МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»
(РУТ (МИИТ)

Рабочая программа дисциплины (модуля),
как компонент образовательной программы
базового высшего образования
по специальности
10.05.01 Компьютерная безопасность,
утвержденной первым проректором РУТ (МИИТ)
Тимониным В.С.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Архитектуры вычислительных систем и комплексов

Специальность: 10.05.01 Компьютерная безопасность

Специализация: Безопасность компьютерных систем и сетей (в сфере связи, информационных и коммуникационных технологий)

Форма обучения: Очная

1. Общие сведения о дисциплине (модуле).

Целями изучения дисциплины являются формирование и закрепление си-стемного подхода к изучению и проектированию сложных систем, систе-матизация знаний и умений по вычислительной технике и сетям через изу-чение различных архитектур параллельных вычислительных систем и компьютерных сетей.

Задачами изучения дисциплины являются освоение основных понятий и видов архитектур вычислительных систем и компьютерных сетей; обуче-ние разбираться в различиях между современными архитектурами, подби-рать оптимальную архитектуру под конкретную задачу и требования за-казчика.

Основными задачами дисциплины являются:

- Изучение принципов организации ЭВМ и вычислительных систем — освоение фоннеймановской и гарвардской архитектур, принципа откры-той архитектуры и программно-управляемого аппарата.

- Анализ классификаций вычислительных систем — рассмотрение архи-тектур по Флинну (SISD, SIMD, MISD, MIMD), а также многомашин-ных и многопроцессорных систем, кластеров и GRID-систем.

- Освоение способов повышения производительности— изучение конвей-еризации, суперскалярности, VLIW-архитектур, векторных вычислений, многоядерности и кэш-памяти различных уровней.

- Изучение организации памяти и ввода-вывода— понимание иерархии запоминающих устройств (регистры, кэш, ОЗУ, внешняя память), режи-мов доступа, каналов и контроллеров ввода-вывода.

- Оценка эффективности и надежности вычислительных комплексов— освоение метрик производительности (MIPS, FLOPS, тактовая частота), коэффициента ускорения, законов Амдала и Густафсона, а также мето-дов обеспечения отказоустойчивости.

- Формирование навыков проектирования и выбора архитектуры— уме-ние обоснованно выбирать тип архитектуры (RISC/CISC, SMP/NUMA, кластер, GPU-системы) для решения конкретных научно-технических и прикладных задач.

2. Планируемые результаты обучения по дисциплине (модулю).

Перечень формируемых результатов освоения образовательной программы (компетенций) в результате обучения по дисциплине (модулю):

ОПК-6 - Способен применять методы научных исследований при проведении разработок в области обеспечения безопасности компьютерных систем и сетей;

ПК-6 - Способность анализировать архитектуру, компоненты и характеристики телекоммуникационных и автоматизированных систем, выявлять потенциальные уязвимости и оценивать информационные риски.

Обучение по дисциплине (модулю) предполагает, что по его результатам обучающийся будет:

Знать:

- основные архитектуры параллельных вычислительных систем, мейнфреймов, GRID- и Cloud-систем;
- архитектуры вычислительных сетей и общие принципы их построения;
- принципы стандартизации вычислительных сетей;
- технологии физического уровня;
- технологии построения локальных вычислительных сетей;
- проблемы сетевой безопасности.

Уметь:

- выбирать структуру ВС и режим ее функционирования;
- разрабатывать структурные и функциональные схемы всех ее составляющих;
- применять методы повышения производительности систем и увеличения ее надежности, устранения проблем сетевой безопасности;
- выбирать необходимый набор и структуру компонентов математического обеспечения.

Владеть:

- навыками использования общепринятых средств эмулирования или симулирования компьютерных сетей;
- навыками использования стандартных программных средств исследования компьютерных сетей на базе протоколов семейства TCP/IP;
- навыками использования общепринятых средств эмулирования или симулирования компьютерных сетей.

3. Объем дисциплины (модуля).

3.1. Общая трудоемкость дисциплины (модуля).

Общая трудоемкость дисциплины (модуля) составляет 4 з.е. (144 академических часа(ов).

3.2. Объем дисциплины (модуля) в форме контактной работы обучающихся с педагогическими работниками и (или) лицами,

привлекаемыми к реализации образовательной программы на иных условиях, при проведении учебных занятий:

Тип учебных занятий	Количество часов	
	Всего	Семестр №6
Контактная работа при проведении учебных занятий (всего):	64	64
В том числе:		
Занятия лекционного типа	32	32
Занятия семинарского типа	32	32

3.3. Объем дисциплины (модуля) в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации составляет 80 академических часа (ов).

3.4. При обучении по индивидуальному учебному плану, в том числе при ускоренном обучении, объем дисциплины (модуля) может быть реализован полностью в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации.

4. Содержание дисциплины (модуля).

4.1. Занятия лекционного типа.

№ п/п	Тематика лекционных занятий / краткое содержание
1	Введение в архитектуру вычислительных систем. Классификация Содержание учебного материала: — Определение архитектуры вычислительной системы (ВС), отличие от организации и реализации — Классификация по Флинну (SISD, SIMD, MISD, MIMD): принципы и примеры — Классификация по назначению (универсальные, специализированные, суперкомпьютеры, встроенные системы) — Уровни представления архитектуры (уровень команд, микроархитектура, системная архитектура) — Основные тенденции развития вычислительных систем
2	Архитектура фон Неймана и гарвардская архитектура Содержание учебного материала: — Принципы фон Неймана (хранимая программа, последовательное исполнение, линейное адресное пространство) — Структура фон-неймановской машины: АЛУ, устройство управления, память, устройства ввода-вывода

№ п/п	Тематика лекционных занятий / краткое содержание
	— Гарвардская архитектура: раздельные шины команд и данных, преимущества и недостатки — Модифицированная гарвардская архитектура (кэш-память инструкций и данных) — Сравнение областей применения: микроконтроллеры vs. универсальные процессоры
3	Центральный процессор: микроархитектура и конвейеризация Содержание учебного материала: — Структура процессора: регистровый файл, АЛУ, блок управления, блок выборки и дешифрации команд — Понятие машинного цикла (fetch-decode-execute-writeback) — Конвейерная обработка команд: стадии конвейера (IF, ID, EX, MEM, WB) — Конфликты в конвейере: структурные, по данным (RAW,WAR,WAW), по управлению — Методы разрешения конфликтов (форвардинг, спекулятивное выполнение, отмена команд)
4	Суперскалярные и VLIW-архитектуры Содержание учебного материала: — Суперскалярный процессор: множественные исполнительные блоки, динамическое планирование — Последовательность: выборка нескольких команд за такт, аппаратное разрешение зависимостей — Архитектура VLIW (Very Long Instruction Word): статическое планирование компилятором — Сравнение суперскалярных и VLIW-процессоров (сложность аппаратуры vs. сложность компилятора) — Примеры: Intel Core (суперскалярный), TI TMS320 (VLIW), IA-64/Itanium
5	Архитектура SIMD: векторные процессоры и графические ускорители Содержание учебного материала: — Принципы SIMD: одна команда — множество данных — Векторные процессоры: векторные регистры, конвейерная обработка векторов — Наборы инструкций SIMD: MMX, SSE, AVX (Intel/AMD), NEON (ARM) — Архитектура GPU: множество ядер SIMD (warp/wavefront), иерархия памяти (shared memory, регистры) — Сравнение CPU и GPU для различных классов задач (графика, машинное обучение, научные расчеты)
6	MIMD-архитектуры: многопроцессорные и многоядерные системы Содержание учебного материала: — Классификация MIMD-систем: SMP (симметричная многопроцессорная), NUMA (неравномерный доступ к памяти), MPP (массивно-параллельные) — Архитектура многоядерного процессора: общий кэш L3, когерентность кэшей — Протоколы когерентности кэшей (MESI, MOESI, MESIF) — Сравнение SMP и NUMA (масштабируемость, стоимость доступа к памяти) — Примеры: Intel Xeon (SMP, NUMA в многопроцессорных конфигурациях), AMD EPYC
7	Иерархия памяти и организация кэш-памяти Содержание учебного материала: — Принцип локальности (временная и пространственная) — Иерархия памяти: регистры > L1/L2/L3 кэш > оперативная память > SSD > HDD > ленты — Организация кэша: прямой, ассоциативный, полностью ассоциативный, set-associative — Алгоритмы вытеснения (LRU, FIFO, Random, LFU) — Запись в кэш: сквозная запись (write-through) и обратная запись (write-back) — Политики выделения места при промахе (read-allocate, write-allocate)

№ п/п	Тематика лекционных занятий / краткое содержание
8	Виртуальная память и механизмы защиты Содержание учебного материала: — Принципы виртуальной памяти: страницы, отображение адресов, TLB (Translation Lookaside Buffer) — Многоуровневые таблицы страниц (x86-64: 4 уровня) — Механизмы защиты памяти: кольца защиты (ring 0–3 в x86), сегментация (устаревшая) vs. страничная организация — NX-бит (Never eXecute) запрет исполнения кода из страниц данных — Изоляция адресных пространств процессов (память ядра vs. пользователя)
9	Системы прерываний и исключений Содержание учебного материала: — Определение прерывания, аппаратное vs. программное (исключения, ловушки) — Классификация прерываний: маскируемые, немаскируемые, векторные, невекторные — Механизмы обработки прерываний: сохранение контекста, таблица векторов прерываний (IDT/IVT) — Приоритеты прерываний и вложенность — Примеры: контроллер прерываний 8259A (PIC), APIC (x86), NVIC (ARM Cortex-M)
10	Архитектура систем ввода-вывода и шины данных Содержание учебного материала: — Классификация шин: системная (FSB, QPI, Infinity Fabric), периферийные (PCIe, USB, SATA) — Архитектура прямого доступа к памяти (DMA): контроллер DMA, режимы работы — Механизм прерываний для оповещения о завершении ввода-вывода (IRQ, MSI/MSI-X) — Конфигурационное пространство PCI/PCIe: BAR, управление питанием — Сравнение топологий шин: общая шина, звезда, кольцо, точка-точка
11	Высокопроизводительные вычислительные системы (суперкомпьютеры) Содержание учебного материала: — Определение суперкомпьютера, показатели производительности (FLOPS, LINPACK) — Классификация по топологии соединений (3D Torus, Fat Tree, Dragonfly, гиперкуб) — Масштабируемые архитектуры: кластеры (Beowulf) и MPP — Ускорители в суперкомпьютерах: GPU (CUDA/HIP), FPGA, нейропроцессоры — Примеры: Top500, Fugaku, Summit, Tianhe, «Ломоносов»
12	Гетерогенные вычислительные системы (CPU + GPU + FPGA + AI-ускорители) Содержание учебного материала: — Проблемы гомогенных систем (производительность на ватт для параллельных задач) — Интеграция CPU и GPU: объединенная память (Unified Memory), шина PCIe, чиплеты — Архитектура FPGA-ускорителей: программируемая логика, баланс гибкость/производительность — Специализированные AI-ускорители (NPU, TPU, нейроморфные системы) — Модели программирования: CUDA, OpenCL, SYCL, OneAPI
13	Архитектуры с общей и распределенной памятью Содержание учебного материала: — Системы с общей памятью (UMA, NUMA, COMA): аппаратная поддержка, синхронизация (мьютексы, семафоры, барьеры) — Системы с распределенной памятью (MPP, кластеры): межсоединения (Ethernet, InfiniBand, Omni-Path) — Модель программирования MPI (передача сообщений) и её соответствие архитектуре

№ п/п	Тематика лекционных занятий / краткое содержание
	— Гибридные системы: распределенная память + разделяемая память внутри узла (MPI+OpenMP) — Достоинства и недостатки, масштабируемость, сложность программирования
14	Энергоэффективные архитектуры и встроенные системы (RISC-V, ARM) Содержание учебного материала: — Показатели энергоэффективности (FLOPS/ватт, MIPS/ватт) — Архитектура ARM: RISC-ядро, режимы (Thumb), big.LITTLE / DynamIQ, управление питанием (P-state, C-state) — Архитектура RISC-V: модульная система (расширения I, M, A, F, D, C, V), открытость, режимы привилегий (M, S, U) — Система на кристалле (SoC): интеграция CPU, GPU, DSP, контроллеров ввода-вывода, SRAM — Примеры энергоэффективных платформ: мобильные устройства (Qualcomm Snapdragon, Apple Silicon), IoT (ARM Cortex-M, ESP32, RISC-V MCU)
15	Сравнительный анализ архитектур x86, ARM, RISC-V Содержание учебного материала: — Архитектура x86 (CISC): переменная длина команд, сложная декодировка, микрокодовое исполнение, обратная совместимость — Архитектура ARM (RISC): фиксированная длина команд, большое количество регистров, условное исполнение — Архитектура RISC-V: открытая спецификация, минимальное базовое ядро, расширяемость, отсутствие лицензионных отчислений — Сравнение производительности, энергопотребления, стоимости, экосистемы и области применения — Перспективы: RISC-V в серверах и суперкомпьютерах, вызовы Intel/AMD
16	Будущие архитектуры вычислительных систем (квантовые, нейроморфные, оптические) Содержание учебного материала: — Квантовые вычисления: кубиты, суперпозиция, запутанность, вентили (Hadamard, CNOT, Toffoli), декогеренция — Квантовые архитектуры: сверхпроводящие кубиты (Google, IBM), ионные ловушки (IonQ), топологические (Microsoft) — Нейроморфные архитектуры: спайковые нейронные сети (SNN), мемристоры, чипы Loihi (Intel) и TrueNorth (IBM) — Оптические вычисления: фотонные процессоры, оптическая память, преимущества скорости и энергопотребления — Сравнение классических и новых архитектур: применимость (криптография, оптимизация, нейросети) и горизонты зрелости

4.2. Занятия семинарского типа.

Лабораторные работы

№ п/п	Наименование лабораторных работ / краткое содержание
1	Исследование фон-неймановской и гарвардской архитектур на эмуляторах В результате выполнения лабораторной работы студент получит навыки сравнения фон-неймановской и гарвардской архитектур с использованием эмуляторов (например, MARIE Simulator и AVR Simulator), анализа разделения шин команд и данных, оценки влияния конфликта доступа к

№ п/п	Наименование лабораторных работ / краткое содержание
	памяти на производительность в фон-неймановской архитектуре при одновременной выборке команды и данных, а также составления отчета с выводами о применимости каждой архитектуры для различных классов задач.
2	Программирование на языке ассемблера для архитектуры x86-64 (базовые команды) В результате выполнения лабораторной работы студент получит навыки написания простейших программ на ассемблере NASM для архитектуры x86-64 (команды MOV, ADD, SUB, MUL, DIV, CMP, JMP), работы с регистрами общего назначения (RAX, RBX, RCX, RDX, RSI, RDI, R8–R15), организации условных переходов и циклов, отладки программ с использованием GDB (просмотр регистров, пошаговое выполнение, точки останова), а также расчета количества тактов при различных способах адресации.
3	Изучение конвейера команд на симуляторе (MIPS или RISC-V) В результате выполнения лабораторной работы студент получит навыки работы с симулятором конвейерного процессора (например, MIPS Pipeline Simulator или Ripes для RISC-V), анализа прохождения команд через стадии конвейера (IF, ID, EX, MEM, WB), выявления конфликтов по данным (RAW, WAR, WAW) и управляющих конфликтов (при условных переходах), применения методов разрешения конфликтов (форвардинг, вставка NOP-операций, спекулятивное выполнение), а также измерения влияния задержек на общую производительность.
4	Исследование SIMD-инструкций (AVX/SSE) на примере обработки массивов В результате выполнения лабораторной работы студент получит навыки написания программ на C/C++ с использованием встроенных SIMD-инструкций (intrinsics для SSE/AVX) для операций над векторами (сложение, умножение, скалярное произведение), сравнения производительности скалярной (поэлементной) и векторной обработки массивов различной длины (от 64 до 10? элементов), измерения ускорения (speedup) и эффективности использования SIMD, а также анализа влияния выравнивания данных (alignment) на скорость выполнения.
5	Программирование на CUDA для архитектуры GPU (матричное умножение) В результате выполнения лабораторной работы студент получит навыки написания ядра CUDA для умножения матриц, организации сетки блоков и потоков (grid и block), использования разделяемой памяти (shared memory) для уменьшения глобальных обращений к памяти, измерения времени выполнения на CPU и GPU для матриц размером от 128?128 до 2048?2048, анализа ускорения (speedup) и узких мест (ограничения по пропускной способности памяти, загрузка вычислительных блоков), а также профилирования кода с использованием NVIDIA Nsight Systems.
6	Исследование когерентности кэшей (протокол MESI) в многопроцессорной системе В результате выполнения лабораторной работы студент получит навыки эмуляции работы многопроцессорной системы с когерентными кэшами (например, в симуляторе Gem5 или простом учебном симуляторе когерентности), анализа состояний кэш-линий (Modified, Exclusive, Shared, Invalid) при чтении и записи из разных ядер, наблюдения за служебным трафиком (инвалидации, обновления) между кэшами, объяснения явления «ложного разделения» (false sharing) на практическом примере (два ядра пишут в разные переменные одной кэш-линии), а также формулирования рекомендаций по оптимизации структур данных для многопоточных приложений.
7	Изучение иерархии памяти и влияния кэша на производительность В результате выполнения лабораторной работы студент получит навыки написания тестовой программы на C/C++ для измерения времени доступа к памяти с различными шаблонами доступа

№ п/п	Наименование лабораторных работ / краткое содержание
	(последовательный, с шагом 2/4/8/16/32/64/128/256 элементов, случайный доступ), построения графика зависимости времени доступа от размера рабочего набора данных (от 1 КБ до 64 МБ), идентификации размеров кэшей L1, L2, L3 по характерным ступенчатым перегибам, анализа влияния выравнивания на использование строки кэша (cache line), а также объяснения эффектов временной и пространственной локальности.
8	Исследование TLB и работы виртуальной памяти В результате выполнения лабораторной работы студент получит навыки написания программы, выполняющей доступ к большому массиву памяти (сотни мегабайт) с различной степенью локальности (последовательный доступ — высокая локальность, случайный доступ — низкая локальность), измерения количества промахов TLB через системные утилиты (perf stat на Linux), анализа роста времени доступа при случайном обходе страниц по сравнению с последовательным, изменения размера страницы (4 КБ, 2 МБ, 1 ГБ) через хуже-страницы (hugepages) и оценки уменьшения количества промахов TLB, а также эмуляции работы многоуровневых таблиц страниц (x86-64: 4 уровня) на упрощенных адресах.
9	Исследование систем прерываний на примере микроконтроллера (Arduino / STM32) В результате выполнения лабораторной работы студент получит навыки настройки аппаратных прерываний (например, по изменению уровня на пине, по таймеру, по завершению АЦП) на микроконтроллере AVR (Arduino Uno) или ARM Cortex-M (STM32), написания функций-обработчиков прерываний (ISR) и правил их минимализма (быстрый вход/выход, избегание длительных операций), анализа приоритетов прерываний и вложенности, оценки задержки от возникновения события до входа в обработчик (латентность), а также создания программы с фоновым циклом и прерыванием для измерения разницы в реакции системы.
10	Исследование прямого доступа к памяти (DMA) на примере микроконтроллера В результате выполнения лабораторной работы студент получит навыки настройки контроллера DMA на микроконтроллере (например, STM32 или ESP32) для передачи данных из АЦП в буфер в ОЗУ без участия процессора, сравнения производительности и энергопотребления при передаче блока данных через DMA и через программный цикл копирования, измерения максимальной частоты дискретизации (sample rate) при использовании DMA, анализа освобождения процессора для выполнения других задач во время DMA-передачи, а также настройки прерывания по завершении DMA-блока.
11	Исследование архитектуры NUMA: задержки доступа к локальной и удаленной памяти В результате выполнения лабораторной работы студент получит навыки выполнения на многопроцессорном NUMA-сервере (например, с двумя процессорами Intel Xeon или AMD EPYC) программы, которая привязывает потоки к разным ядрам (taskset / numactl на Linux) и измеряет задержку доступа к памяти, выделенной на локальном процессоре (local memory) и на удаленном (remote memory), сравнения латентности (обычно удаленная в 1.5–3 раза медленнее), анализа ухудшения производительности при неоптимальном распределении памяти (использование numastat, профилировщика), а также написания кода с NUMA-осведомленным выделением памяти (libnuma, mbind, set_mempolicy).
12	MPI-программирование для систем с распределенной памятью (кластер) В результате выполнения лабораторной работы студент получит навыки написания параллельной программы на языке C/C++/Fortran с использованием библиотеки MPI (Message Passing Interface) для вычисления числа ? методом Монте-Карло или численного интегрирования (метод

№ п/п	Наименование лабораторных работ / краткое содержание
	прямоугольников), организации обмена сообщениями между процессами (MPI_Send / MPI_Recv и коллективные операции MPI_Reduce, MPI_Bcast), запуска программы на нескольких узлах кластера (виртуального — на одном ПК через mpirun с несколькими процессами, либо на реальном удаленном кластере), измерения времени выполнения при разном количестве процессов (1, 2, 4, 8, 16) и расчета ускорения (speedup) и эффективности, а также анализа влияния накладных расходов на передачу данных.
13	OpenMP-программирование для систем с общей памятью (многоядерный CPU) В результате выполнения лабораторной работы студент получит навыки распараллеливания цикла (for pragma) на языке C/C++/Fortran с использованием директив OpenMP, применения различных режимов распределения итераций (static, dynamic, guided) и их влияния на балансировку нагрузки, организации критических секций (critical) и атомарных операций (atomic) для избежания гонок данных (data race), использования переменных с приватной копией (private, firstprivate, lastprivate), измерения ускорения на задачах с разной вычислительной плотностью (обработка изображений, умножение матриц, сортировка массива), а также анализа эффекта псевдо-распараллеливания (overhead при малом объеме вычислений на итерацию).
14	Сравнительный анализ производительности архитектур (x86 vs. ARM vs. RISC-V) В результате выполнения лабораторной работы студент получит навыки запуска одинакового набора тестов (например, бенчмарки CoreMark, Dhrystone, LINPACK, STREAM) на виртуальных или физических машинах с разными архитектурами (x86-64, ARM64 (например, Raspberry Pi или AWS Graviton), RISC-V (в эмуляторе QEMU или на плате SiFive)), сбора показателей производительности (операций в секунду, FLOPS, пропускной способности памяти), вычисления отношения производительность на ватт (при наличии данных о потреблении), построения сравнительных таблиц и гистограмм, а также формулирования выводов о применимости каждой архитектуры для веб-серверов, научных расчетов, встраиваемых систем.
15	Проектирование простого RISC-V процессора на языке Verilog/VHDL (в симуляторе) В результате выполнения лабораторной работы студент получит навыки описания базового RISC-V процессора (подмножество команд RV32I: ADD, SUB, LW, SW, BEQ, JAL) на языке описания аппаратуры (Verilog или VHDL) с использованием симулятора (например, Icarus Verilog или ModelSim), написания тестового окружения (testbench) для проверки выполнения программы из нескольких инструкций (например, вычисление суммы чисел от 1 до N), моделирования работы процессора (просмотр сигналов, состояний регистров, памяти), анализа работы конвейера (если реализован), а также сравнения полученного RISC-V ядра с эталонными реализациями (например, PicoRV32).
16	Анализ архитектуры вычислительной системы с помощью профилирования (perf, VTune) В результате выполнения лабораторной работы студент получит навыки профилирования реального приложения (например, умножение матриц, сжатие данных, компиляция кода) на архитектуре x86-64 с использованием инструментов Linux perf (perf stat, perf record, perf report) и/или Intel VTune Profiler (доступна бесплатная версия для некоммерческого использования), сбора счетчиков производительности (промахи кэша L1/L2/L3, промахи TLB, количество взятых/не взятых условных переходов, количество тактов, инструкций за такт — IPC), идентификации узких мест (bottlenecks) в коде (частые промахи кэша, неэффективная работа конвейера, слишком много ветвлений), формулирования предложений по оптимизации исходного кода (изменение структуры данных, перестановка циклов, развертка циклов (loop unrolling)), а также оформления отчета о профилировании с графиками и рекомендациями.

4.3. Самостоятельная работа обучающихся.

№ п/п	Вид самостоятельной работы
1	Работа с лекционным материалом
2	Подготовка к лабораторным работам
3	Выполнение курсовой работы.
4	Подготовка к промежуточной аттестации.
5	Подготовка к текущему контролю.

5. Перечень изданий, которые рекомендуется использовать при освоении дисциплины (модуля).

№ п/п	Библиографическое описание	Место доступа
1	Программирование микроконтроллеров семейства CORTEX-M: Учебное пособие для студентов направлений «Информатика и вычислительная техника» и «Информационная безопасность» / Шамров М. И., 2020. - 88 с.	URL: https://e.lanbook.com/book/175969(Электронно-библиотечная система Лань)(дата обращения 03.03.2026)
2	Архитектура и структурная организация микроконтроллеров семейства CORTEX-M: Учебное пособие для студентов направлений «Информатика и вычислительная техника» и «Информационная безопасность» / Шамров М. И.,; 2019. - 62 с.	URL: https://e.lanbook.com/book/175725(Электронно-библиотечная система Лань)(дата обращения 03.03.2026)
3	Применение коммутаторов в современных сетях передачи информации: Учебно-методическое пособие для направления 27.03.04 «Управление в технических системах» профиля «Управления и информатика в технических системах» (бакалавры) очная и очно-заочная (вечерняя) формы обучения/ Антонов Д. А., Ермакова А. Е., Иконников С. Е.. РУТ(МИИТ), 2021. - 94 с.	URL: https://e.lanbook.com/book/269750 Электронно-библиотечная система Лань)(дата обращения 03.03.2026)

6. Перечень современных профессиональных баз данных и информационных справочных систем, которые могут использоваться при освоении дисциплины (модуля).

Форум специалистов по информационным технологиям http://citforum.ru/

Интернет-университет информационных технологий http://www.intuit.ru/

Тематический форум по информационным технологиям http://habrahabr.ru/

7. Перечень лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства, необходимого для освоения дисциплины (модуля).

Microsoft Windows
Microsoft Office

8. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю).

Для проведения аудиторных занятий и самостоятельной работы требуются:

- Рабочее место преподавателя с персональным компьютером, подключённым к сетям INTERNET.
- Специализированная лекционная аудитория с мультимедиа аппаратурой и интерактивной доской.
-Компьютерный класс с кондиционером. Рабочие места студентов в компьютерном классе, подключённые к сетям INTERNET

Для проведения практических занятий:

-компьютерный класс; кондиционер; компьютеры.

-В случае проведении занятий с применением электронного обучения и дистанционных образовательных технологий необходимо наличие компьютерной техники, для организации коллективных и индивидуальных форм общения педагогических работников со студентами, посредством используемых средств коммуникации.

Допускается замена оборудования его виртуальными аналогами.

9. Форма промежуточной аттестации:

Зачет в 6 семестре.

10. Оценочные материалы.

Оценочные материалы, применяемые при проведении промежуточной аттестации, разрабатываются в соответствии с локальным нормативным актом РУТ (МИИТ).

Авторы:

доцент, к.н. кафедры
«Вычислительные системы и
квантовые коммуникации» Я.М. Голдовский

Согласовано:

Заведующий кафедрой ВССиИБ Б.В. Желенков

Председатель учебно-методической
комиссии Н.А. Андриянова