The Ultimate Acorn Archimedes Talk

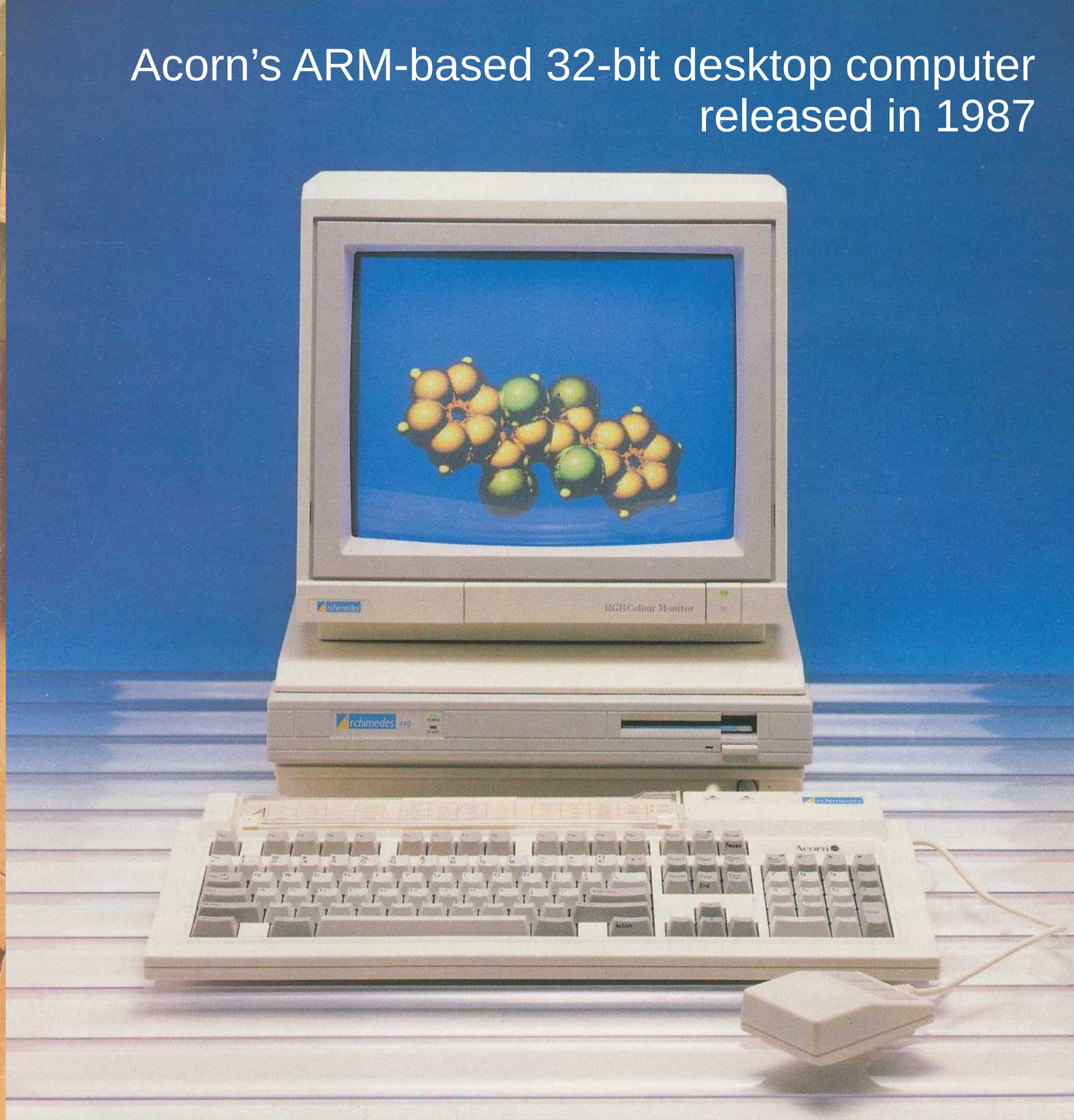
Matt Evans

2019

36C3

What is an Acorn Archimedes (Arc)?

Acorn’s ARM-based 32-bit desktop computer released in 1987




Acorn
Esc
F1
F2
F3
F4
F5
F6
F7
F8
F9
F10
F11
F12
Print
Scroll Lock
Break
Pause
BRITISH BROADCASTING CORPORATION
MICROCOMPUTER SYSTEM
A3000
POWER
DISC
Tab
Q
W
E
R
T
Y
U
I
O
P
Return
Ctrl
A
S
D
F
G
H
J
K
L
Shift
Z
X
C
V
B
N
M
Shift
CapsLock
Alt
Alt
Ctrl
Action
Insert
Home
Page Up
Delete
Copy
End
Page Down
Num Lock
Enter

“Classic” Arcs

1987 – A300 series

- 8MHz ARM2, 0.5-4MB RAM, no HD ~£700

1987 – A400 series

- 1-4MB RAM, option for 20MB HD ~£2300

1989 – (A400/1 series,) A3000

- Now with MEMC1a ~£600

1990 – A540

- 30MHz ARM3, 4-16MB RAM, 100MB SCSI ~£3000

Other 1980s machines...

1983	Lisa	~£6500
1984	Mac 512K	~£2000
1985	Amiga 1000	~£1000
1986	Compaq 386 (16MHz)	~£4000
1987	Amiga 500	£500
1987	Mac II (68020 @16MHz)	~£3500 w/ monitor
1987	Sun 4/110 (SPARC @14.7MHz)	~£7500 w/o monitor or HD...
1988	NeXT Cube (68030 @25MHz)	~£3800
1989	Mac IIci (68030 @25MHz)	~£4800 w/ monitor

Why was it built?

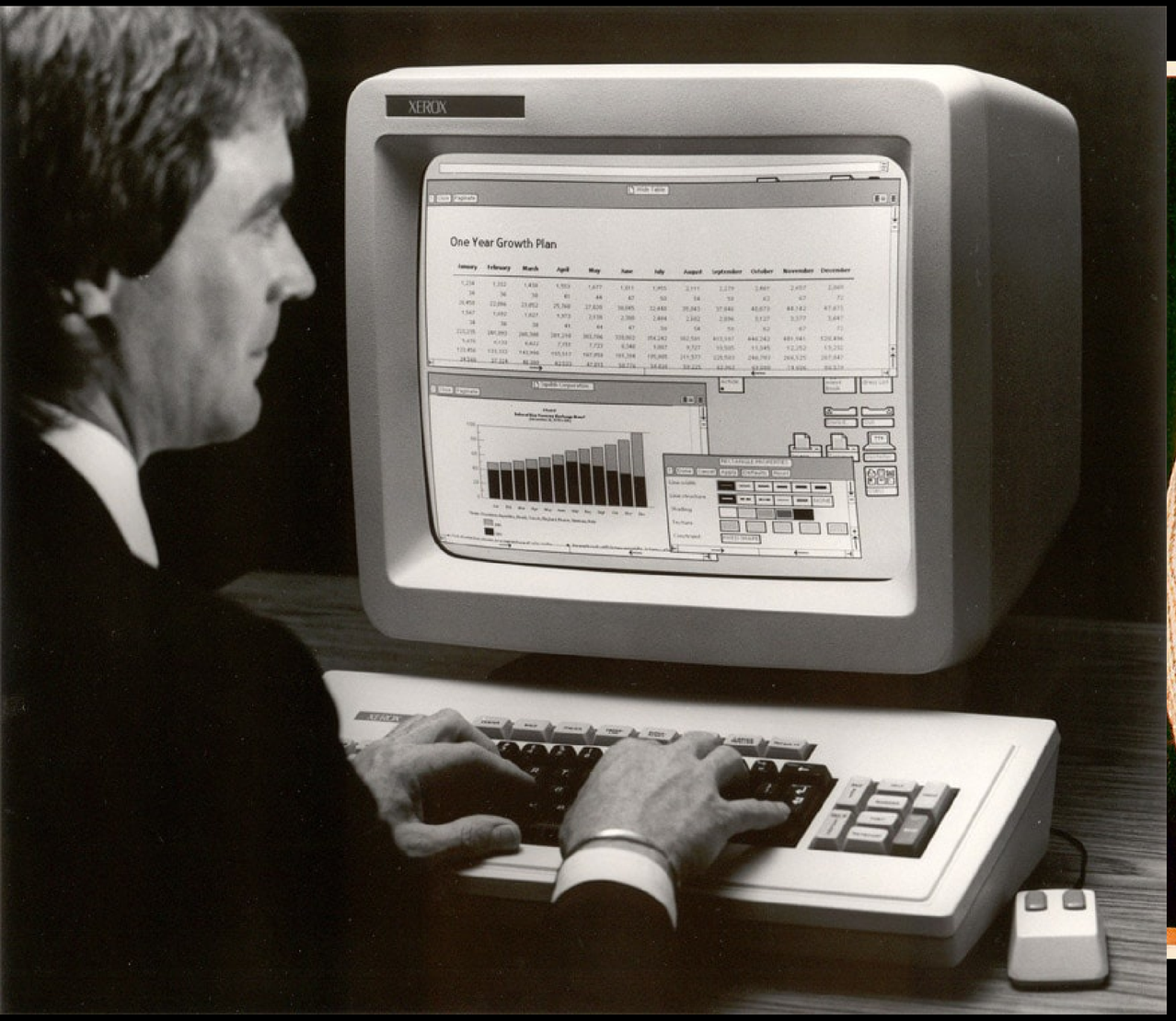
Successor to 8-bit BBC Micro →

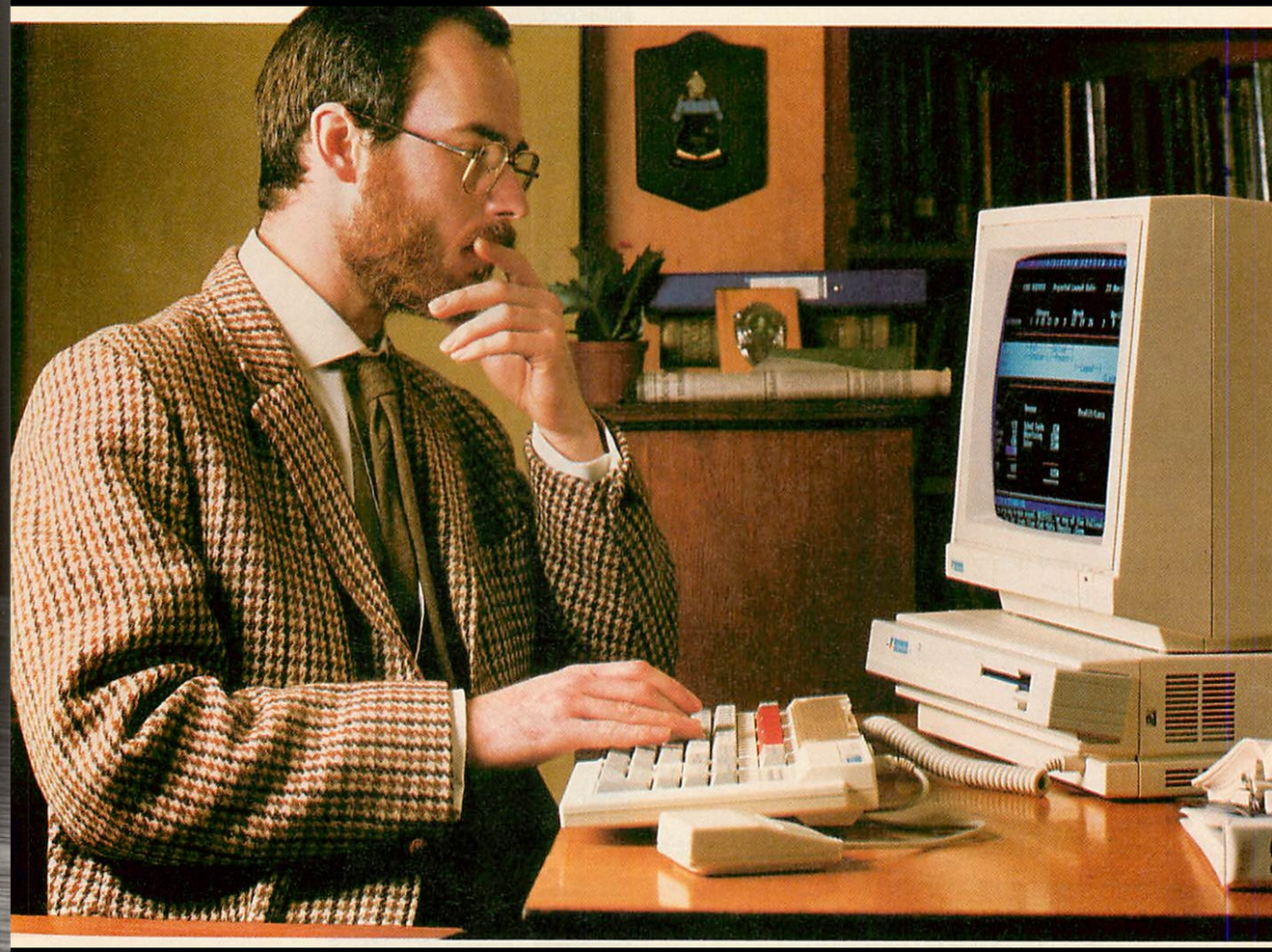
Goal: **10x** the performance at same price

Business/office computer, workstation, education

- Intended to be like a *Xerox Star* (1981, price ~3-series BMW)
- Or *Apple Lisa* (1983, price ~VW Golf)

… but a LOT faster and a lot cheaper





Delightful Acorn Advert, 1987

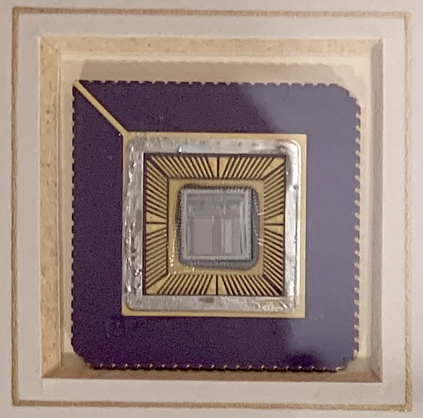
“Project A”, Acorn’s RISC machine

Existing 16-bit CPUs were “a bit crap” – not *much* faster than the existing 8-bit systems!

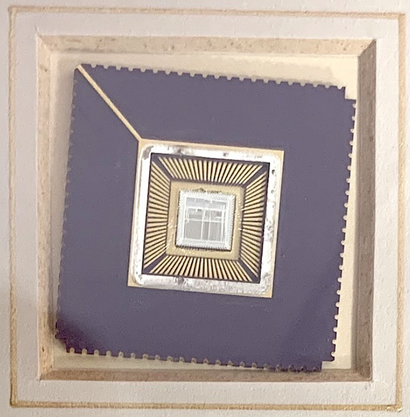
- These CPUs also required an expensive system, DMA controllers

Acorn designed their own custom computer entirely from scratch

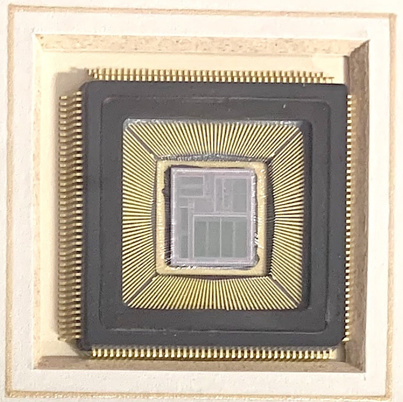
- Small team – about a dozen designing chipset and computer
- Needed to be as simple as possible
- It had to be RISC – CISC chips were difficult even for big companies!
- Simple was their key advantage



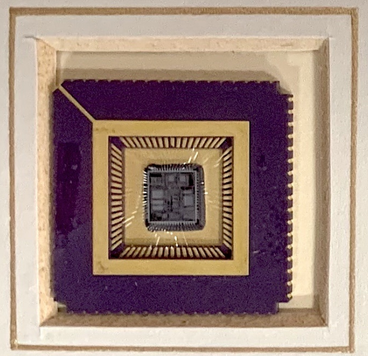
ARM



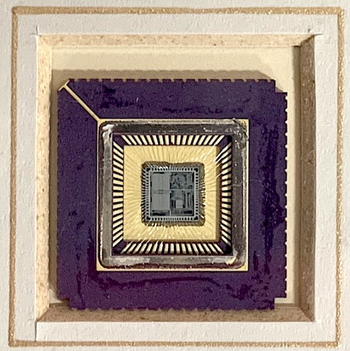
ARM 2



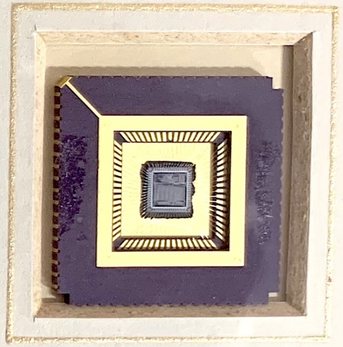
ARM 3



VIDC



MEMC



IOC

ACORN RISC CHIPSET

Arc specifications

Archimedes machines all have the same basic architecture:

- ARM2 8MHz, 1-4MB of RAM, no cache, 26-bit addresses
  - A540 has 30MHz+ ARM3 with 4KB cache, up to 16MB of RAM @12MHz
- MEMC with MMU (1989+ machines had “MEMC1a”, 10% faster)
- 8-channel 8-bit (u-law) stereo audio
- Internal WD1772 DD floppy controller, 3.5” 800K, optional ST506 HD
- PC-ish keyboard, 3-button mouse
- Printer port, serial, 16-bit I/O “podule” slots – A3000 also has 8-bit I/O slot

Arc graphics

All video from a *simple, flat* framebuffer:

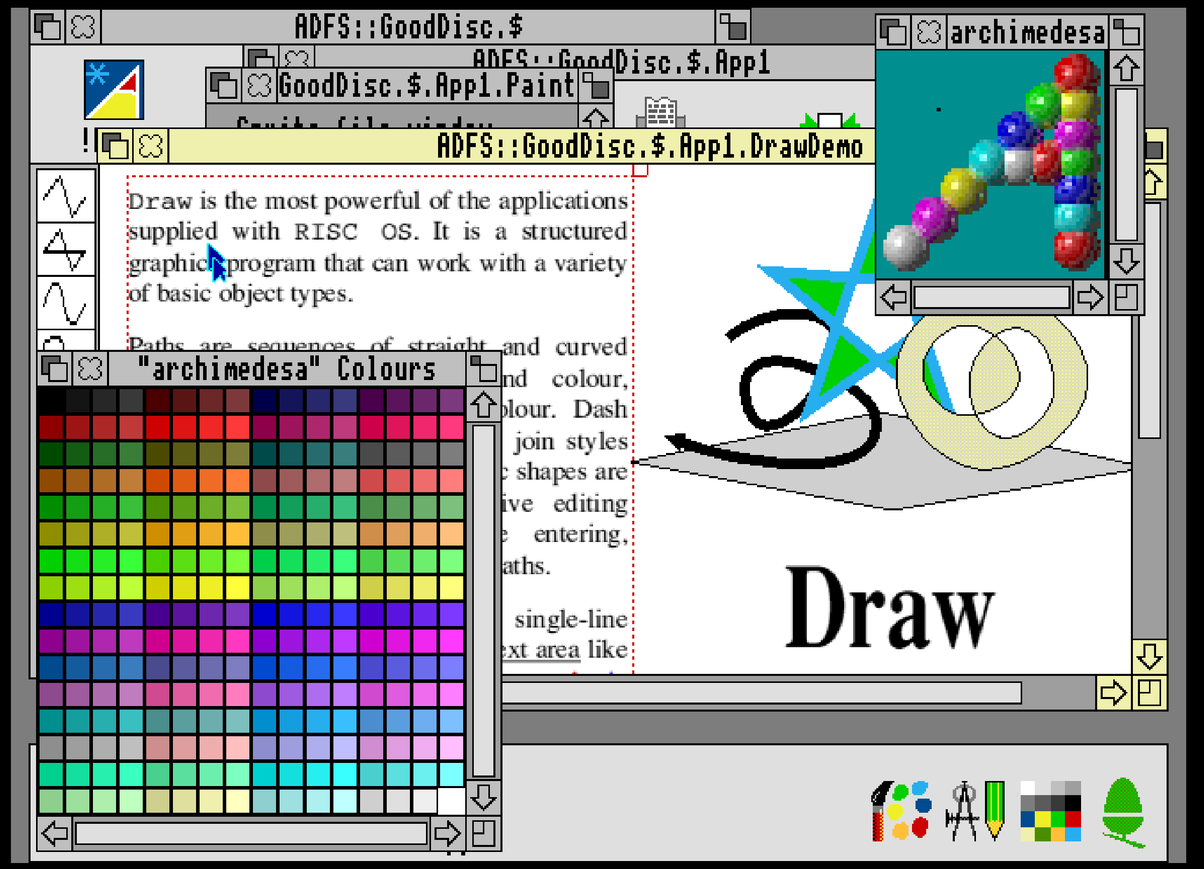
- No bitplanes or weird addressing
- No blitter or sprites :(
- (OK, 1 sprite for mouse cursor)

Resolution up to *640x512*

A540 supports *800x600*

Up to *256 colours*, or 2/4/16 from a palette of 4096

A400 and A540 can do ***1152x900x1bpp*** !



It’s all about the (cost)performance

ARM2 at 8MHz is about 5000 Dhrystones* 1.1, which is:

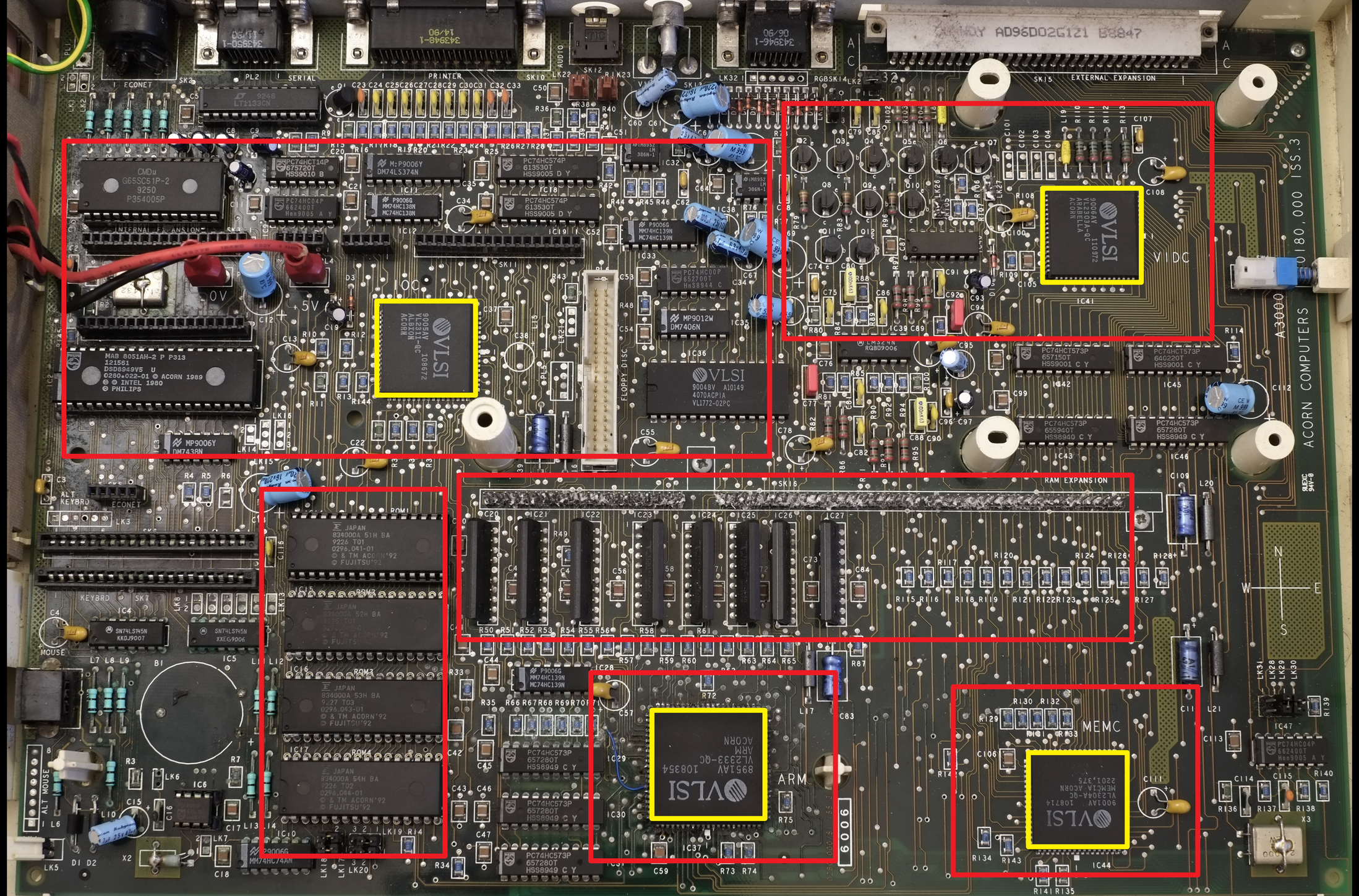
- >7x faster than an Amiga 1000 or Macintosh (68000 @ 7.xxMHz)
- ~50% faster than a Sun3/160 or Macintosh II (68020 @ 16.6MHz)
- 2x faster than a 386 @ 16MHz
- About the speed of a VAX11/784
- Half the speed of a MIPS R2000 (which is *quite expensive*)
- About ¼ the speed of a Sun 4/110 (which is >>4x the cost)

Really quick for the cost!

A540 w/ ARM3 30-36MHz is about 24000-26000 (4.8x ARM2)

* Yes! Dhrystones for benchmarking! Be grateful I’m not quoting “BASIC performance” like a real 1980s kid.


HALO
FDI2-1016
HIPOTTED
MX
MX98902AQC
i-cubed ltd
Serial no:00:29:82
CSO509N
CTC
9516
FLASH ROM
BEDEN 761
CITIZEN
MADE IN TAIWAN R.O.C.
Esc
F1
F2
F3
F4
F5
F6
F7
F8
F9
F10
F11
F12
Print
Scroll Lock
Break
Pause
Insert
Home
Page Up
Delete
Copy
End
Page Down
Num Lock
Tab
Ctrl
Return
Shift
CapsLock
Alt
Action
Enter


ECONET
SERIAL
PRINTER
AUDIO
EXTERNAL EXPANSION
INTERNAL EXPANSION
FLOPPY DISC
RAM EXPANSION
KEYBRD
ALT KEYBRD
MOUSE
ALT MOUSE
ROM1
ROM3
ROM4
VIDC
MEMC
ARM
ACORN COMPUTERS
A3000
ISS.3
CMOS
G65SC51P-2
9250
P354005P
MAB 8051AH-2 P P313
12151
DSD8949V5 U
0280-022-01 © ACORN 1989
© © INTEL 1980
© PHILIPS
9006AV VL2211-QC ACORN
9004BV A10149
4070ACPIA
VL1772-02PC
9006AV VL2333-QC ARM ACORN
9006AV VL2364-QC MEMCIA ACORN
9006AV A10972 ACORN
LT1133CN
MP9012W
DM7406N
MP9006Y
DM743RN
SN74LS145N
SN74LS145N
JAPAN
834000A 51H BA
834000A 52H BA
834000A 53H BA
834000A 54H BA
© & TM ACORN'92
© FUJITSU'92

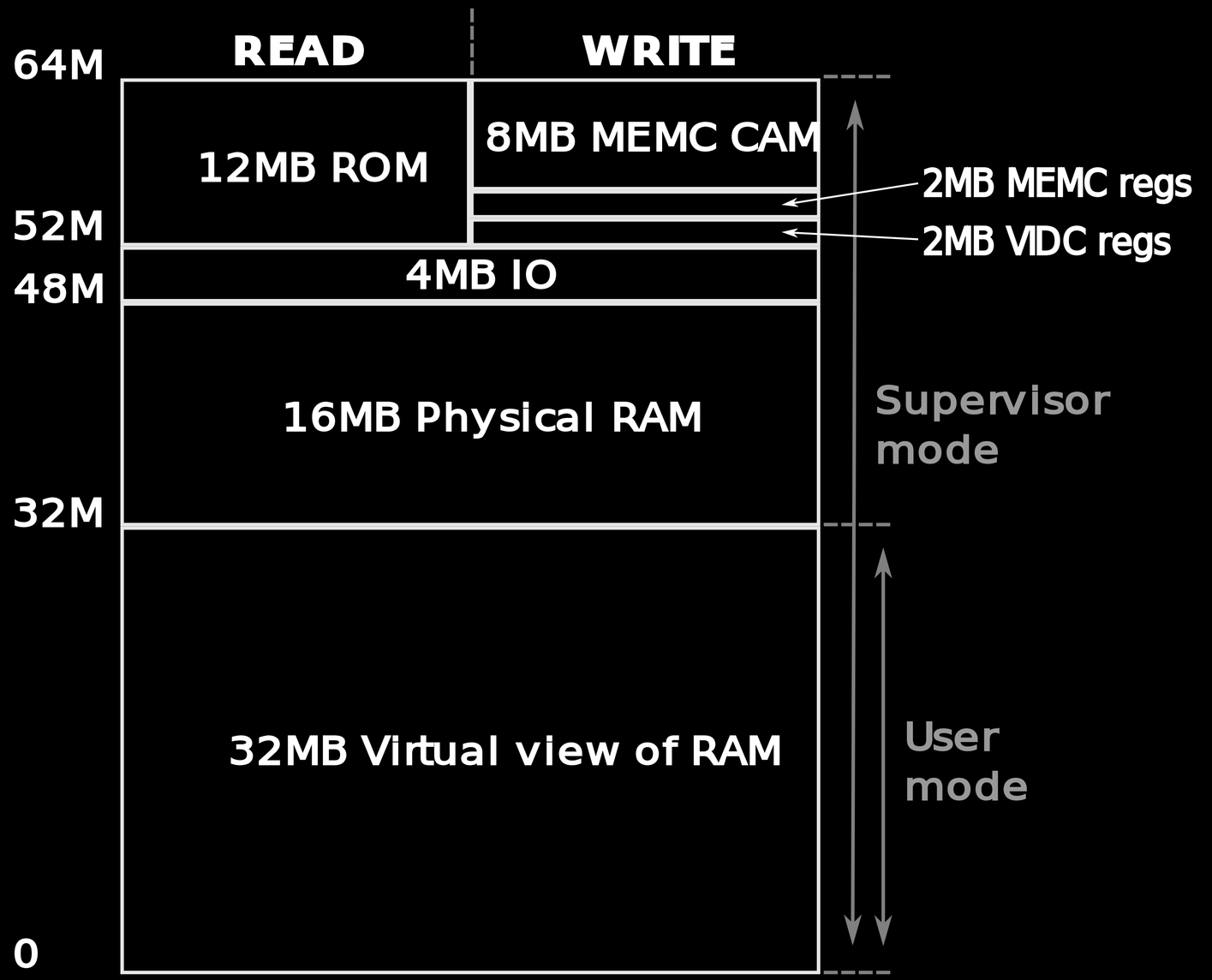
64MB address space

Top/bottom for OS/userspace

MMU translates User pages

Inherently limited to 16MB RAM

I/O is R/W, but MEMC/VIDC regs are banked *write-only* behind ROM



ARM1, 1985

3-stage pipelined 32-bit RISC, 6MHz

32-bit instructions, 3-operand, load-store architecture

All instructions conditional

32-bit regs R0-R14, R15 is PC

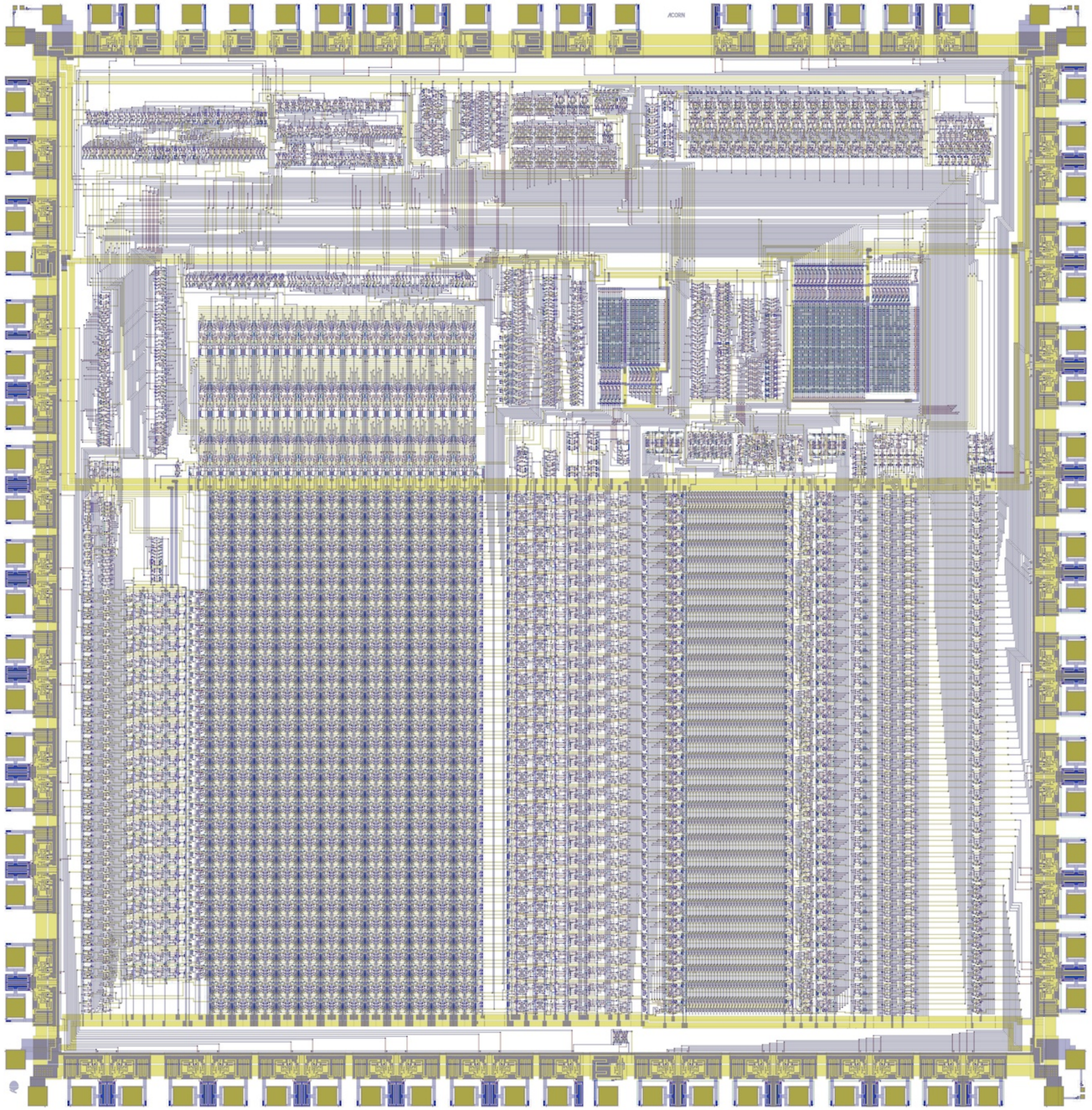
Shifted-operand

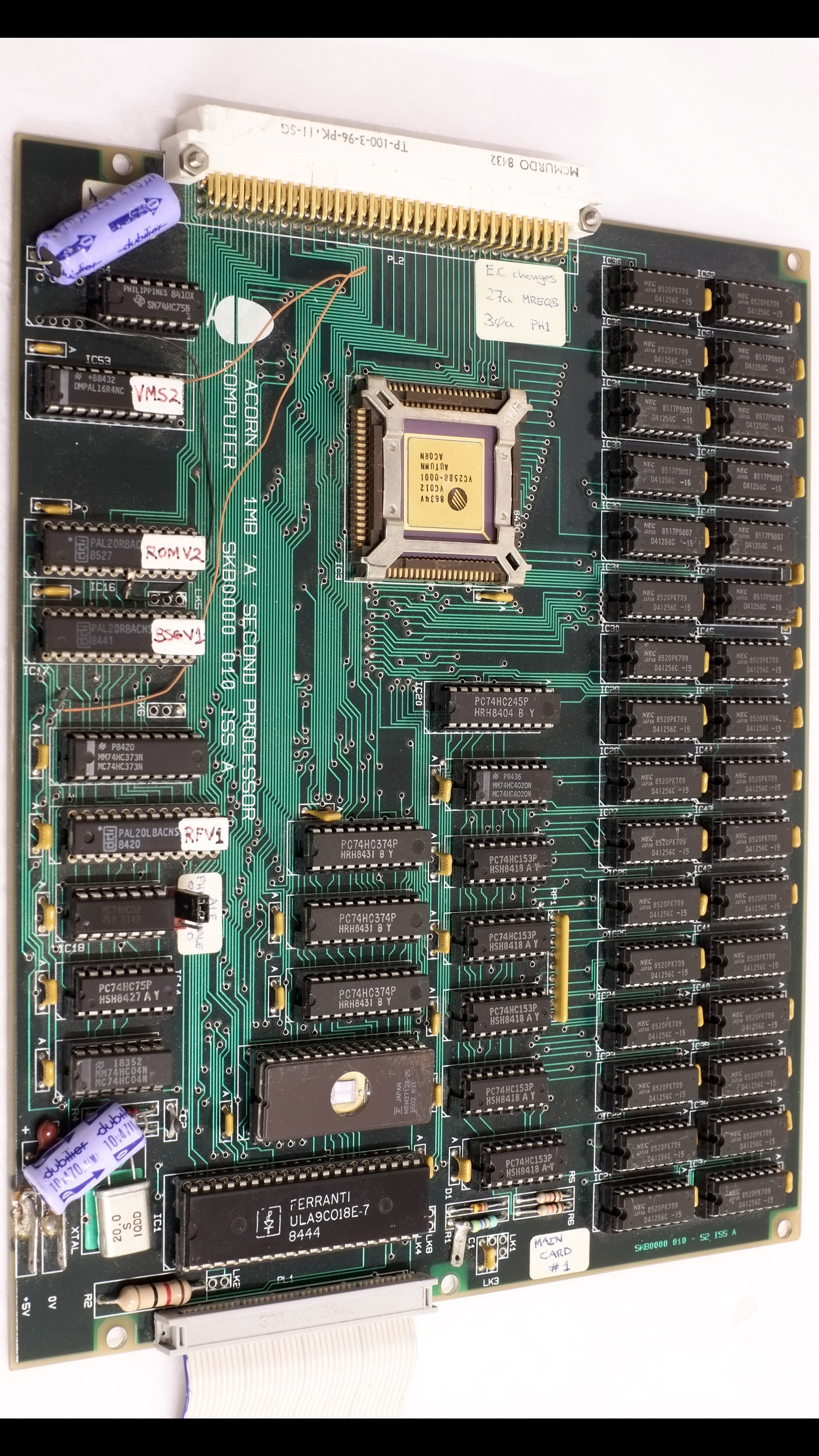
Load/store multiple

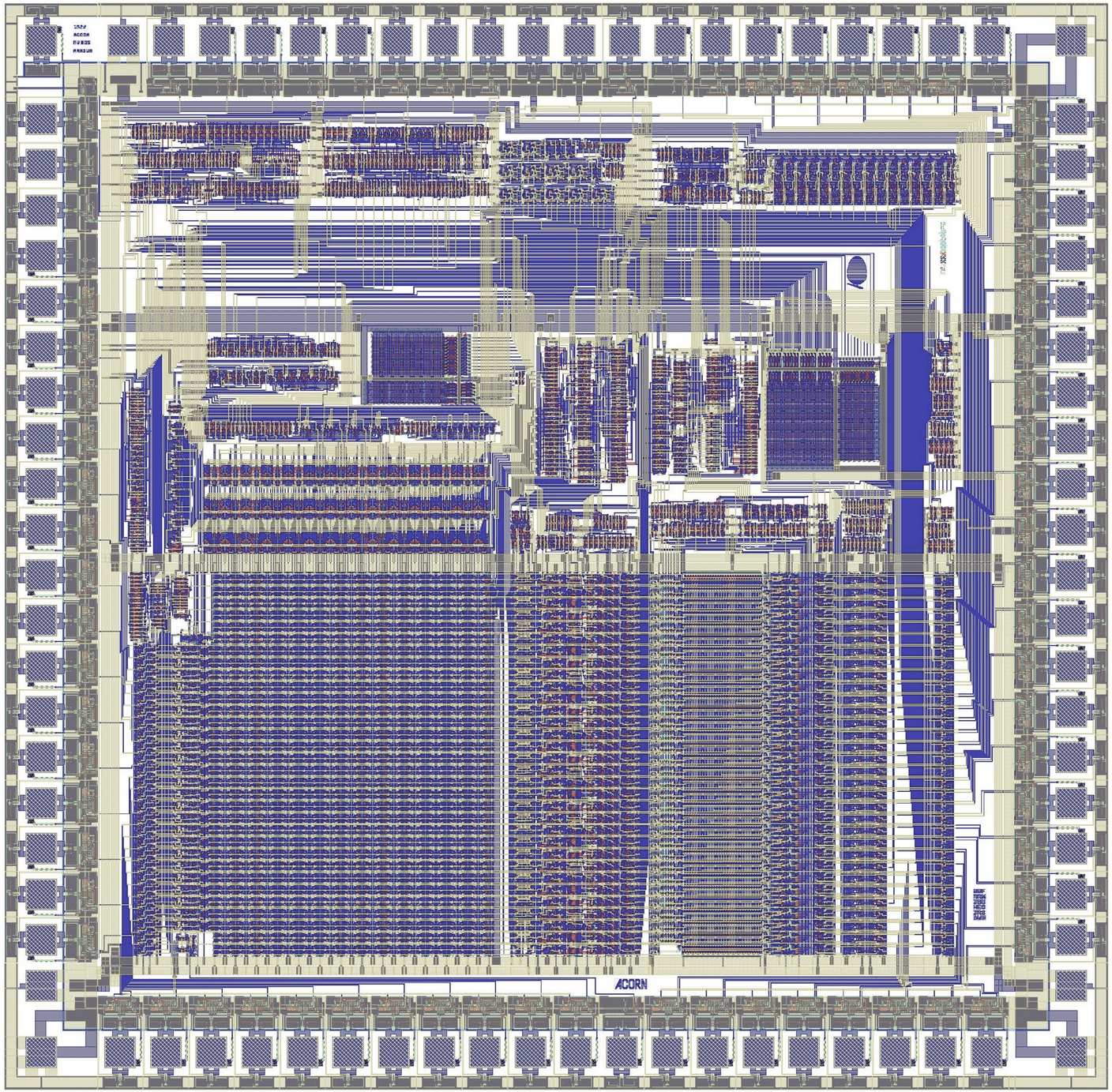
26-bit addresses

50mm^2, 3um 2LM

CMOS 24,000 transistors




ACORN
COMPUTER
IMB 'A' SECOND PROCESSOR
SKB0000 010 ISS A
E.C changes
27a MREQB
3Cpa PH1
VMS2
ROM V2
3S6V1
RFV1
MAIN CARD #1
SKB0000 010 - S2 ISS A
PHILIPPINES 8410X
SN74HC75N
+B8432
DMPAL16R4NC
PAL20R8A
8527
PAL20R8ACNS
8441
PAL20L8ACNS
8420
P8420
MM74HC373N
MC74HC373N
P8436
MM74HC4020N
MC74HC4020N
PC74HC245P
HRH8404 B Y
PC74HC374P
HRH8431 B Y
PC74HC374P
HRH8431 B Y
PC74HC374P
HRH8431 B Y
PC74HC153P
HSH8418 A Y
PC74HC153P
HSH8418 A Y
PC74HC153P
HSH8418 A Y
PC74HC153P
HSH8418 A Y
PC74HC75P
HSH8427 A Y
I8352
MM74HC04N
MC74HC04N
FERRANTI
ULA9C018E-7
8444
20.0
S
IQDD
dubilier
10/470
dubilier
10/47
NEC
JAPAN
8520PK709
D41256C -15
8517P5007
D41256C -15
IC1
IC2
IC16
IC17
IC18
IC20
IC29
IC30
IC31
IC32
IC33
IC34
IC35
IC36
IC37
IC38
IC39
IC40
IC41
IC42
IC43
IC44
IC45
IC46
IC47
IC48
IC49
IC50
IC51
IC52
PL1
PL2
XTAL
LK1
LK2
LK3
LK4
LK5
LK6
LK7
LK8
R1
R2
R5
R6
C1
C2
D1
RP1
+5V
0V



ARM2, 1986

Shrink of ARM1, including Booth multiplier, mode-banked registers

Late 1986, VL2333 8MHz

Early 1987, VL86C010 10/12MHz

30mm², 2um 2LM

CMOS 27,000 transistors

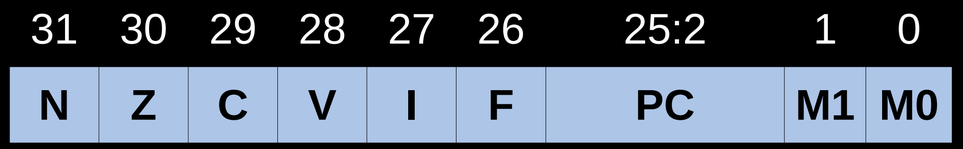
arm26 vs arm32

ARM2 has R8-R14 banked in Fast Interrupt reQuest (FIQ) mode

- Designed for *minimal latency* for FIQ

But *unlike* arm32, R15 contains PC, flags, CPU mode and interrupt masks:

- 24 bits left for (32b-aligned) PC



State all in one register, less to push/pop on exception

Return from interrupt simply using:

- SUBS PC, R14, #4

MEMC1, 1986

“Anna”

Address translation & protection

Clock generation

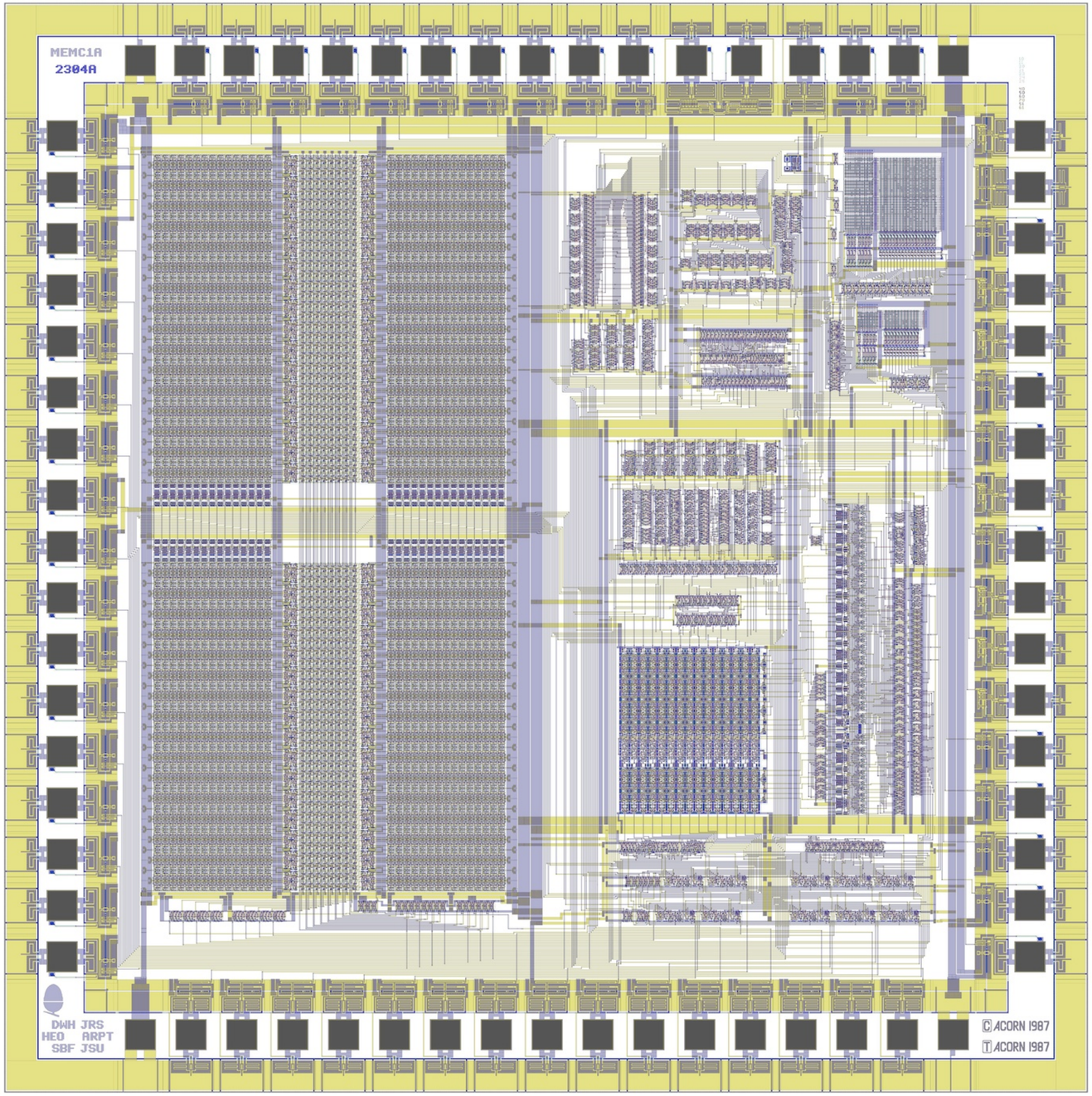
DRAM address generation & refresh

DMA address generation

MEMC1a revision in 1987

31mm², 2um 2LM

CMOS 24,000 transistors



ARM + MEMC

Making best use of DRAM bandwidth is **prime factor for performance**:

- DRAM is 32 bits wide
- ARM is pipelined (1 instruction per clock)
- ARM indicates Sequential cycles for *page-mode* DRAM access
- Balanced design!

N-cycles are non-sequential, 250ns

S-cycles are sequential, 125ns

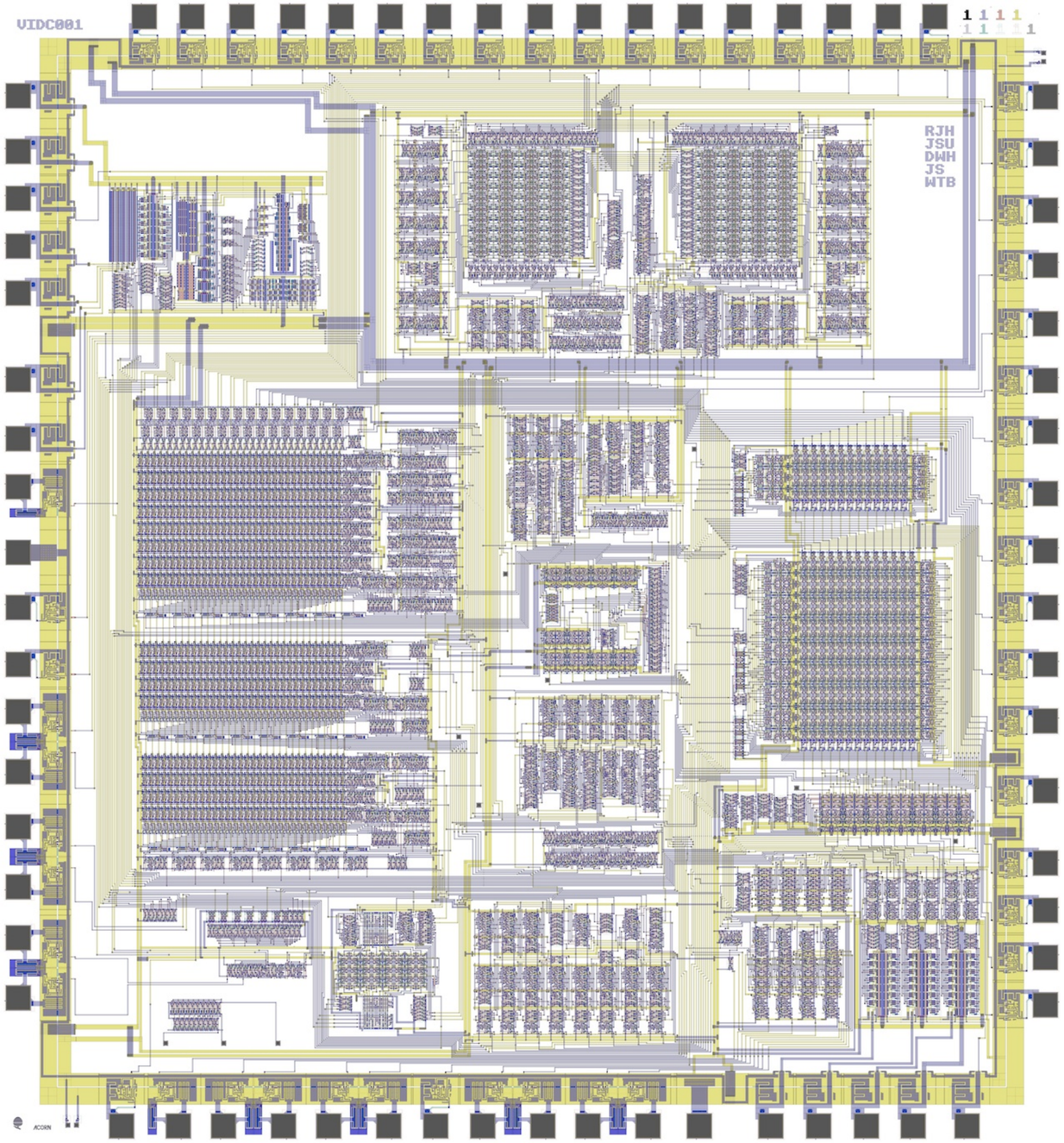
Example: STM store multiple instructions take 2N+(1+n)S for n registers

- Store 14 registers (56 bytes) in 2.375us (~25MB/s memzero)

MEMC MMU

MEMC contains a software-loaded CAM that matches a Virtual Address to one of 128 physical pages

- There are always 128 physical pages per MEMC
  - But page size changes!
- Limits memory
- There can be ONLY ONE Virtual mapping of a physical page
- Only translates RAM addresses (no IO remapping)



VIDC1, 1985

“Arabella”

RGB video DACs, video FIFO, frame timing

16-entry palette to 12-bit
RGB+transparency, 256 colour mode

Audio FIFO & 8-bit log stereo DAC, 8ch mix

24MHz pixel clock (~VGA)

VIDC1a revision in 1986

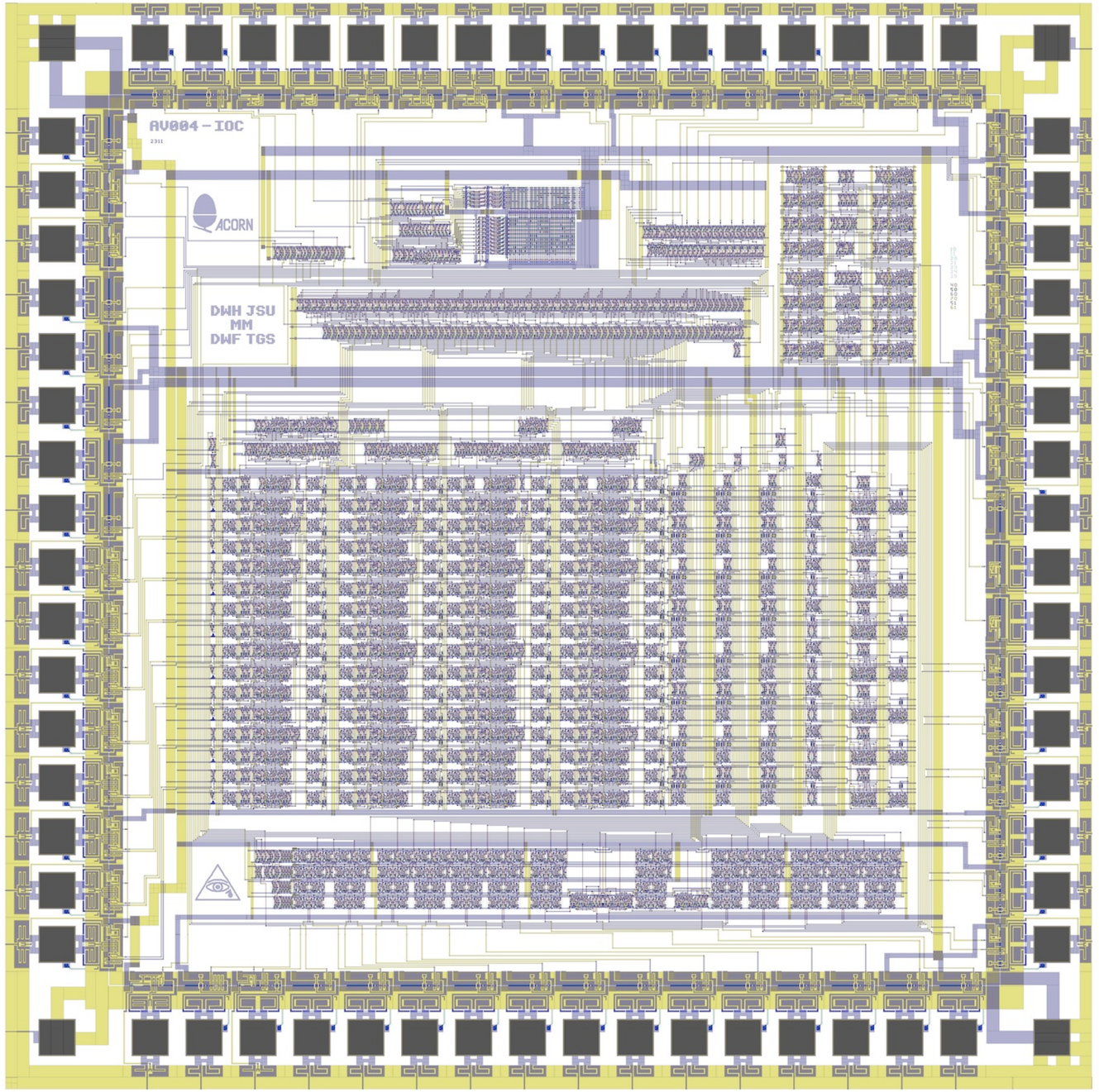
31mm^2, 2.4um 2LM
CMOS 18,000 transistors

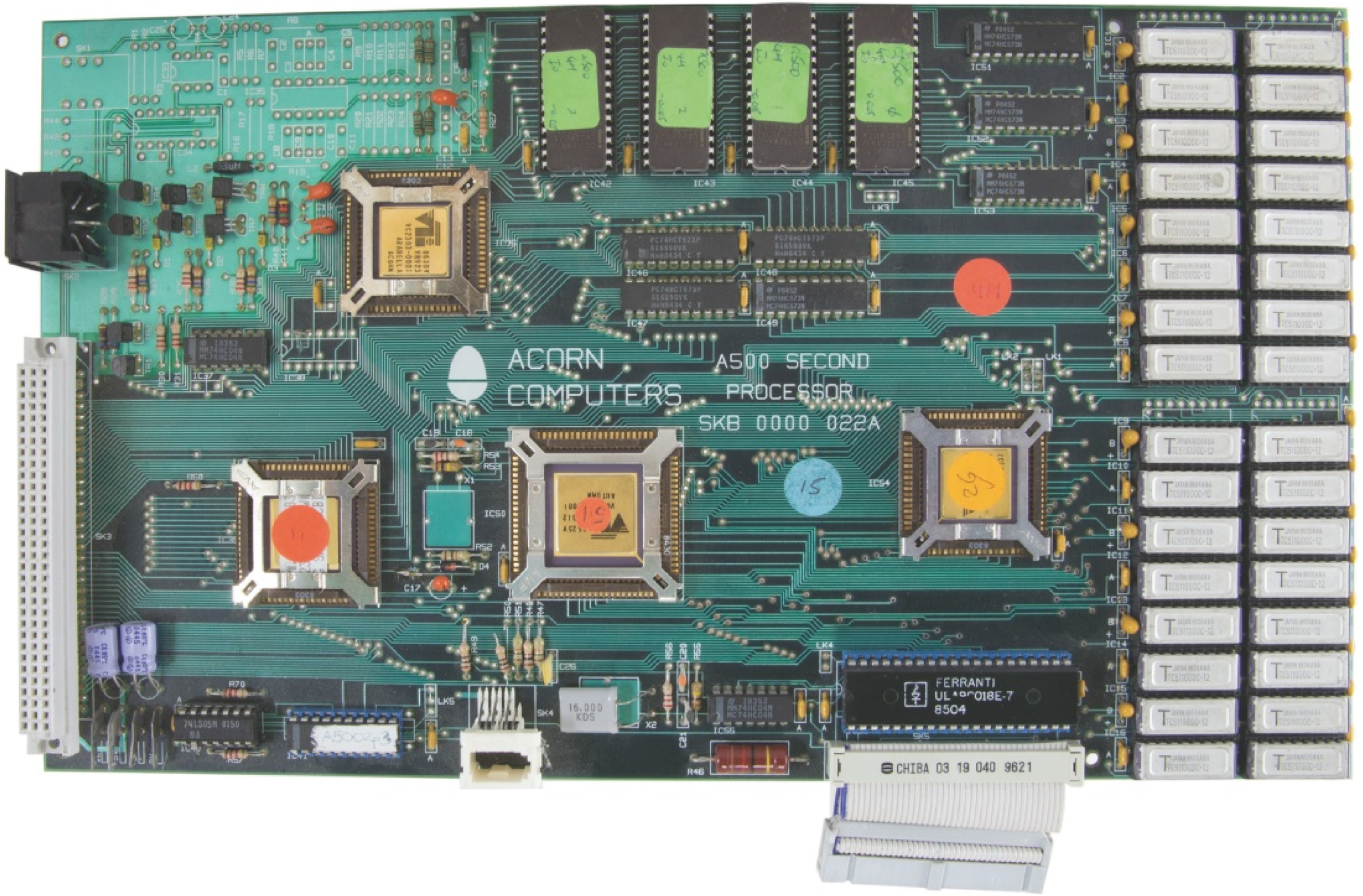
IOC, 1986

“Albion”

IO cycle timing
KART
Interrupt masks + status
4x 2MHz timers/counters
GPIO/I2C
Illuminati Eye of Providence

21mm2 2um 2LM
CMOS 11K transistors



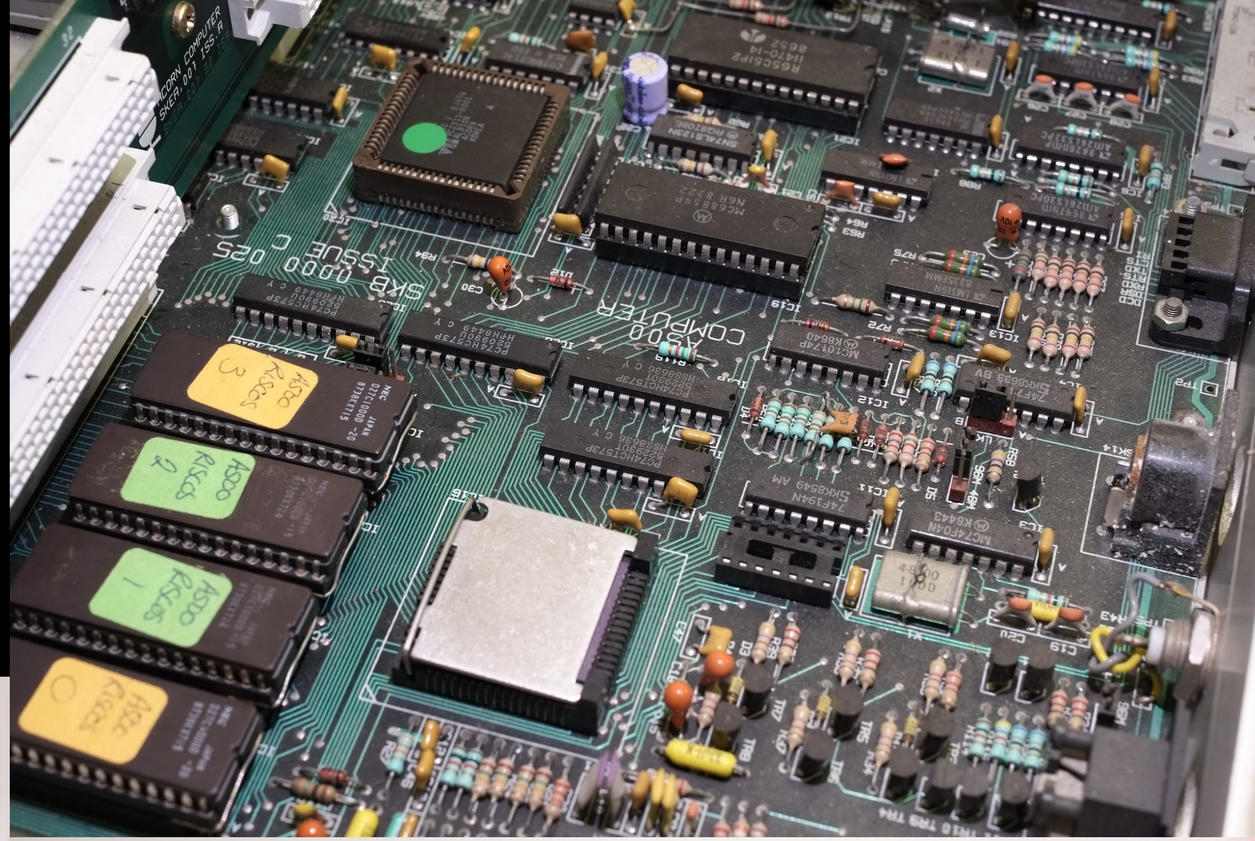

ACORN
COMPUTERS
A500 SECOND
PROCESSOR
SKB 0000 022A
FERRANTI
8504
CHIBA 03 19 040 9621
16.000
KDS

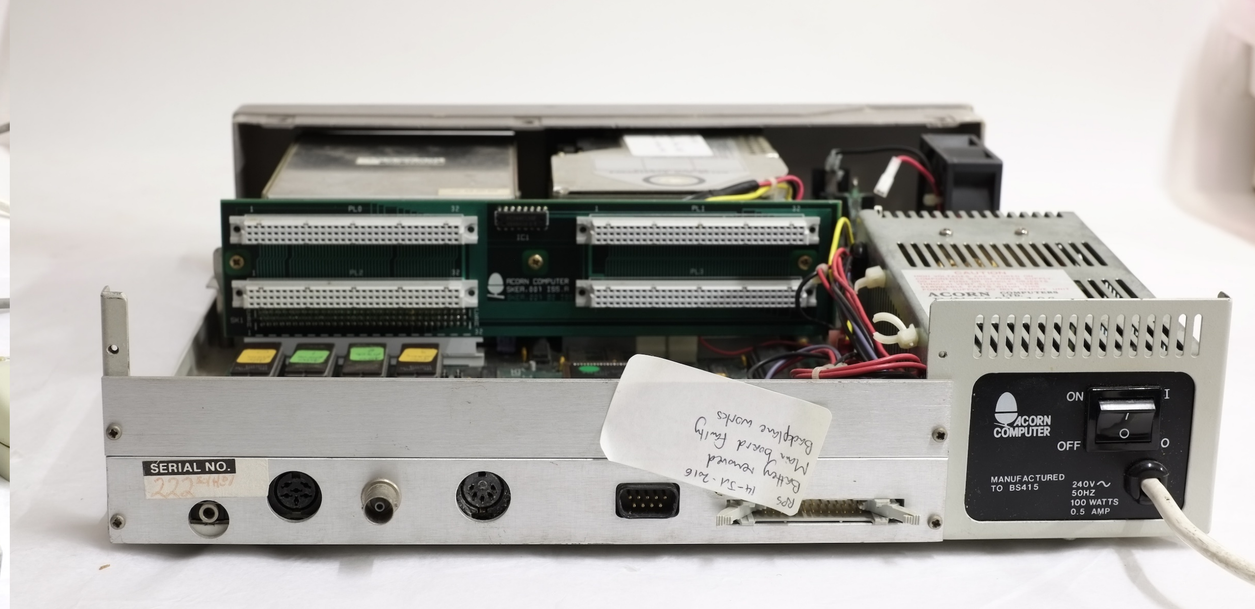
A500, 1986

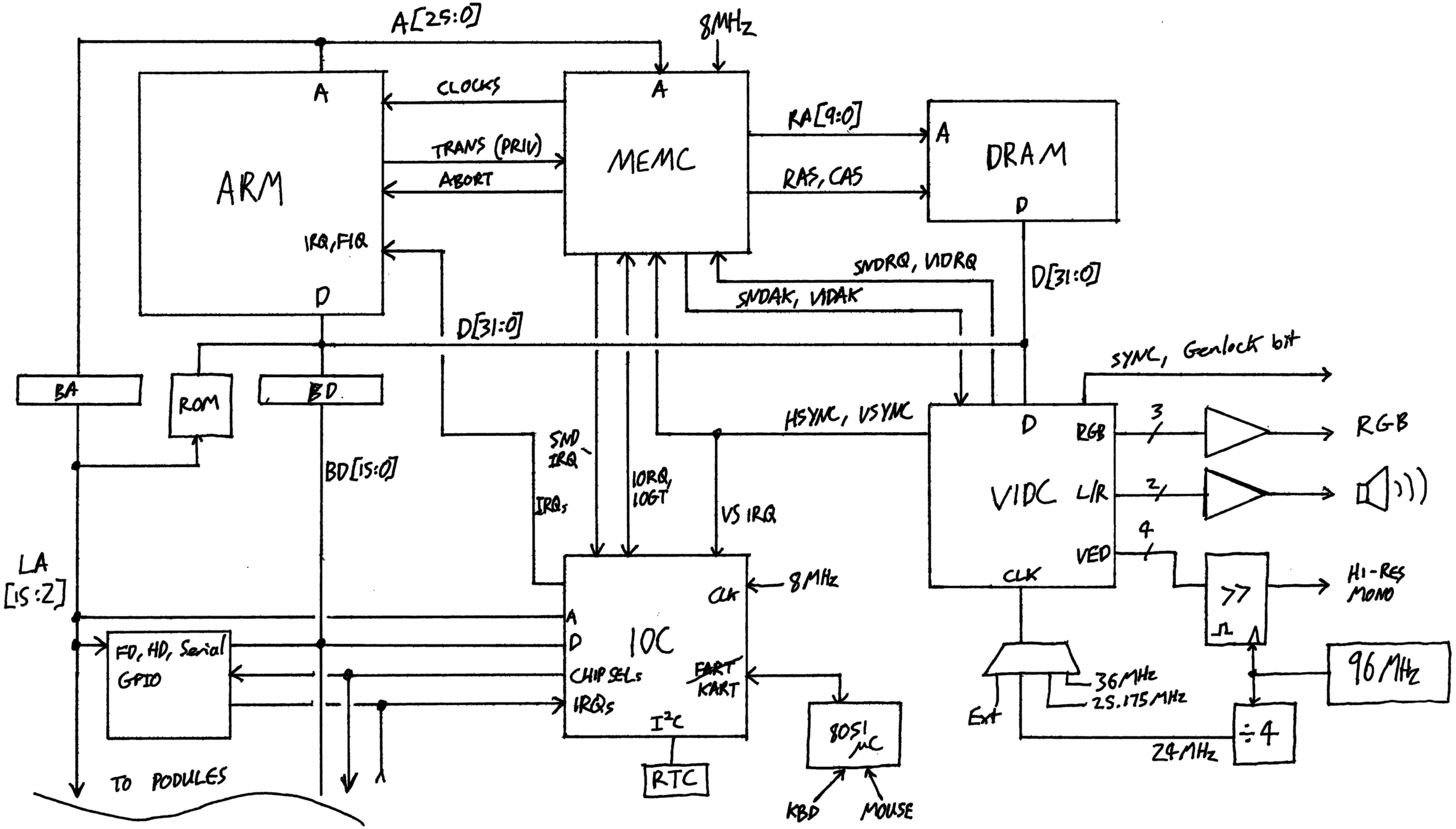
Prototype Archimedes

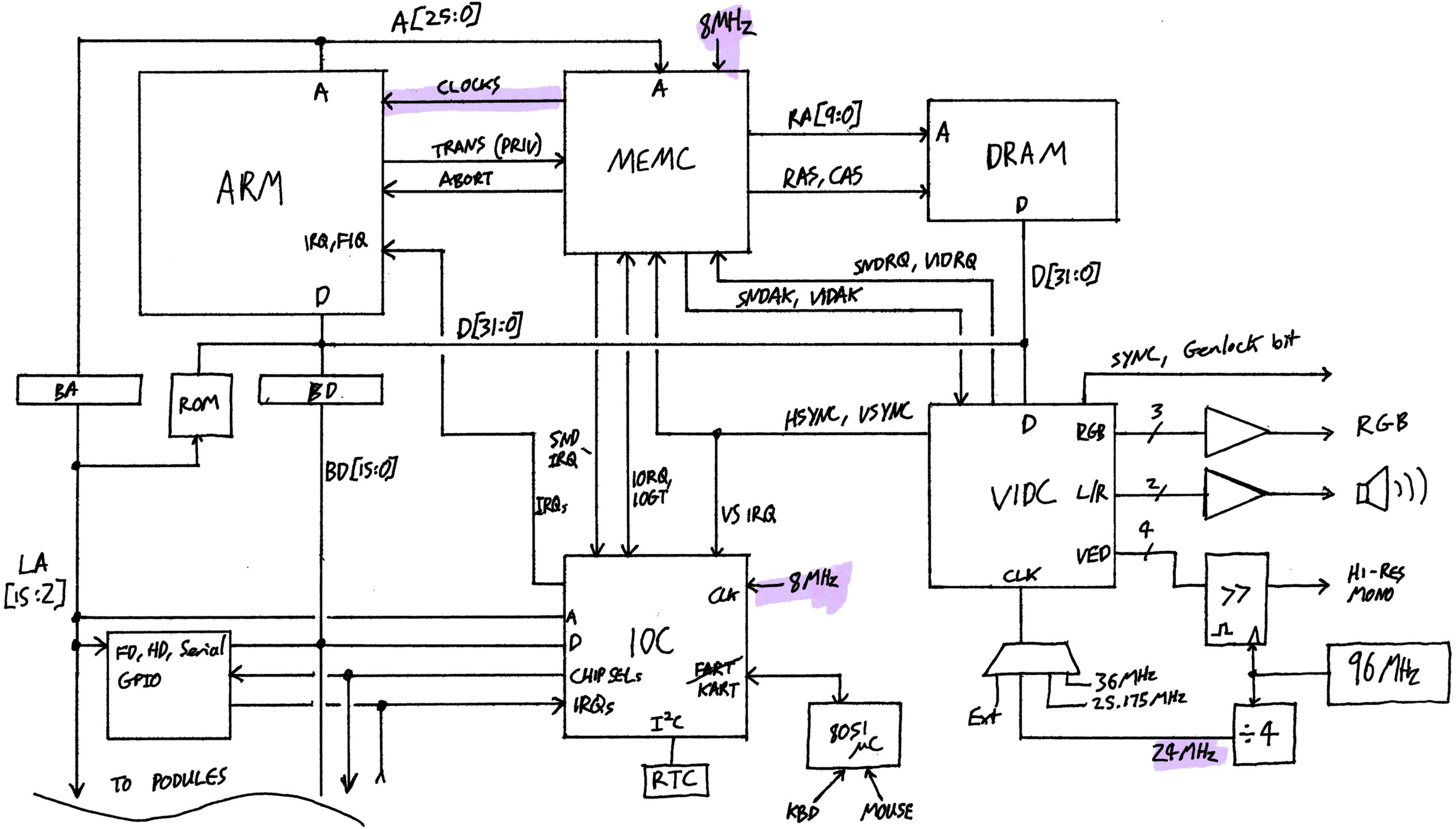
ARM2, 4MB, ST506 HD

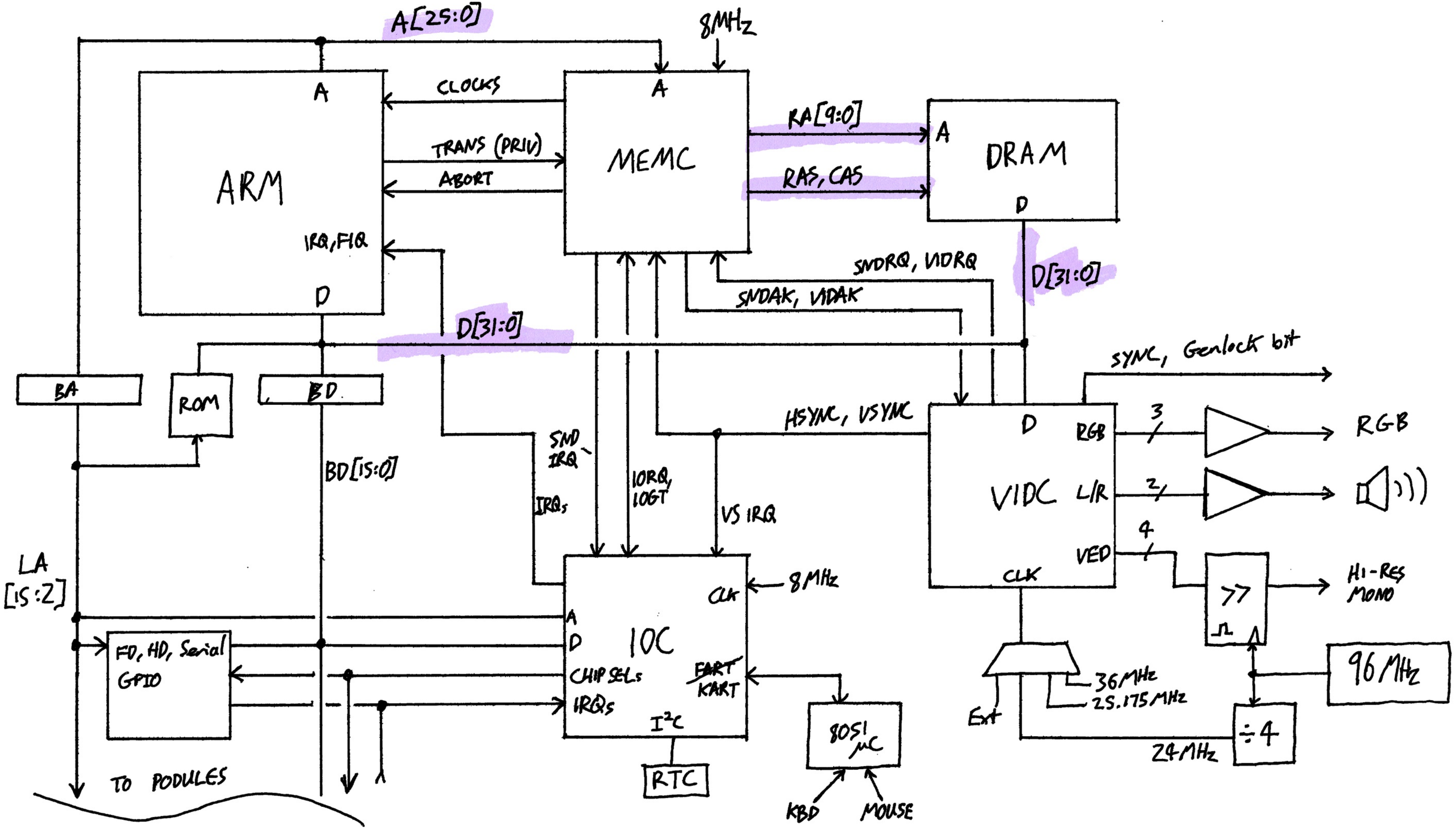
OS development

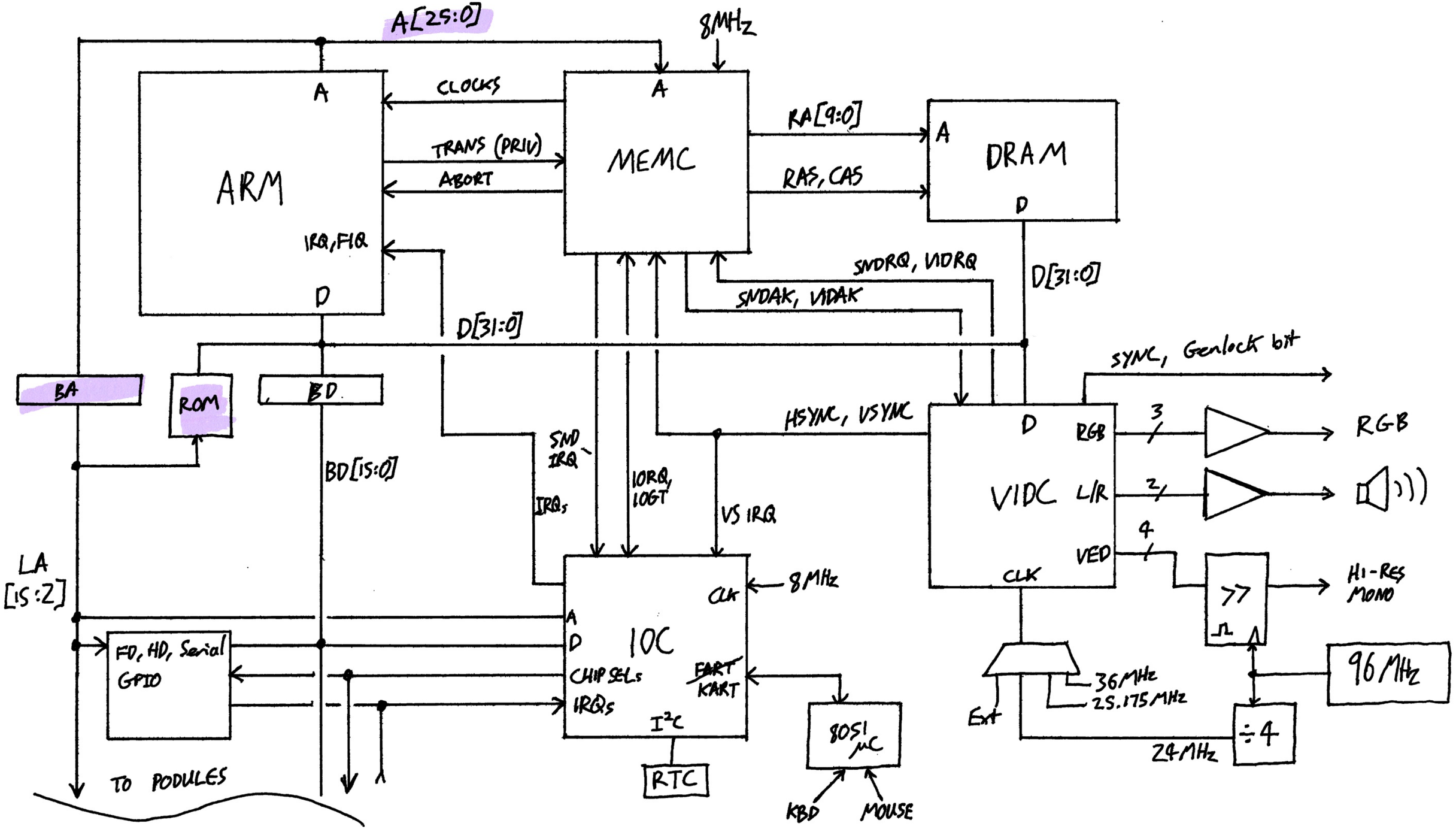


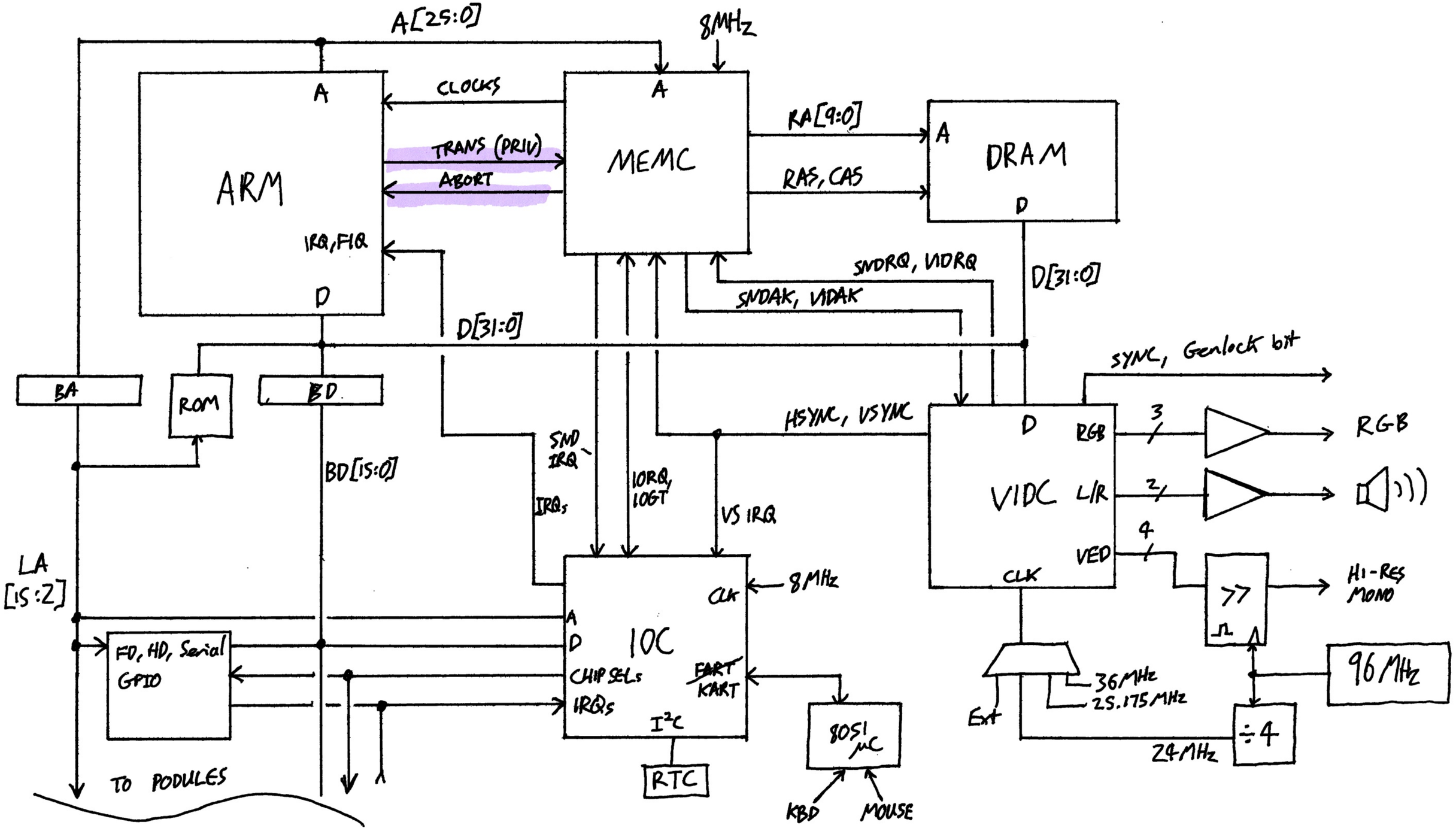


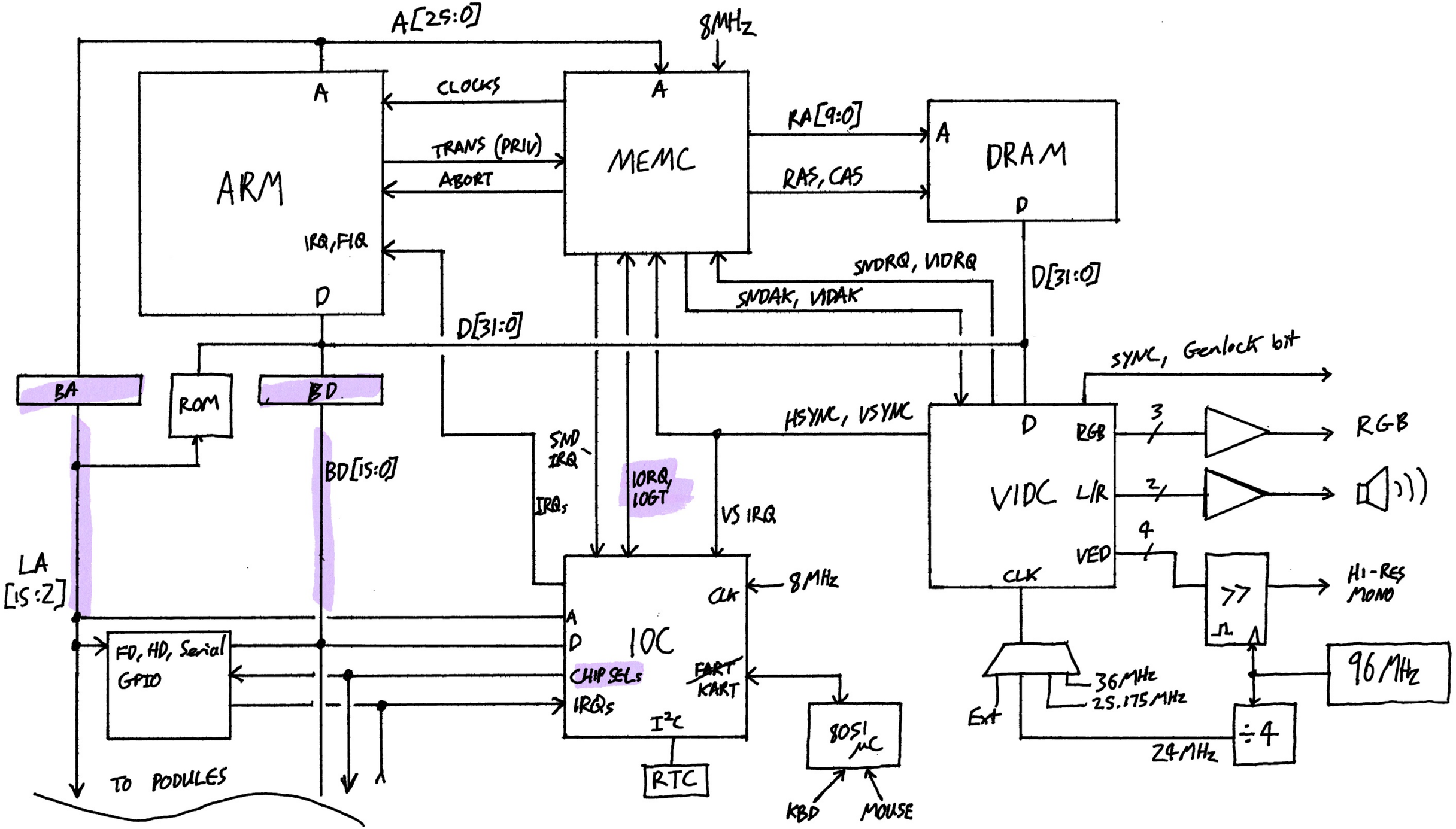

A[25:0]
8MHz
A
ARM
IRQ,FIQ
D
CLOCKS
TRANS (PRIV)
ABORT
A
MEMC
RA[9:0]
RAS, CAS
A
DRAM
D
D[31:0]
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
BA
ROM
BD
BD[15:0]
SND IRQ
IRQs
IORQ, LOGT
VS IRQ
HSYNC, VSYNC
SYNC, Genlock bit
D
RGB
VIDC
L/R
VID
CLK
3
2
4
RGB
77
HI-RES MONO
LA [15:2]
A
D
IOC
CHIP SELs
IRQs
I²C
CLK
8MHz
FART
KART
FD, HD, Serial
GPIO
TO PODULES
RTC
8051 µC
KBD
MOUSE
Ext
36MHz
25.175MHz
24MHz
÷4
96MHz

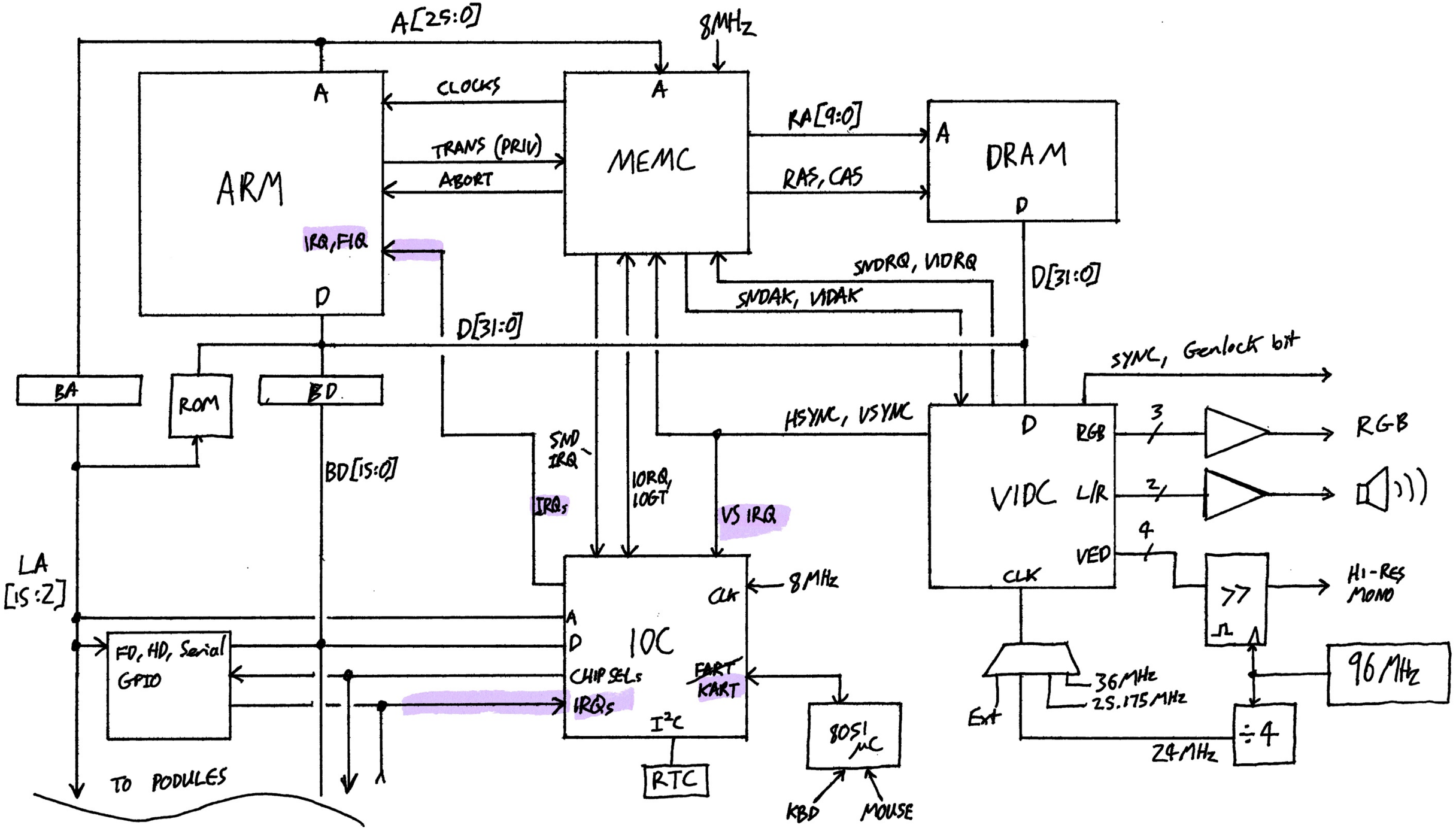

A[25:0]
8MHz
A
CLOCKS
A
ARM
TRANS (PRIV)
ABORT
MEMC
RA[9:0]
RAS, CAS
A
DRAM
D
IRQ, FIQ
D
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
D[31:0]
BA
ROM
BD
SYNC, Genlock bit
HSYNC, VSYNC
D
RGB
3
RGB
SND IRQ
BD[15:0]
VIDC
L/R
2
IORQ, LOGT
IRQs
VS IRQ
4
VID
LA [15:2]
CLK
77
HI-RES MONO
8MHz
CLK
A
D
IOC
FD, HD, Serial GPIO
CHIP SELs
IRQs
FART KART
I²C
36MHz
25.175MHz
Ext
96MHz
÷4
24MHz
8051 µC
RTC
KBD
MOUSE
TO PODULES

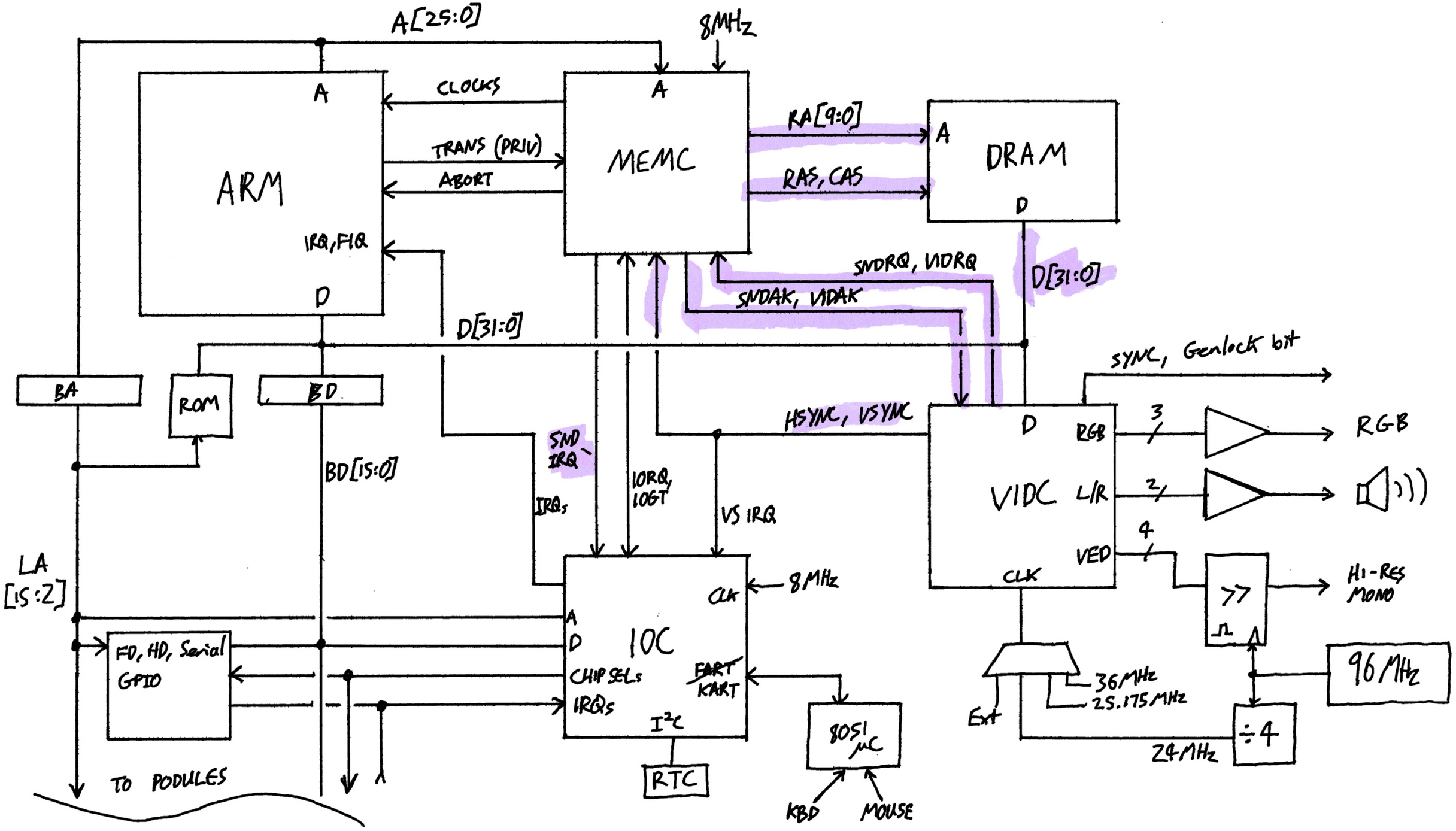

A[25:0]
8MHz
A
ARM
CLOCKS
TRANS (PRIV)
ABORT
IRQ,FIQ
D
A
MEMC
RA[9:0]
RAS, CAS
A
DRAM
D
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
D[31:0]
BA
ROM
BD
BD[15:0]
SND IRQ
IRQs
IORQ, LOGT
VS IRQ
HSYNC, VSYNC
SYNC, Genlock bit
D
RGB
VIDC
L/R
VED
CLK
3
2
4
RGB
77
HI-RES MONO
LA [15:2]
A
D
IOC
CHIP SELs
IRQs
I²C
CLK
8MHz
FART
KART
FD, HD, Serial
GPIO
TO PODULES
RTC
8051 µC
KBD
MOUSE
Ext
36MHz
25.175MHz
24MHz
÷4
96MHz

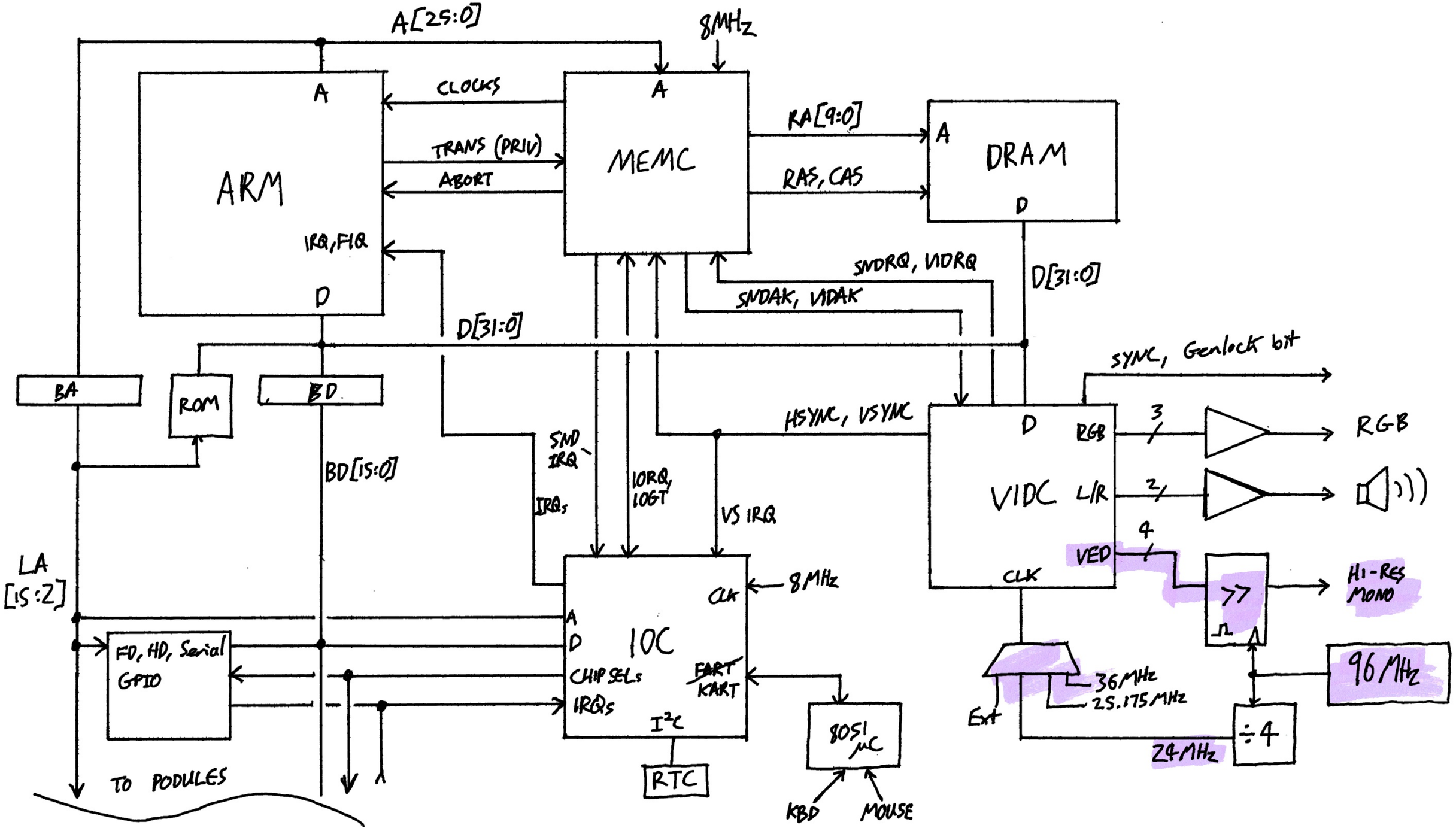

A[25:0]
8MHz
A
ARM
CLOCKS
TRANS (PRIV)
ABORT
IRQ,FIQ
D
A
MEMC
RA[9:0]
RAS, CAS
A
DRAM
D
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
D[31:0]
BA
ROM
BD
BD[15:0]
SND IRQ
IRQs
IORQ, LOGT
VS IRQ
HSYNC, VSYNC
SYNC, Genlock bit
D
RGB
VIDC
L/R
VID
CLK
3
2
4
RGB
HI-RES MONO
77
LA [15:2]
A
D
IOC
CHIP SELs
IRQs
I²C
CLK
8MHz
FART
KART
FD, HD, Serial
GPIO
TO PODULES
RTC
8051 µC
KBD
MOUSE
Ext
36MHz
25.175MHz
24MHz
÷4
96MHz


A[25:0]
8MHz
A
ARM
CLOCKS
TRANS (PRIV)
ABORT
IRQ,FIQ
D
A
MEMC
RA[9:0]
RAS, CAS
A
DRAM
D
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
D[31:0]
BA
ROM
BD
BD[15:0]
SND IRQ
IRQs
IORQ, LOGT
VS IRQ
HSYNC, VSYNC
SYNC, Genlock bit
D
RGB
VIDC
L/R
VED
CLK
3
2
4
RGB
77
HI-RES MONO
LA [15:2]
A
D
IOC
CHIP SELs
IRQs
I²C
CLK
8MHz
FART
KART
FD, HD, Serial
GPIO
TO PODULES
RTC
8051 µC
KBD
MOUSE
36MHz
25.175MHz
Ext
24MHz
÷4
96MHz


A[25:0]
8MHz
ARM
A
D
IRQ,FIQ
CLOCKS
TRANS (PRIV)
ABORT
MEMC
A
RA[9:0]
RAS, CAS
DRAM
A
D
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
D[31:0]
BA
ROM
BD
BD[15:0]
SND IRQ
IRQs
IORQ, LOGT
VS IRQ
HSYNC, VSYNC
SYNC, Genlock bit
D
RGB
VIDC
L/R
VED
CLK
3
2
4
RGB
Hi-Res MONO
77
LA [15:2]
A
D
IOC
CLK
8MHz
CHIP SELs
IRQs
I²C
FART
KART
FD, HD, Serial
GPIO
TO PODULES
RTC
8051 µC
KBD
MOUSE
Ext
36MHz
25.175MHz
24MHz
÷4
96MHz


A[25:0]
8MHz
A
ARM
D
CLOCKS
TRANS (PRIV)
ABORT
IRQ,FIQ
A
MEMC
RA[9:0]
RAS, CAS
A
DRAM
D
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
D[31:0]
BA
ROM
BD
BD[15:0]
SND IRQ
IRQs
IORQ, IOGT
VS IRQ
HSYNC, VSYNC
SYNC, Genlock bit
D
RGB
VIDC
L/R
VID
CLK
3
2
4
RGB
HI-RES MONO
LA [15:2]
A
D
IOC
CHIP SELs
IRQs
I²C
CLK
8MHz
FART
KART
RTC
8051 µC
KBD
MOUSE
FD, HD, Serial
GPIO
TO PODULES
36MHz
25.175MHz
Ext
24MHz
÷4
96MHz


A[25:0]
8MHz
A
ARM
IRQ,FIQ
D
CLOCKS
TRANS (PRIV)
ABORT
A
MEMC
RA[9:0]
RAS, CAS
A
DRAM
D
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
D[31:0]
BA
ROM
BD
BD[15:0]
SND IRQ
IRQs
IORQ, IOGT
VS IRQ
HSYNC, VSYNC
SYNC, Genlock bit
D
RGB
VIDC
L/R
VED
CLK
3
2
4
RGB
Hi-Res Mono
77
LA [15:2]
A
D
IOC
CLK
8MHz
CHIP SELs
IRQs
I²C
FART
KART
FD, HD, Serial
GPIO
TO PODULES
RTC
8051 µC
KBD
MOUSE
Ext
36MHz
25.175MHz
24MHz
÷4
96MHz

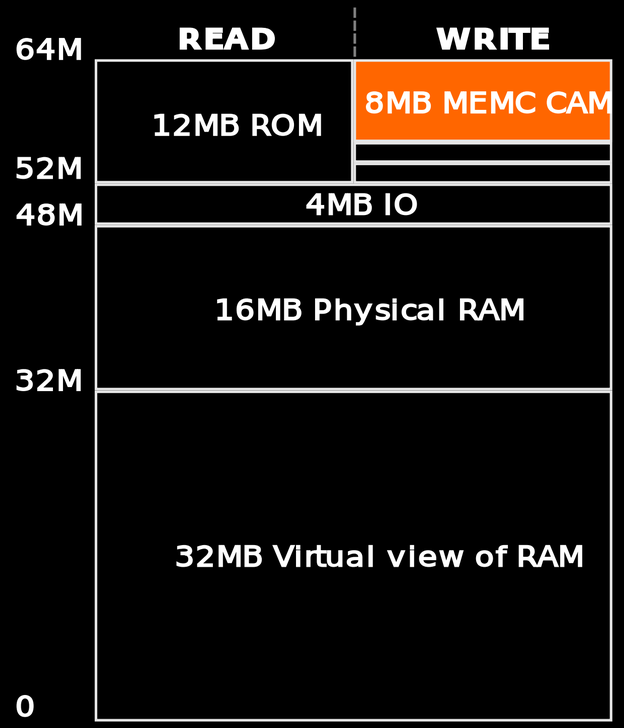

A[25:0]
8MHz
A
ARM
IRQ,FIQ
D
CLOCKS
TRANS (PRIV)
ABORT
A
MEMC
RA[9:0]
RAS, CAS
A
DRAM
D
D[31:0]
SNDRQ, VIDRQ
SNDAK, VIDAK
D[31:0]
BA
ROM
BD
BD[15:0]
SND IRQ
IRQs
IORQ, IOGT
VS IRQ
HSYNC, VSYNC
SYNC, Genlock bit
D
RGB
VIDC
L/R
VED
CLK
3
2
4
RGB
LA [15:2]
A
D
IOC
CLK
8MHz
CHIP SELs
IRQs
FART
KART
I²C
RTC
FD, HD, Serial
GPIO
TO PODULES
8051 µC
KBD
MOUSE
÷2
Hi-Res MONO
96MHz
36MHz
25.175MHz
Ext
÷4
24MHz

MEMC isn’t on the data bus

But it has registers that need programming!

Oh my… Writes to 10MB of address space affect MEMC…

E.g. to program MEMC Address Translator, write any value to an address 0x38XXXXX-0x3fXXXXX:



25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	LPN (Logical page 0-1023)											LPN		Perms			PPN (Physical page 0-127)						

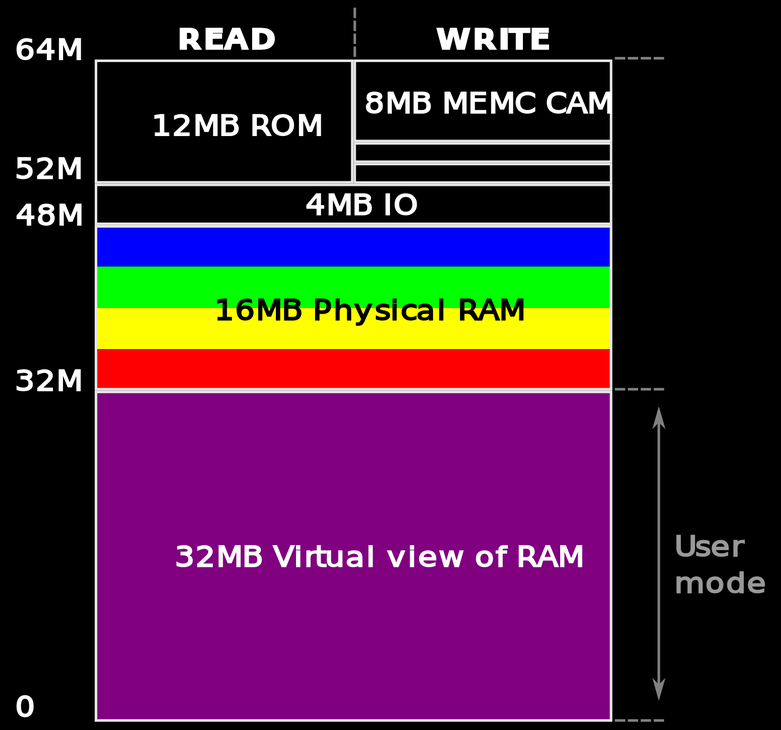
A540: >1 MEMC, ARM3 gets cache

Four MEMCs in parallel – all MMUs translate, one matches the address

Address translation happens outside ARM3, *after cache access*

4KB 64-way (!) unified I+D cache, 16B lines

→ Changing Virtual-to-Physical mappings requires a cache invalidate



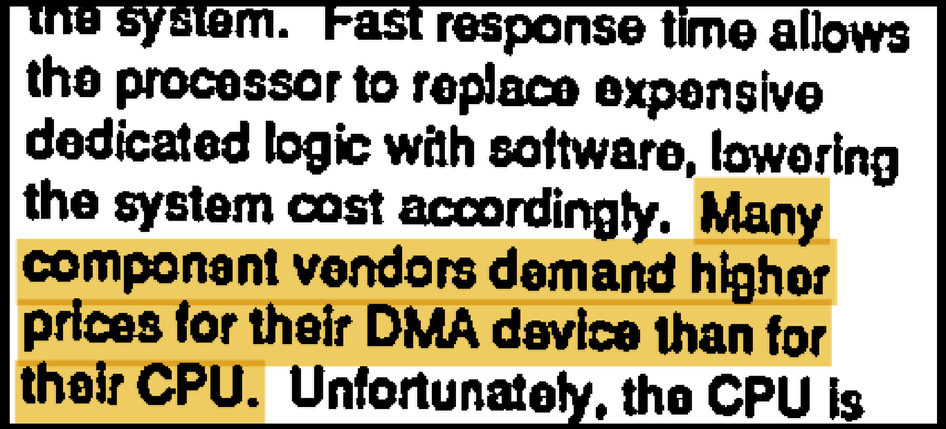
Programmed I/O

IO done using PIO/CPU

- Floppy disc, hard disc, serial, printer
- External cards (“podules”)

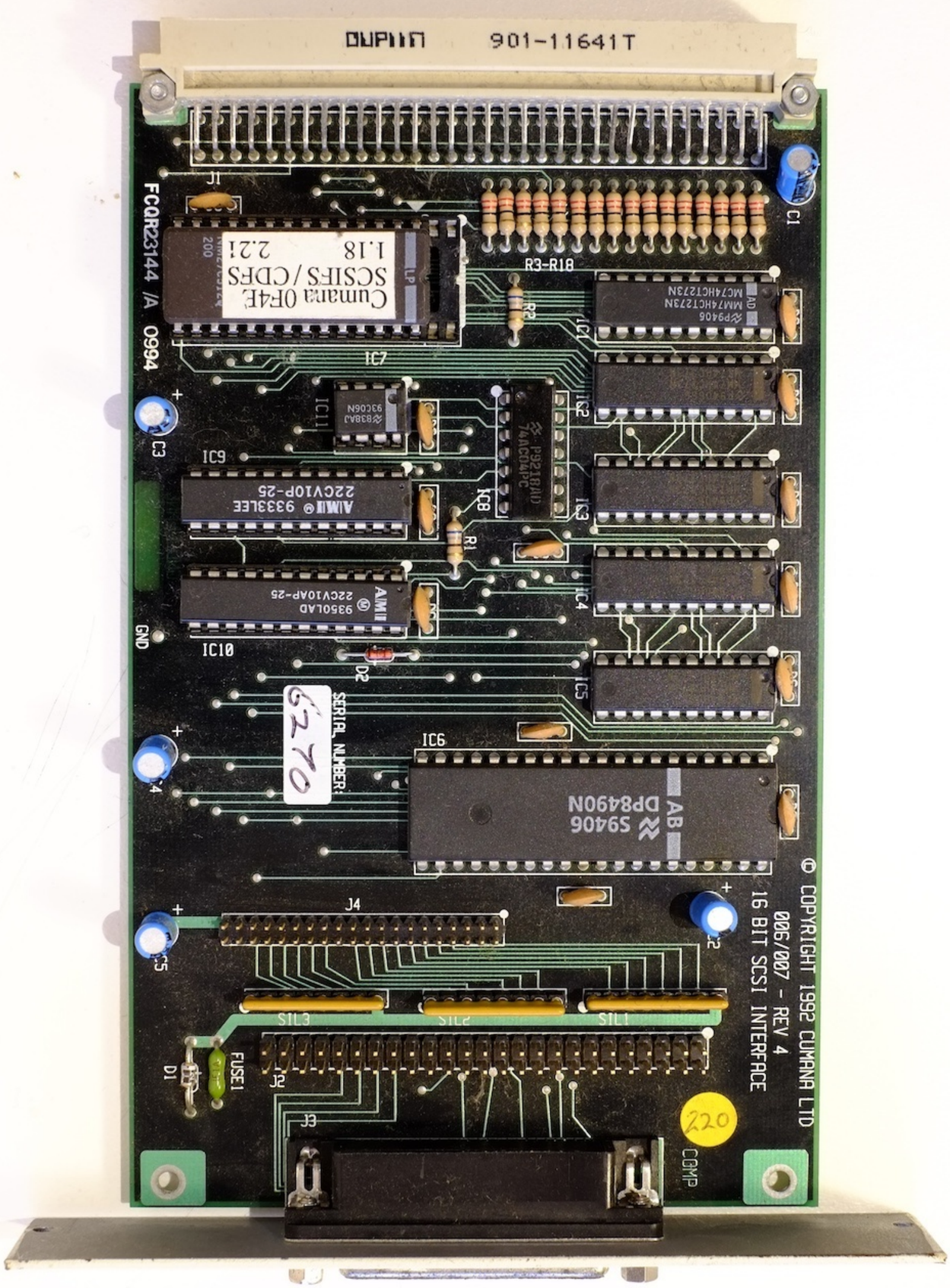
FIQ interrupts enable fast “software IO”

Right choice for performance vs cost


the system. Fast response time allows the processor to replace expensive dedicated logic with software, lowering the system cost accordingly. Many component vendors demand higher prices for their DMA device than for their CPU. Unfortunately, the CPU is

VLSI: ARM Family Data Manual

Brings “RISC principles” to system: Don’t do it in HW if you can do it in SW


DUPLIN 901-11641T
FCQR23144 /A 0994
J1
C1
R3-R18
R2
IC7
IC1
IC2
IC3
IC4
IC5
IC6
IC8
IC9
IC10
IC11
R1
C3
C4
C5
C2
GND
D2
SERIAL NUMBER
0270
J4
J2
J3
SIL3
SIL2
SIL1
D1
FUSE1
220
COMP
© COPYRIGHT 1992 CUMANA LTD
006/007 - REV 4
16 BIT SCSI INTERFACE

Hardware: A+++ great seller would buy again

ARM pipelining and LDM/STM make best use of DRAM

Simple as possible: lower power, lower chip cost

No cache: Right choice for simplicity

No DMA: PIO and 16-bit are good for cost

System partitioning: Not all buses on all chips, cheap 68-pin packages

MMU: Pretty weird inverted CAM-style

→ Cheap, high performance, low power, worked at all...

Now we just need an OS

ARX, the almost OS

Acorn Research Centre (ARC) started in 1984 in Palo Alto, to develop ARX:

- Multitasking, microkernel-based, virtual memory
- GUI (and A500 keyboard has mystery Looks/Menu/Again/Cmd keys)
- Written in Modula-2
- Resource-hungry – not fast on a 4MB A500 with HD

With the hardware nearing completion in 1986, ARX wasn’t complete (or fast) enough

ARX got killed, ARC closed. So it goes.

Plan B, Arthur, RISC OS

In 1986 (with hardware looming), Acorn started a new OS with well under a year of development time before the Arc’s launch

- Star-commands like BBC Micro MOS
- “Relocatable Modules” – loadable services/libraries
- 512K ROM, hand-written assembler
- BBC BASIC V (procedures, FP, graphics/fonts support)
- RISC OS adds co-operative multitasking, drag & drop, anti-aliased outline fonts
- “Modules from podules”

Arc launched with Arthur in 1987, then RISC OS in 1988

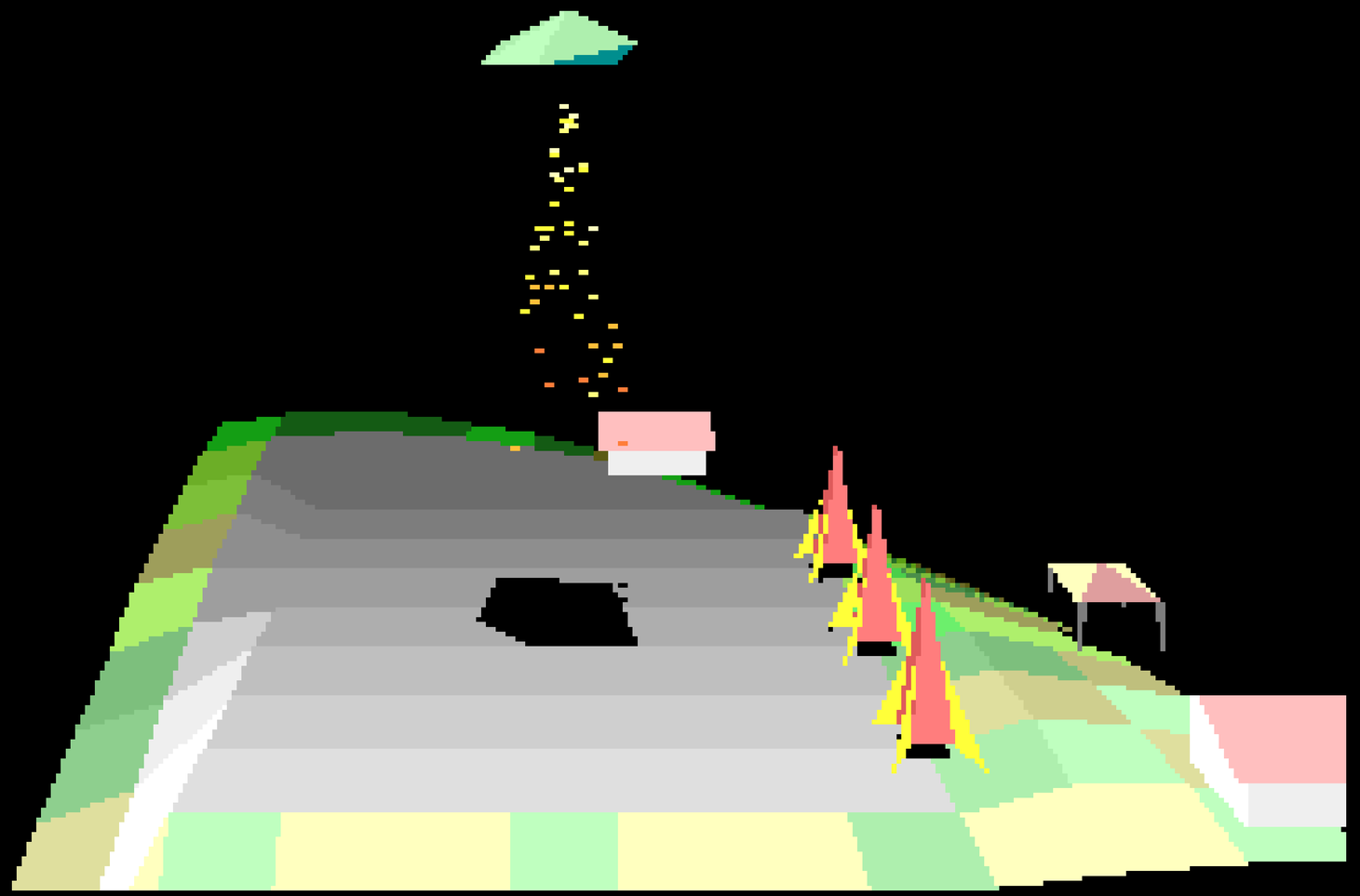
Arthur 0.30 (17 Jun 1987) 1024K

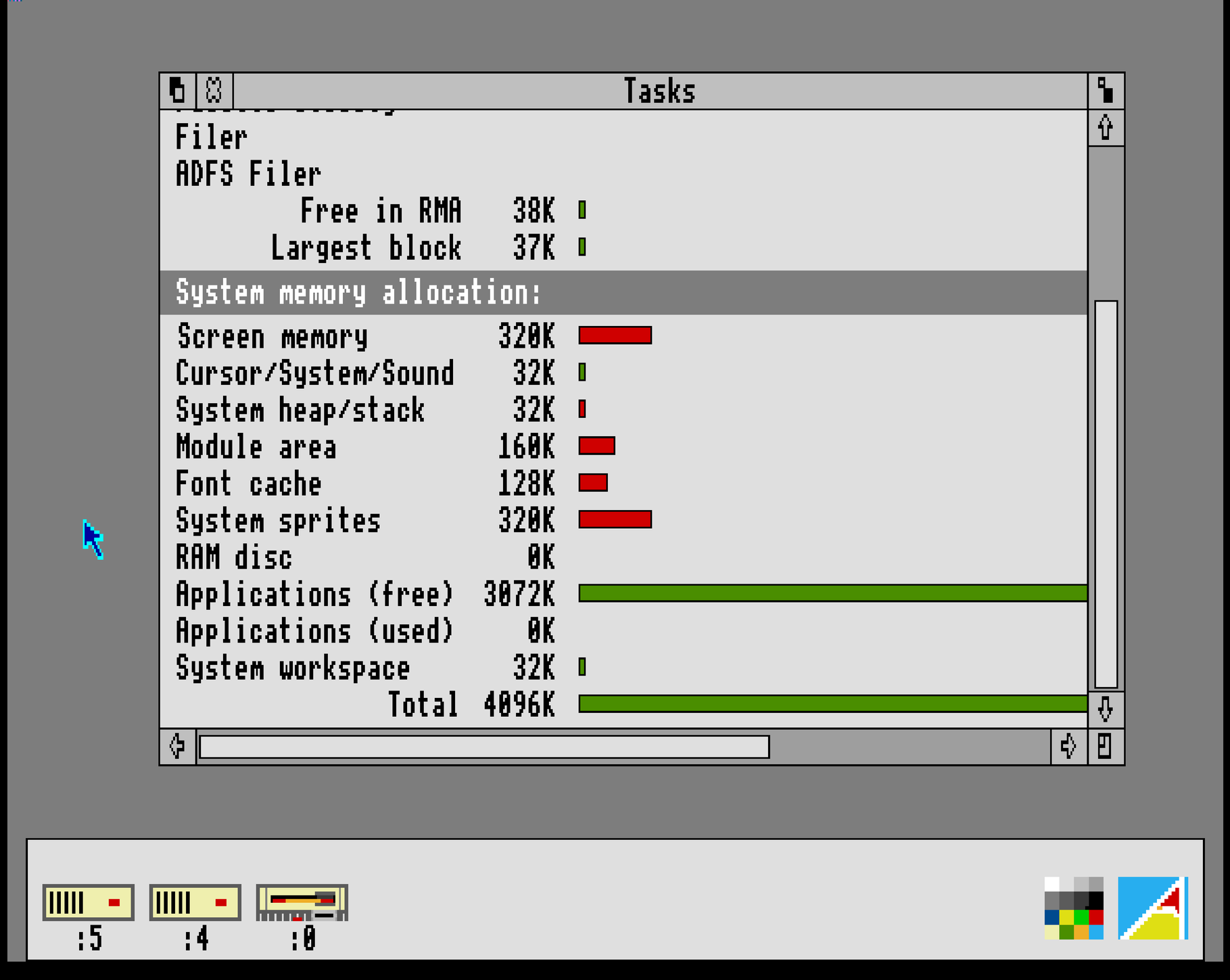
Acorn ADFS

Arthur Supervisor

*

Lander Demo/Practice (C) D.J.Braben 1987
500 3 500




Tasks
Filer
ADFS Filer
Free in RMA 38K
Largest block 37K
System memory allocation:
Screen memory 320K
Cursor/System/Sound 32K
System heap/stack 32K
Module area 160K
Font cache 128K
System sprites 320K
RAM disc 0K
Applications (free) 3072K
Applications (used) 0K
System workspace 32K
Total 4096K
:5
:4
:0

Resources:$.Apps

ADFS::GoodDisc.$.App1.DrawDemo

!Configure

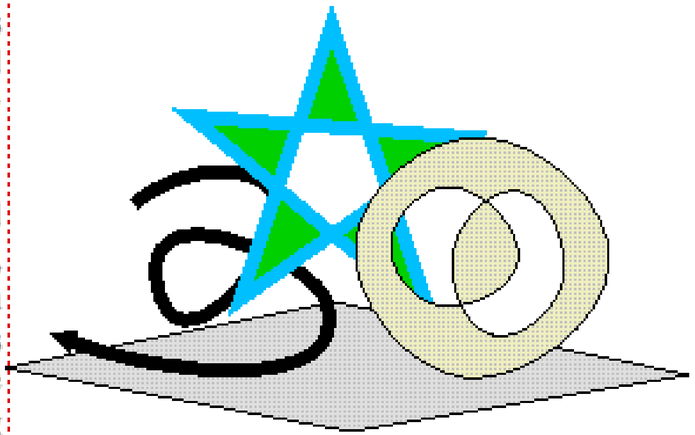
!Paint

Draw is the most powerful of the applications supplied with RISC OS. It is a structured graphics program that can work with a variety of basic object types.

Paths are sequences of straight and curved lines of arbitrary thickness and colour, optionally filled with another colour. Dash patterns, arrow-heads and various join styles are all supported. All basic graphic shapes are constructed from paths. Extensive editing facilities are provided for the entering, construction and manipulation of paths.

Text can be supplied either as a single-line text object, or as a multi-column text area like the one you are reading now. Text can be in any colour, and is rendered using anti-aliasing where possible.

Images can be included in a Draw document as bitmaps. Such bitmaps (which can be coloured and partially transparent) are known as sprites. Extensive facilities for editing sprites are provided by Paint.



Draw

Characters

```
 ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [ \ ] ^ _
` a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~
```

HostFS GoodDisc RISCiXBase :1 :0 Apps

1988/9: RISCiX – Acorn’s BSD 4.3 UNIX

A range of workstations:

- R140 (really an A440/1)
- R225/R260 (really an A540)

A680 unreleased, pre-dates these

M4 very rare development machine

RISCiX development started *after chipset was complete*

32KB page size very inconvenient but “challenge = opportunity”:

- RISCiX used novel *on-demand decompression* for binaries — sparse filesystem in 1988!
- The only way a 50MB R140 HD would hold a usable distro……

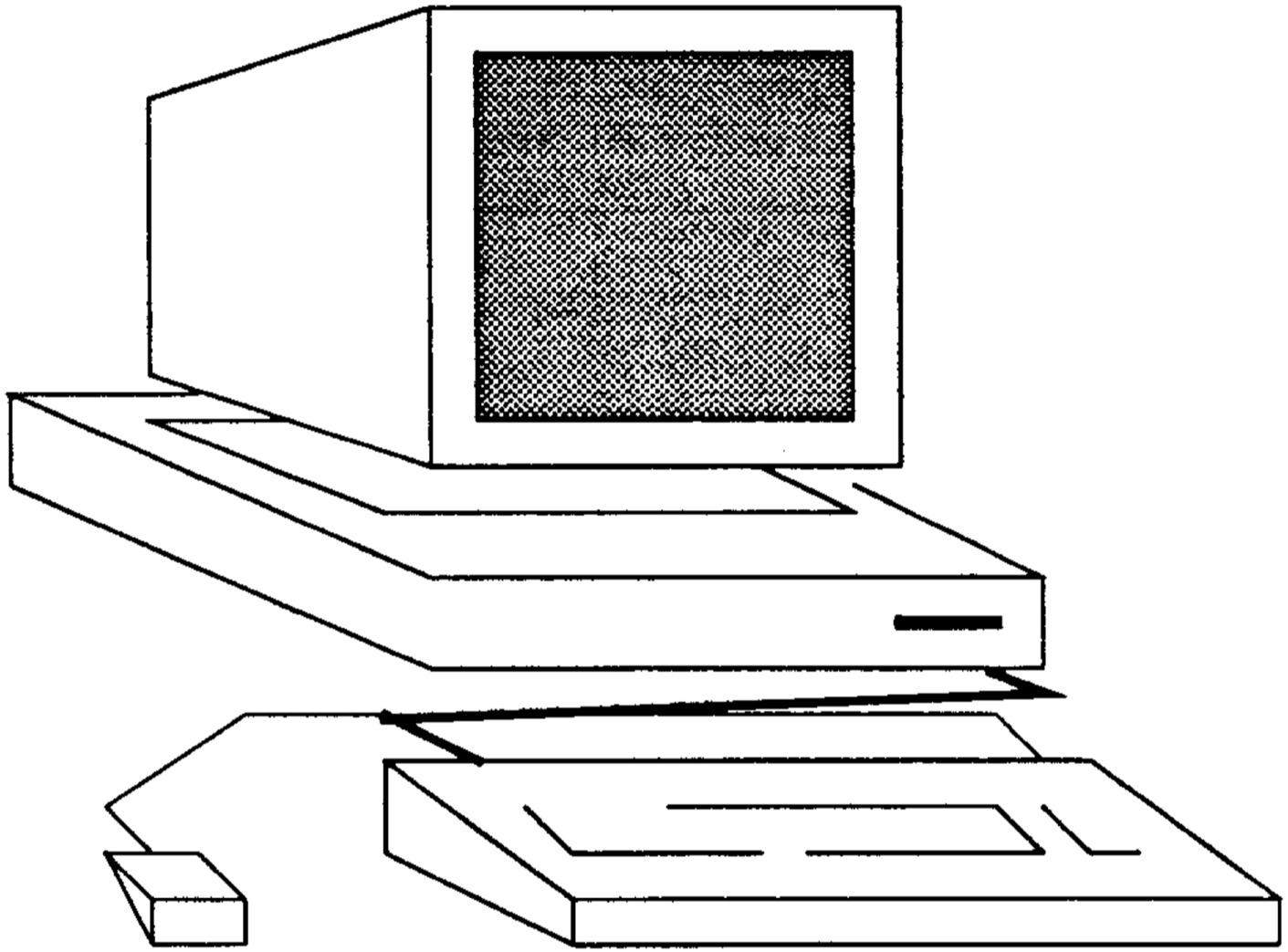


Figure 1.1 : Acorn Technical Publishing System

A680
Technical
Publishing
System

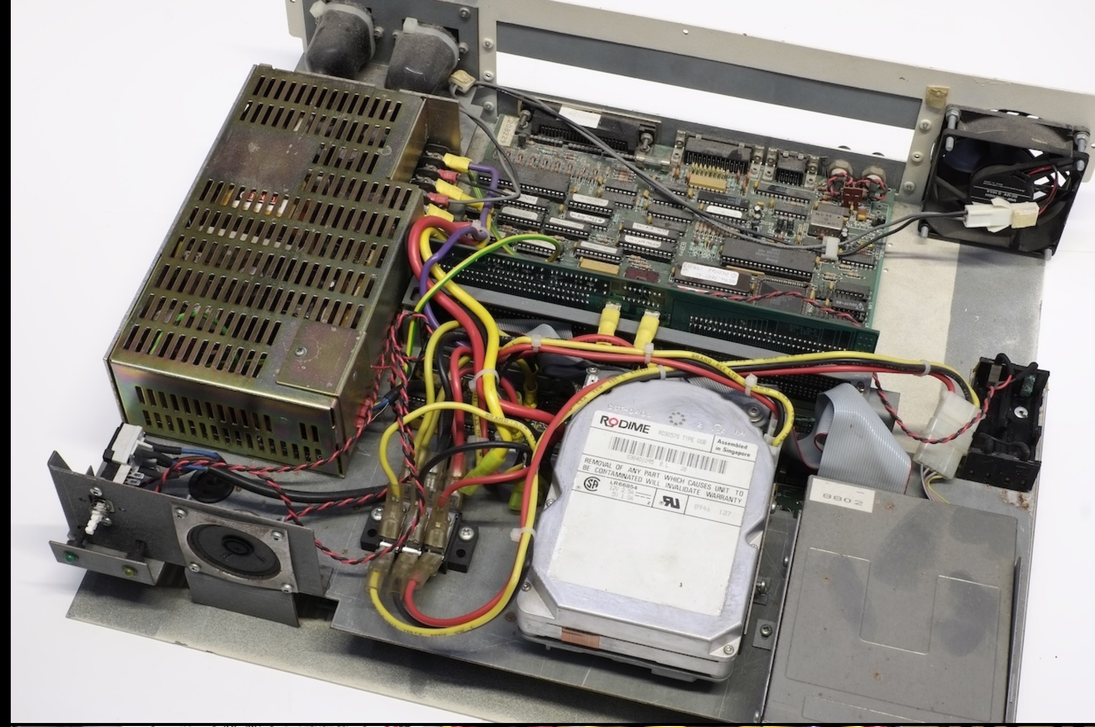
ARM2

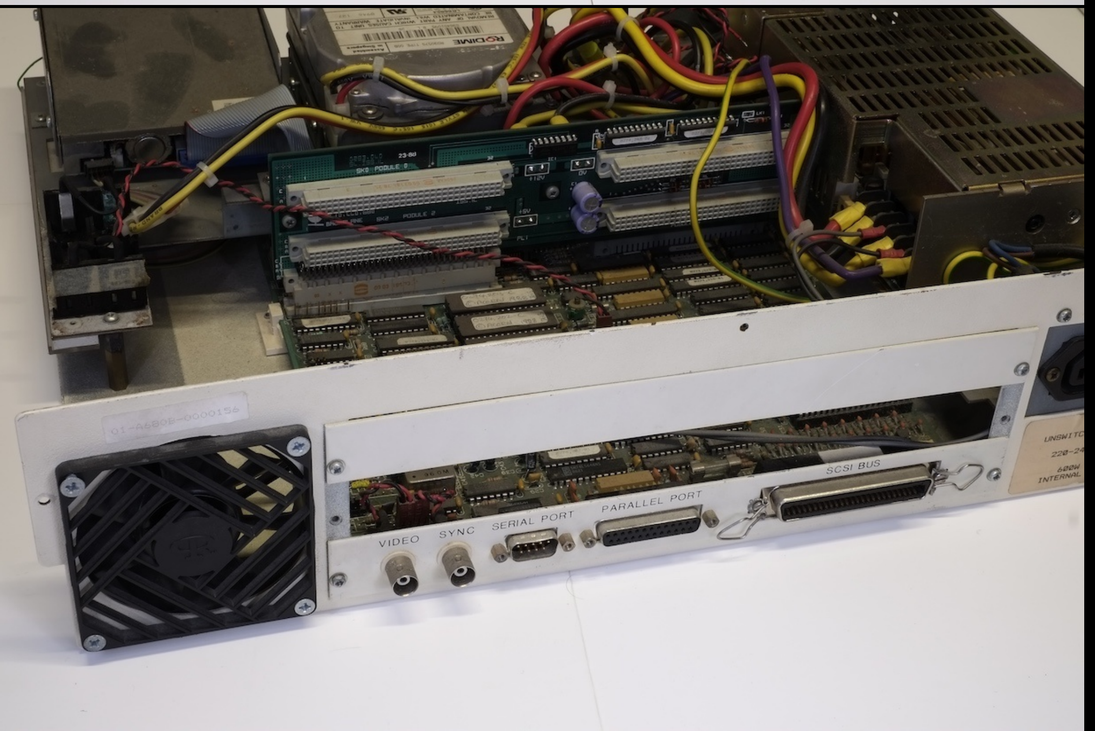
Dual MEMC,
8MB RAM

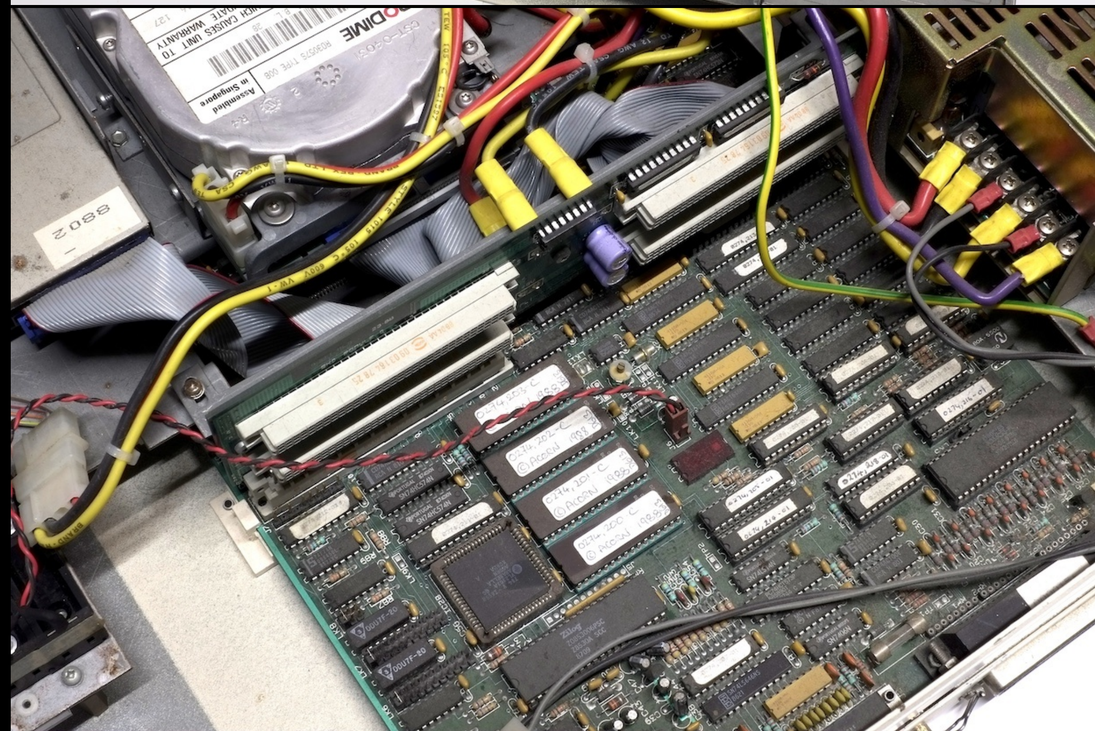
On-board
SCSI

RISCiX only

1152x900
*mono video
only*







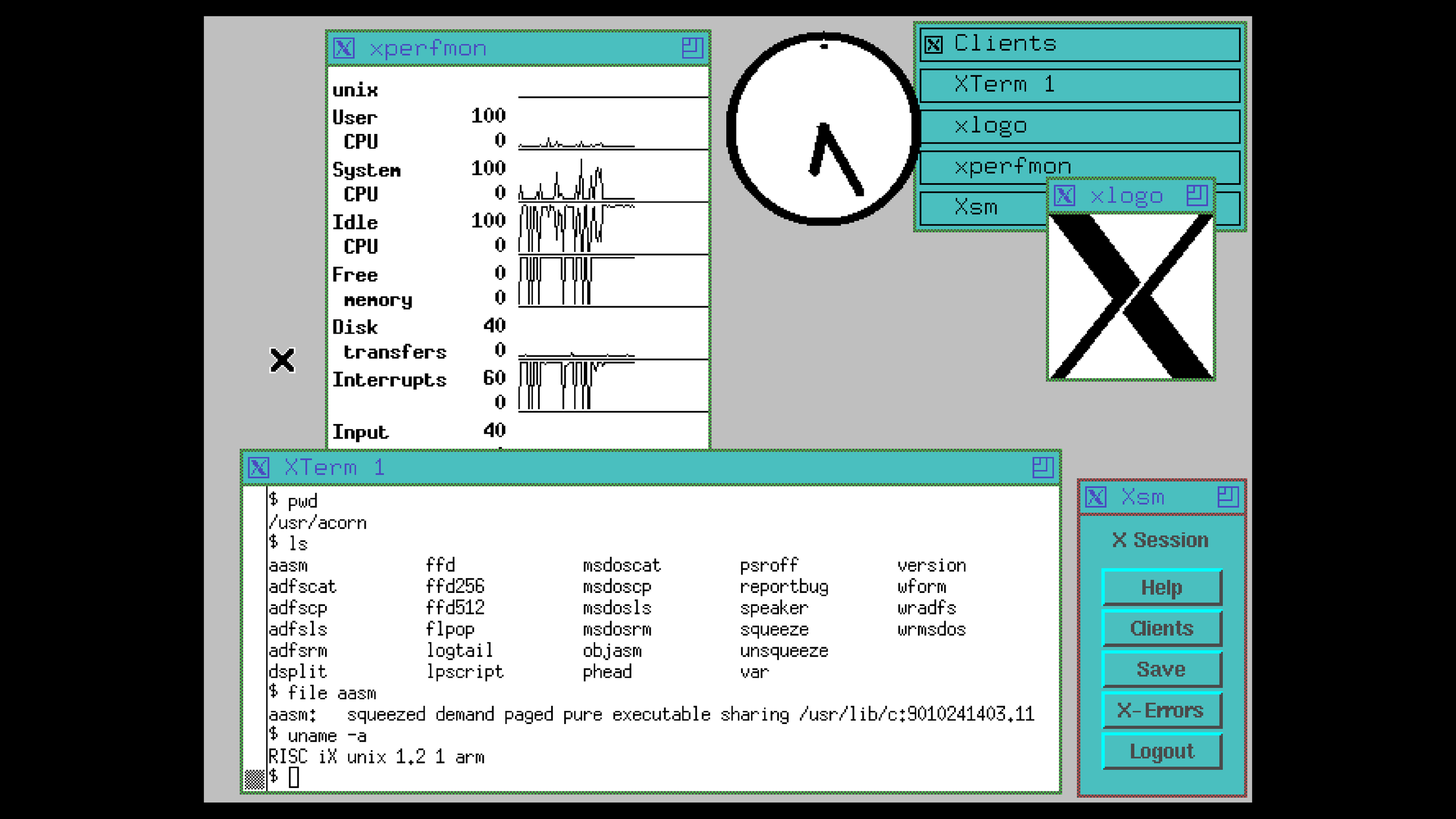
```
RISC iX 1.21a made Wed Apr 17 20:25:31 1991
real mem  = 16777216
avail mem = 15499264
72 buffers (576 Kbytes)
st[0-1]: internal controller
xcbman: NOTICE: no backplane interrupt hardware fitted
et0: can't initialise, status 0x0
et0: slot 0: iss 1, address 00:00:a4:de:ad:69
WARNING: system has not booted for 11724 days -- CHECK THE DATE!
Swap size = 32.7 Mb
root fstype  4.3, name /dev/st0a
swap fstype spec, name /dev/st0S
Dec  9 17:58:26 init: single user boot
#
```

```
# ^D
Checking discs...please wait
Thu Dec  9 17:58:56 GMT 1999
/dev/st0a: Used: 7024 files, 63958 kbytes
/dev/st0a: Free: 147161 kbytes, 3777 frags, 17923 blocks, 1.8% fragmentation
Thu Dec  9 17:59:00 GMT 1999
starting system logger
starting local daemons: portmap statd lockd.
preserving editor files & clearing /tmp
standard daemons: update cron accounting printer.
starting local services:/etc/rc.local: route .
.
Thu Dec  9 17:59:01 GMT 1999

Multi-user system started


RISC iX release 1.21
5:59pm on Thurs, 9 Dec 1999 on console of unix
unix login: root
Password:
Last login: Fri Nov  3 07:11:09 on console
RISC iX 1.21a made Wed Apr 17 20:25:31 1991
# _
```


Welcome to the X Window System
Login: user
Password:


xperfmon
unix
User 100
CPU 0
System 100
CPU 0
Idle 100
CPU 0
Free 0
memory 0
Disk 40
transfers 0
Interrupts 60
0
Input 40
Clients
XTerm 1
xlogo
xperfmon
Xsm
xlogo
XTerm 1
$ pwd
/usr/acorn
$ ls
aasm ffd msdoscat psroff version
adfscat ffd256 msdoscp reportbug wform
adfscp ffd512 msdosls speaker wradfs
adfsls flpop msdosrm squeeze wrmsdos
adfsrm logtail objasm unsqueeze
dsplit lpscript phead var
$ file aasm
aasm: squeezed demand paged pure executable sharing /usr/lib/c:9010241403.11
$ uname -a
RISC iX unix 1.2 1 arm
$
Xsm
X Session
Help
Clients
Save
X-Errors
Logout

Lots of useful stuff in ROM

Cheaper floppy-only Arcs were intended for school/home use

Program it right out of the box! Just need a floppy disc

Powerful *assembler* in BASIC!

```
>LOAD "ASM"
>LIST
   10 DIM space% 1024
   20
   30 FOR pass%=0 TO 2 STEP 2
   40  P%=space% : REM P% is assemble address
   50  [OPT pass%
   60
   70   .my_func
   80         ;; Called from BASIC, R0,R1 are parameters
   90         ;; from magic variables A%, B%:
  100
  110         MUL     R2, R0, R1
  120
  130         ;; R0 returns a value via USR()
  140         MOV     R0, R2
  150
  160         MOVS    PC, R14
  170
  180   ]:NEXT
  190
  200 PRINT "Assembled "; (P%-space%); " bytes"
  210
  220 A% = 123
  230 B% = 456
  240 result% = USR(my_func)
  250
  260 PRINT A%; " times "; B%, " is "; result%
>RUN
Assembled 12 bytes
       123 times 456 is 56088
>_
```

The Arc legacy

(Well, Arm…)

The machine helped democratise GUI/WYSIWIG and *did* bring ‘MIPS to the Masses’

RISC OS GUI influenced other GUI systems

- The icon bar :-)
- Anti-aliased outline fonts

People realised that designing a custom computer was possible even if not a megacorp → SoCs, eventually

ADFS::GoodDisc.$.Thanks Lock

Thanks to:

Sophie Wilson

Steve Furber

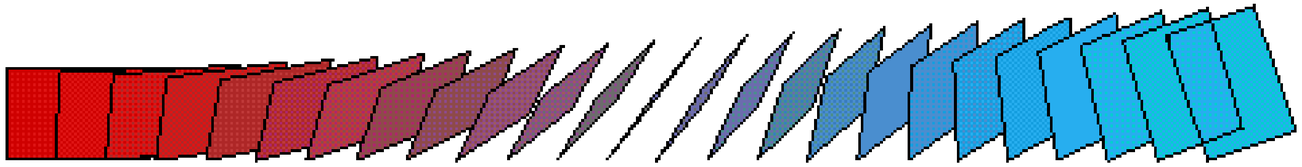
Tudor Brown

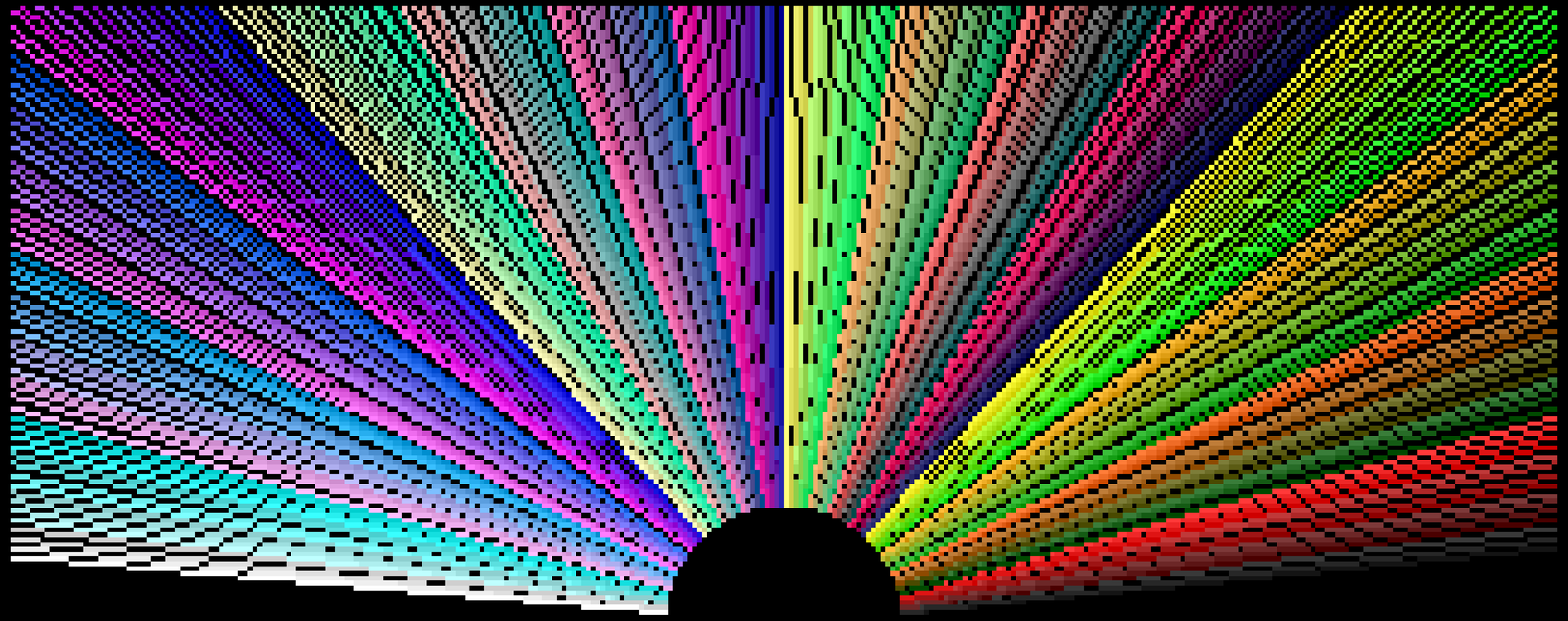
Mike Muller

John Biggs

Jamie Urquhart

The Centre for Computing History





Thank you! Any questions?

https://axio.ms

Press SPACE or click mouse to continue

Backup

Not the Risc PC

- This talk is about the **“classic” Archimedes** machines 1987-1990:
  - 1987 – **A300, A400** series
  - 1989 – **A400/1** series, **A3000**
  - 1990 – **A540**
-
- The later machines are the same in most of the interesting ways:
  - **A5000 & A4** (PC-style floppy/IO, IDE)
  - **A3010/A3020** (ARM250 SoC, but *very similar*)

...and eventually got fancy 32-bit addressing in 1994:

- **Risc PC** (ARM610/ARM710/StrongARM, familiar MMU, IDE, DMA)

1983	1984	1985	1986	1987	1988	1989	1990	1991
BBC Micro	Design begins			A300/ A400		A3000 A400/1 R140	R540 R225 R260	A5000
		BASIC V		Arthur 1.2	RISC OS 2	RISCiX 1.13	RISC OS 2.01 RISCiX 1.2	RISC OS 3
		ARM1		ARM2		ARM3		
		VIDC	MEMC & IOC					

On ARM1 (6MHz),

The ARM chip packs 25,000 transistors onto a small 50-mm² chip. In contrast, Motorola packed about 192,000 devices onto an 80-mm² chip using 2-μm design rules to build the 2.5-mips 68020 microprocessor. Acorn's smaller chip improves yields and lowers chip costs by about a factor of four. The company

The chip is about twice as fast as a VAX-11/780

"Electronics", Aug 1985

Trivia

- Designed in the ‘Silver building’, next to Arm’s Cambridge HQ
- Designed on Apollo Domain 68000 workstations (DN600, DN300). VLSI Compass tools.
- Titles for various BBC broadcast TV shows
- WE32206 80-bit FPU was used in AT&T 3B2 “UNIX PC”
- ARM2 was running at 12MHz in some A500s (with fast memory) and up to 18MHz reported with static memory!


ACORN COMPUTERS
F.P. CO-PROCESSOR
WE32206 FLOATING POINT CO-PROCESSOR
AKA20
AT&T
IC1
IC2
C1
DAMAGED
100u

Linux for arm26

Bitrot until 2007, when it was removed

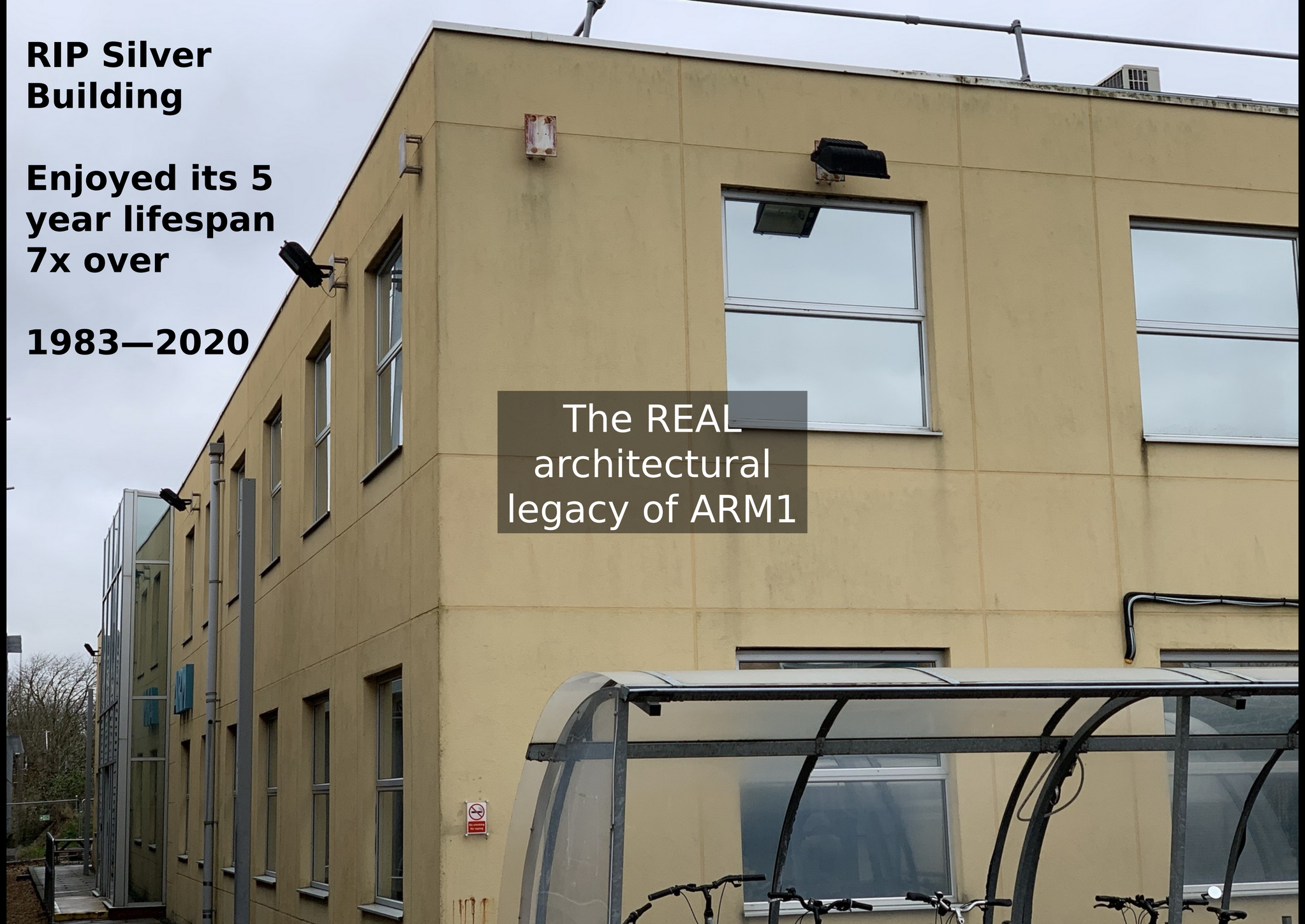
GCC 4.0 quite rightly stopped supporting arm26

...even NetBSD’s acorn26 port was removed in 2015... :(

My favourite quotes

MM: “Nobody told us it was difficult”

SW: “Luckily there was no internet or it couldn’t have been done in the time scale”


RIP Silver Building
Enjoyed its 5 year lifespan 7x over
1983—2020
The REAL architectural legacy of ARM1