STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

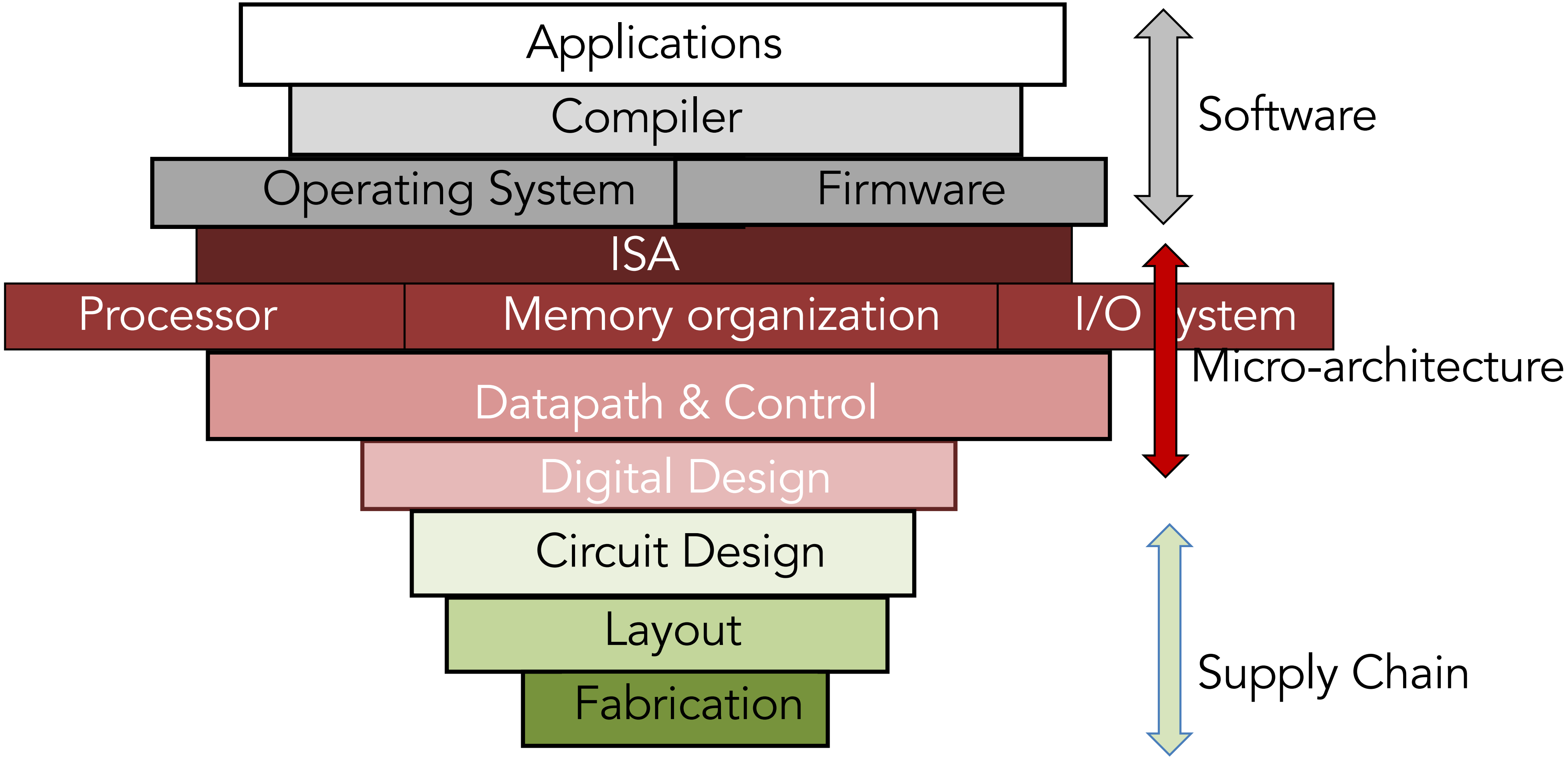
Ira A. Fulton Schools of Engineering
Arizona State University

CSE/CEN 598
Hardware Security & Trust

Information Channels, Covert Channels, & Side Channels

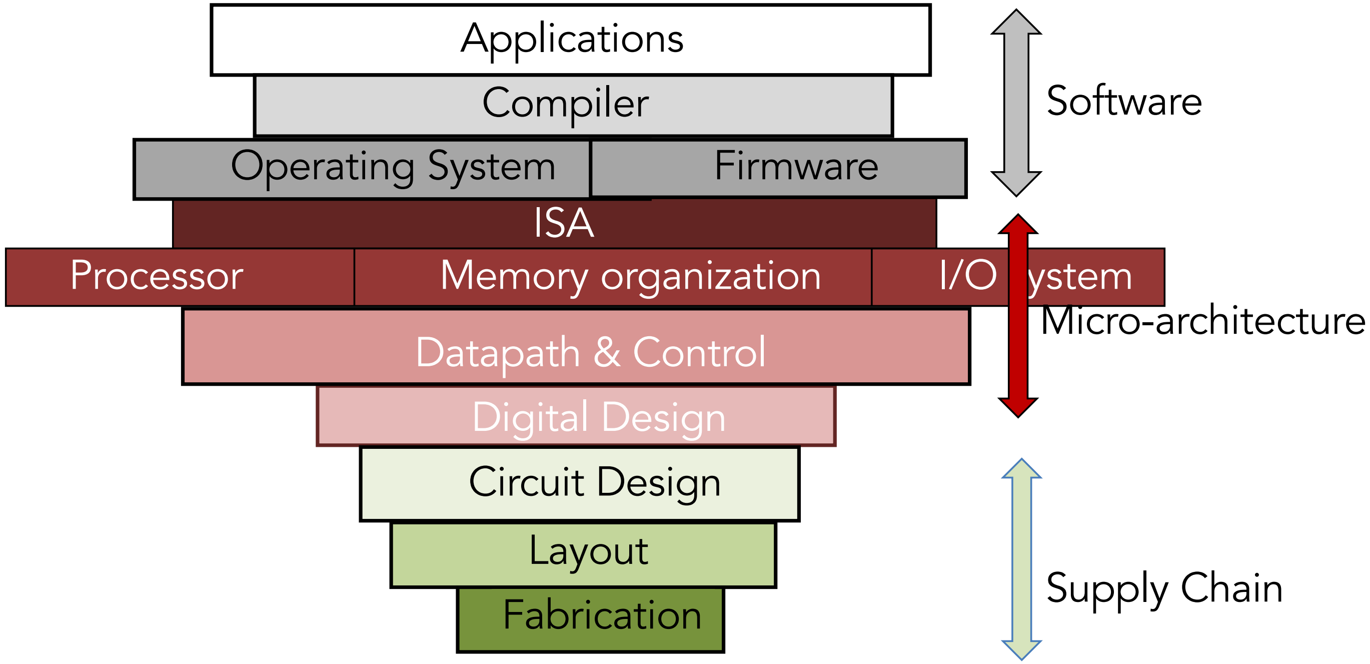
Prof. Michel A. Kinsy

Secure Computing System Patterns



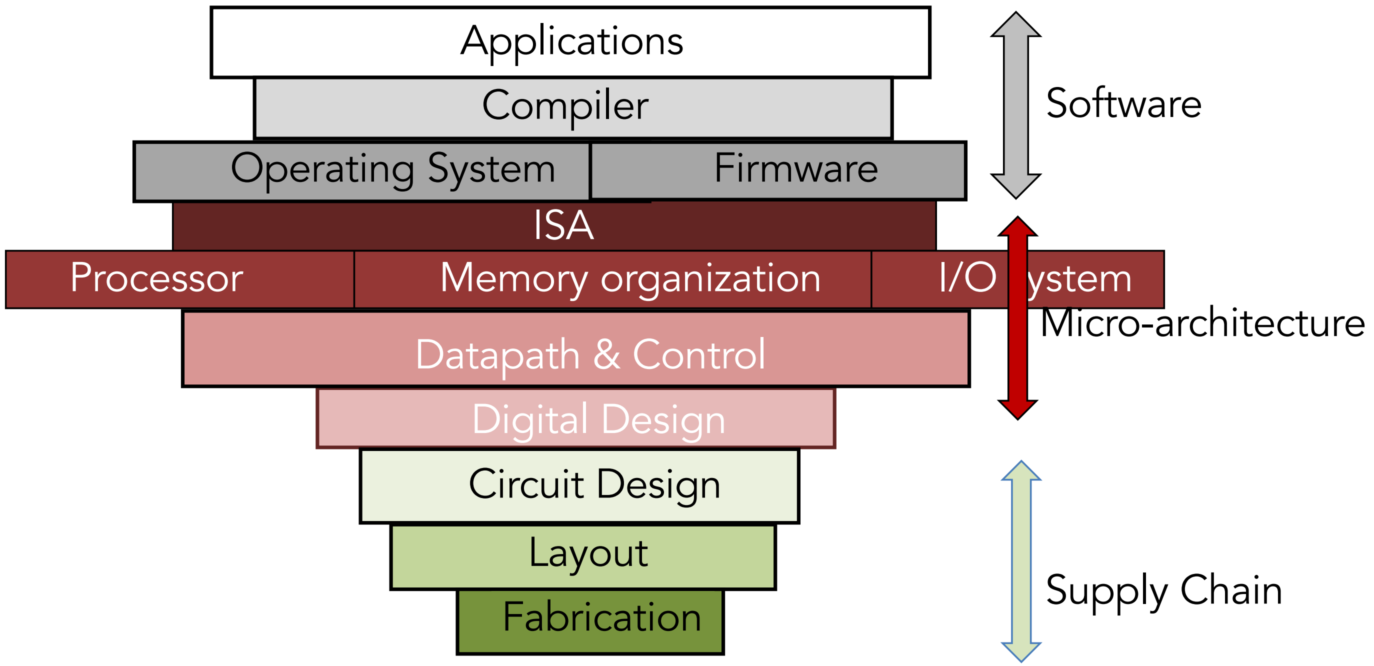
Computing System Security Patterns

- Patterns
  - Procedural
    - Confidentiality
    - Integrity
    - Availability
    - Nonrepudiation
    - Authentication
    - Authorization
    - Authentication and authorization are complementary
  - Algorithmic
    - Public-Key encryption
    - Key exchange protocols
    - Hash functions
    - Digital signatures
  - Structural
    - Software and hardware
    - Storage
    - Communication
    - Virtualization

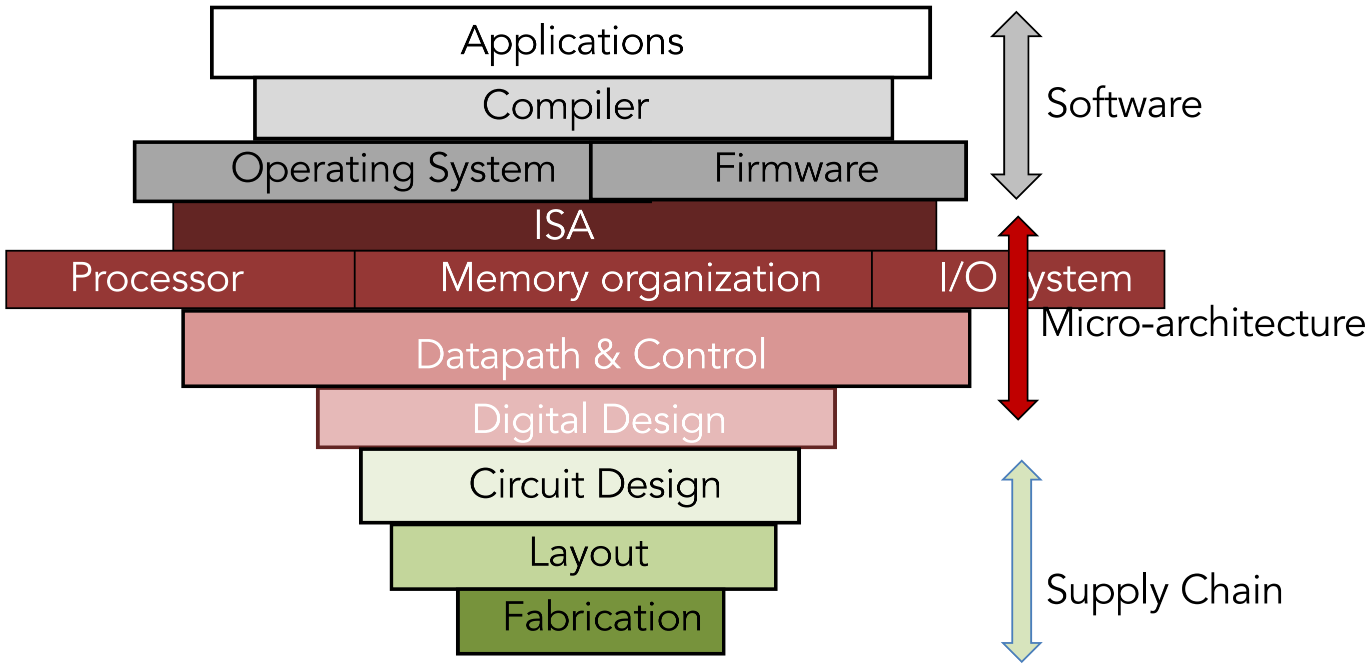


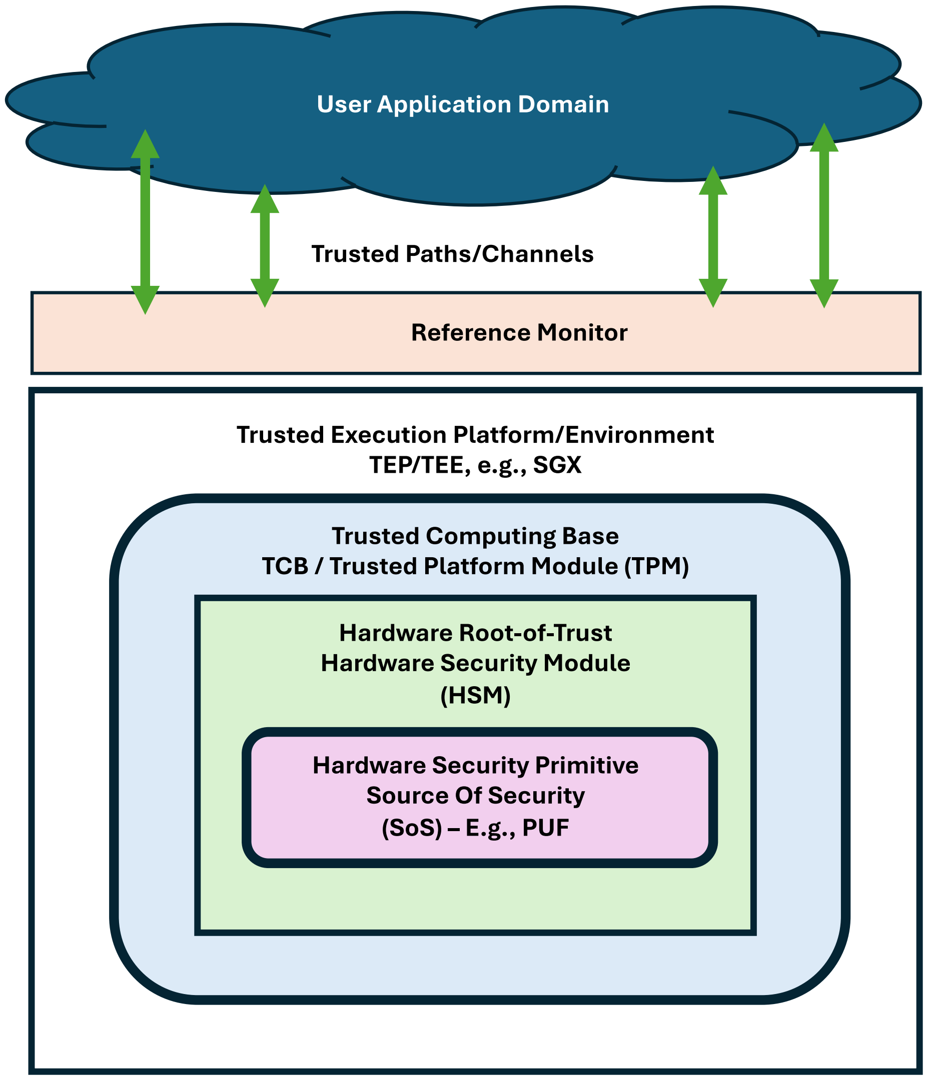
Computing System Security Patterns

- Patterns
  - Procedural
    - Confidentiality
    - Integrity
    - Availability
    - Nonrepudiation
    - Authentication
    - Authorization
    - Authentication and authorization are complementary
  - Algorithmic
    - Public-Key encryption
    - Key exchange protocols
    - Hash functions
    - Digital signatures
  - Structural
    - Software and hardware
    - Storage
    - Communication
    - Virtualization



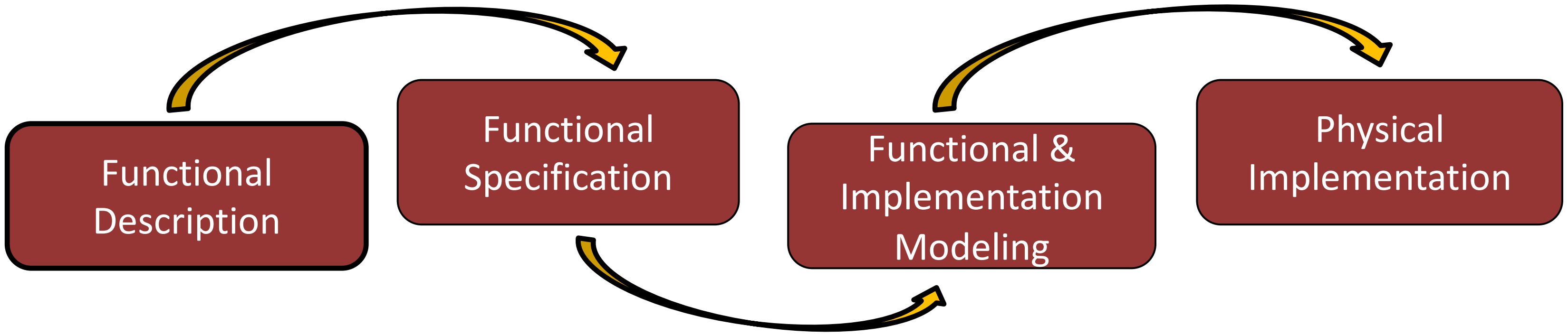
Secure Computing System Patterns





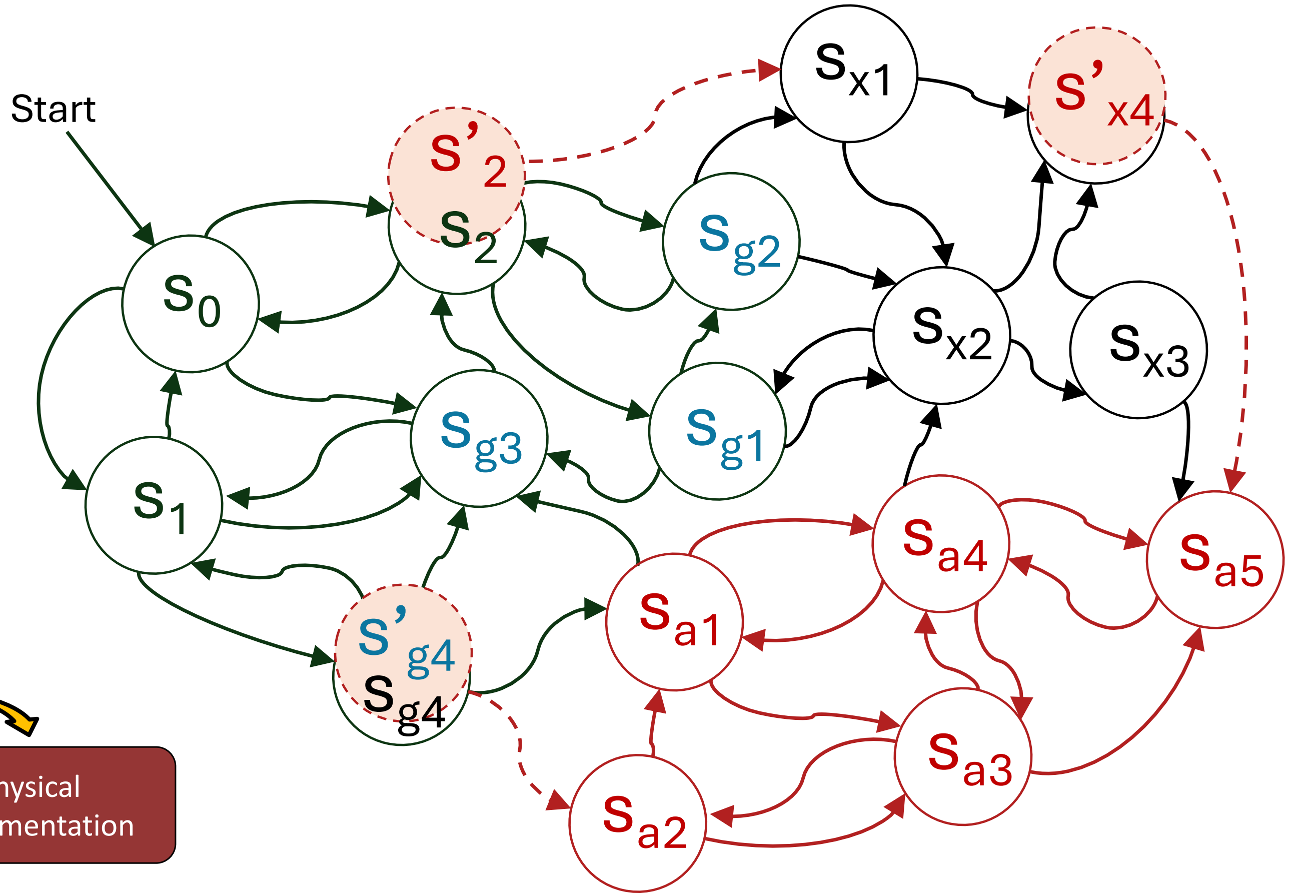
Functional to Design Implementation

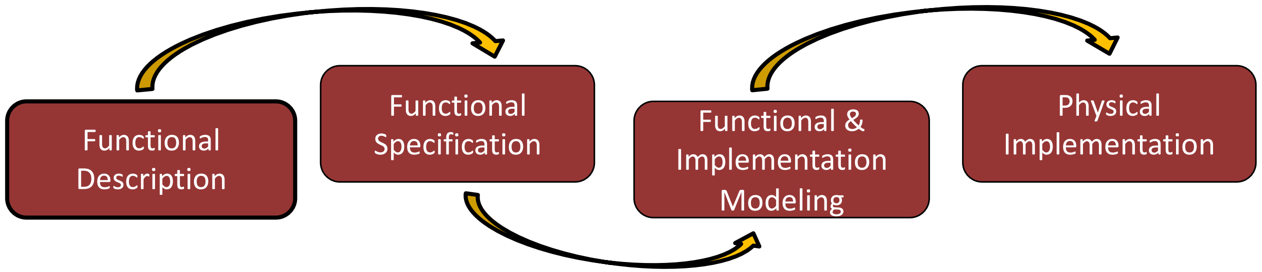
- From functional description to the physical design implementation
  - Functional correctness
  - Including security features
  - Reliability issues
  - Side-channels



Functional to Design Implementation

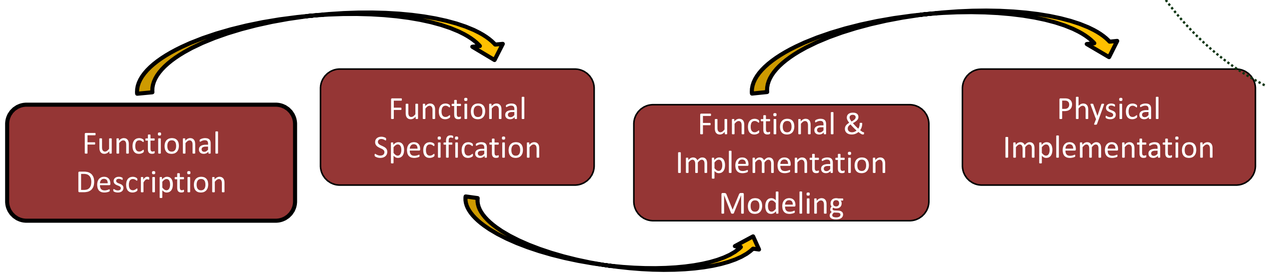
- From functional description to the physical design implementation
  - Functional correctness
  - Including security features
  - Reliability issues
  - Side-channels

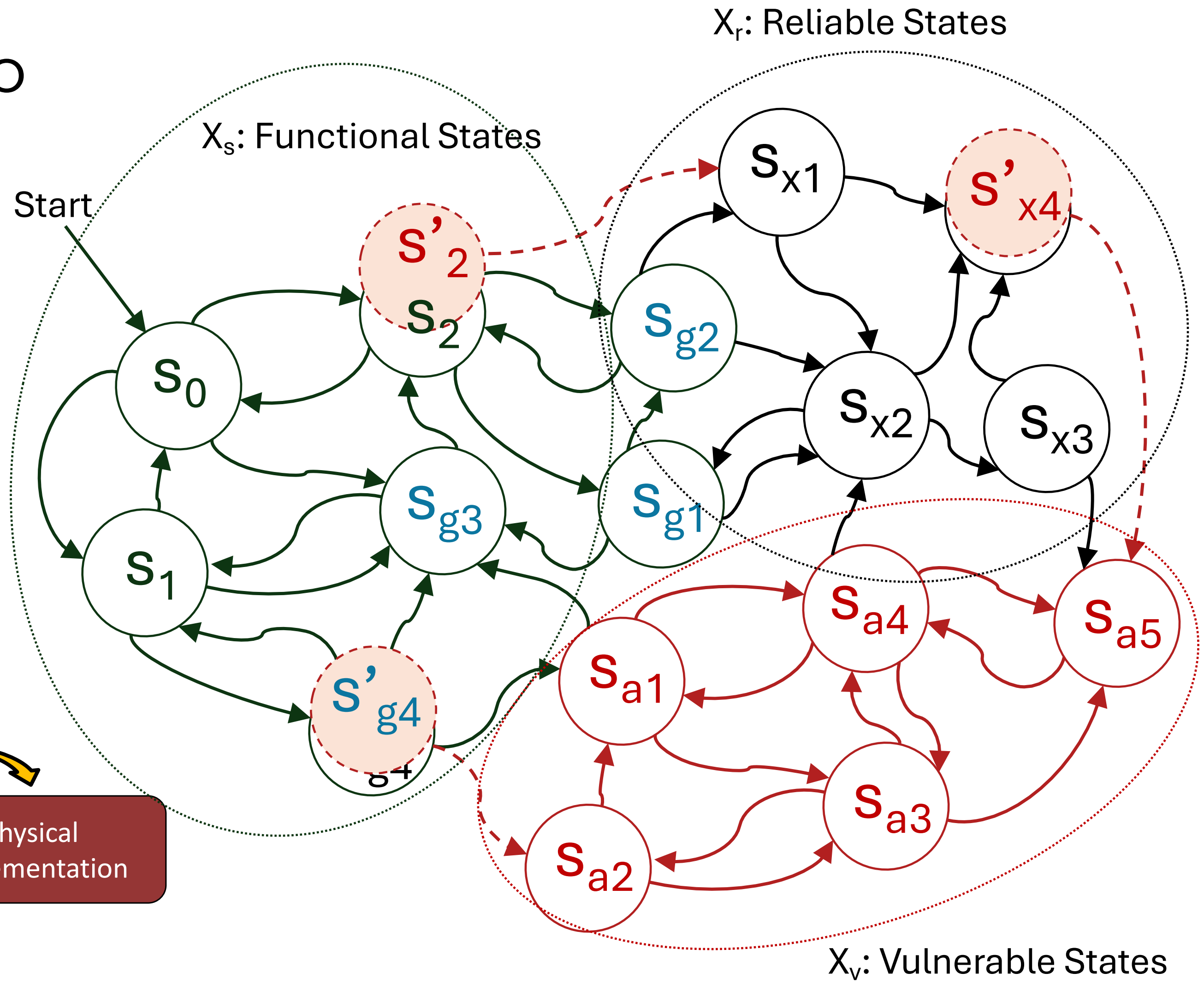




Functional to Design Implementation

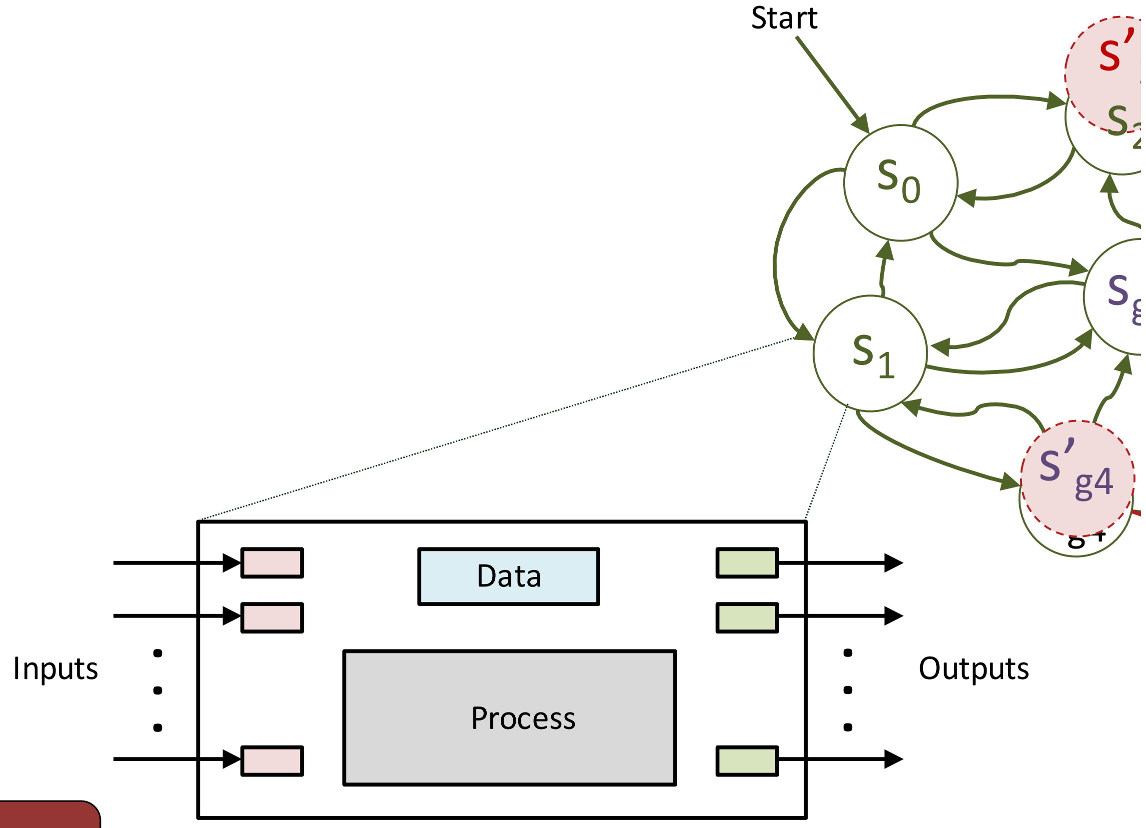
- From functional description to the physical design implementation
  - Functional correctness
  - Including security features
  - Reliability issues
  - Side-channels

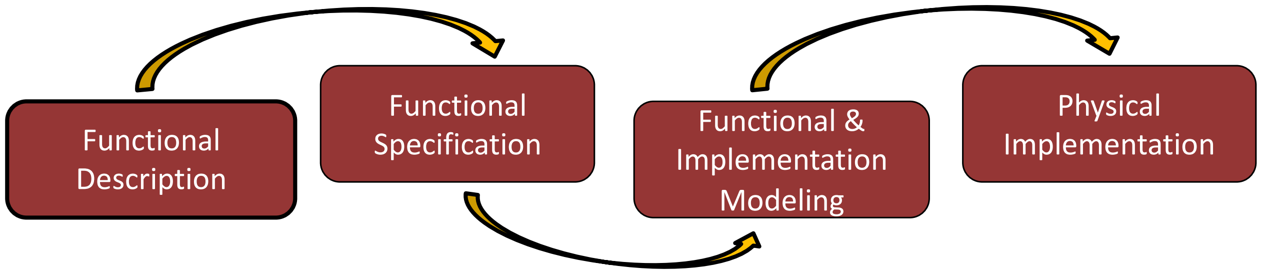




Functional to Design Implementation

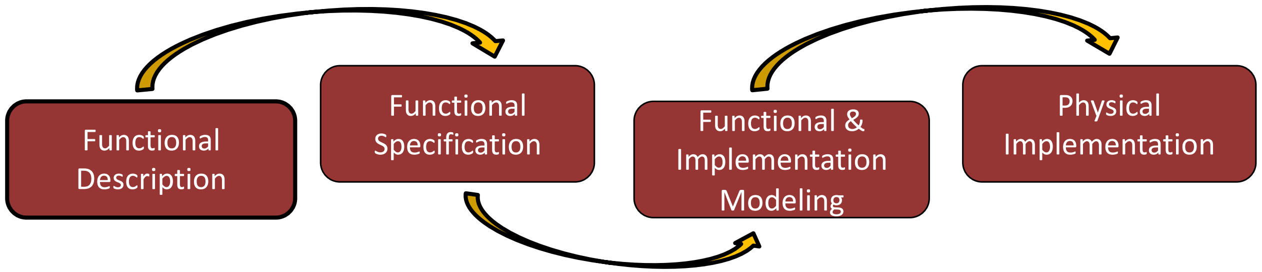
- From functional description to the physical design implementation
  - Functional correctness
  - Including security features
  - Reliability issues
  - Side-channels

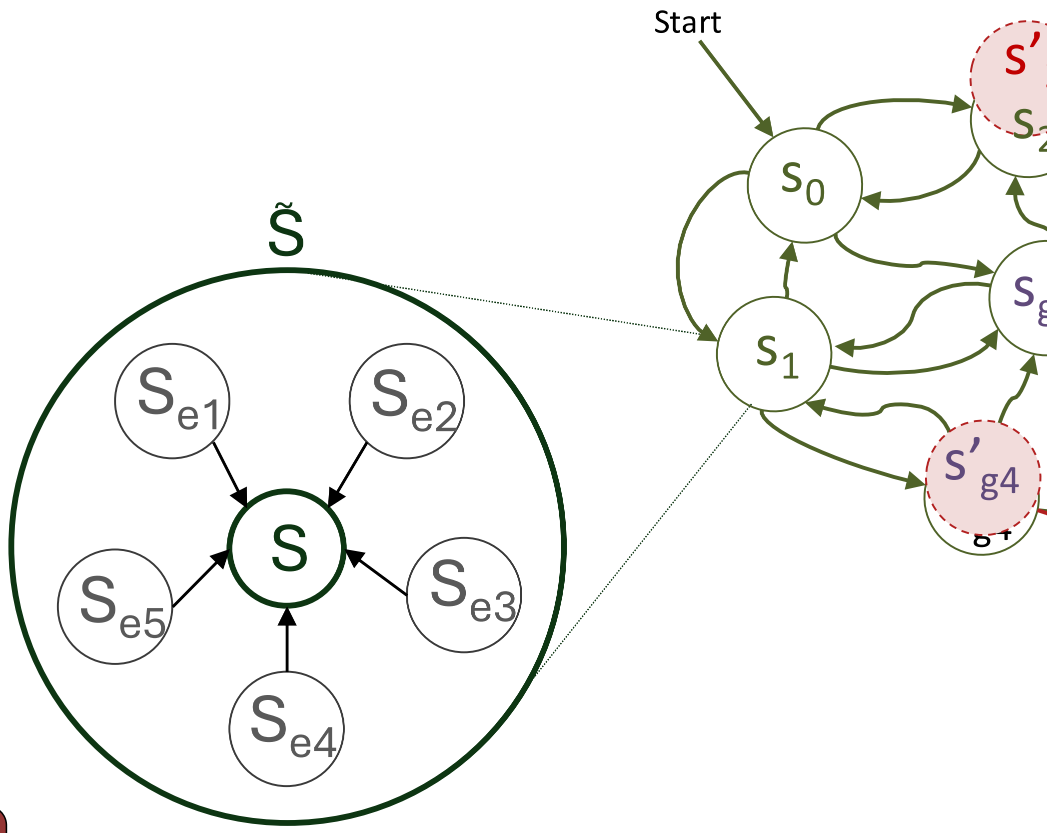




Functional to Design Implementation

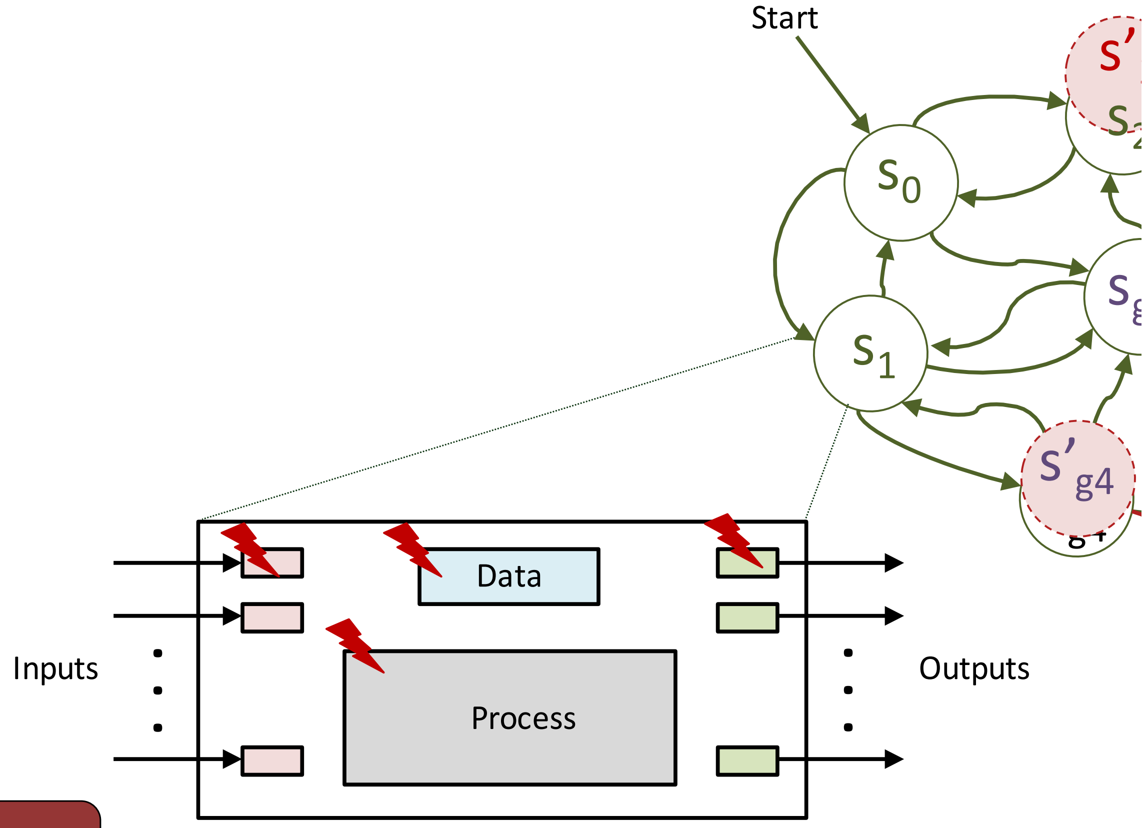
- From functional description to the physical design implementation
  - Functional correctness
  - Including security features
  - Reliability issues
  - Side-channels

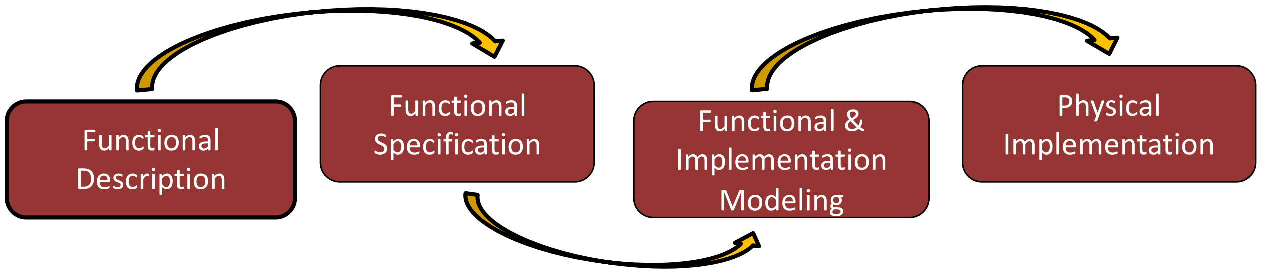




Functional to Design Implementation

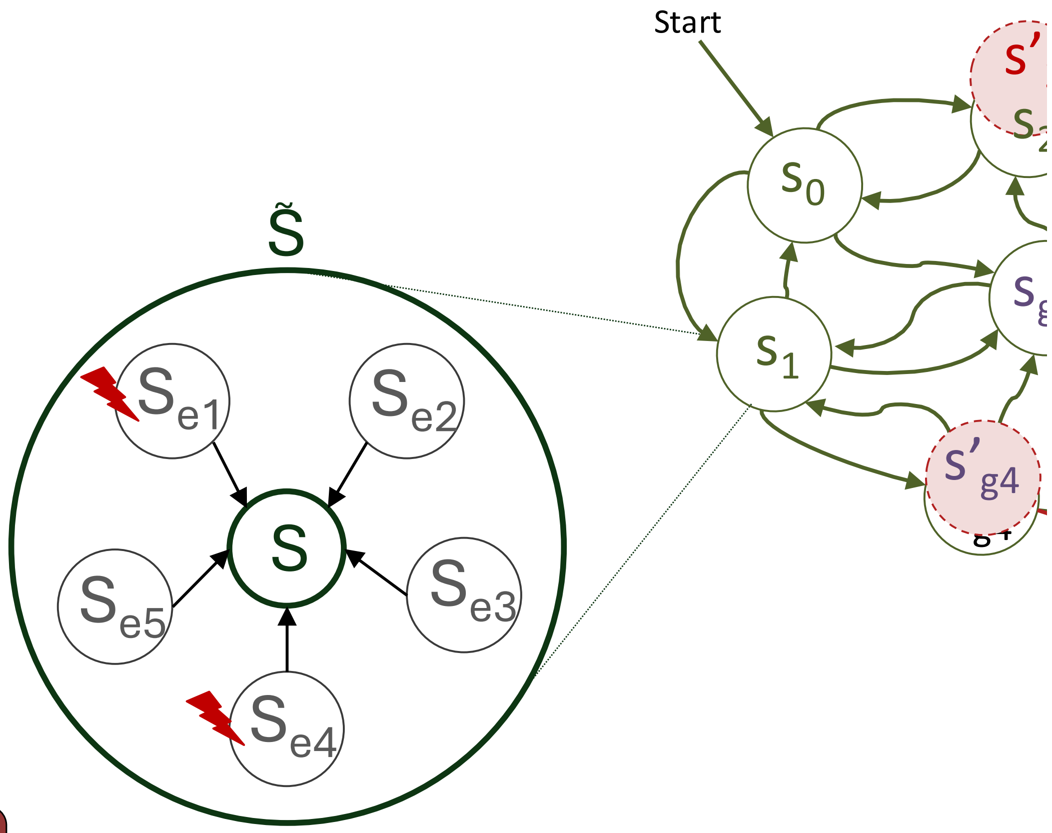
- From functional description to the physical design implementation
  - Functional correctness
  - Including security features
  - Reliability issues
  - Side-channels

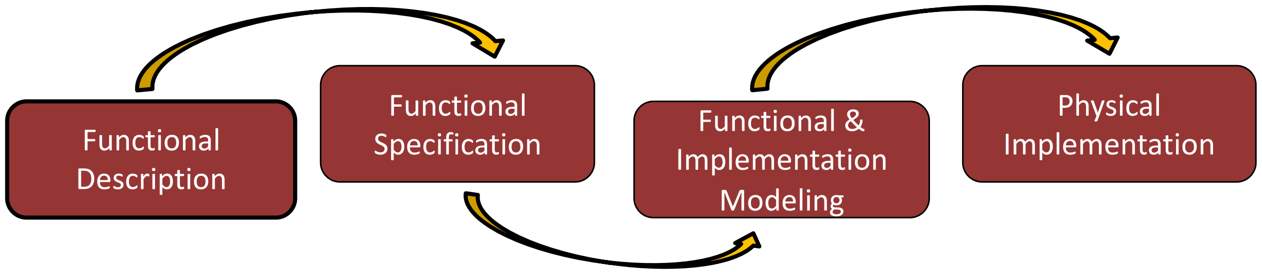




Functional to Design Implementation

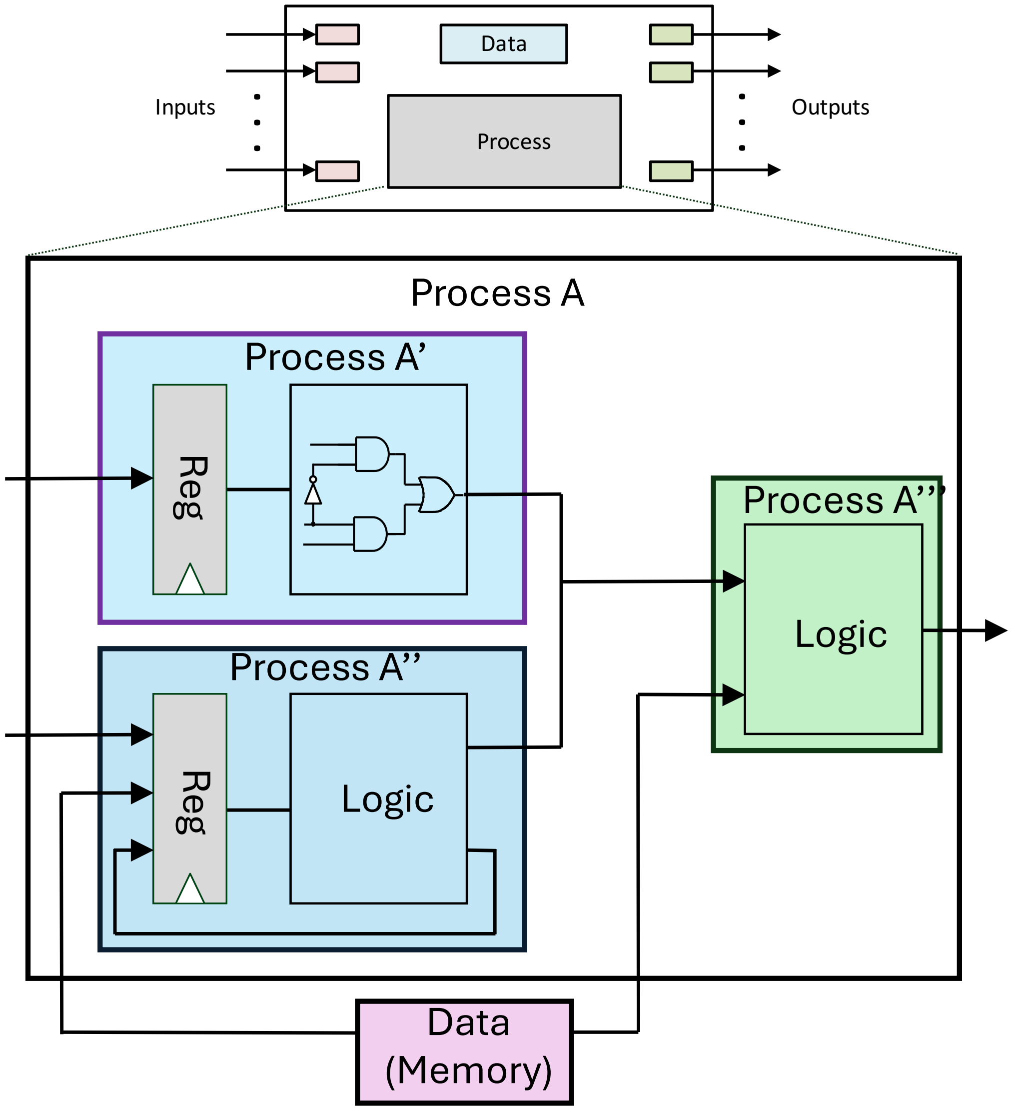
- From functional description to the physical design implementation
  - Functional correctness
  - Including security features
  - Reliability issues
  - Side-channels

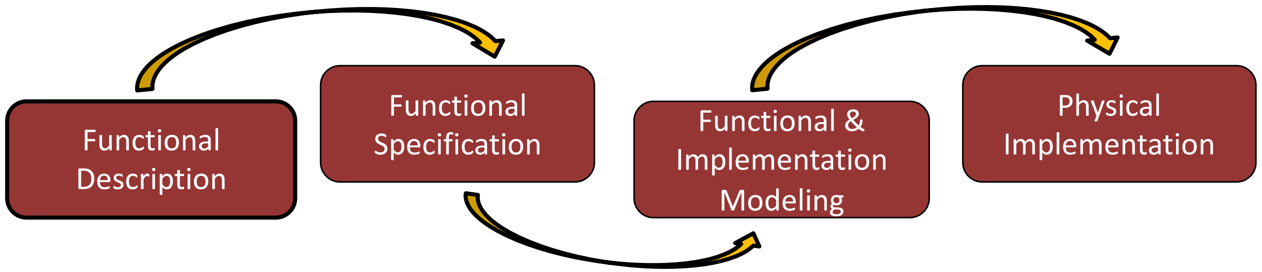




Functional to Design Implementation

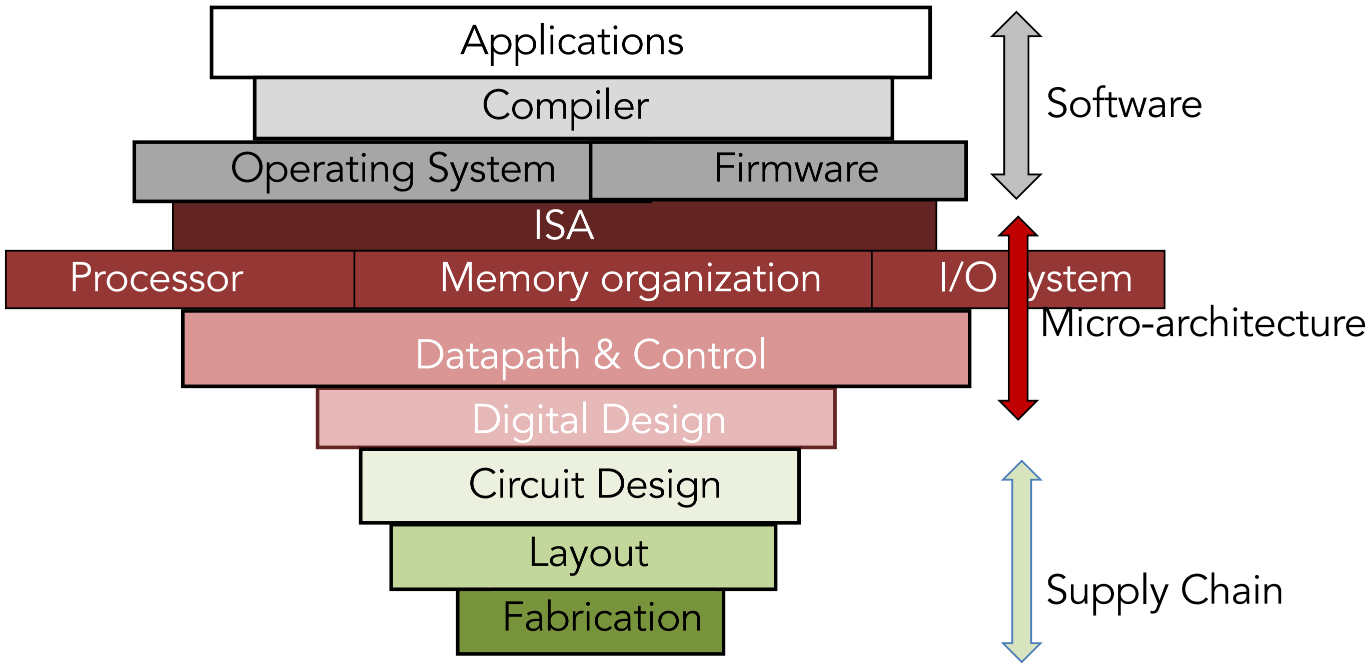
- From functional description to the physical design implementation
  - Functional correctness
  - Including security features
  - Reliability issues
  - Side-channels





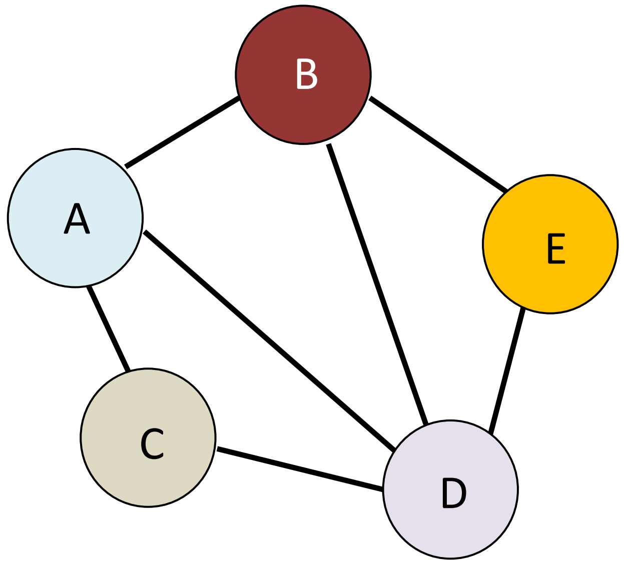
Computing System Security Patterns

- Patterns
  - Procedural
    - Confidentiality
    - Integrity
    - Availability
    - Nonrepudiation
    - Authentication
    - Authorization
    - Authentication and authorization are complementary
  - Algorithmic
    - Public-Key encryption
    - Key exchange protocols
    - Hash functions
    - Digital signatures
  - Structural
    - Software and hardware
    - Storage
    - Communication
    - Virtualization



Information Channels

- Information flow is an intrinsic part of computing both in a single computer system and in a distributed system
  - Information processing
    - Data in manipulation
  - Information communication
    - Data in motion
  - Information storage
    - Data at rest
  - Information sharing
    - Data sharing
- There are intended, specified, implemented/ established channels for the flow of information
  - With all the required security guarantees
    - Integrity, availability, nonrepudiation, authentication, etc.



Covert Channels

- A covert channel is a path for an illegal flow of information within a system
- Any communication channel that can be exploited by a process to transfer information in a manner that violates the system's security policy
  - National Institute of Standards and Technology

Covert Channels

- A covert channel is a path for an illegal flow of information within a system
- Any communication channel that can be exploited by a process to transfer information in a manner that violates the system's security policy
  - National Institute of Standards and Technology
- There are many types of covert channels within a computing system:
  - Timing covert channels
    - Methods to extract how much time a computation or a computational task takes?
  - Termination covert channels
    - Methods to detect if a computation terminates?
  - Probability covert channels
    - Methods to determine what the distribution of certain system events is? What control path does the program take?
  - Resource utilization covert channels
    - Approaches to establish some resource utilization level or if the resource is depleted?
  - Power covert channels
    - Method to determine the amount energy consumed or required by a computational task?

Covert Channels

- There are many usage of covert channels to both improve and undermine the security of a computing system
  - Exfiltrate data from an otherwise secure system
  - Avoid detection of unauthorized access
  - Install, spread, or control malware on compromised systems
  - Circumvent content or resource filters
  - Bypass firewalls for unrestricted access
  - Malware authors use timing to detect analysis sandboxes

Covert Channels

- Important characteristics of a covert channel
- Existence
  - Is a channel present?
- Bandwidth
  - How much information can be transmitted?
- Noiseless/noisy
  - Can the information be transmitted without loss or distortion?
- Main example of covert channels
  - Covert storage channel
  - Covert timing channel

Covert Channels

- Main example of covert channels
  - Covert storage channel
    - Conditions
      - For actors A (sender) and B (receiver) to use a covert storage channel
        1. Both A and B must have access to some attribute/state of a shared storage
        2. A must be able to change the storage state
        3. B must be able to reference or view the modified storage state.
        4. Mechanism for initiating both processes, and sequencing their accesses to the shared storage must be present
        - **Note: A and B could the same actor**
  - Covert timing channel

Covert Channels

- Main example of covert channels
  - Covert storage channel
  - Covert timing channel
    - Conditions
      - For actors A (sender) and B (receiver) to use a covert timing channel
        1. Both A and B must have access to some attribute/state of a shared object
        2. Both A and B have access to a time reference (real-time clock, timer, ordering of events)
        3. A must be able to control the timing of the detection of a change in the attribute/state of B
        4. Mechanism for initiating both processes, and sequencing their accesses to the shared resource
        - **Note: A and B could the same actor.**
  - Hardware Trojans
    - To be covered later in a couple lectures

Covert Channels

- It is usually infeasible for realistic systems to eliminate every potential covert channel
- Mitigation techniques for covert channels
  - Eliminate or minimize it by modifying or refining the system implementation
  - Reduce potential covert channel bandwidth through noise injection into the channel
  - Monitor it for patterns of usage that indicate potential exploitation
    - E.g., Intrusion detection

Side Channels

- Unintended conduit/channel to gain information about the state or operation about the computing system
  - Power usage
  - Electromagnetic radiation
  - Execution time
  - Timing behavior
  - Optical characteristics
  - Traffic analysis
  - Acoustic effects
  - Thermal signature

Side Channels

- Side channel: Unintended conduit to gain information about the state or operation about the computing system
  - Power utilization profile
  - Timing behavior
  - Acoustic characteristics
  - Fault patterns
- Side channel attack (SCA): attacks based on the analysis and exploitation of side channel information
  - Cache attack: attacks based on attacker's ability to monitor cache accesses made by the processing element
  - Fault injection attacks
  - Power analysis attacks
  - …

M. Backes, M. Dürmuth, S. Gerling, M. Pinkal, and C. Sporleder, "Acoustic side-channel attacks on printers." in USENIX Security sym- posium, 2010, pp. 307–322

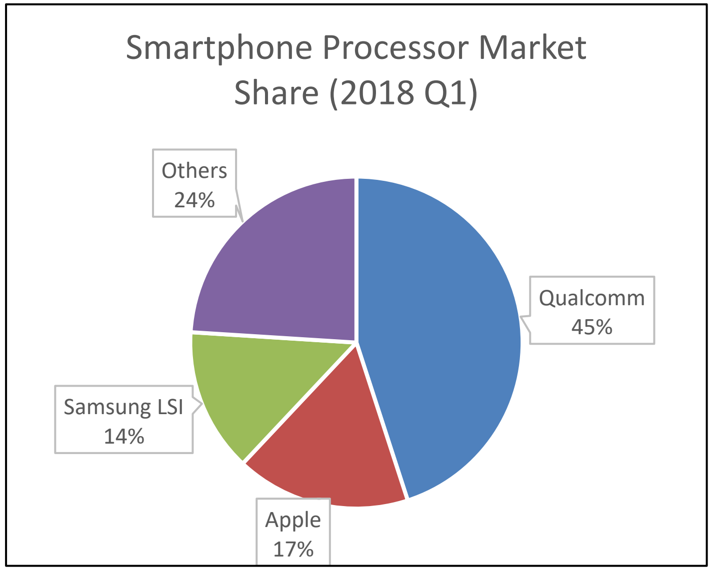
M. Hutter and J.-M. Schmidt, "The temperature side channel and heating fault attacks," in International Conference on Smart Card Research and Advanced Applications. Springer, 2013, pp. 219–235

Side Channels

- Side-channel attacks are current and real threats

Security flaw lets attackers recover private keys from Qualcomm chips

Firmware patches have been released earlier this month, 46 Qualcomm chipsets impacted.



[1] https://www.businesswire.com/news/home/20180808005464/en/Strategy-Analytics-Q1-2018-Smartphone-Apps-Processor

Side Channels

- Circumvent security measures
  - Qualcomm Secure Execution Environment (QSEE)
    - Hardware-isolated execution
    - Leaks private data, encryption keys, etc.
  - A cache side-channel is used to retrieve sensitive information

Intel CPUs impacted by new Zombieload side-channel attack

Researchers, academics detail new Microarchitectural Data Sampling (MDS) attacks.

Side Channels

- Exploits speculative execution
- Steals information from SGX secure enclave
- Extracts the machine's private attestation key

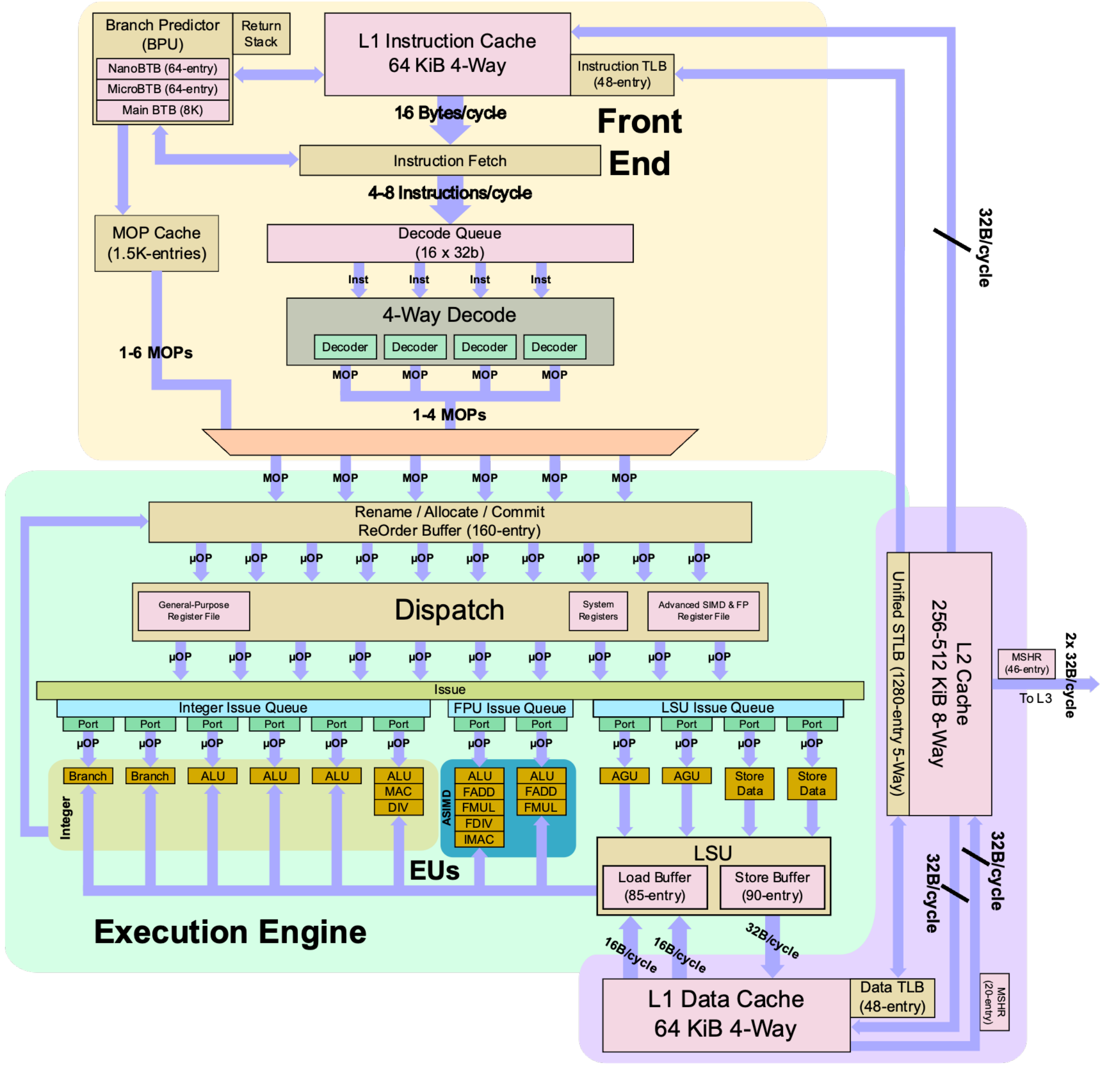
LILY HAY NEWMAN SECURITY 08.14.18 01:00 PM

SPECTRE-LIKE FLAW UNDERMINES INTEL PROCESSORS' MOST SECURE ELEMENT

FORESHADOW

WIRED (https://www.wired.com/story/foreshadow-intel-secure-enclave-vulnerability/)

Modern Out-of-Order Superscalar Processor



https://en.wikichip.org/wiki/File:cortex-a77_block_diagram.svg

- Branch prediction
  - Pipeline depth increased with time (14, 20 stages, and more)
  - Out-of-Order pipeline might be able to execute 100's of independent instructions following a branch till a resolution is obtained

Branch →

```
     lw    a5,296(zero) # 128 <a>
     bge   a5,zero,dc
     sw    zero,304(zero) # 130 <y>
     jal   zero,e4
dc:  sw    a5,304(zero) # 130 <y>
e4:  addi  zero,zero,0
```

Side Channels: Microarchitectural Features

- Speculation
  - Branch prediction
  - Speculative execution
  - Out-of-order execution
  - Return Stack Buffer (RSB)
- Shared resources
  - Caches
  - Translation Lookaside Buffer (TLB)
  - Cache coherence
  - Multi-threading

Microarchitecture Side Channels

- Spectre
  - Exploits speculative execution
  - Mistrains the branch predictor
  - Breaks isolation between different applications
- Meltdown
  - Exploits out-of-order execution
  - Allows an unprivileged application to read kernel memory

- P. Kocher, J. Horn, A. Fogh, , D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," in 40th IEEE Symposium on Security and Privacy (S&P'19), 2019.
- M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Melt-down: Reading kernel memory from user space," in 27th USENIX Security Symposium (USENIX Security 18), 2018.

Side-Channel Attacks Evolution

Variant	Description	CVE	Codename	Affected CPUs	More info
Variant 1	Bounds check bypass	CVE-2017-5753	Spectre v1	Intel, AMD, ARM	Website
Variant 1.1	Bounds check bypass on stores	CVE-2018-3693	Spectre 1.1	Intel, AMD, ARM	Paper
Variant 1.2	Read-only protection bypass	CVE unknown	Spectre 1.2	Intel, AMD, ARM	Paper
Variant 2	Branch target injection	CVE-2017-5715	Spectre v2	Intel, AMD, ARM	Website
Variant 3	Rogue data cache load	CVE-2017-5754	Meltdown	Intel, ARM	Website
Variant 3a	Rogue system register read	CVE-2018-3640	-	Intel, AMD, ARM, IBM	Mitre
Variant 4	Speculative store bypass	CVE-2018-3639	SpectreNG	Intel, AMD, ARM, IBM	Microsoft blog post
-	Return Mispredict	-	SpectreRSB	Intel, AMD, ARM	Paper

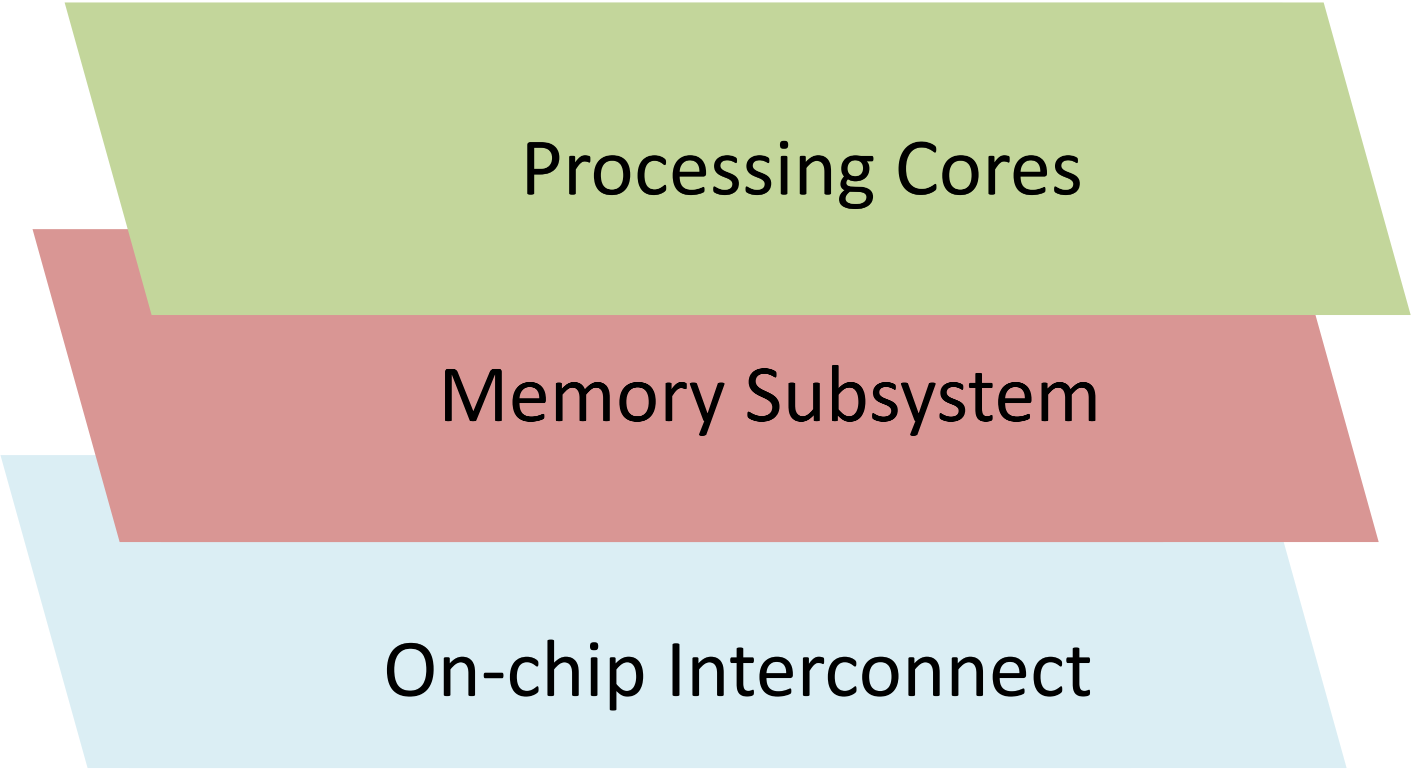
https://www.bleepingcomputer.com/news/security/researchers-detail-new-cpu-side-channel-attack-named-spectrersb/

Side Channels: Physical Access

Physical access required	Physical access not required
Power analysis	Timing side-channels
Electromagnetic side-channels	Traffic analysis
Optical side-channels	
Acoustic side-channels	
Thermal side-channels	

- Most side-channel attacks require physical access to the system
- Cache side-channels are based on timing

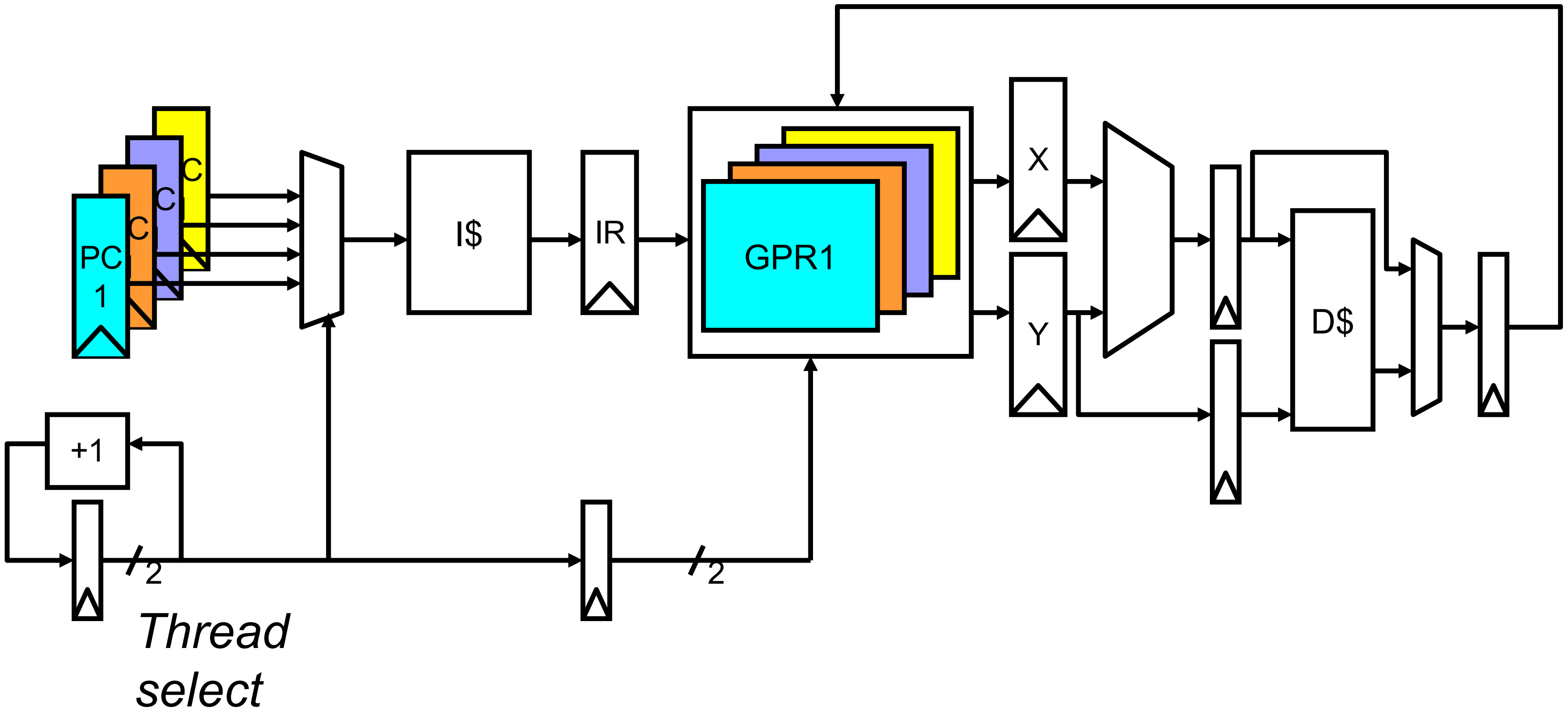
Modern Computer Architecture Layers



- Design to support optimal data manipulations, storage, and movements

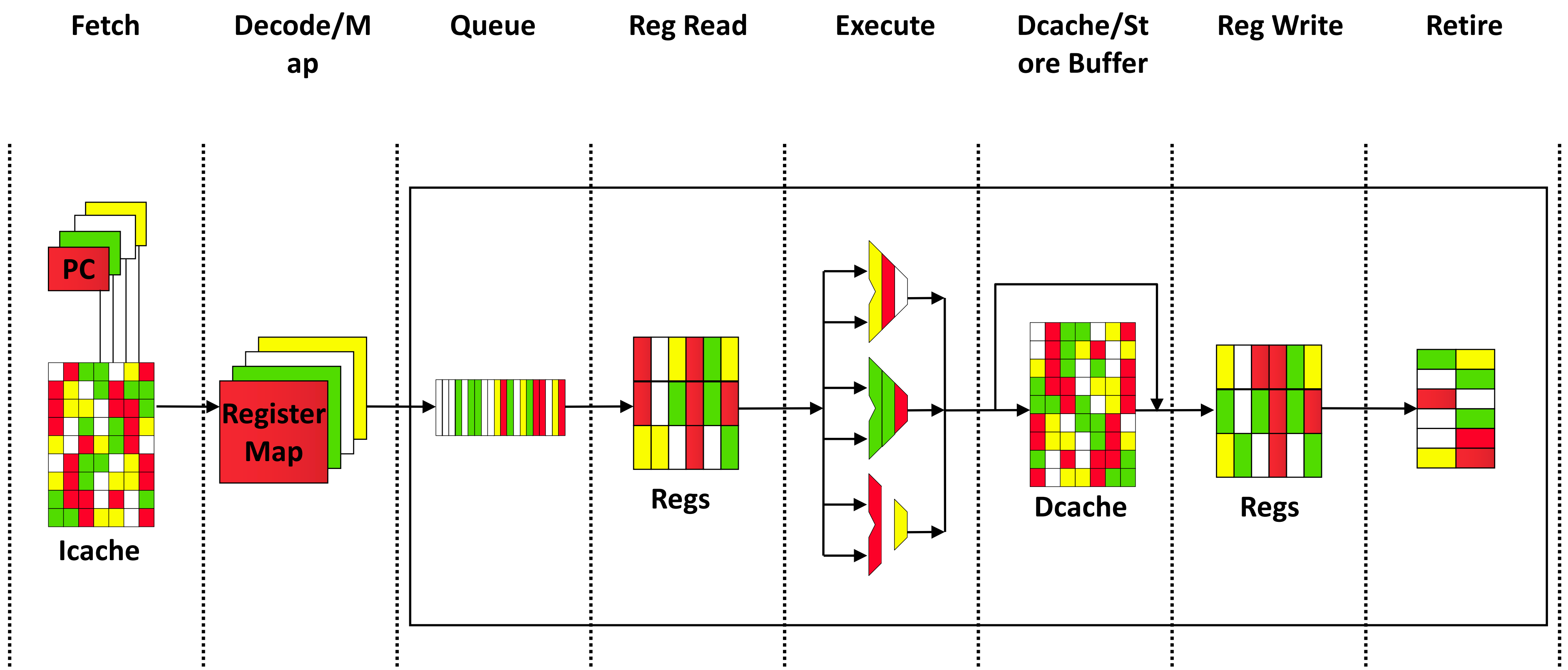
Multithreaded Pipeline

- Must carry thread select down pipeline to ensure correct state bits read/written at each pipe stage



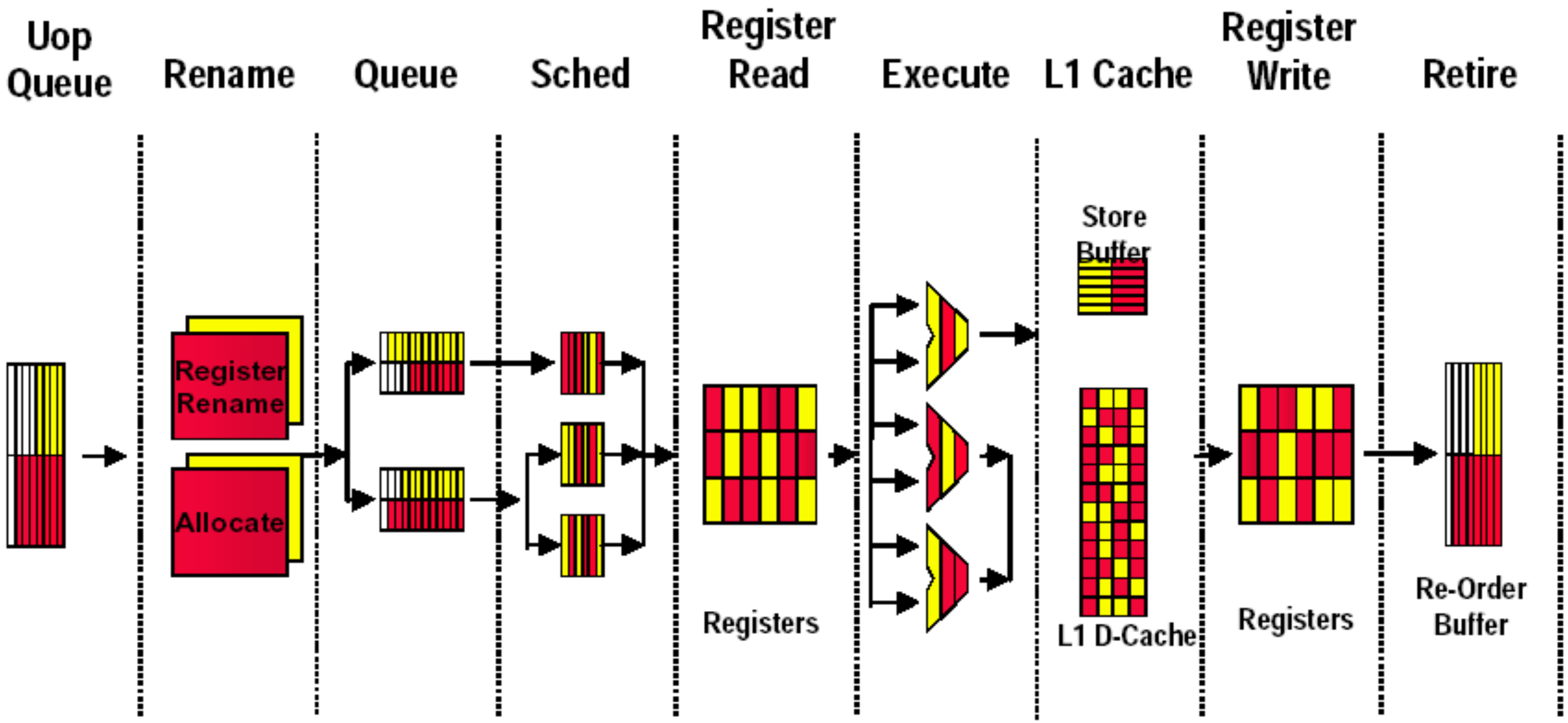
- These interactions create side-channels that can be exploited

Hardware Multithread Pipeline



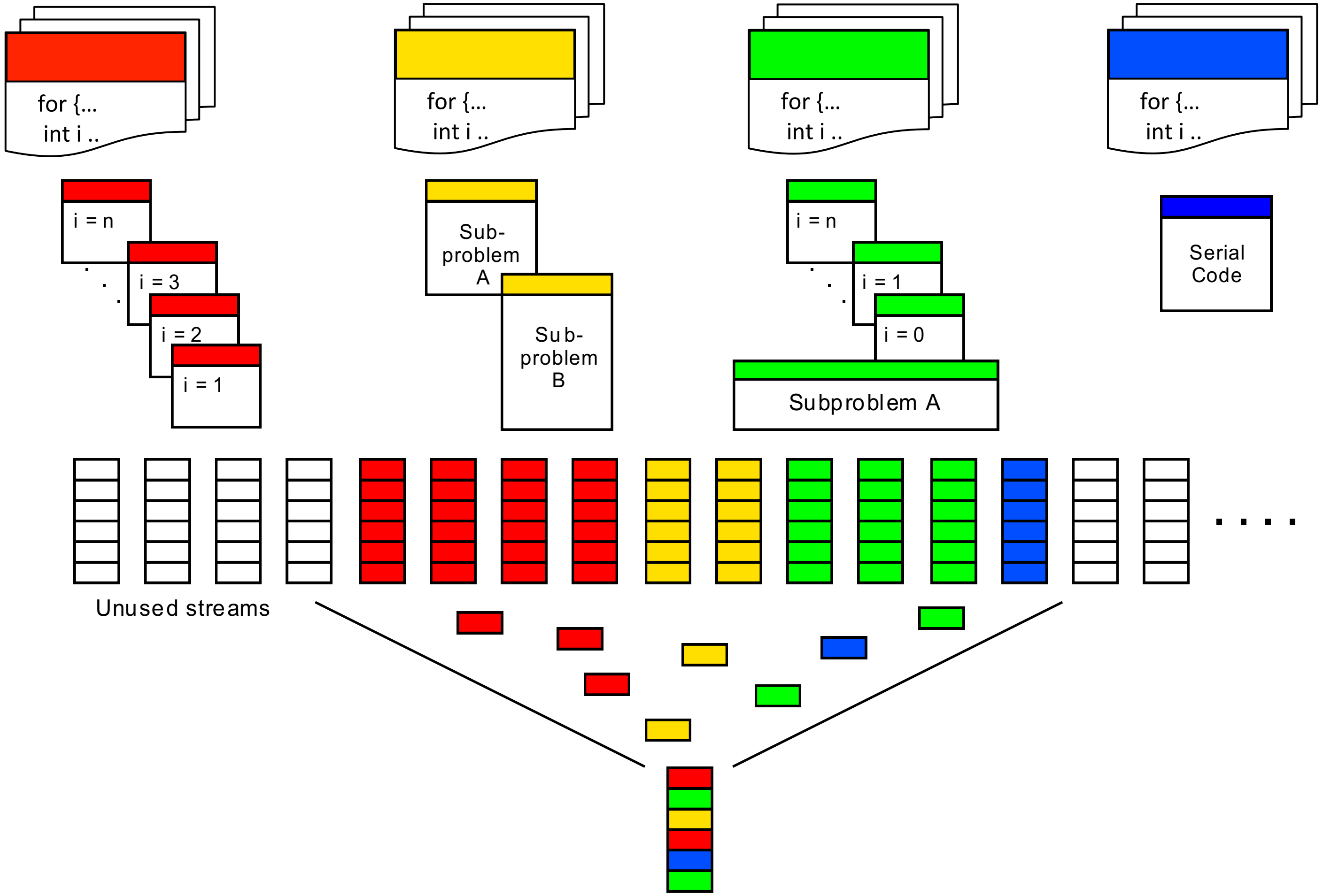
- These interactions create side-channels that can be exploited

Pentium-4 Hyperthreading (2002)



- These interactions create side-channels that can be exploited

Processing: Multithreading



Program running in Parallel

Threads of computation

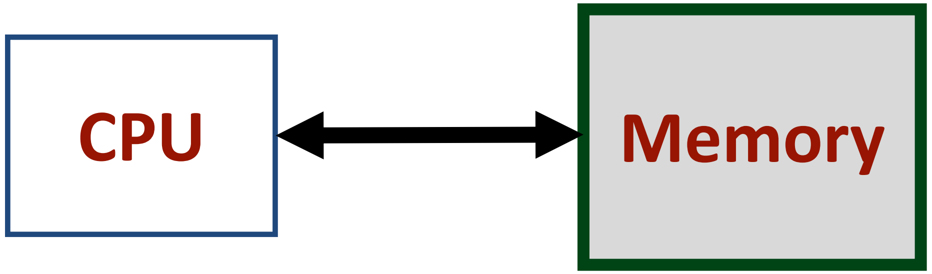
Supported hardware threads (e.g., 16)

Instruction ready pool

Interleaving of instruction executions

- These interactions create side-channels that can be exploited

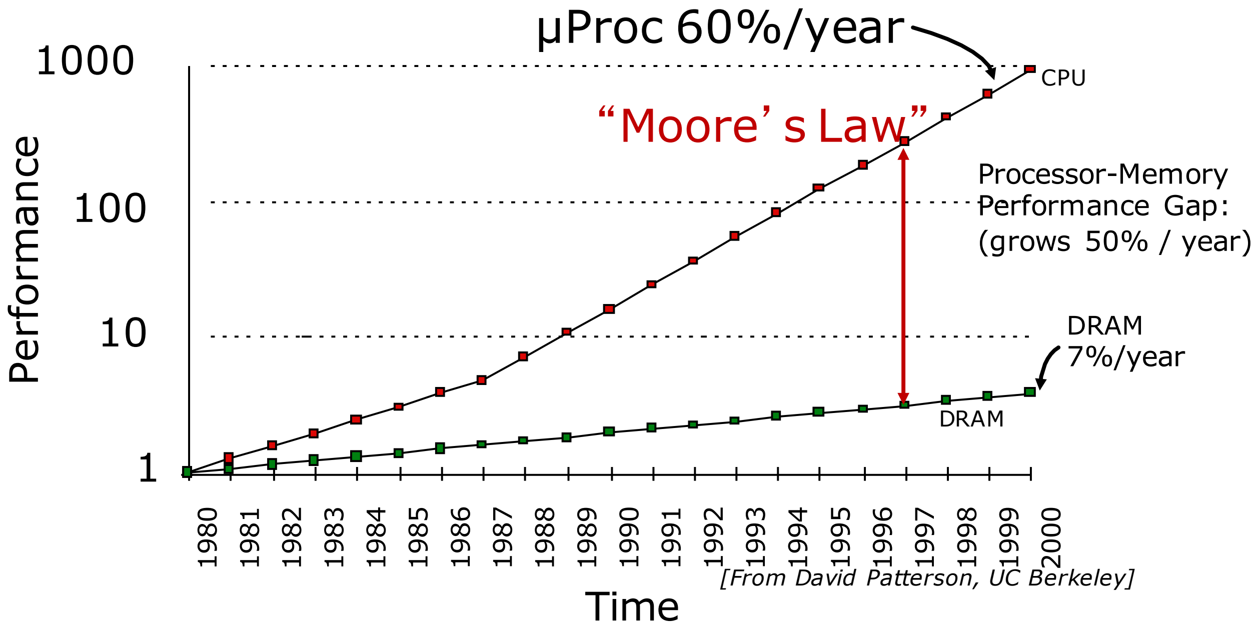
CPU-Memory Bottleneck



- Performance of high-speed computers is usually limited by memory bandwidth & latency
  - Latency (time for a single access) Memory access time >> Processor cycle time
  - Bandwidth (number of accesses per unit time) if fraction m of instructions access memory,
    - 1+m memory references / instruction
      - Ghost of the stored-program architecture
    - CPI = 1 requires 1+m memory refs / cycle

Processor- Memory Gap

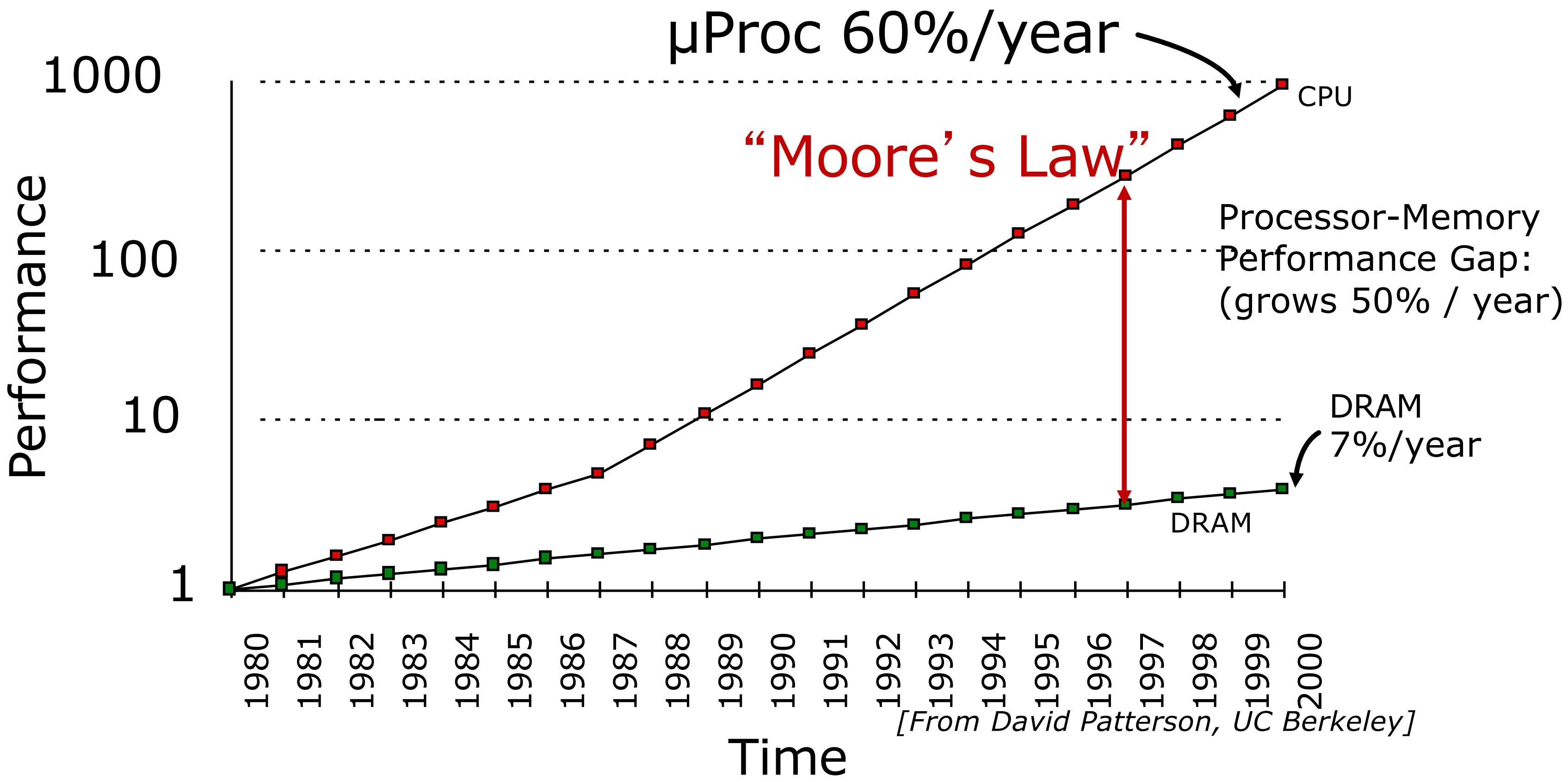
- Performance gap: CPU (55% each year) vs. DRAM (7% each year)
  - Processor operations take of the order of 1 ns
  - Memory access requires 10s or even 100s of ns
  - Each instruction executed involves at least one memory access



[From David Patterson, UC Berkeley]

Processor-DRAM Gap (latency)

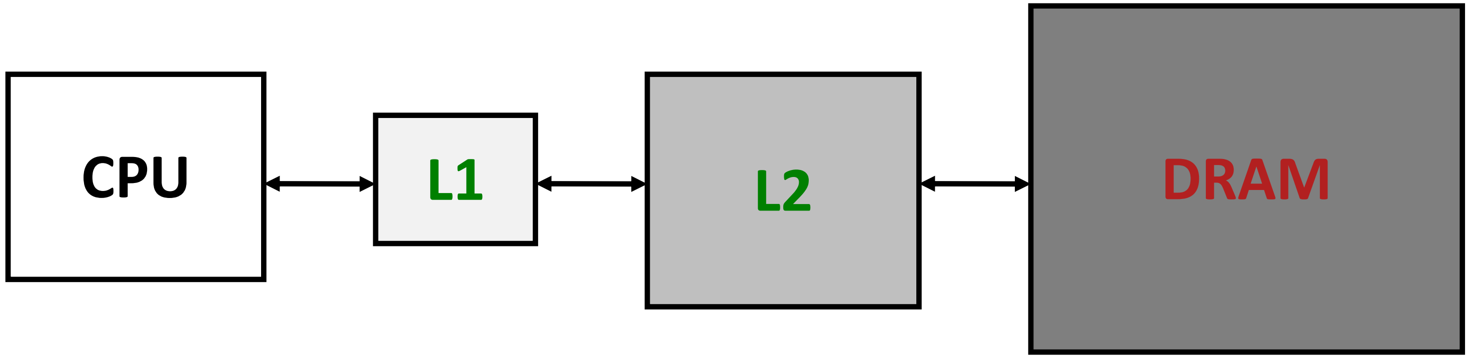
- Four-issue 2GHz superscalar accessing 100ns DRAM could execute 800 instructions during time for one memory access!



[From David Patterson, UC Berkeley]

Memory Organization

- A memory cannot be large and fast
- Increasing sizes of cache at each level



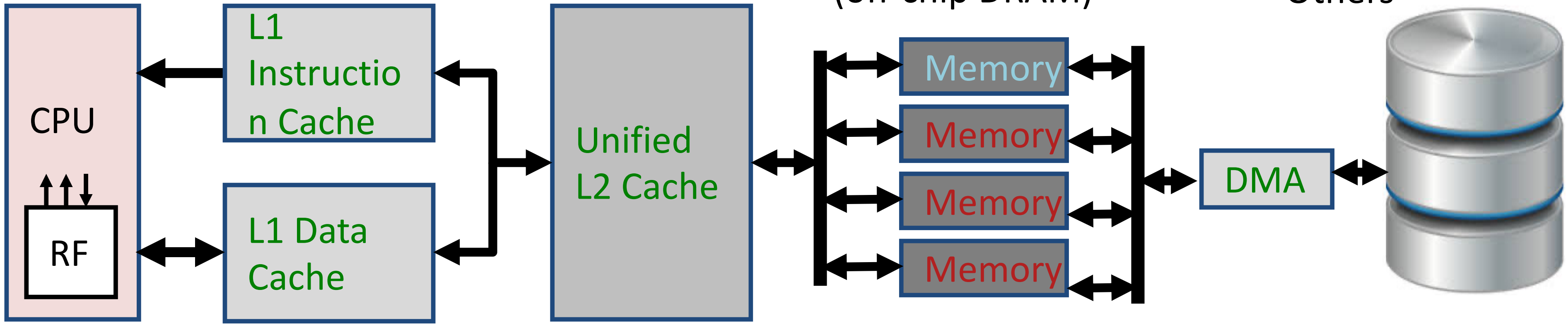
  - A hit at a level occurs if that level of the memory contains the data needed by the CPU
  - A miss occurs if the level does not contain the requested data

A Typical Memory Hierarchy

Split instruction
& data primary caches
(on-chip SRAM)

Multiple interleaved
memory banks
(off-chip DRAM)

Disks/External
Memory/Devices/
Others



Multi-ported
register file
(part of CPU)

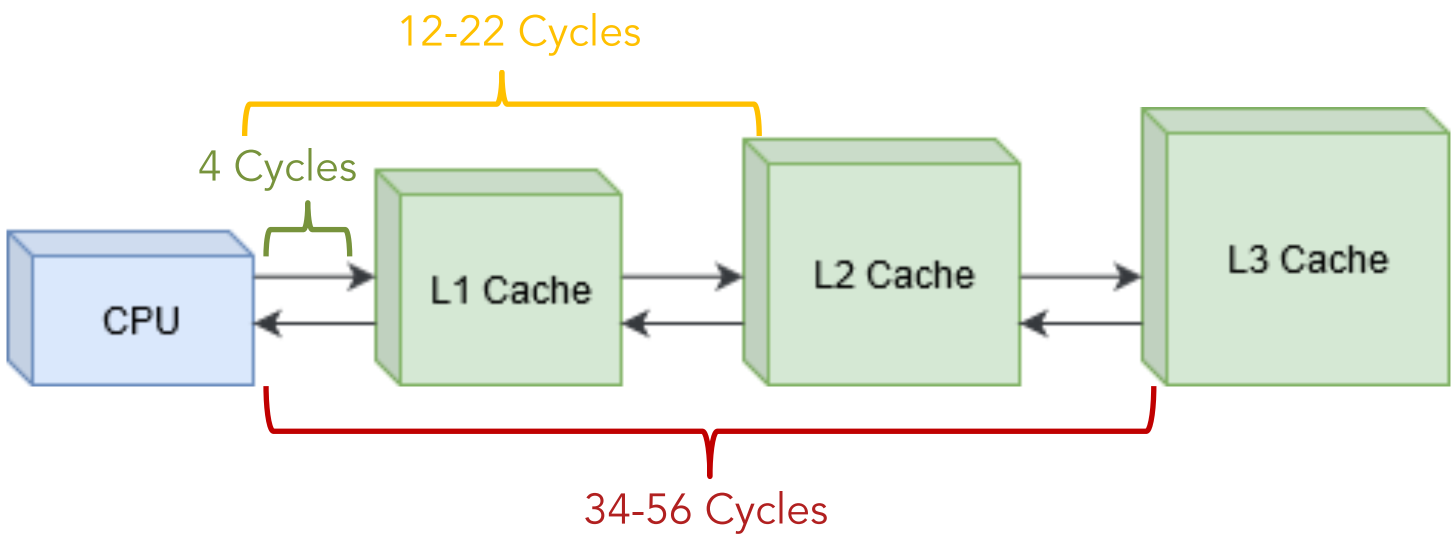
Large unified
secondary cache
(on-chip SRAM)

Cache Side-Channel Attacks

- "Consider securing a smart card is harder than securing the hardware of an offsite server against side-channel attacks"
- Threat model
  - Given: F: K X M -> D
    - Where K is a finite set of key
    - M is a finite set of messages
    - D is an arbitrary set of ciphertext
    - The attacker is assumed to have no access to the values of k and F (k,m) but he can measure/observe the characteristics of the physical implementation of F

Cache Side-Channel Attacks

- Caches used as the covert channel in many microarchitectural attacks (Spectre, Meltdown, Foreshadow, etc.)

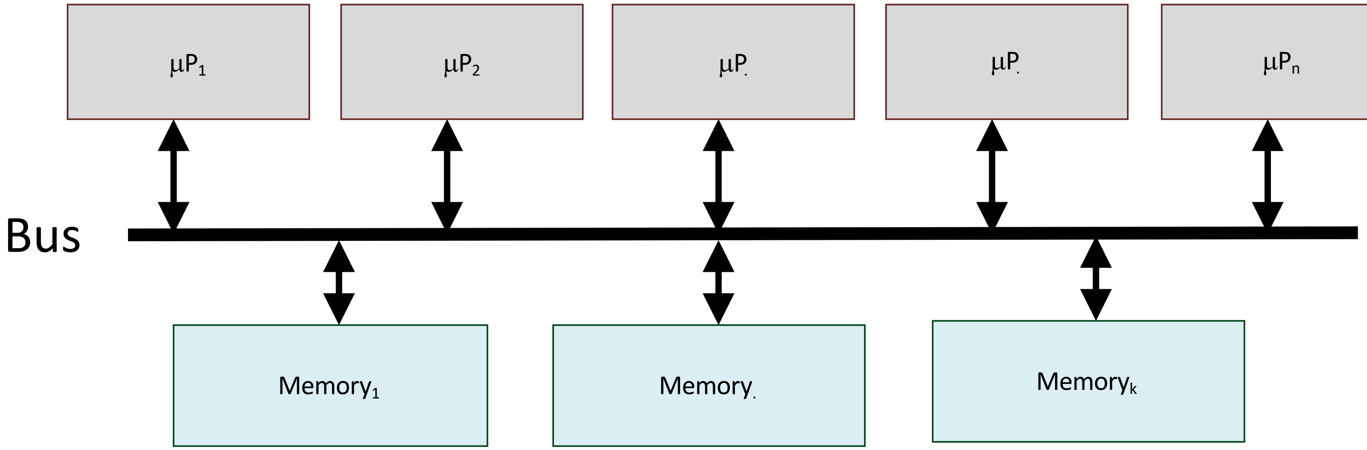


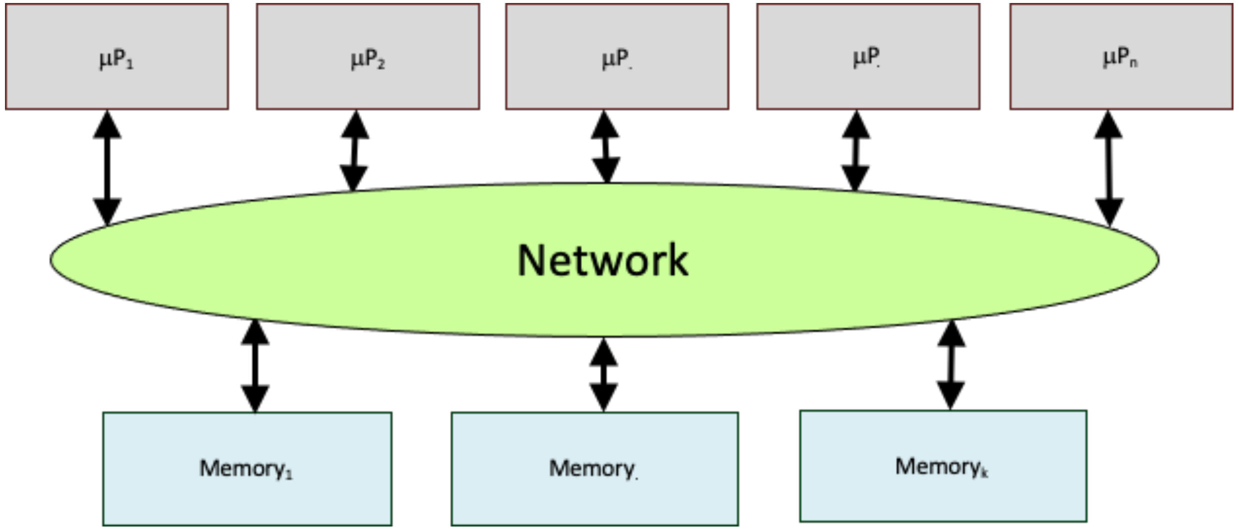
Cache Side-Channel Attacks

- Caches used as the covert channel in many microarchitectural attacks (Spectre, Meltdown, Foreshadow, etc.)
- Cache side-channel attacks exploit intrinsic cache characteristics
  - Caches are shared among processes
  - Hit/miss latency
  - Cache way and set organization
  - Coherence invalidations
- Can circumvent security measures such as privilege checks, browser sandboxing, Address Space Layout Randomization, etc.

Communication Networks

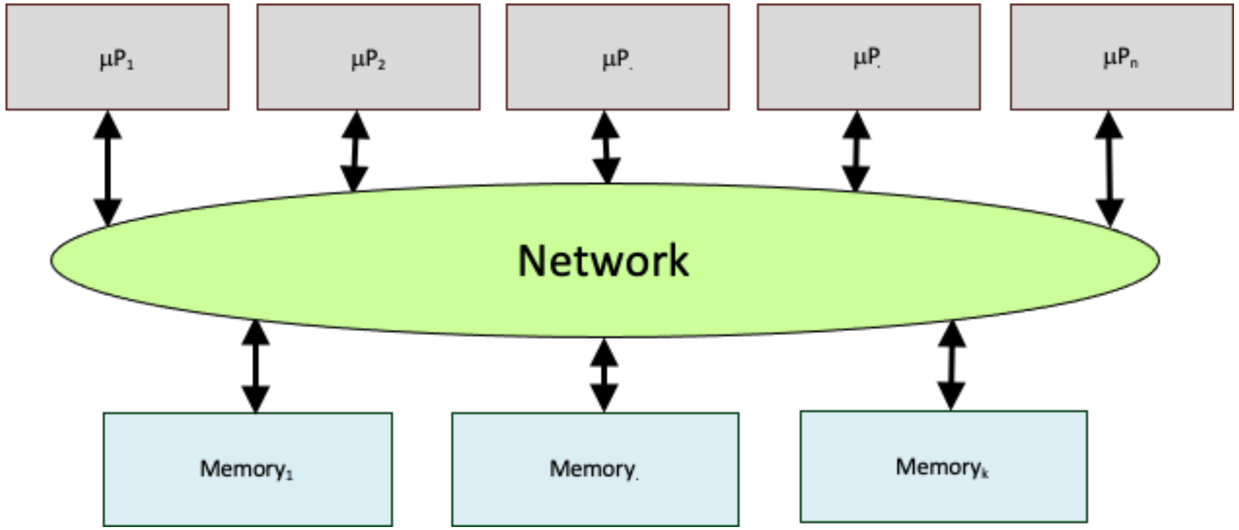
- Bus-based communication and side-channel effects
- Network-on-chip based communication and side-channel effects

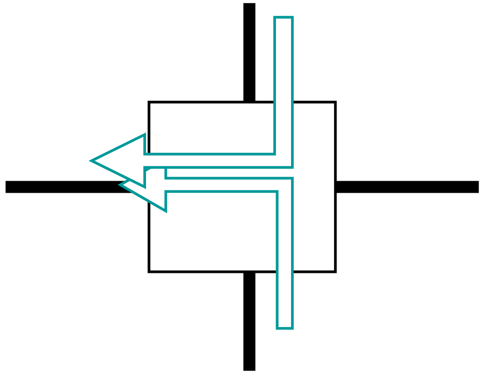




Communication Networks

- Bus-based communication and side-channel effects
- Network-on-chip based communication and side-channel effects

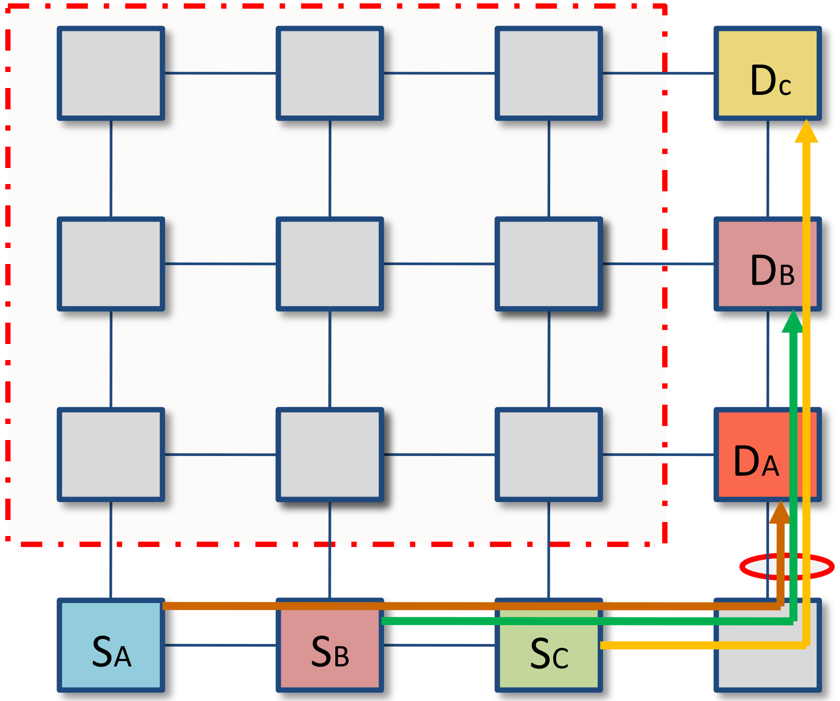




On-Chip Network Routing

- Oblivious Routing
  - Statically determined given the source and destination addresses

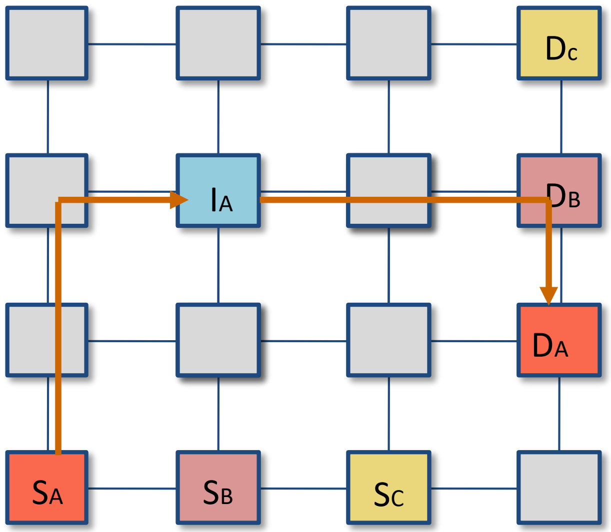
XY Routing



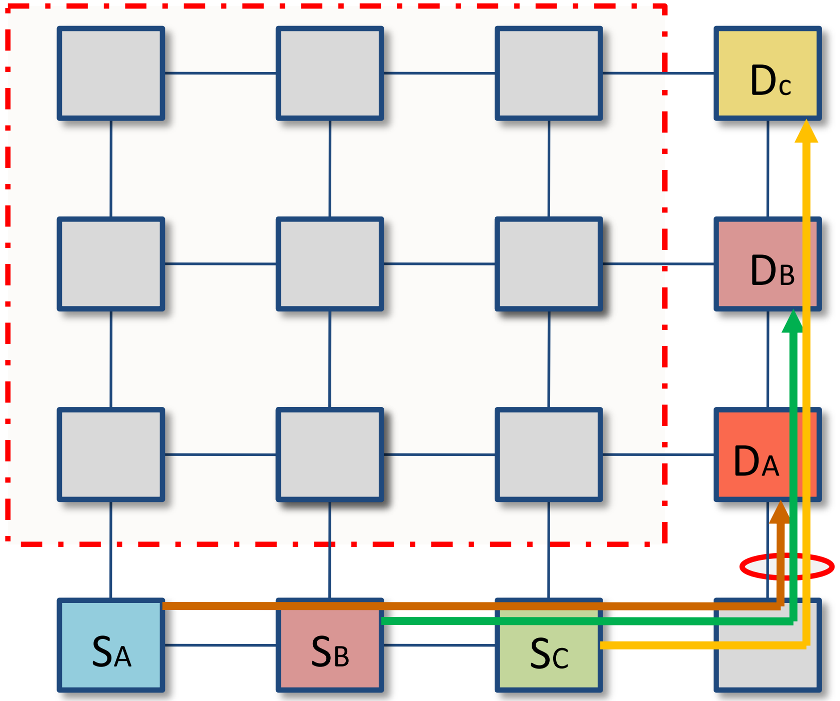
Link Capacity 75 Mbytes/sec
Each flow has 25 Mbytes/sec bandwidth demand

On-Chip Network Routing

- Oblivious Routing
  - Statically determined given the source and destination addresses
- Randomized Routing
  - Can add security



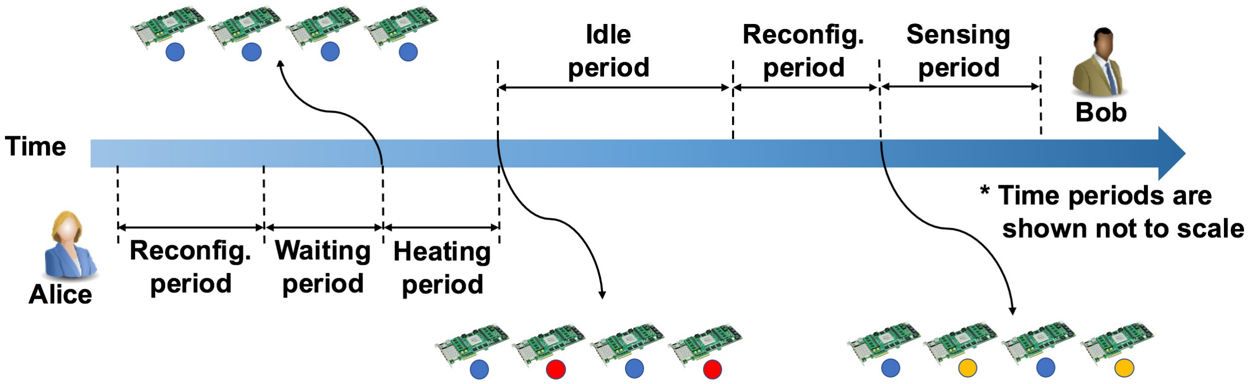
XY Routing



Link Capacity 75 Mbytes/sec
Each flow has 25 Mbytes/sec bandwidth demand

Cloud/FPGA Side-Channel Attacks

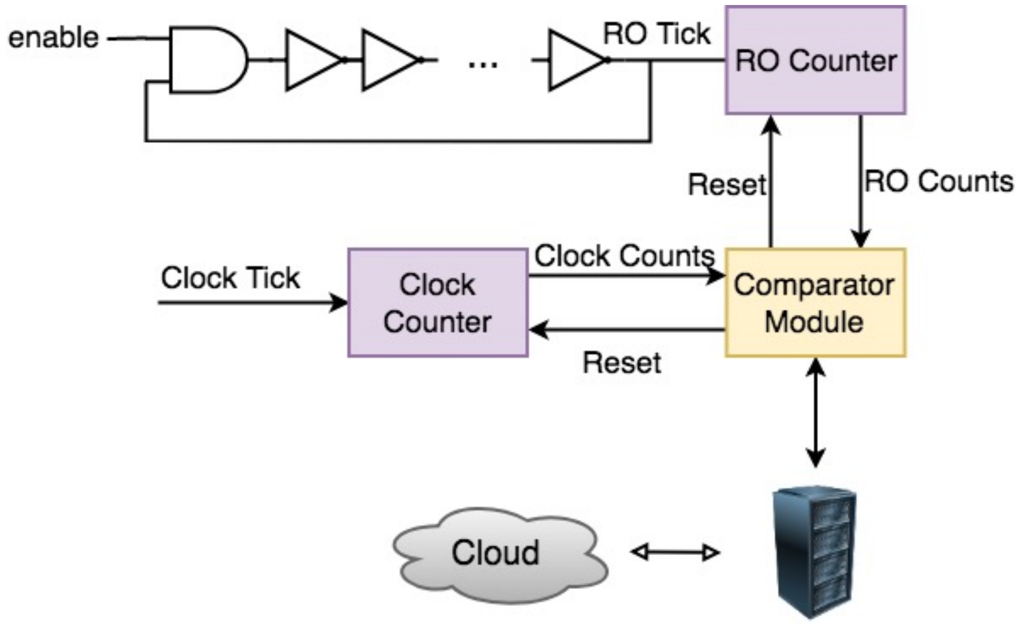
- Thermal Covert Channel in FPGAs
  - Cloud FPGAs allow for the sharing of the FPGA resources among users
  - This can lead to covert channels and information leakage
- Example
  - To transmit information, simple on-off keying (OOK) is used
  - The sender and receiver share or can access the same set of FPGAs
  - High Temperature = 1 and Low Temperature = 0



Temporal Thermal Covert Channels in Cloud FPGAs by Shanquan Tian and Jakub Szefer, International Symposium on Field-Programmable Gate Arrays (FPGA) 2018

Cloud/FPGA Side-Channel Attacks

- Ring Oscillator Temperature Sensor
  - Ring Oscillator (RO) is a temperature-to-frequency transducer suitable for thermal monitoring on FPGAs
  - Comparing RO counts (affected by temperature) to reference clock counts (not affected by temperature) allows one to measure relative RO frequency



Eduardo Boemo and Sergio López-Buedo. 1997. Thermal monitoring on FPGAs using ring-oscillators. In International Workshop on Field Programmable Logic and Applications. Springer, 69–78.

Side Channel Defenses

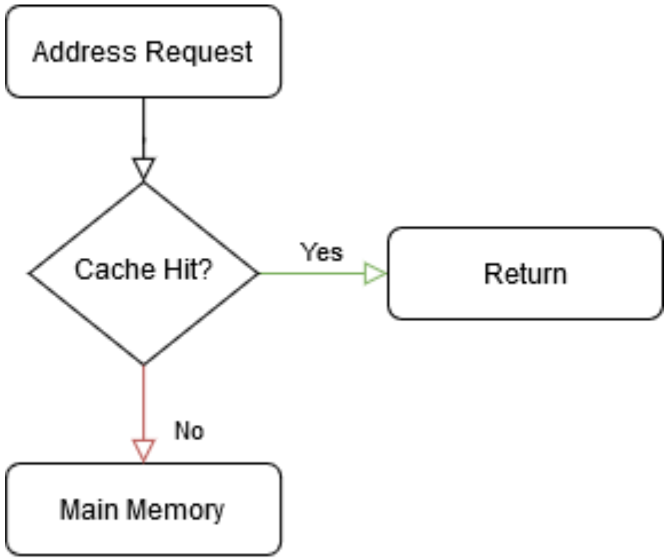
- Defense
  - Leakage Reduction
  - Noise Injection
  - Key Update
  - Side channel resistant PUFs
  - Secure scan chains
- Metrics
  - The amount of secret information that is vulnerable
  - The number of samples from side channels needed to extract the secret information

Process Isolation

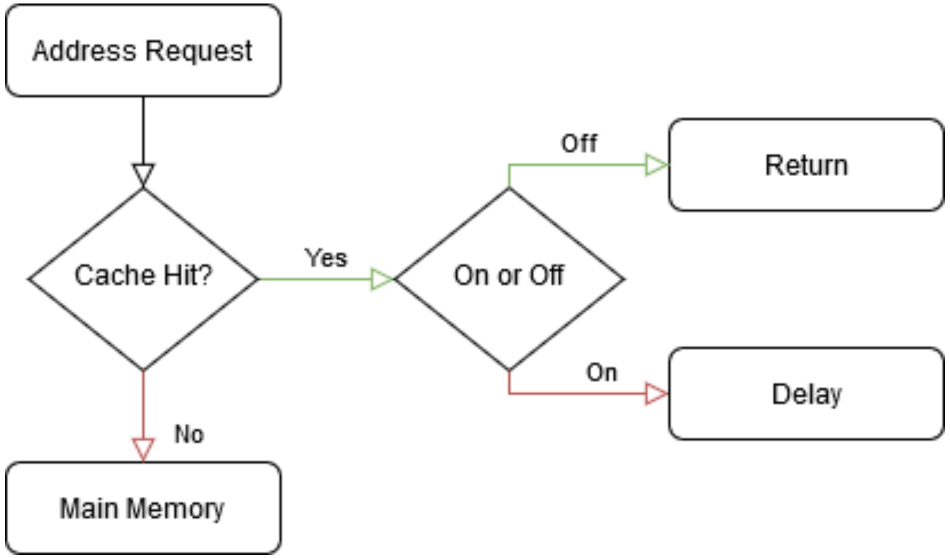
- Isolating a process is not trivial, and requires architectural, OS, compiler/runtime, and software support
- Isolating processes (e.g., using SGX) sacrifices hardware utilization
- Currently there is no simple way of using multi-core systems for secure computation
  - Users have to choose between multiple powerful but unsecure cores, and slow, secure enclaves
  - The tradeoff is coarse-grained and at design-time

Obfuscation: Cache Behavior Example

Standard Behavior

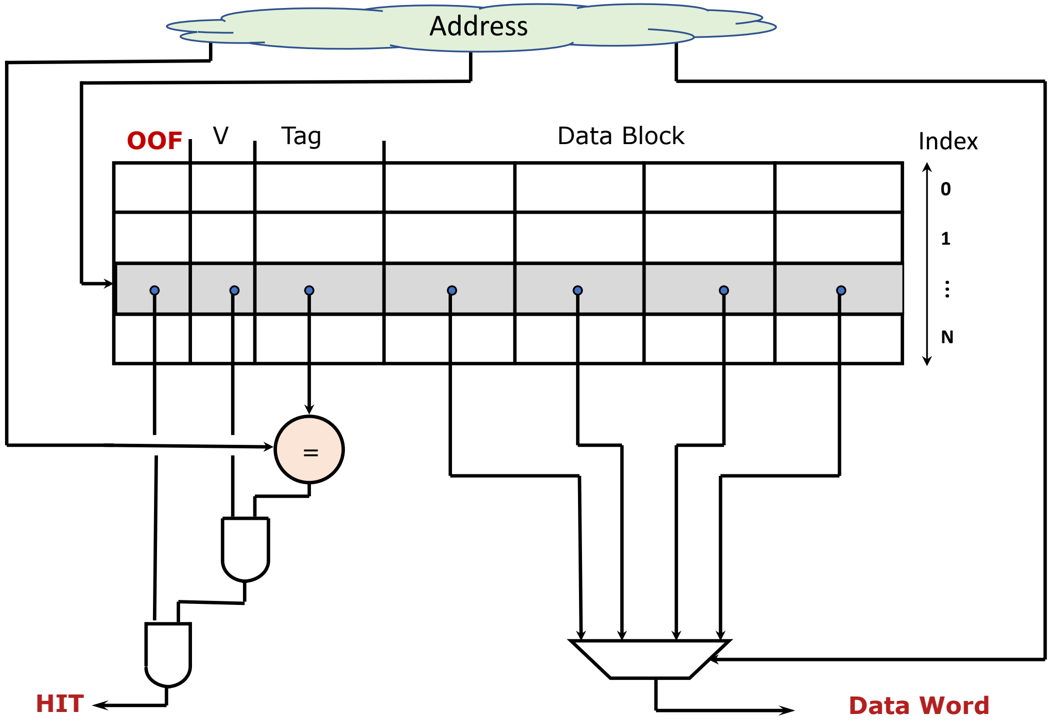


Modified Behavior



Obfuscation: Janus Cache Behavior Example

- Each line of cache is augmented with a 1-bit "ON_OFF Flag"
  - When ON_OFF FLAG=0 the block works normal
  - When ON_OFF FLAG=1 the block cannot be accessed



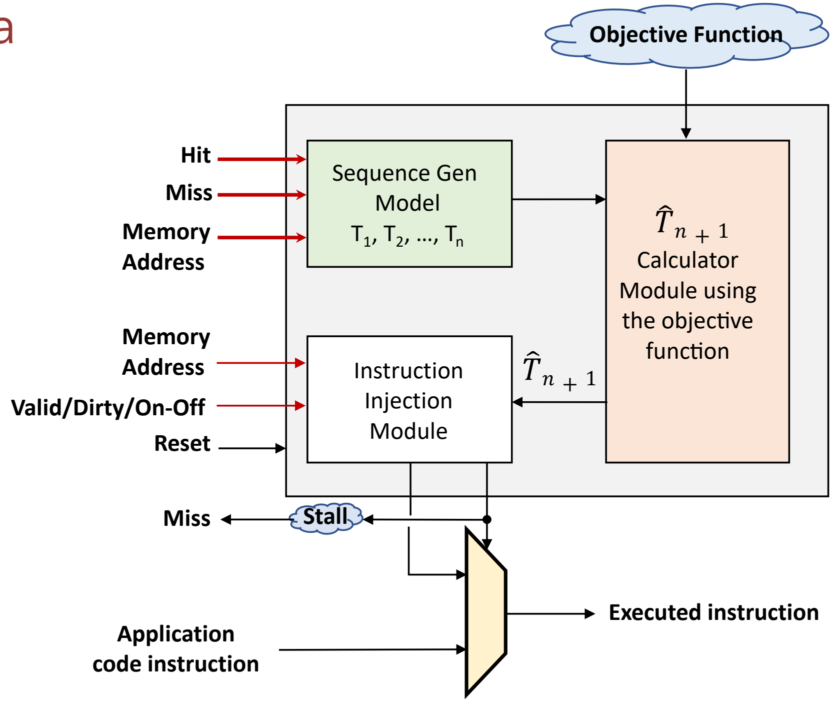
- **Four new instructions to support runtime cache block ON/OFF operation**
  - ***cacheblock-on-i*** and ***cache-block-off-i***
  - Load or Store with ***cacheblock-on-i*** or ***cache-block-off-i***

H. Hosseinzadeh, M. Isakov, M. Darabi, A. Patooghy, and M. Kinsy: "Janus: An uncertain cache architecture to cope with side channel attacks". In 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS) Aug 2017

Obfuscation: Janus Cache Behavior Example

- Which data block (s) of the cache should be turn ON of OFF at time slot t?
  1. Compute the P_i values for different block at the slot
  2. Compute the power changes for all the states of data blocks
  3. Compute *n(t)*
  4. Choose the block, *j*, which changes the power to the closest value of *n(t)*

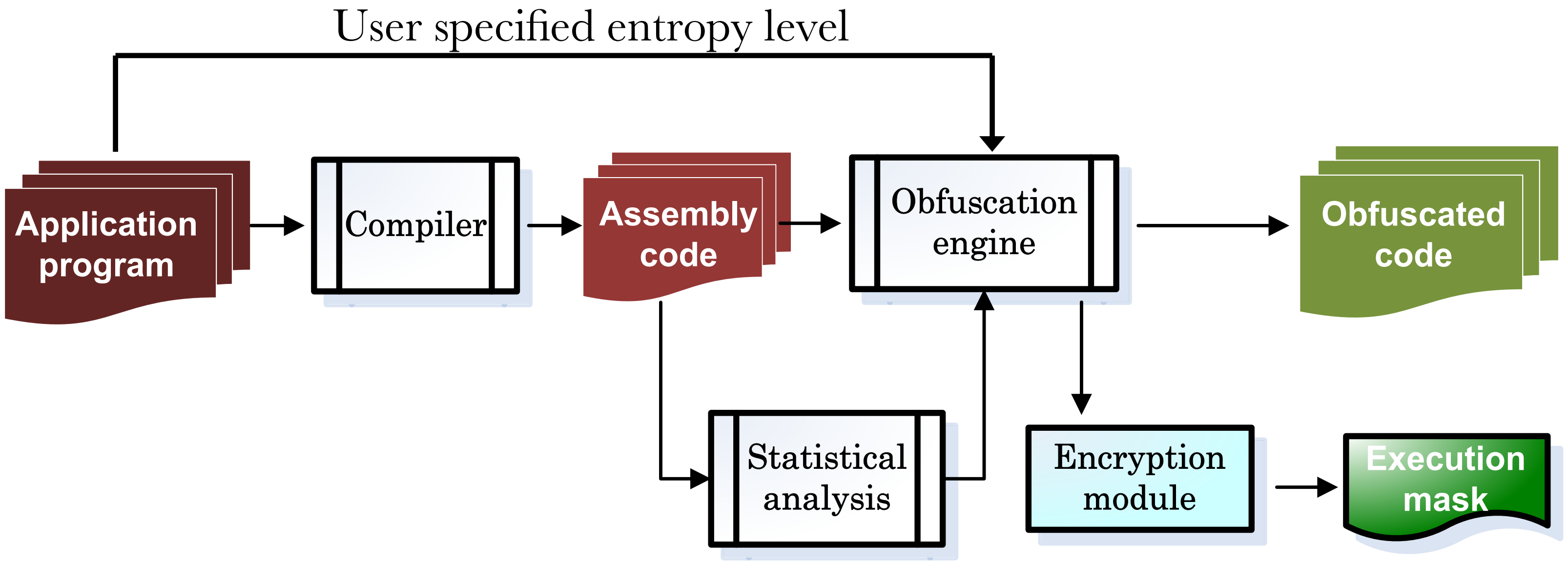
$$n(t) = \sqrt{\frac{-2\log(S)}{S}} V_1$$



	Baseline	With Obfuscation	
	Normal Exe. Time	Average Exe. Time (ns)	Max. Exe. Time (ns)
Bubble sort	290.2	325.8	547.9
Quick sort	223.6	237.2	313.5
Fibonacci	58.2	58.2	55.6

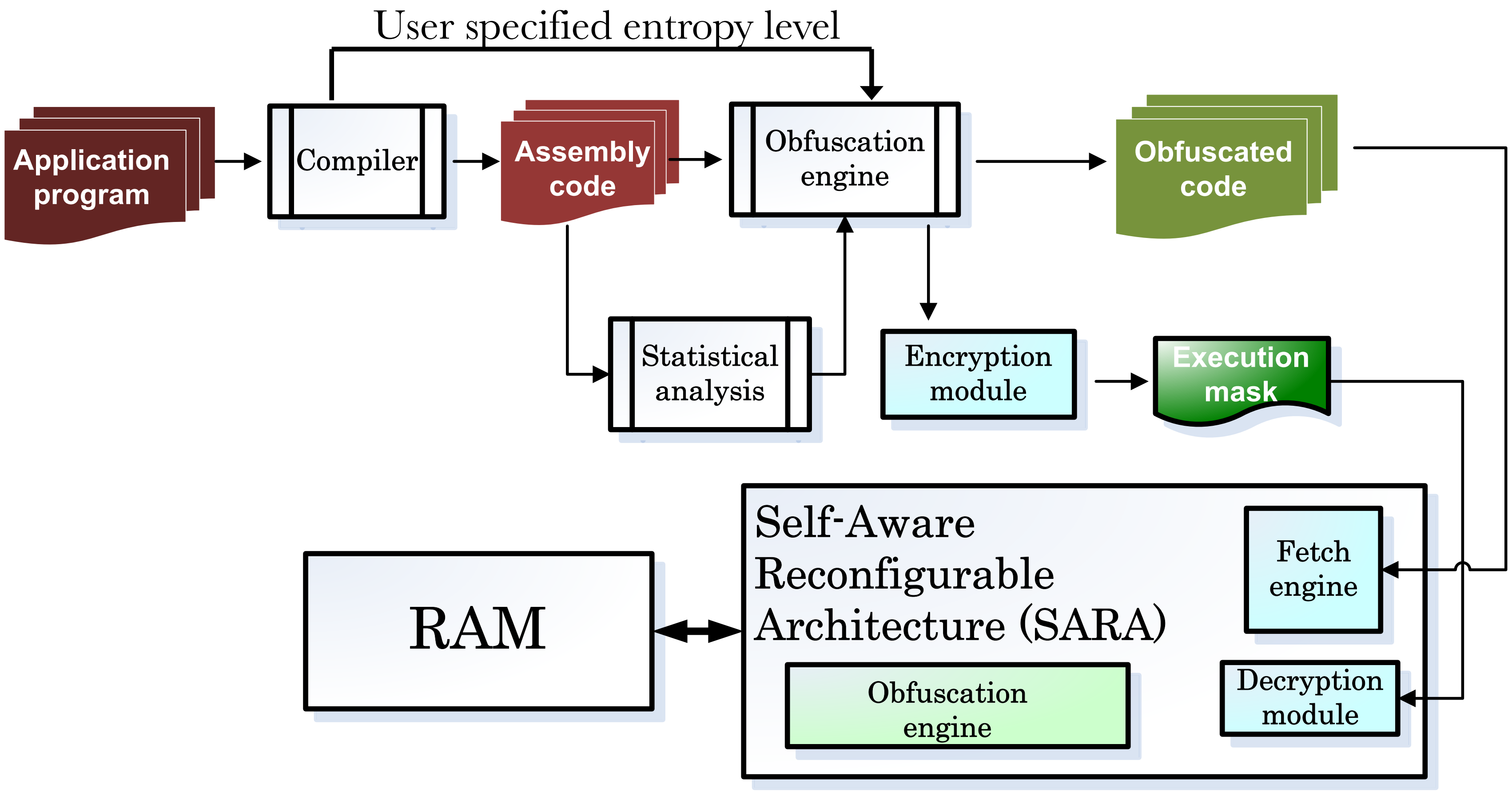
Sphinx Architecture: Co-Design Obfuscation

- Software-side Obfuscation
  - Dynamic obfuscation added to allow increase security for speed tradeoff



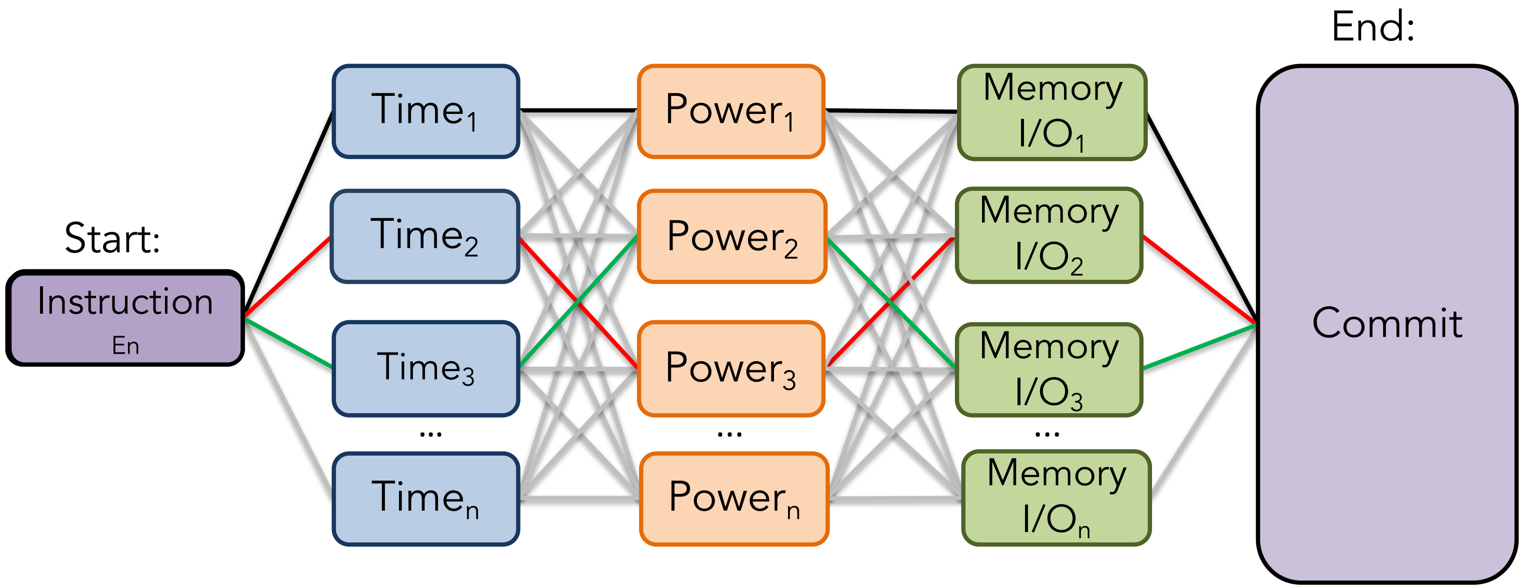
Sphinx Architecture: Co-Design Obfuscation

- Hardware-side Obfuscation



Sphinx Architecture: Co-Design Obfuscation

- Allows the hardware to have multiple ways of executing the same instruction with different time, power, and memory/IO profiles
  - Performance/timing-awareness for timing obfuscation
  - Power awareness for power obfuscation
  - Self-organized data storage for memory and I/O obfuscation



Upcoming Lectures

- Hardware Root-of-Trust Design