Curriculum 2013 Knowledge Units Pertaining to PDC

KA	KU	Tier	Level	Nur	PDC	Learning Outcome
AR	Assembly level machine organization	2	Familiarity	2	p	Describe how an instruction is executed in a classical von Neumann machine, with extensions for threads, multiprocessor synchronization, and SIMD execution
AR	Assembly level machine organization	2	Familiarity	3	p	Describe instruction level parallelism and hazards, and how they are managed in typical processor pipelines
AR	Digital logic and digital systems	2	Familiarity	2	p	Comprehend the trend of modern computer architectures towards multi-core and that parallelism is inherent in all hardware systems
AR	Digital logic and digital systems	2	Familiarity	3	p	Explain the implications of the "power wall" in terms of further processor performance improvements and the drive towards harnessing parallelism
AR	Functional organization	3	Familiarity	3	p	Explain basic instruction level parallelism using pipelining and the major hazards that may occur
AR	Interfacing and communication	2	Familiarity	4	d	Compare common network organizations, such as ethernet/bus, ring, switched vs. routed
AR	Multiprocessing and alternative architectures	3	Familiarity	3	d	Explain the concept of interconnection networks and characterize different approaches
AR	Multiprocessing and alternative architectures	3	Familiarity	5	d	Describe the differences between memory backplane, processor memory interconnect, and remote memory via networks
AR	Multiprocessing and alternative architectures	3	Familiarity	1	p	Discuss the concept of parallel processing beyond the classical von Neumann model
AR	Multiprocessing and alternative architectures	3	Familiarity	2	p	Describe alternative architectures such as SIMD and MIMD
AR	Multiprocessing and alternative architectures	3	Familiarity	4	p	Discuss the special concerns that multiprocessing systems present with respect to memory management and describe how these are addressed
AR	Performance enhancements	3	Familiarity	5	p	Discuss the performance advantages that multithreading offered in an architecture along with the factors that make it difficult to derive maximum benefits from this approach
CN	Processing	3	Familiarity	9	p	Describe the levels of parallelism including task, data, and event parallelism.
CN	Processing	3	Assessme	10	p	Compare and contrast parallel programming paradigms recognizing the strengths and weaknesses of each.
CN	Processing	3	Usage	12	p	Design, code, test and debug programs for a parallel computation.
GV	Basic Rendering	3	Familiarity	2	p	Describe the basic graphics pipeline and how forward and backward rendering factor in this.
HC	Collaboration and communication	3	Familiarity	1	d	Describe the differences between synchronous and asynchronous communication
IAS	Defensive Programming	1	Usage	4	p	Demonstrate using a high-level programming language how to prevent a race condition from occurring and how to handle an exception
IAS	Network Security	2	Familiarity	1	d	Describe the different categories of network threats and attacks
IAS	Network Security	2	Familiarity	3	d	Describe virtues and limitations of security technologies at each layer of the network stack

Curriculum 2013 Knowledge Units Pertaining to PDC

IAS	Security Policy and Governance	3	Familiarity	7	d	Understand the risks and benefits of outsourcing to the cloud
IAS	Web Security	3	Familiarity	1	d	Understand the browser security model including same-origin policy and threat models in web security
IAS	Web Security	3	Familiarity	2	d	Understand the concept of web sessions, secure communication channels such as TLS and importance of secure certificates, authentication including single sign-on such as OAuth and SAML
IAS	Web Security	3	Usage	3	d	Understand common types of vulnerabilities and attacks in web applications and defenses against them.
IAS	Web Security	3	Usage	4	d	Understand how to use client-side security capabilities
IAS	Web Security	3	Usage	5	d	Understand how to use server-side security tools.
IM	Distributed Databases	3	Familiarity	1	d	Explain the techniques used for data fragmentation, replication, and allocation during the distributed database design process
IM	Distributed Databases	3	Assessmer	2	d	Evaluate simple strategies for executing a distributed query to select the strategy that minimizes the amount of data transfer
IM	Distributed Databases	3	Familiarity	3	d	Explain how the two-phase commit protocol is used to deal with committing a transaction that accesses databases stored on multiple nodes
IM	Distributed Databases	3	Familiarity	4	d	Describe distributed concurrency control based on the distinguished copy techniques and the voting method
IM	Distributed Databases	3	Familiarity	5	d	Describe the three levels of software in the client-server model
IM	Information Management Concepts	2	Familiarity	12	d	approaches that scale up to globally networked systems
IM	Information Storage and Retrieval	3	Usage	4	d	Perform Internet-based research
NC	Introduction	1	Familiarity	1	d	Articulate the organization of the Internet
NC	Introduction	1	Familiarity	2	d	List and define the appropriate network terminology
NC	Mobility	2	Familiarity	2	d	Describe how wireless networks support mobile users
NC	Networked Applications	1	Usage	3	d	Implement a simple client-server socket-based application
NC	Reliable Data Delivery	2	Familiarity	1	d	Describe the operation of reliable delivery protocols
NC	Reliable Data Delivery	2	Usage	3	d	Design and implement a simple reliable protocol
NC	Resource Allocation	2	Familiarity	1	d	Describe how resources can be allocated in a network
NC	Resource Allocation	2	Familiarity	2	d	Describe the congestion problem in a large network
OS	Concurrency	2	Usage	2	c	Demonstrate the potential run-time problems arising from the concurrent operation of many separate tasks
OS	Concurrency	2	Familiarity	3	c	Summarize the range of mechanisms that can be employed at the operating system level to realize concurrent systems and describe the benefits of each
OS	Concurrency	2	Familiarity	5	c	Summarize techniques for achieving synchronization in an operating system (e.g., describe how to implement a semaphore using OS primitives)

Curriculum 2013 Knowledge Units Pertaining to PDC

OS	Concurrency	2	Familiarity	6	c	Describe reasons for using interrupts, dispatching, and context switching to support concurrency in an operating system
OS	Operating System Principles	2	Familiarity	1	c	Describe the need for concurrency within the framework of an operating system
OS	Overview of Operating Systems	1	Familiarity	4	d	Discuss networked, client-server, distributed operating systems and how they differ from single user operating systems
PBD	Web Platforms	3	Usage	1	d	Design and Implement a simple web application
PBD	Web Platforms	3	Familiarity	4	d	Describe the differences between Software-as-a-Service and traditional software products
PD	Cloud Computing	3	Familiarity	1	d	Discuss the importance of elasticity and resource management in cloud computing.
PD	Cloud Computing	3	Usage	4	d	Deploy an application that uses cloud infrastructure for computing and/or data resources
PD	Cloud Computing	3	Familiarity	2	pd	Explain strategies to synchronize a common view of shared data across a collection of devices
PD	Communication and Coordination	1	Usage	1	p	Use mutual exclusion to avoid a given race condition
PD	Communication and Coordination	2	Familiarity	2	c	Give an example of an ordering of accesses among concurrent activities that is not sequentially consistent
PD	Communication and Coordination	2	Usage	5	c	Write a program that correctly terminates when all of a set of concurrent tasks have completed
PD	Communication and Coordination	2	Usage	6	c	Use a properly synchronized queue to buffer data passed among activities
PD	Communication and Coordination	2	Familiarity	7	c	Explain why checks for preconditions, and actions based on these checks, must share the same unit of atomicity to be effective
PD	Communication and Coordination	2	Usage	8	c	Write a test program that can reveal a concurrent programming error; for example, missing an update when two activities both try to increment a variable
PD	Communication and Coordination	2	Familiarity	9	c	Describe at least one design technique for avoiding liveness failures in programs using multiple locks or semaphores
PD	Communication and Coordination	2	Familiarity	10	c	Describe the relative merits of optimistic versus conservative concurrency control under different rates of contention among updates
PD	Communication and Coordination	2	Usage	3	d	Give an example of a scenario in which blocking message sends can deadlock
PD	Communication and Coordination	2	Familiarity	4	d	Explain when and why multicast or event-based messaging can be preferable to alternatives
PD	Communication and Coordination	3	Usage	12	c	Use semaphores or condition variables to block threads until a necessary precondition holds
PD	Distributed Systems	3	Familiarity	1	d	Distinguish network faults from other kinds of failures
PD	Distributed Systems	3	Familiarity	2	d	Explain why synchronization constructs such as simple locks are not useful in the presence of distributed faults

Curriculum 2013 Knowledge Units Pertaining to PDC

PD	Distributed Systems	3	Usage	3	d	Give examples of problems for which consensus algorithms such as leader election are required
PD	Distributed Systems	3	Usage	4	d	Write a program that performs any required marshalling and conversion into message units, such as packets, to communicate interesting data between two hosts
PD	Distributed Systems	3	Usage	5	d	Measure the observed throughput and response latency across hosts in a given network
PD	Distributed Systems	3	Familiarity	6	d	Explain why no distributed system can be simultaneously consistent, available, and partition tolerant
PD	Distributed Systems	3	Usage	7	d	Implement a simple server -- for example, a spell checking service
PD	Distributed Systems	3	Familiarity	8	d	Explain the tradeoffs among overhead, scalability, and fault tolerance when choosing a stateful v. stateless design for a given service
PD	Distributed Systems	3	Familiarity	9	d	Describe the scalability challenges associated with a service growing to accommodate many clients, as well as those associated with a service only transiently having many clients
PD	Formal Models and Semantics	3	Usage	1	c	Model a concurrent process using a formal model, such as pi-calculus
PD	Formal Models and Semantics	3	Familiarity	2	c	Explain the characteristics of a particular formal parallel model
PD	Formal Models and Semantics	3	Usage	3	c	Formally model a shared memory system to show if it is consistent
PD	Formal Models and Semantics	3	Usage	4	c	Use a model to show progress guarantees in a parallel algorithm
PD	Formal Models and Semantics	3	Usage	5	c	Use formal techniques to show that a parallel algorithm is correct with respect to a safety or liveness property
PD	Formal Models and Semantics	3	Usage	6	c	Decide if a specific execution is linearizable or not
PD	Parallel Algorithms, Analysis, and Programming	2	Usage	2	p	Compute the work and span, and determine the critical path with respect to a parallel execution diagram
PD	Parallel Algorithms, Analysis, and Programming	2	Familiarity	3	p	Define “speed-up” and explain the notion of an algorithm’s scalability in this regard
PD	Parallel Algorithms, Analysis, and Programming	2	Usage	4	p	Identify independent tasks in a program that may be parallelized
PD	Parallel Algorithms, Analysis, and Programming	2	Familiarity	5	p	Characterize features of a workload that allow or prevent it from being naturally parallelized

Curriculum 2013 Knowledge Units Pertaining to PDC

PD	Parallel Algorithms, Analysis, and Programming	2	Usage	6	p	Implement a parallel divide-and-conquer and/or graph algorithm and empirically measure its performance relative to its sequential analog
PD	Parallel Algorithms, Analysis, and Programming	3	Familiarity	8	d	Provide an example of a problem that fits the producer-consumer paradigm
PD	Parallel Algorithms, Analysis, and Programming	3	Familiarity	10	d	Identify issues that arise in producer-consumer algorithms and mechanisms that may be used for addressing them
PD	Parallel Algorithms, Analysis, and Programming	3	Familiarity	9	pd	Give examples of problems where pipelining would be an effective means of parallelization
PD	Parallel Architecture	1	Familiarity	1	d	Explain the differences between shared and distributed memory
PD	Parallel Architecture	2	Familiarity	2	p	Describe the SMP architecture and note its key features
PD	Parallel Architecture	2	Familiarity	3	p	Characterize the kinds of tasks that are a natural match for SIMD machines
PD	Parallel Architecture	3	Familiarity	6	d	Describe the key features of different distributed system topologies
PD	Parallel Architecture	3	Familiarity	5	p	Describe the challenges in maintaining cache coherence
PD	Parallel Architecture	3	Familiarity	4	pd	Explain the features of each classification in Flynn's taxonomy
PD	Parallel Decomposition	1	Usage	1	p	Explain why synchronization is necessary in a specific parallel program
PD	Parallel Decomposition	2	Usage	2	p	Write a correct and scalable parallel algorithm
PD	Parallel Decomposition	2	Usage	3	p	Parallelize an algorithm by applying task-based decomposition
PD	Parallel Decomposition	2	Usage	4	p	Parallelize an algorithm by applying data-parallel decomposition
PD	Parallel Performance	3	Usage	1	p	Calculate the implications of Amdahl's law for a particular parallel algorithm
PD	Parallel Performance	3	Usage	4	p	Detect and correct an instance of false sharing
PD	Parallel Performance	3	Familiarity	5	p	Explain the impact of scheduling on parallel performance
PD	Parallel Performance	3	Familiarity	7	p	Explain the impact and trade-off related to power usage on parallel performance
PD	Parallel Performance	3	Familiarity	2	pd	Describe how data distribution/layout can affect an algorithm's communication costs
PD	Parallel Performance	3	Usage	3	pd	Detect and correct a load imbalance
PD	Parallel Performance	3	Familiarity	6	pd	Explain performance impacts of data locality
PD	Parallelism Fundamentals	1	Familiarity	3	p	Distinguish data races from higher level races
PL	Concurrency and Parallelism	3	Usage	1	c	Write correct concurrent programs using multiple programming models.
PL	Concurrency and Parallelism	3	Familiarity	2	p	Explain why programming languages do not guarantee sequential consistency in the presence of data races and what programmers must do as a result.
SE	Software Verification Validation	2	Familiarity	7	pd	Describe the issues and approaches to testing distributed and parallel systems.

Curriculum 2013 Knowledge Units Pertaining to PDC

SF	Computational Paradigms	1	Familiarity	3	p	Articulate the differences between single thread vs. multiple thread, single server vs. multiple server models, motivated by real world examples (e.g., cooking recipes, lines for multiple teller machines, couple shopping for food, wash-dry-fold, etc.).
SF	Computational Paradigms	1	Usage	7	p	Write a simple sequential problem and a simple parallel version of the same program.
SF	Computational Paradigms	1	Assessmer	7	p	Evaluate performance of simple sequential and parallel versions of a program with different problem sizes, and be able to describe the speed-ups achieved.
SF	Evaluation	1	Familiarity	2	p	Describe Amdahl's law and discuss its limitations.
SF	Evaluation	1	Usage	3	p	Design and conduct a performance-oriented experiment, e.g., benchmark a parallel program with different data sets in order to iteratively improve its performance.
SF	Parallelism	1	Familiarity	1	p	For a given program, distinguish between its sequential and parallel execution, and the performance implications thereof.
SF	Parallelism	1	Familiarity	2	p	Demonstrate on an execution time line that parallelism events and operations can take place simultaneously (i.e., at the same time). Explain how work can be performed in less elapsed time if this can be exploited.
SF	Parallelism	1	Familiarity	3	p	Explain other uses of parallelism, such as for reliability/redundancy of execution.
SF	Parallelism	1	Familiarity	4	p	Define the differences between the concepts of Instruction Parallelism, Data Parallelism, Thread Parallelism/Multitasking, Task/Request Parallelism.
SF	Parallelism	1	Usage	5	p	Write more than one parallel program (e.g., one simple parallel program in more than one parallel programming paradigm; a simple parallel program that manages shared resources through synchronization primitives; a simple parallel program that performs simultaneous operation on partitioned data through task parallel (e.g., parallel search terms; a simple parallel program that performs step-by-step pipeline processing through message passing).
SF	Parallelism	1	Assessmer	6	p	Use performance tools to measure speed-up achieved by parallel programs in terms of both problem size and number of resources.
SE	Software Design	2	Familiarity	10	x	Given a high-level design, identify the software architecture by differentiating among common software architectures such as 3-tier, pipe-and-filter, and client-server.
SE	Software Design	3	Usage	20	x	Apply component-oriented approaches to the design of a range of software, such as using components for concurrency and transactions, for reliable communication services, for database interaction including services for remote query and database management, or for secure communication and access.
SE	Software Construction	3	Usage	8	x	Rewrite a simple program to remove common vulnerabilities, such as buffer overflows, integer overflows and race conditions