Arquiteturas com Memória Universal

Emilio Francesquini
e.francesquini@ufabc.edu.br

Centro de Matemática, Computação e Cognição
Universidade Federal do ABC

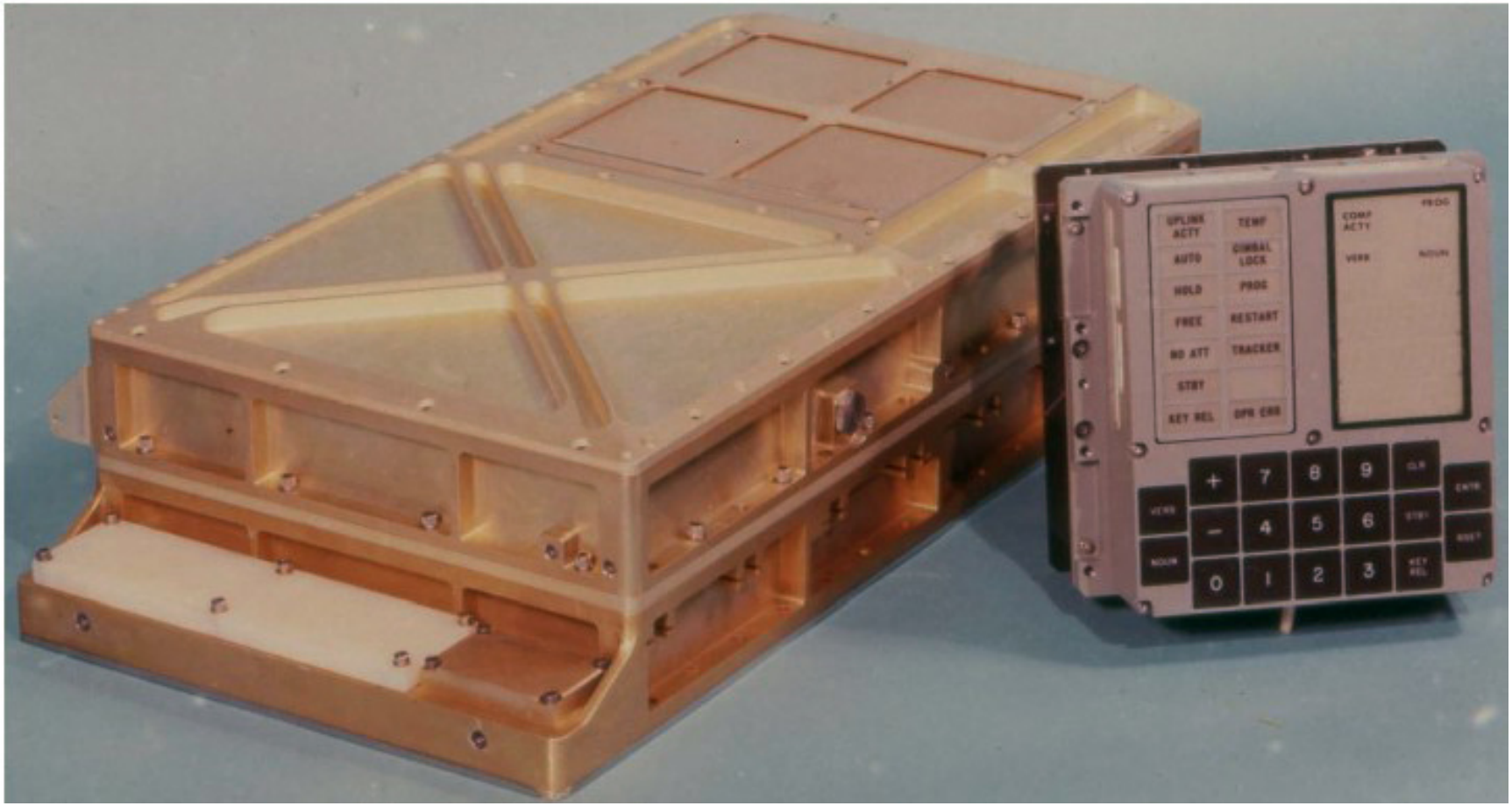
Arquitetura de Computadores
2020.Q1

Roteiro

UFABC

- Introdução
- Tecnologias de hardware
- Perspectivas de pesquisa em software envolvendo NVMs e Memória Universal
- Pesquisas em desenvolvimento no LSC do IC-Unicamp
- Conclusão

Introdução


UPLINK ACTY
TEMP
AUTO
GIMBAL LOCK
HOLD
PROG
FREE
RESTART
NO ATT
TRACKER
STBY
KEY REL
COMP ACTY
PROG
VERB
NOUN
+
7
8
9
CLR
ENTR
−
4
5
6
RSET
0
1
2
3
KEY REL

Sistema de navegação Missões Apollo

UFABC

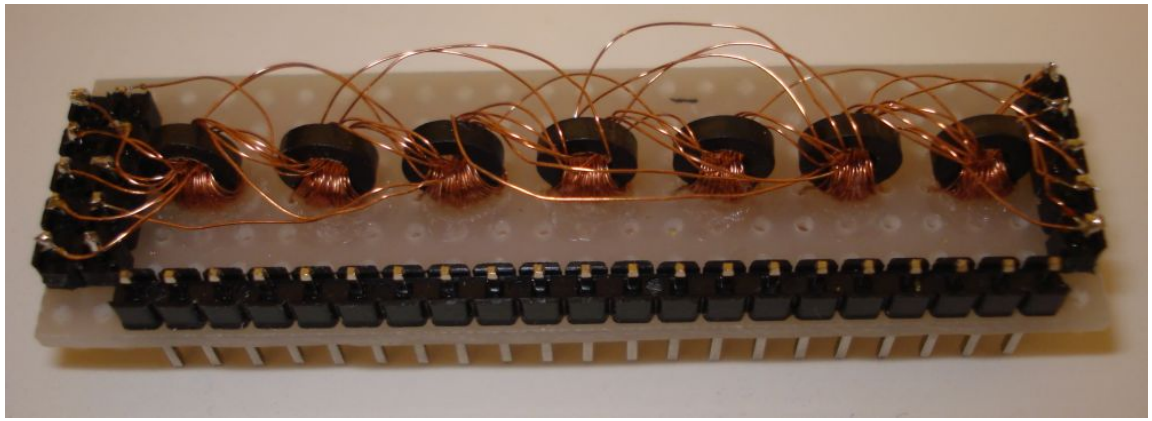
Desenvolvido na década de 60

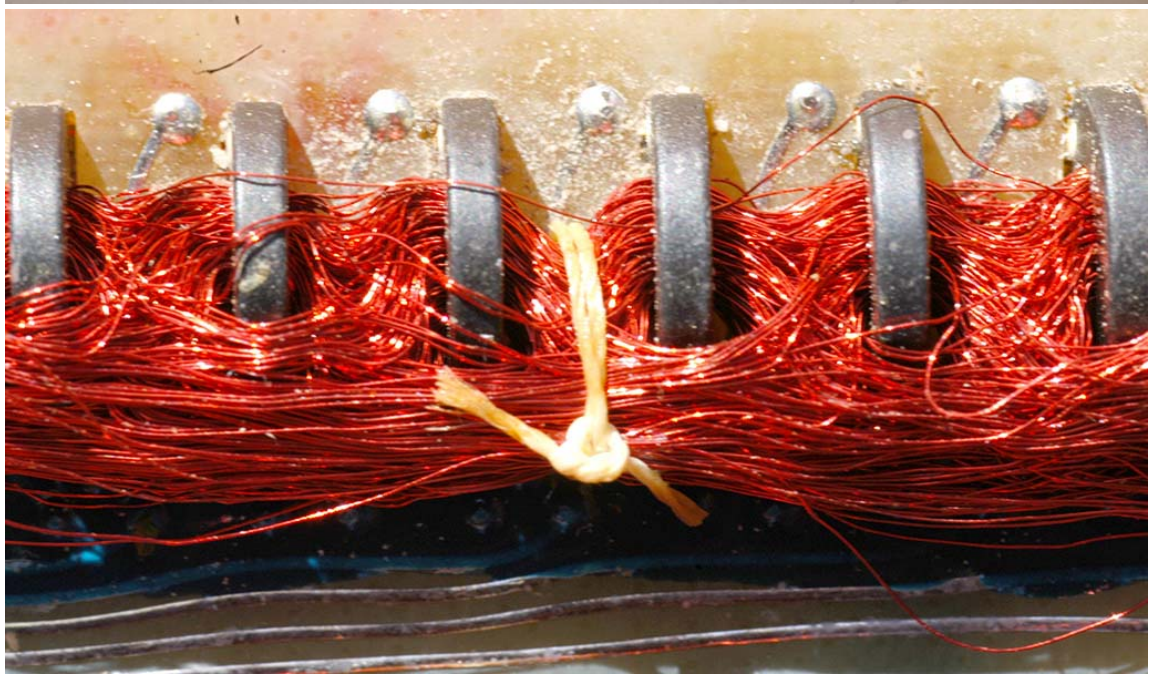
- Tinha ideias revolucionarias para a época como: escalonamento preemptivo com prioridades e comunicação assíncrona

Sistema de memória

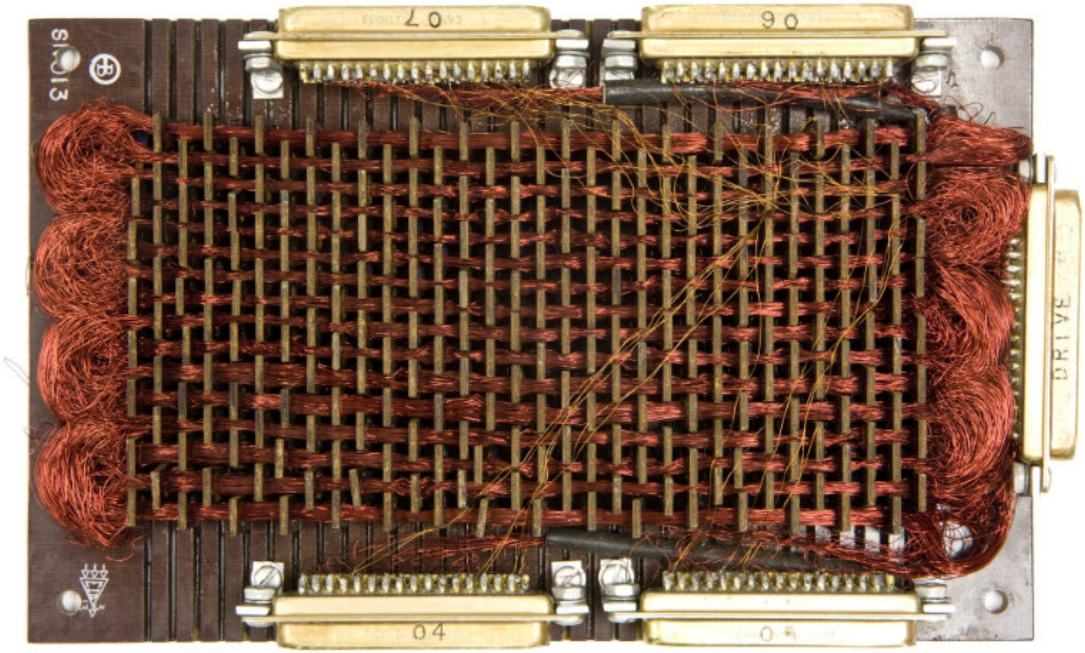
- "Memória de corda" (*core rope memory*) para a *fixed memory*
  - AKA, "LOL Memory"
  - 36K palavras, ou 72KB de ROM
- Memória de ferrite (*magnetic-core memory*) para a *erasable memory*
  - 2K palavras, ou 4KB de RAM

Memória de corda

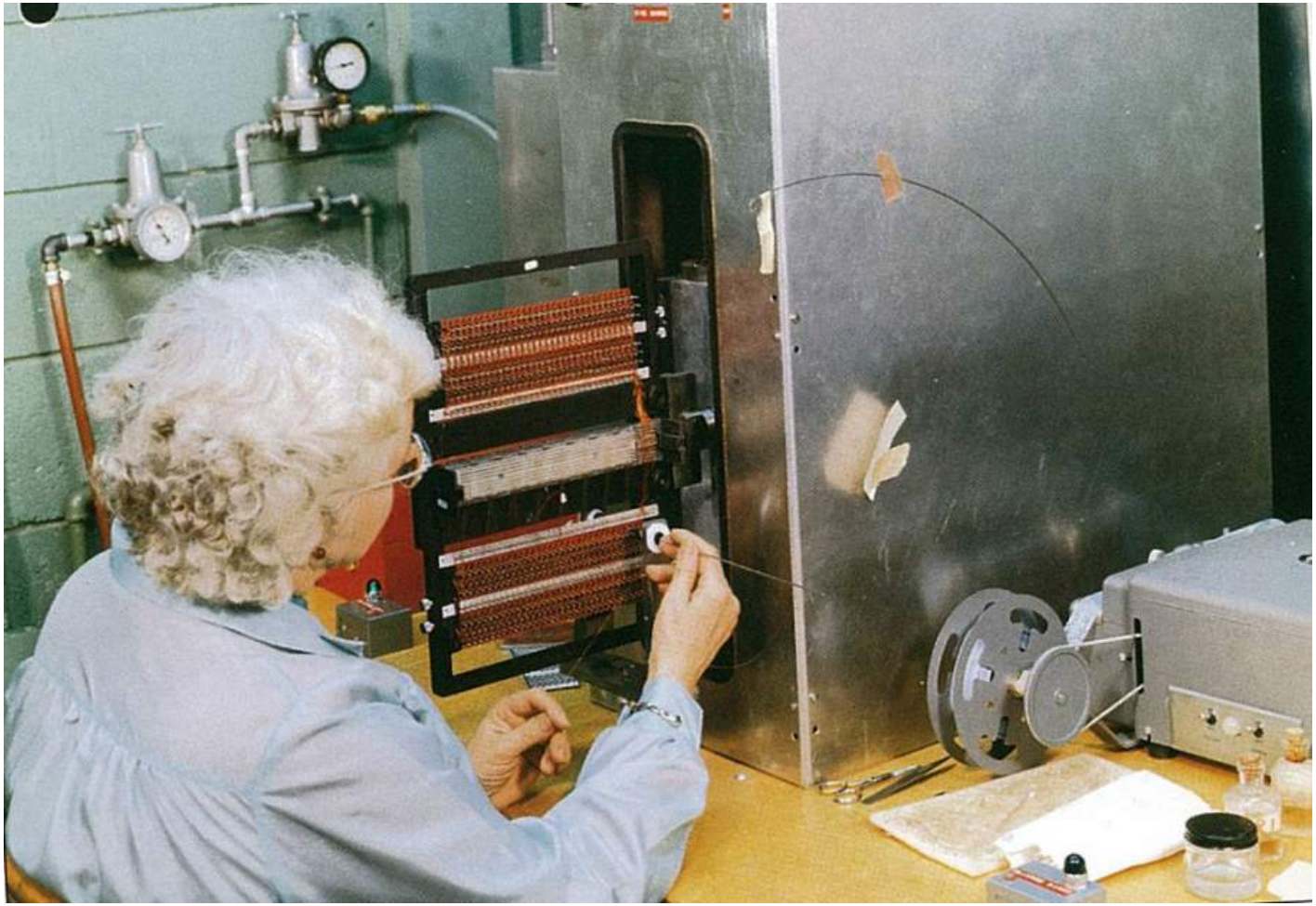




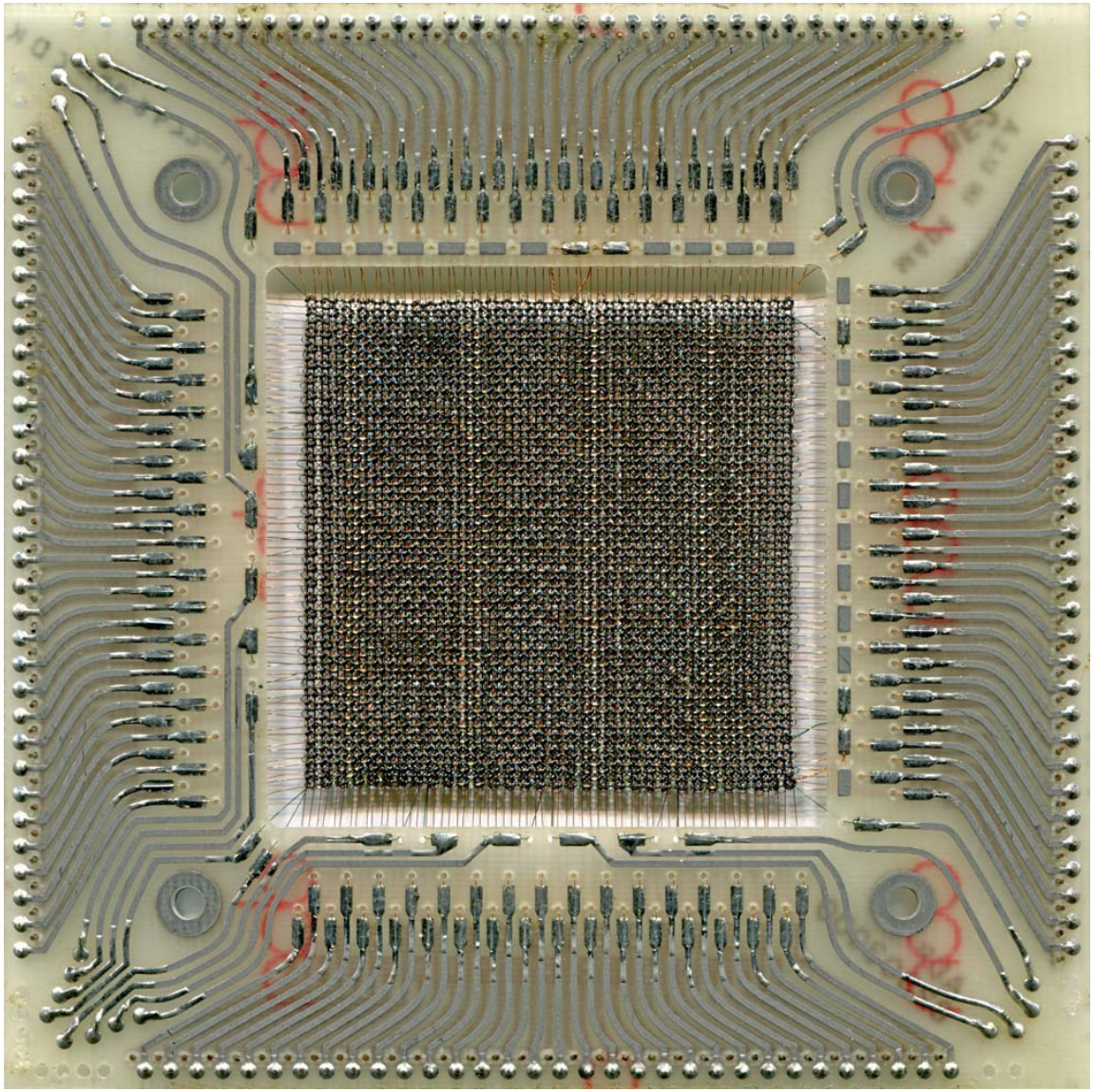
Memória de corda



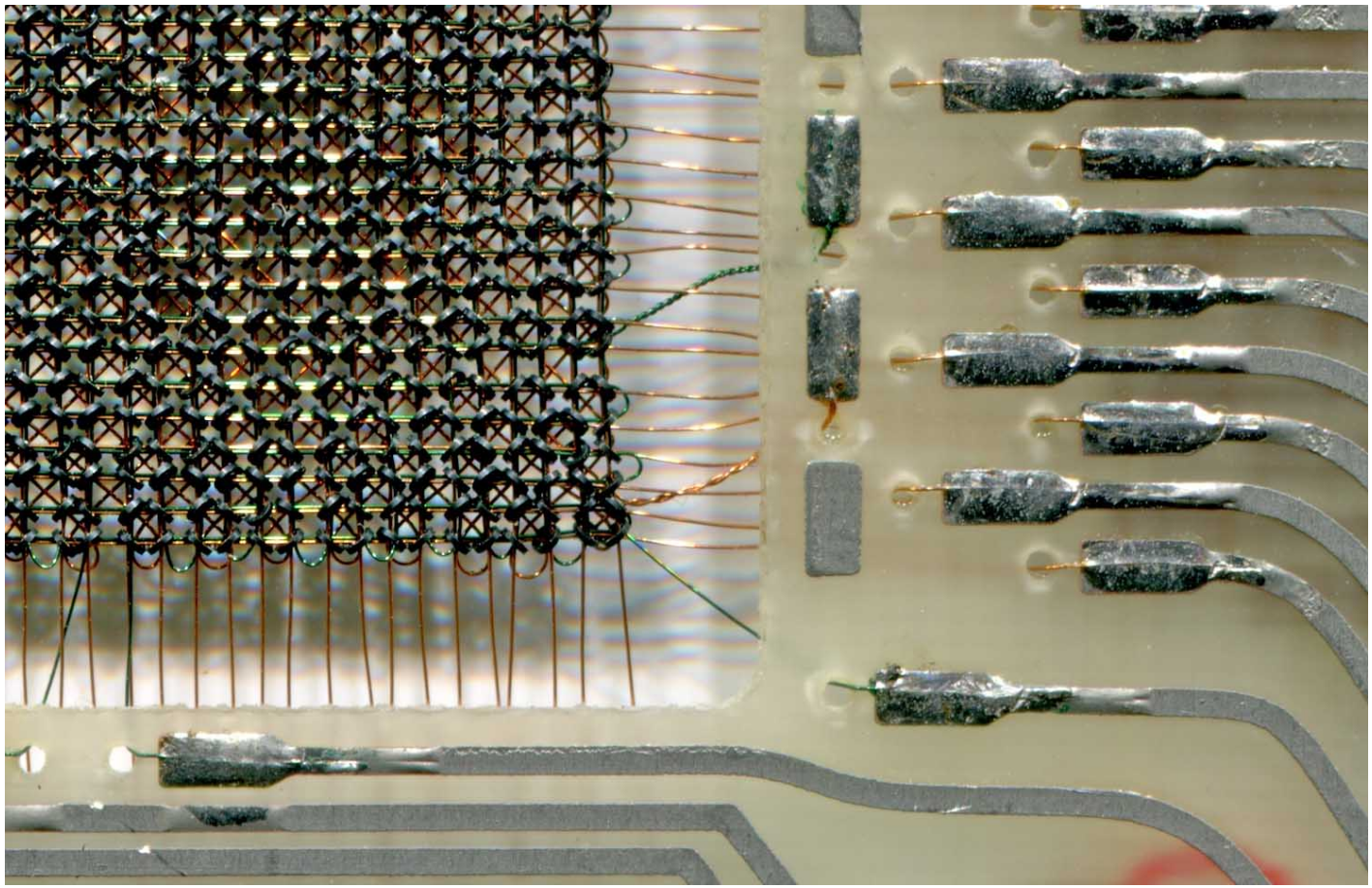
Memória de corda



Memória de Ferrite

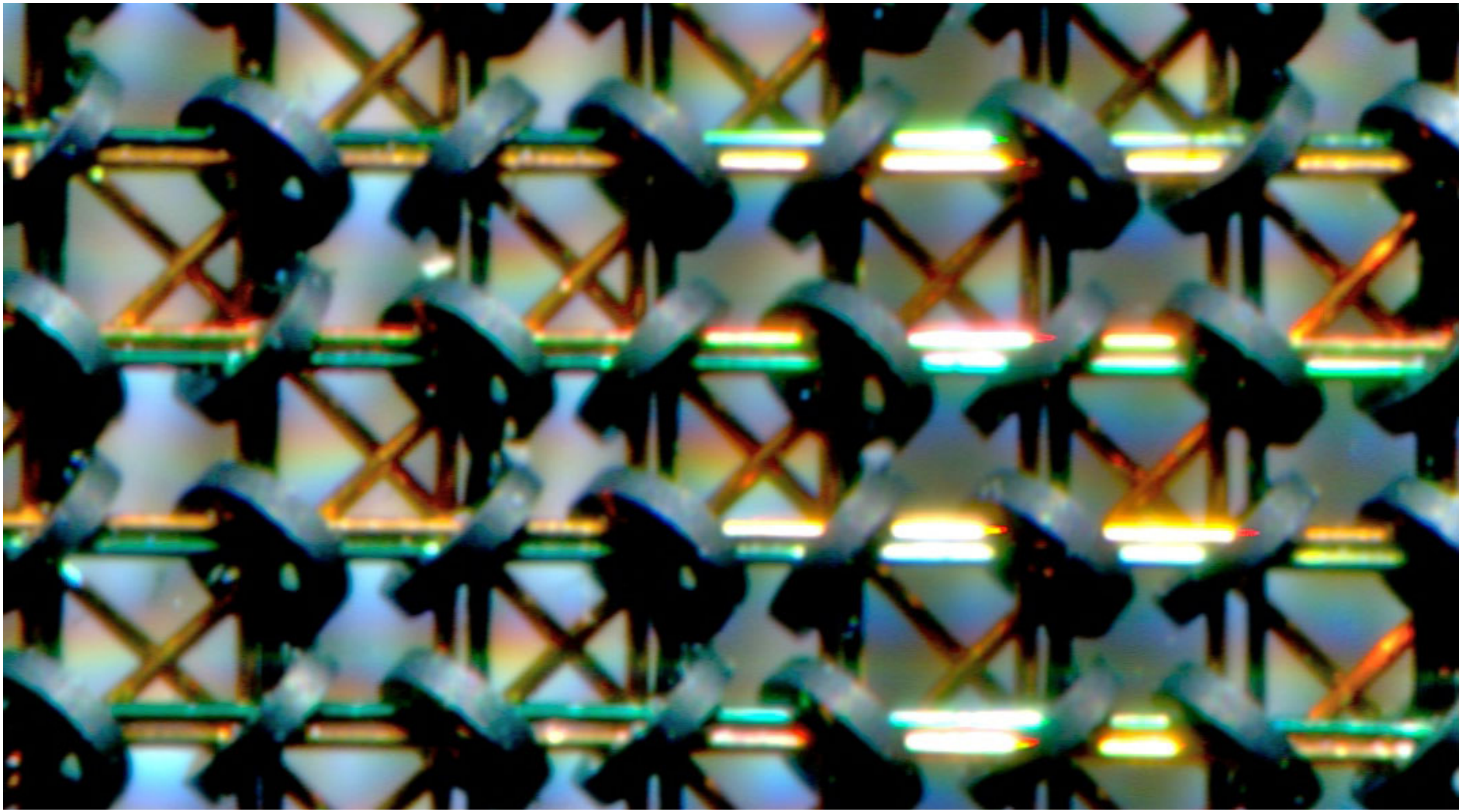


Memória de Ferrite

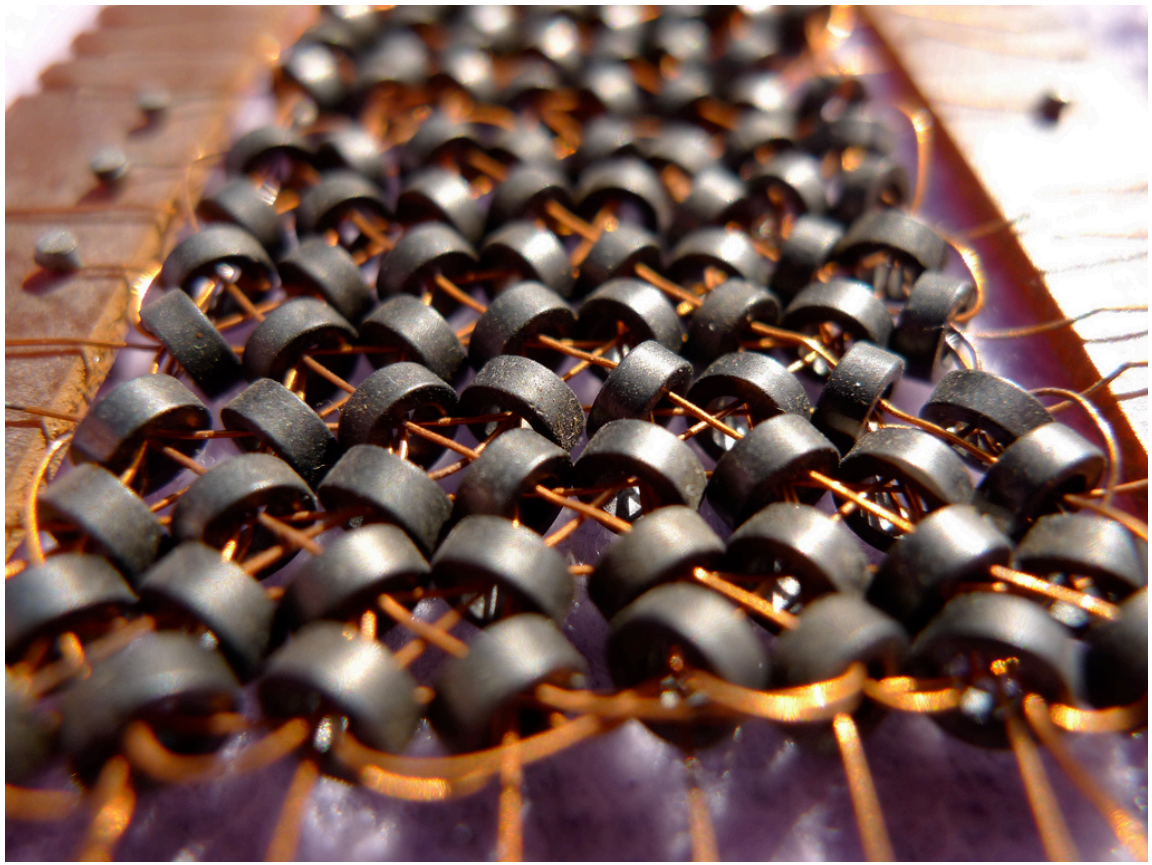


Memória de Ferrite

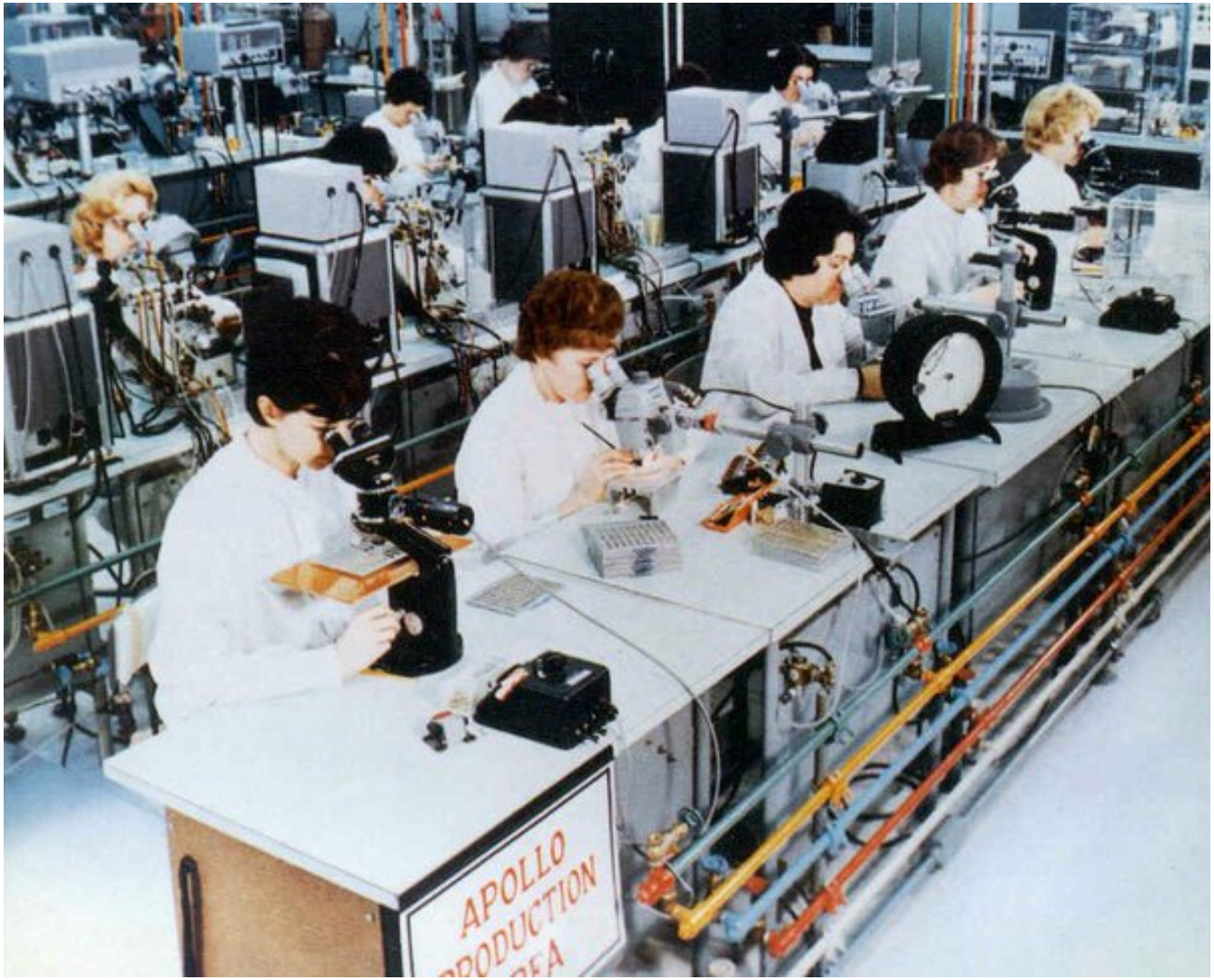
UFABC



Memória de Ferrite



Memória de Ferrite



Tecnologias de memória

UFABC

Diversas tecnologias:

- Tubo de raios catódicos (46) — Anos 40
- Core Memory (47) — Anos 50 e 60
- FE-RAM (52) — Desenvolveu-se no fim dos anos 80
- DRAM (68) SRAM (64) — Anos 70 até hoje
- PCM (69) — Apenas recentemente explorada
- FLASH (84) — Popularizou-se nos anos 90
- STT-RAM (96)— Início da exploração comercial nos anos 00

Tecnologias de memória

UFABC

Diversas tecnologias:

- Tubo de raios catódicos (46) — Anos 40
- Core Memory (47) — Anos 50 e 60
- FE-RAM (52) — Desenvolveu-se no fim dos anos 80
- DRAM (68) SRAM (64) — Anos 70 até hoje
- PCM (69) — Apenas recentemente explorada
- FLASH (84) — Popularizou-se nos anos 90
- STT-RAM (96)— Início da exploração comercial nos anos 00

SRAM e DRAM

- Ampla popularidade: desde dispositivos móveis até supercomputadores
- Tecnologia madura, sendo desenvolvida há décadas
  - SRAM - 1964
  - DRAM - 1968
- SRAM $\rightarrow$ maior velocidade
- DRAM $\rightarrow$ menor consumo de energia e maior densidade

Tecnologias de memória

Características	DRAM	FeRAM	STT-RAM	PCM	NOR Flash	NAND Flash
Tamanho da célula (F^2)	6–8	15–32	36	4–12	10	4
Tamanho da página	64b	64b	64b	64b	64b	512b–4KB
Latência R/W	10s ns	10s ns	10s ns	10s/100s ns	100s/1000s ns	10/100s μs
Volátil	sim	não	não	não	não	não
Durabilidade (Nº. escritas)	10^{16}	10^{14}	10^{15}	10^{8}–10^{9}	10^{5}	10^{5}

Fontes: DRAM (Int07; Int09; Mic06a), FeRAM (Doh+10; Esh10;Kim+07), STT-RAM (GIS10; Mis+11; Wu+09), PCM (Lee+09; QSR09) e Flash (CY09;Mic06b; Tos06).

Tecnologias de memória

Características	DRAM	FeRAM	STT-RAM	PCM	NOR Flash	NAND Flash
Tamanho da célula (F^2)	6–8	15–32	36	4–12	10	4
Tamanho da página	64b	64b	64b	64b	64b	512b–4KB
Latência R/W	10s ns	10s ns	10s ns	10s/100s ns	100s/1000s ns	10/100s μs
Volátil	sim	não	não	não	não	não
Durabilidade (Nº. escritas)	10^{16}	10^{14}	10^{15}	10^{8}–10^{9}	10^{5}	10^{5}

Fontes: DRAM (Int07; Int09; Mic06a), FeRAM (Doh+10; Esh10;Kim+07), STT-RAM (GIS10; Mis+11; Wu+09), PCM (Lee+09; QSR09) e Flash (CY09;Mic06b; Tos06).

Tecnologias de memória

Características	DRAM	FeRAM	STT-RAM	PCM	NOR Flash	NAND Flash
Tamanho da célula (F^2)	6–8	15–32	36	4–12	10	4
Tamanho da página	64b	64b	64b	64b	64b	512b–4KB
Latência R/W	10s ns	10s ns	10s ns	10s/100s ns	100s/1000s ns	10/100s μs
Volátil	sim	não	não	não	não	não
Durabilidade (Nº. escritas)	10^{16}	10^{14}	10^{15}	10^8–10^9	10^5	10^5

Fontes: DRAM (Int07; Int09; Mic06a), FeRAM (Doh+10; Esh10;Kim+07), STT-RAM (GIS10; Mis+11; Wu+09), PCM (Lee+09; QSR09) e Flash (CY09;Mic06b; Tos06).

Memória Universal

UFABC

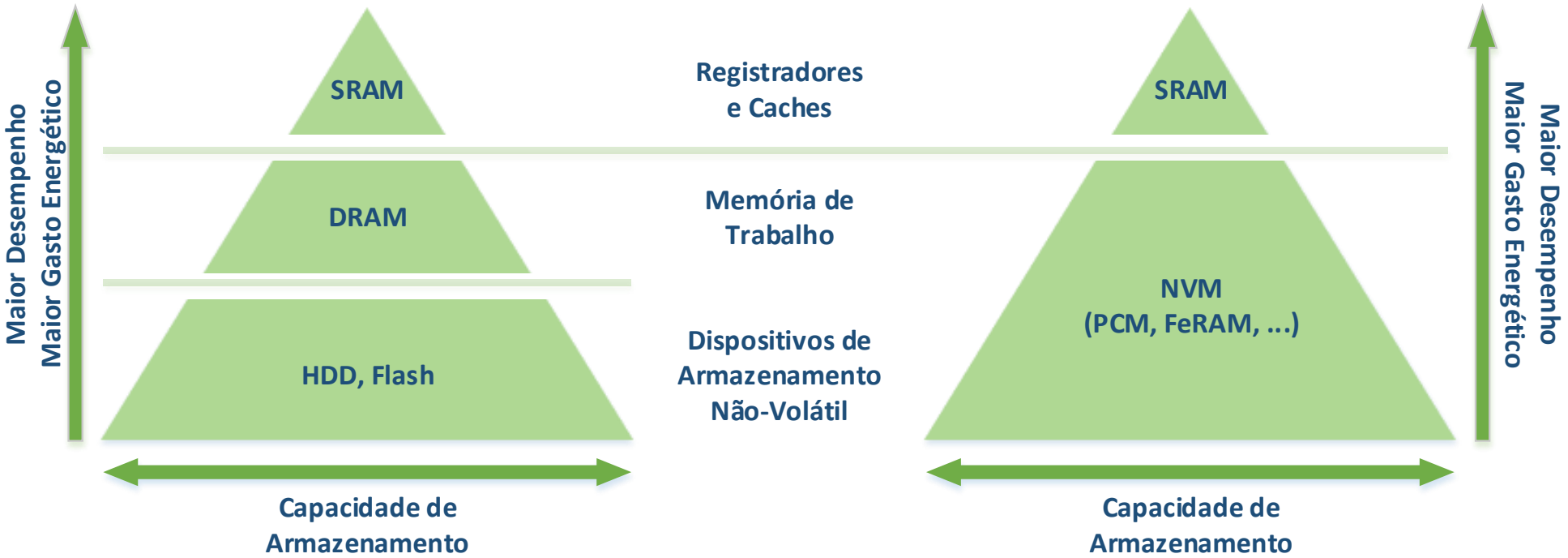
Novas tecnologias de memória vs. DRAM

- Coletivamente conhecidas como Storage-class Memory (SCM)
- Desempenho ainda inferior, mas em alguns casos similar
- Maior densidade
- Majoritariamente não voláteis

Ciclo *carga* $\rightarrow$ *execução* $\rightarrow$ *descarga*

- Perde o sentido com uma memória de trabalho não volátil grande o suficiente
- Coloca em xeque a necessidade de possuir um espaço secundário de armazenamento (discos, SSDs)

Memória Universal



Memória Universal

UFABC

Desenvolvimento e adoção

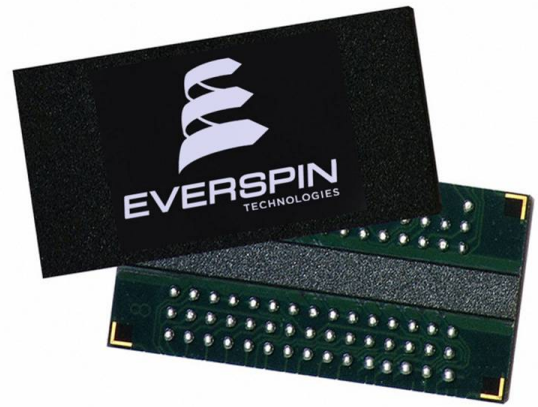
- Inicialmente sistemas híbridos ou parciais
  - Amadurecimento tanto do hardware quanto do software
  - Desenvolvimento dos processos de fabricação dessas novas tecnologias

Que tipo de hardware já está disponível hoje?

UFABC

Memória não volátil

- Sandisk ULLtraDIMM SSDs - Flash (200-400GB)
- Everspin - ST-RAM (64MB)
- Viking ArxCis-NV - DRAM + Bateria + Flash (16GB)
- Netlist NVvault NVDIMM - DRAM + Bateria + Flash (8GB)



Hardware — Promessas

UFABC

HP — The Machine (2014)

- Originalmente baseada em memristors
- Agora, baseada em DRAM e PCM
- Protótipo previsto para 2016
  - 320TB memória
  - 2500 CPU cores

Intel e Micron — 3D XPoint (2015)

Comparado à Flash NAND, promete melhorias de:

- Desempenho — 1000x
- Durabilidade — 1000x
- Densidade — 10x

Perspectivas de pesquisa em software

UFABC

Caso existisse uma máquina com MU hoje

- Sistema de arquivos em memória
- SO zerando memória antes de repassar para as aplicações
- Não aproveita as características das novas arquiteturas

Perspectivas de pesquisa

- Ciclo de vida das aplicações
- Sistemas de arquivos e memória
- Otimizações específicas para NVMs
- Persistência de dados
- Segurança

Ciclo de vida das aplicações

UFABC

- Diversas tarefas desempenhadas pelas aplicações não são intrínsecas ao seu funcionamento e são na verdade impostas pelas características do hardware
- *Serialization, marshalling e unmarshalling* são exemplos disso
- Em máquinas com MU a aplicação quando reexecutada, com suporte do SO, já vem com toda a memória "preenchida"

Reinicialização/Atualização de Sw.

UFABC

- Tradicionalmente feito com o ciclo: instalação $\rightarrow$ reinicialização
- Em máquinas tradicionais, o ciclo garante que dados de execução eventualmente corrompidos são zerados numa próxima execução
- Em um SO para máquinas com MU, o sistema precisa ser capaz de evitar e lidar com inconsistências. A memória não será reinicializada.

Sistemas de Arquivos

UFABC

- São realmente necessários?
- Sistemas de arquivo e memória $\rightarrow$ revisão e unificação

Acesso à Memória

Diferentes tecnologias $\rightarrow$ diferentes mecanismos de acesso

- Mapeados
  - RAM, registradores de controle de dispositivos, RAM de vídeo
  - Memória virtual e controle de acesso com suporte de hardware
- Baseados em drivers
  - Discos rígidos, SSDs (SATA e PCI), ...
  - Nível de indireção e controle de acesso por software

Acesso à Memória

UFABC

Paginação e arquivos mapeados em memória

- Possibilitam às aplicações extrapolar o limite do tamanho da memória RAM
- Trazem parte da vantagem oferecida por memórias não-voláteis
- Entretanto trazem um grande overhead por conta de múltiplas camadas

Acesso à Memória $\rightarrow$ Mem. Universal

UFABC

NVM grande o suficiente $\rightarrow$ ausência de discos

- Sistemas de paginação, tais como são atualmente, deixam de ser necessários
- Simplificação de mecanismos de acesso ao hardware
- Otimização do SO para as novas tecnologias

SW - Otimizações para NVMs

UFABC

Funcionamento peculiar de cada uma das tecnologias

- Flash
  - Não é tão rápida como DRAM, mas é muito maior
  - Baixa durabilidade
  - Dispositivos híbridos com NOR e NAND Flash
  - É possível minimizar os impactos
    - Evitando write-backs desnecessários (tipicamente dados na pilha)
    - Inicialização de variáveis x Durabilidade das células

SW - Otimizações específicas para NVMs

UFABC

PCM

- Tempo e custo energético para escrita e leitura diferentes (10x)
- Tempo variável de escrita se utiliza mais de um bit por célula
- Tempo e custo energético variável de escrita dependendo da durabilidade desejada

PCM - Possíveis otimizações

- *Differential Write*
- Computação aproximada
- Ballooning

PCM - Ballooning

Ideia originalmente usada por VMs

- Guest OS - Balloon driver aloca memória (infla o balão) e força a paginação para disco
- Host OS - Analisa páginas que foram para disco e usa aquele espaço de memória para outras VMs

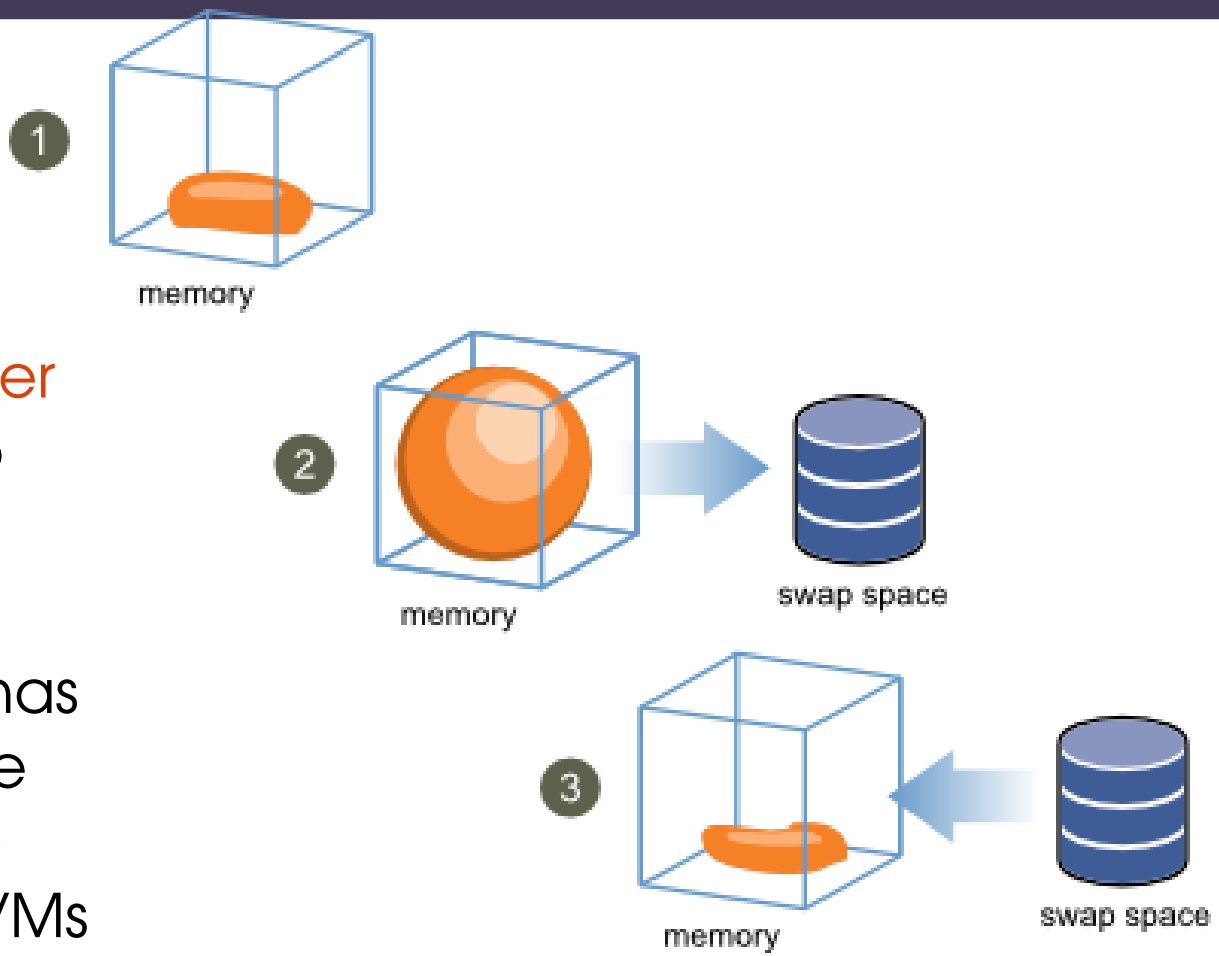


Figura: vSphere Resource Management Guide

PCM - Ballooning

PCM - Cada célula é capaz de guardar até 4 bits

- Trade-off entre velocidade e capacidade de armazenamento
- Sistema começa a execução com uma capacidade de memória e pode chegar até 4x maior
  - Graceful performance degradation
  - Ainda assim, muito mais rápido do que paginação em disco!

Segurança

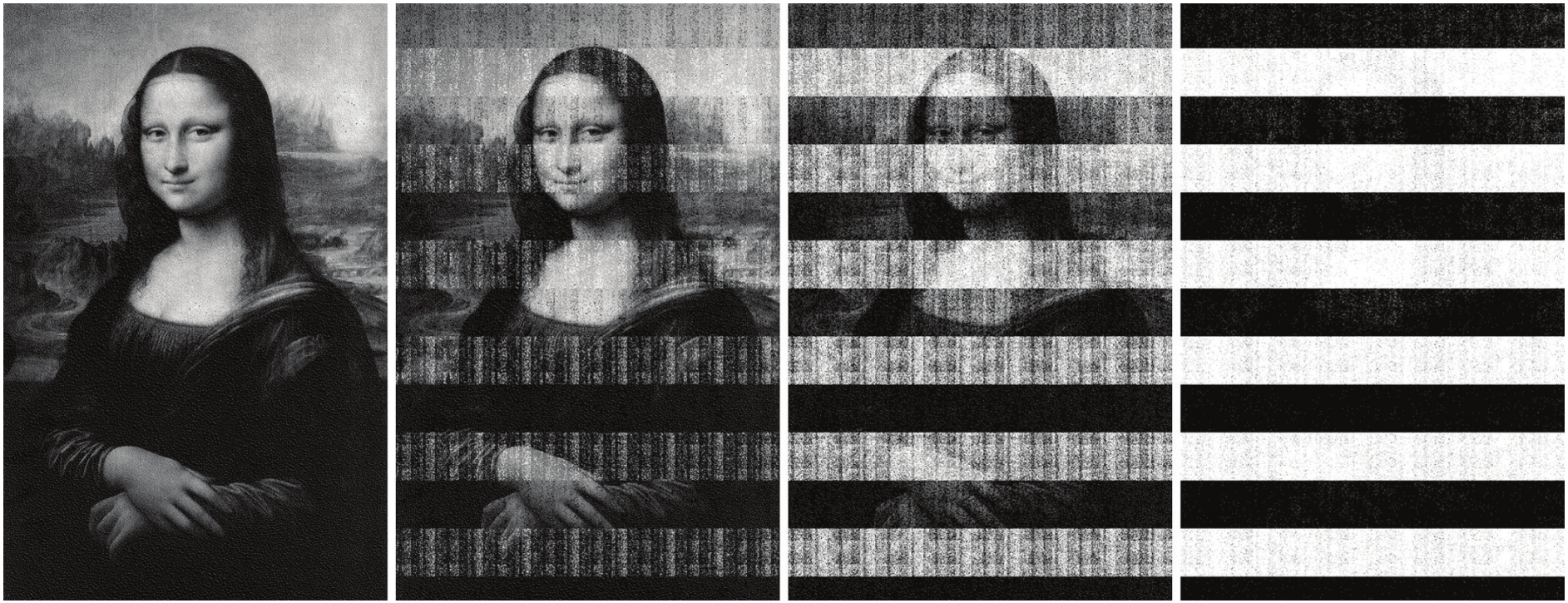
UFABC

- Informações sigilosas tais como chaves criptográficas e senhas são armazenados na RAM
- Seguro, já que o acesso ao espaço de memória é controlado e a memória é volátil

- Informações sigilosas tais como chaves criptográficas e senhas são armazenados na RAM
- Seguro, já que o acesso ao espaço de memória é controlado e a memória é volátil

Será mesmo?

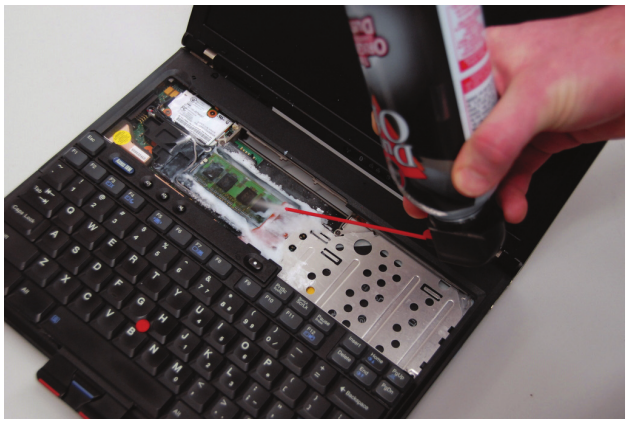
Segurança



- Degradação após 5s, 30s, 60s e 5 minutos
- Os pesquisadores foram capazes de obter chaves criptográficas da memória e ler dados criptografados no disco

J. A. Halderman, et al. Lest we remember: cold-boot attacks on encryption keys. *Communications of the ACM 52, 5 (May 2009), 91-98.*

Segurança



- Resfriando as memórias com um tubo de ar comprimido
  - Após 6 minutos 0% de erros
  - Após 10 minutos 0.000036% de erros

J. A. Halderman, et al. Lest we remember: cold-boot attacks on encryption keys. *Communications of the ACM 52, 5 (May 2009), 91-98.*

UFABC

NVM

- Arquiteturas baseadas em NVM precisam dar uma atenção ainda maior a esses tipos de ataques
- Problemas semelhantes ao processamento de dados sigilosos em ambientes não seguros, e.g., cloud
- Utilizar o suporte de hardware (se disponível) para regiões de memória que sao mantidas nas caches e nunca vão para RAM. Alguns trabalhos preliminares lançam mão de
  - Computação apenas nos registradores
  - Scratchpad memory

Persistência de dados

UFABC

- Controle de acesso concorrente.
- Dados na memória principal precisam ser consistentes
- Processos veem apenas modificações "completas"
- Em memórias voláteis utiliza-se locks/semáforos/memória transacional para garantir tudo isso
- A adição da característica de durabilidade por NVMs, torna tudo mais interessante

Persistência de dados - Falhas

Não há garantias de que o dado tenha sido escrito na RAM

- Pode estar nas caches
- Pode estar no *write-buffer* do processador
- Pode estar no buffer do controlador de memória

Também não há garantia implícita da ordem ou atomicidade das operações

- Escritas incompletas
- Inversão da ordem correta das operações

UFABC

Barreiras de memória (*memory fences*) e `CLFLUSH`

- Barreiras garantem que todas as instruções de escrita até então não serão executadas após novas instruções do mesmo tipo
- Instrução `CLFLUSH`
  - Força escrita da linha de cache (suja ou não) para a memória
  - Como efeito colateral também invalida a linha
  - Não garante esvaziamento do *write-buffer*
- Usando-se esses dois mecanismos juntos é possível ter algumas garantias em troca de desempenho
- Ainda não há controle sobre o subsistema de memória

Persistência de dados

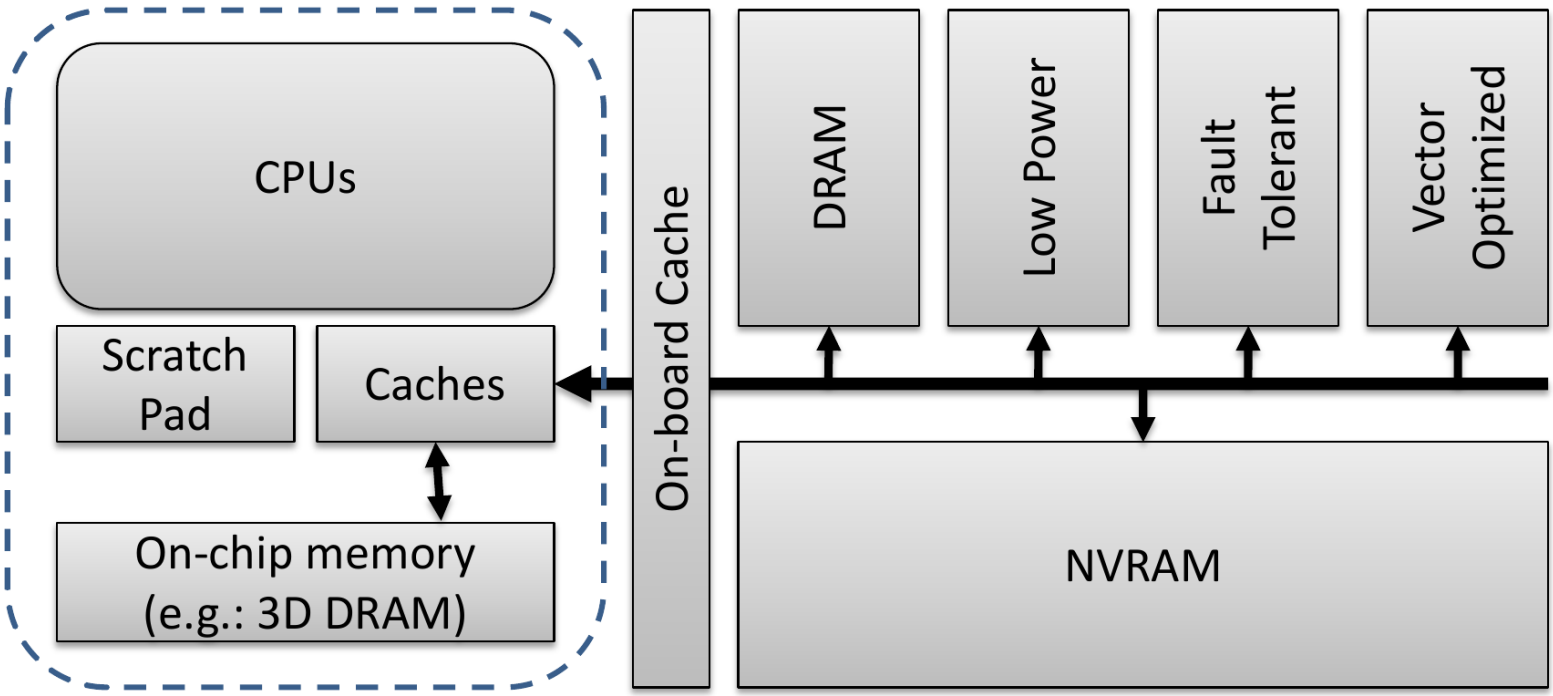
Em outubro de 2014 a Intel anunciou* novas instruções

- `CLFLUSHOPT` — uma versão otimizada do `CLFLUSH` que apenas faz a escrita na memória se a linha estiver suja, mas ainda a invalida
- `CLWB` — semelhante à `CLFLUSHOPT` porém apenas marca a linha como limpa
  - Minimiza os cache-misses compulsórios associados às instruções `CLFLUSH` e `CLFLUSHOPT`
- `PCOMMIT` — força o *flush* de quaisquer *write buffers* ou caches do subsistema de memória

** Intel Architecture Instruction Set Extensions Programming Reference, Chapter 11 "Memory Instructions". October 2014*

Modelos de programação

Forte tendência que as primeiras máquinas sejam híbridas



.

A. J. Pena and P. Balaji. Toward the efficient use of multiple explicitly managed memory subsystems. *IEEE International Conference on Cluster Computing (CLUSTER), 2014*

Memória transacional

UFABC

- Ambientes como NVHeaps e Mnemosyne oferecem mecanismos transacionais para a modificação de memória
- Alguns trabalhos mostram, como aproveitar as instruções transacionais de hardware para acelerar mecanismos de persistência baseados em memória
- NVHeaps trata, inclusive, de maneira diferente ponteiros para dados voláteis ou permanentes. Um ponteiro partindo de uma região não volátil para uma volátil é tratado como erro de compilação

Ambientes promissores

- Pmem.io
  - Desenvolvido pela Intel
    ↑ Rico conjunto de APIs para a utilização de NVMs
    ↓ As APIs poderosas são de baixíssimo nível e as demais têm importantes limitações
- Atlas
  - Desenvolvido pela HP Labs
    ↑ API simples e funcional
    ↓ Oferece apenas operações básicas se comparado ao Pmem.io

Já estamos prontos?

UFABC

- Cada vez mais tecnologias de hardware estão surgindo
- Ainda há muito a fazer...
  - Sistemas operacionais
  - Aplicações
  - Ferramentas de desenvolvimento
- Momento especialmente interessante para pesquisar nesta área