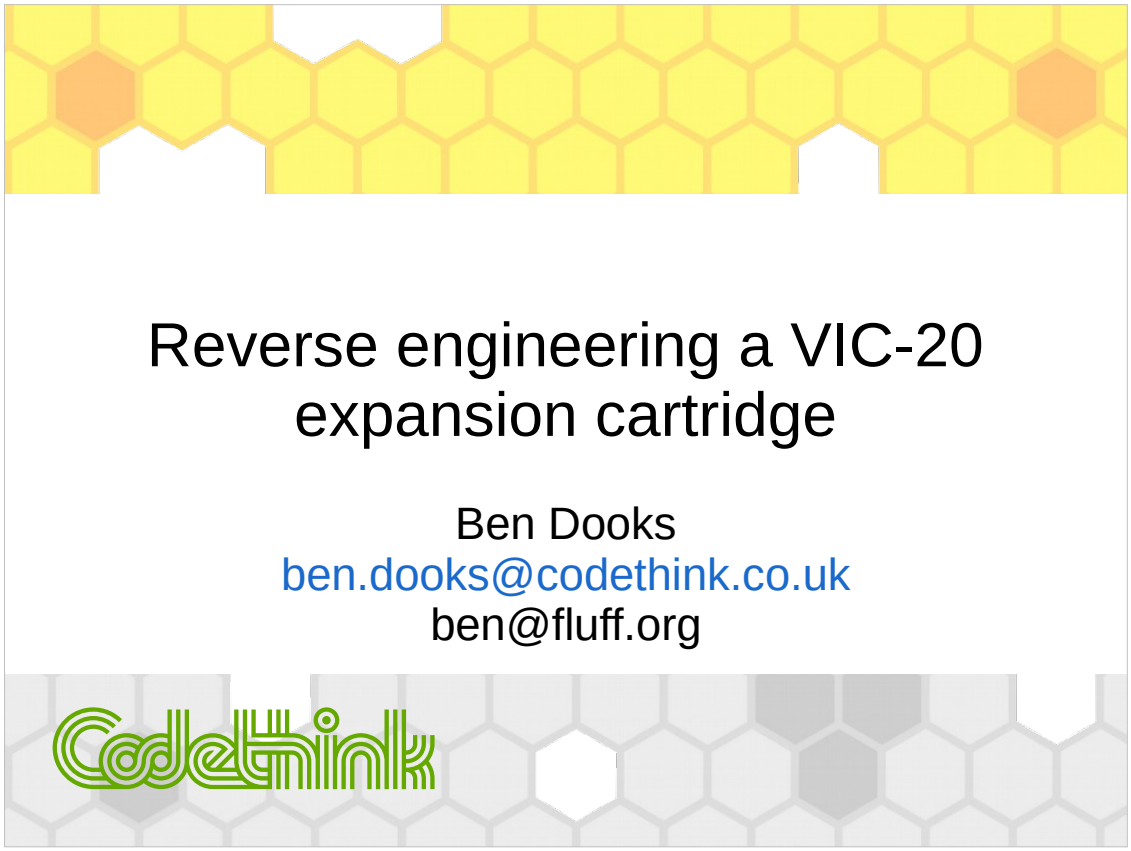
- Try optimisation of design
  - It is very close to optimal
- Try with PLL and state machine
  - Simple PLL ICs were available
  - Possibly PLL + shift-register based state

The end?

- Project files and documentation will be on:
  - https://github.com/bendooks/vixen

- Thank you. Any questions?

Ben Dooks <ben@fluff.org>



Reverse engineering a VIC-20 expansion cartridge

Ben Dooks
ben.dooks@codethink.co.uk
ben@fluff.org

I started off in the days of the 8 bit computer, this is a bit of a trip back to the pre PC days

Overview:

- the hardware involved
- quick overview of electrical interfaces
- some techniques on how to reverse engineer

The VIC-20

- Released 1980
- 20KiB ROM
- 5KiB RAM
- $299 price

CodeThink

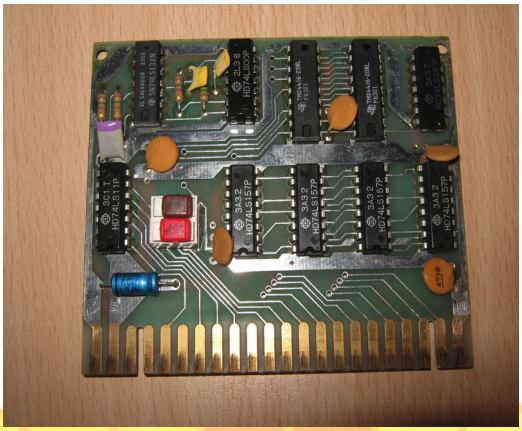
Ben Dooks <ben@fluff.org>

That’s around $778 in current money (approx £600)
https://en.wikipedia.org/wiki/Commodore_VIC-20

5KiB of RAM leaves about 3300 bytes free for user

The Vixen RAM expansion

- 16KiB RAM expansion (8bit x 16K)



CodeInk

Ben Dooks <ben@fluff.org>

Source :
https://www.nightfallcrew.com/18/05/2010/vixen-switchable-16k-ram-for-commodore-vic-20/

- found this device whilst browsing on line
- would have loved this as a youth
- no idea of the price of this device retailed
- does not fill the memory map (but lots more)

But why?

- My second ever computer.
- Browsing peripherals and found the Vixen RAM
  - And photos of the internals
  - But how did it work?
  - Is there sufficient information in the pictures?

Ben Dooks <ben@fluff.org>

-

RAM

- Two main types
  - Dynamic (DRAM) and Static (SRAM)
- Most other expansions used static RAM
  - But the vixen uses dynamic RAM
  - And the circuit looks far too simple

Ben Dooks <ben@fluff.org>

- Many cartridges used easier SRAM

Dynamic vs Static RAM

Dynamic RAM	Static RAM
Fast access time	Slower access time (250ns)
Multiplexed address bus	Simple address bus
External refresh cycle	No refresh cycle
Cheap	Expensive

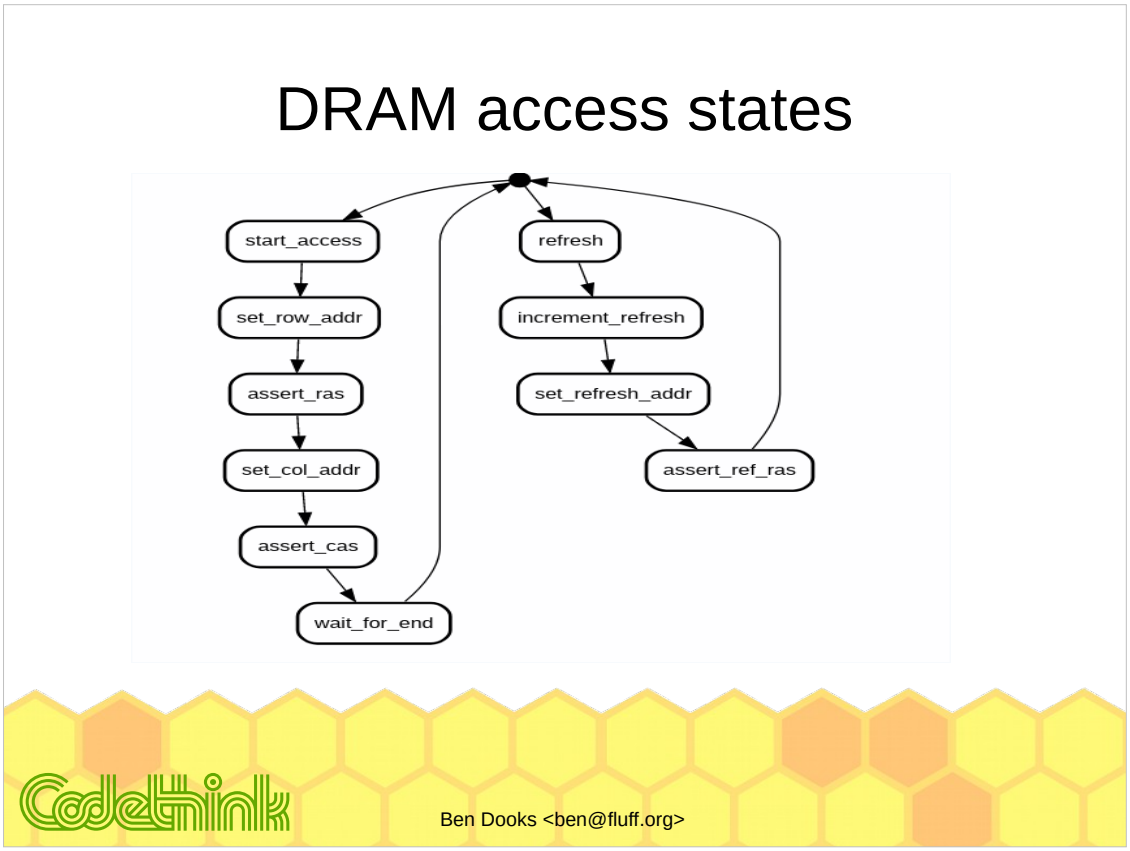
Ben Dooks <ben@fluff.org>

Quick table for dynamic vs static

- DRAM is much cheaper by about 8 times but requires multiplexed address and two stage address cycle.

- I think 2-3 times cheaper per 1KiB, so for 16KiB this is a significant saving.

- DRAM also needs an external refresh needs 128 rows refreshed every 4ms



https://state-machine-cat.js.org/

Quick view of DRAM access states for normal data and for a refresh.

In a modern system this would be a simple state machine you could build in a CPLD or FPGA but would require a 8-16 multiplied clock

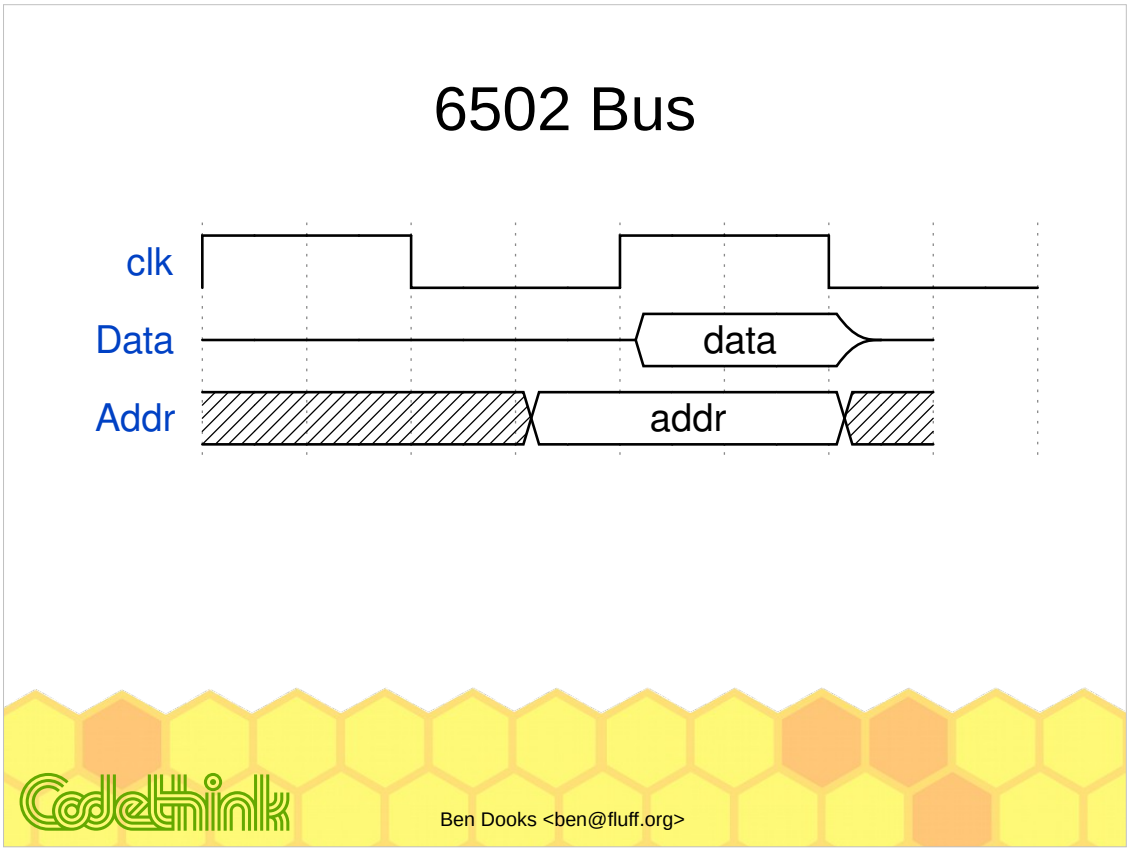
VIC-20 expansion bus

- Simple extension of 6502 main bus
  - 8 data and 13 address bits
  - Read and write
  - 9 pre-decoded chip selects
  - CPU clock
  - Soft and hard interrupt inputs

Ben Dooks <ben@fluff.org>

This is a fairly simple parallel bus
Cycle time is about 1MHz (1000ns)

- note there is no way to stall memory cycles for refresh

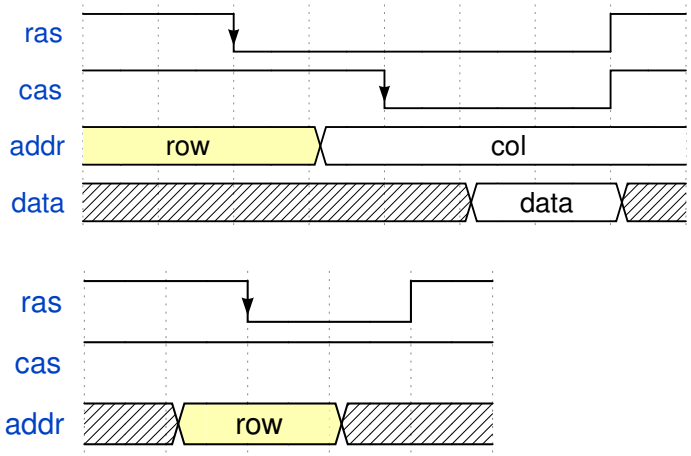


Simplified diagram of 6502 memory access cycle

Address is setup before the high cycle of the clock
Data is read or written during high part of clock.

The clock cycle is around 1MHz so approx 1000ns per cycle. This we will come back to later

DRAM timing



Ben Dooks <ben@fluff.org>

First is simple read/write single access.

Second is RAS only refresh cycle of one row

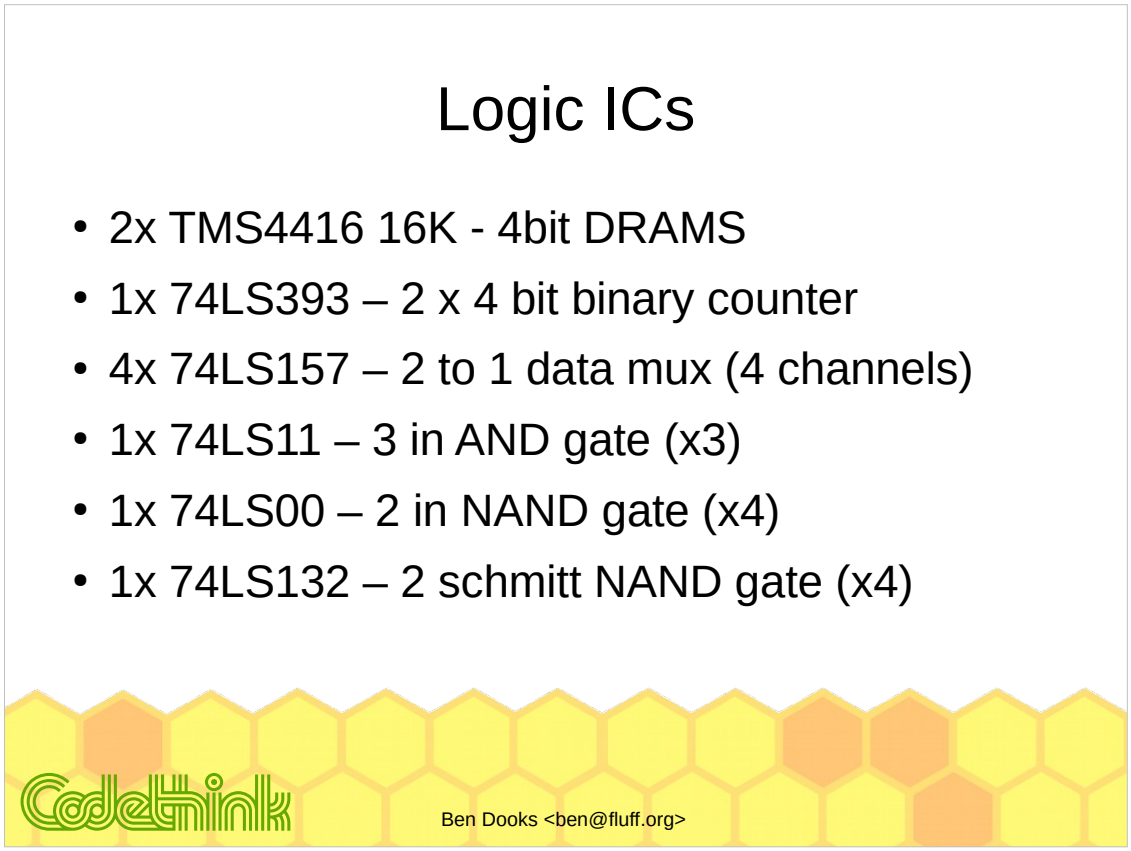
First thoughts

- The picture tells a lot
  - Components used
  - Basic track info
- All standard logic gates
  - Datasheets available
- Simple components
  - Resistors, capacitors
  - So no room for PLL/state machine

Ben Dooks <ben@fluff.org>

- most are simple logic gates
    - combinatorial logic and simple
    - full list in later slide
- can sort of read some of the values from passives
- list of ICs on the next slide

- therefore no complicated state machine
- certainly no PLL here

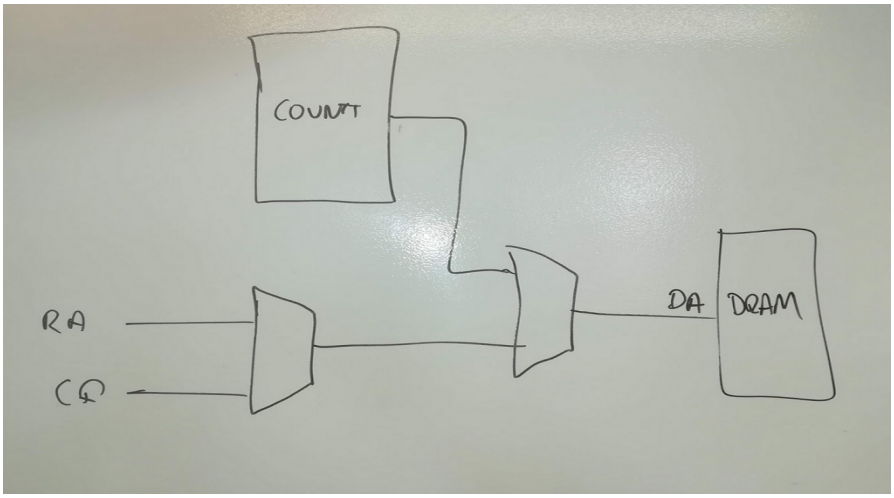


So we have about 13 usable simple logic gates and bits for data muxing

TMS4416 =
74LS157 = 89¢
74LS393 = $2.49

Prices at Jan 1981
https://www.americanradiohistory.com/Archive-Poptronics/80s/1981/Poptronics-1981-01.pdf

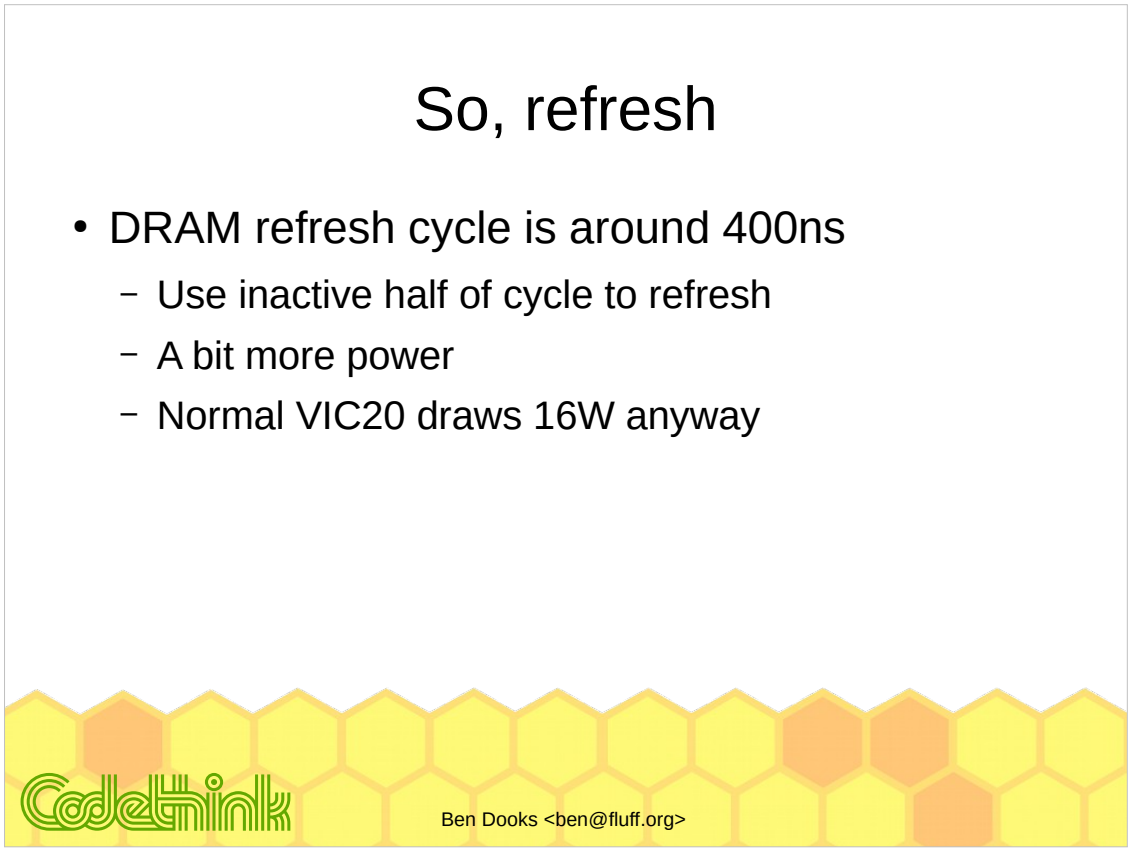
Simple start (address mux)



Ben Dooks <ben@fluff.org>

The 4-by-2 muxes are a good indication they are part of the address row/column and refresh. So it is quite easy to work out the few possibilities for the address mux.

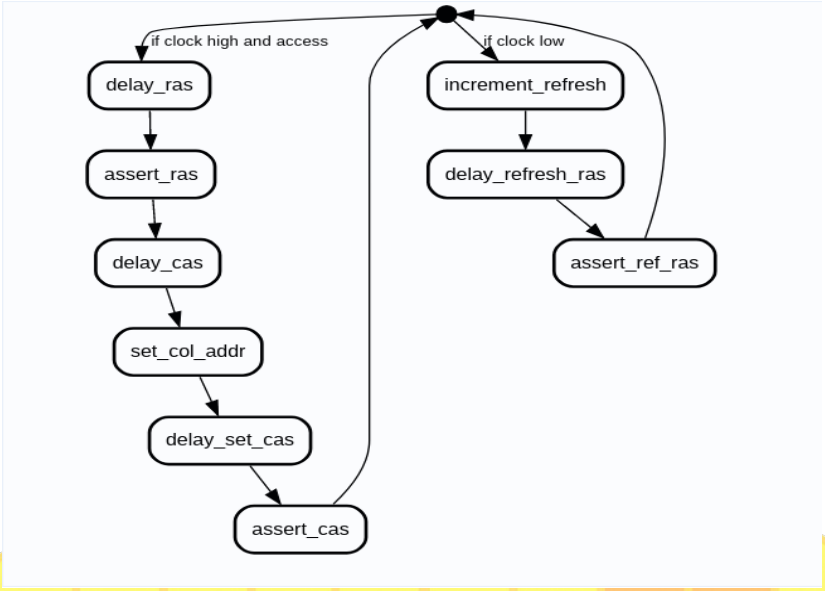
We'll end up refreshing around 32 times more than we need to.



Since the DRAM requires 400ns to refresh and we have 500ns of inactive cycle of the CPU clock we can simply refresh each cycle.

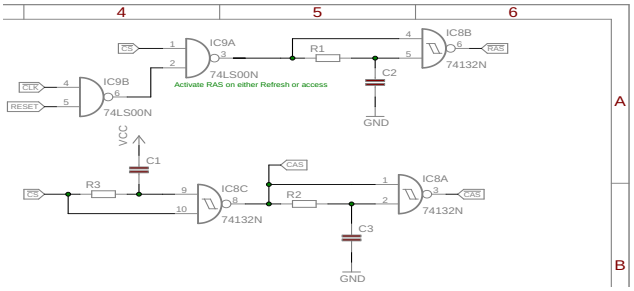
Worst case refresh current is 46mA per device, average is about 23mA.

New state machine



Ben Dooks <ben@fluff.org>

First schematic attempt



02/05/2019 23:30 f=2.00 /home/ben/eagle/vixen/vixen2/vixen1.sch (Sheet: 1/1)

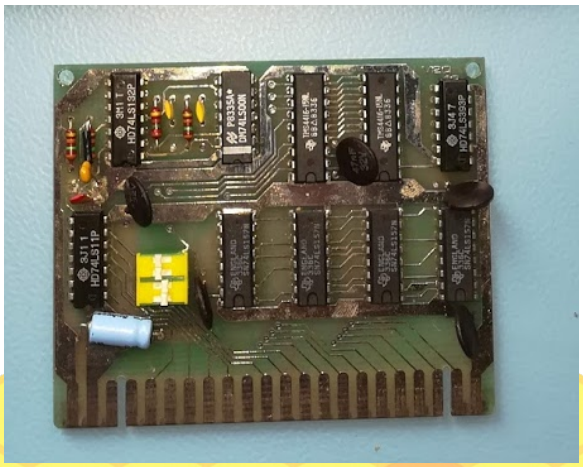
CodeThink

Ben Dooks <ben@fluff.org>

Simple logic gate RC delays to produce RAS/CAS from the clock and the access.

The real hardware

- Track tracing
- Photograph and draw
- Multimeter
- And sneaky cap!



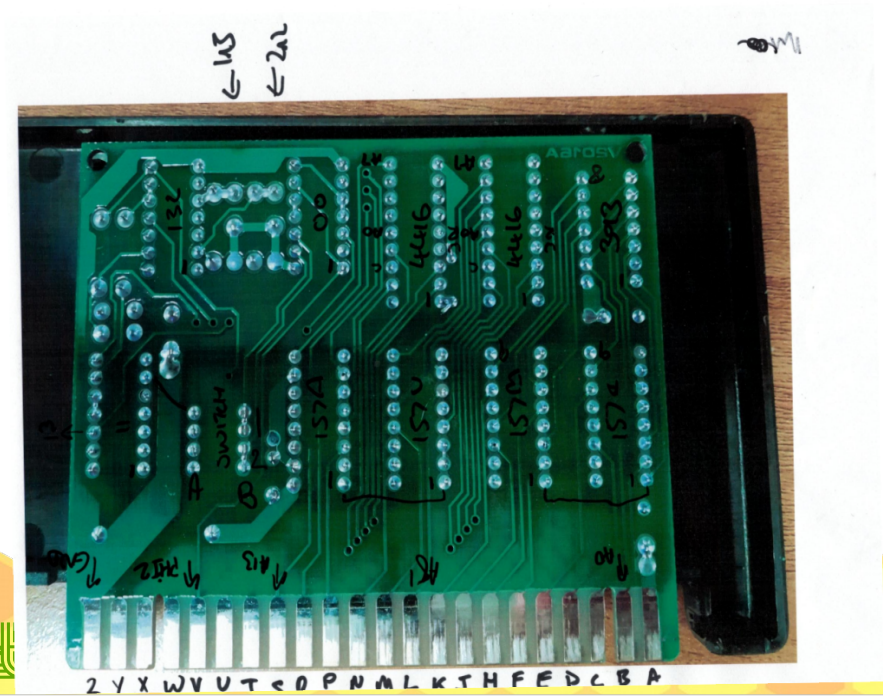
CodeThink

Ben Dooks <ben@fluff.org>

- searching find there is one on ebay
→ so bid and win

The photo shows the hardware better and there is an extra capacitor hidden in the original.

Tracing tracks



Take photo of underside
Flip it so same way as looking down on board
Add drawings of parts

- use black marker to show components and connector details

My clone board

Ben Dooks <ben@fluff.org>

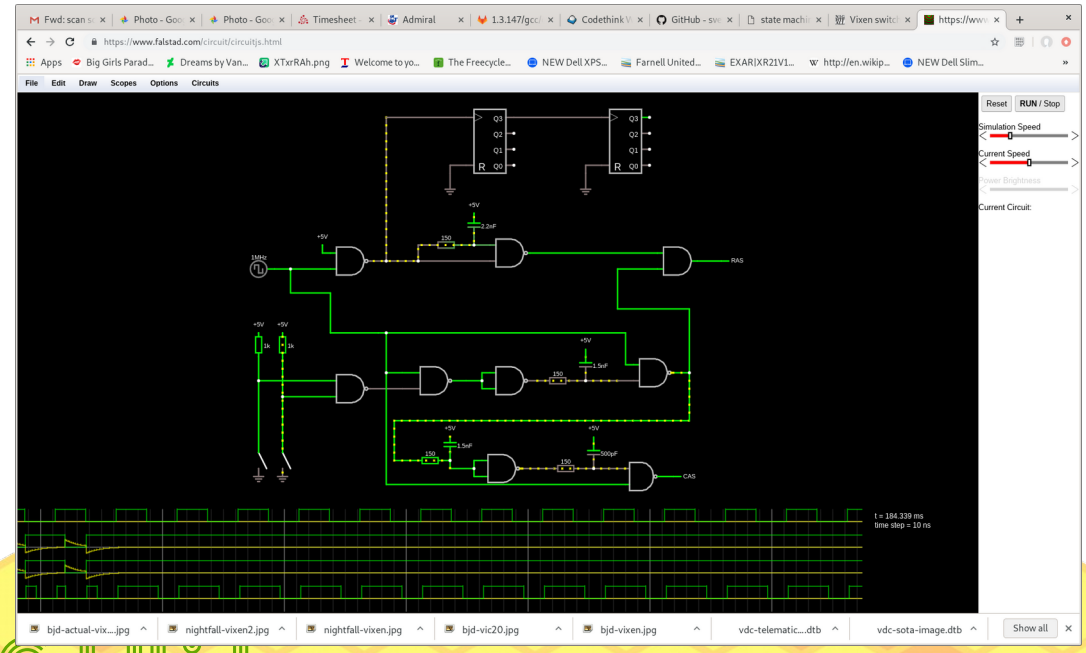
- spend two evenings using eagle
- sent the gerber files to JLC
- approx £30 in components
- £13 for the PCBs

How close was I

- Almost
- Refresh circuit wrong

Ben Dooks <ben@fluff.org>

Simulation



Ben Dooks <ben@fluff.org>

Should have thought of simulation sooner
(Might have saved some money)
This is circuit.js browser based simulation

Future

- Try optimisation of design
  - It is very close to optimal
- Try with PLL and state machine
  - Simple PLL ICs were available
  - Possibly PLL + shift-register based state

Ben Dooks <ben@fluff.org>

The end?

- Project files and documentation will be on:
  - https://github.com/bendooks/vixen
- Thank you. Any questions?

Ben Dooks <ben@fluff.org>

About 10 days of work into the reverse engineering