Приложение 4

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Нижегородский государственный университет
им. Н.И. Лобачевского»

Институт информационных технологий, математики и механики

УТВЕРЖДЕНО

решением Ученого совета ННГУ

протокол № 10 от 02.12.2024 г.

Рабочая программа дисциплины

Введение в сопряженное проектирование программного обеспечения и аппаратуры на программируемых логических интегральных схемах

Уровень высшего образования
Бакалавриат

Направление подготовки / специальность
02.03.02 - Фундаментальная информатика и информационные технологии

Направленность образовательной программы
Сопряженная разработка программного и аппаратного обеспечения

Форма обучения
очная

г. Нижний Новгород

2025 год начала подготовки

1. Место дисциплины в структуре ОПОП

Дисциплина Б1.В.12.05 Введение в сопряженное проектирование программного обеспечения и аппаратуры на программируемых логических интегральных схемах относится к части, формируемой участниками образовательных отношений образовательной программы.

2. Планируемые результаты обучения по дисциплине, соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями и индикаторами достижения компетенций)

Формируемые компетенции (код, содержание компетенции)	**Планируемые результаты обучения по дисциплине (модулю), в соответствии с индикатором достижения компетенции**		**Наименование оценочного средства**	
	Индикатор достижения компетенции (код, содержание индикатора)	**Результаты обучения по дисциплине**	**Для текущего контроля успеваемости**	**Для промежуточной аттестации**
ПК-3: Способен создавать и исследовать новые математические модели в естественных науках, промышленности и бизнесе, с учетом возможностей современных информационных технологий и программирования и *компьютерной техники*	*ПК-3.1: Знает методы анализа и исследования математических моделей в области фундаментальной информатики и информационных технологий* *ПК-3.2: Умеет определять ключевые свойства и ограничения системы*	*ПК-3.1:* *Знать:* *- принципы аппаратно-программной декомпозиции, архитектуру ПЛИС (LUT, BRAM, DSP, PLL) и SoC FPGA (процессор + логика);* *- базовые конструкции Verilog (модули, always, конечные автоматы) и методы верификации;* *- интерфейсы AXI/Avalon, порядок создания пользовательских IP-блоков;* *- средства разработки (Vitis, Quartus, Libero), приёмы профилирования и временного анализа;* *- основы High-Level Synthesis (директивы, прагмы, генерация RTL из C/C++).* *ПК-3.2:* *Уметь:* *- оценивать выигрыш от переноса вычислительной модели в аппаратуру и выбирать границу HW/SW;* *- проектировать на Verilog конвейерные схемы, конечные автоматы, память; писать тестовые стенды;* *- разрабатывать IP с AXI-интерфейсом, интегрировать его в блочный*	*Тест* *Практическое задание*	*Экзамен:* *Контрольные вопросы* *Практическое задание*

		дизайн и вызывать из C-кода под bare-metal/Linux; *- профилировать систему (счётчики, ILA, временные метки) и оптимизировать код (конвейеризация, параллелизм, замена на DSP/BRAM);* *- применять HLS для быстрого прототипирования ядер и анализировать отчёты по ресурсам/частоте;* *- создавать математические модели (фильтры, нейросети, симуляции) с учётом возможности ускорения на ПЛИС.*		

3. Структура и содержание дисциплины

3.1 Трудоемкость дисциплины

	очная
Общая трудоемкость, з.е.	**4**
Часов по учебному плану	**144**
в том числе	
аудиторные занятия (контактная работа):	
- занятия лекционного типа	**32**
- занятия семинарского типа (практические занятия / лабораторные работы)	**32**
- КСР	**2**
самостоятельная работа	**42**
Промежуточная аттестация	**36** **Экзамен**

3.2. Содержание дисциплины

(структурированное по темам (разделам) с указанием отведенного на них количества академических часов и виды учебных занятий)

Наименование разделов и тем дисциплины	Всего (часы)	в том числе			
		Контактная работа (работа во взаимодействии с преподавателем), часы из них			Самостоятельная работа обучающегося, часы
		Занятия лекционного типа	Занятия семинарского типа (практические занятия/лабораторные работы), часы	Всего	
	очно	очно	очно	очно	очно
Тема 1. Введение в hardware/software co-design	13	4	4	8	5

Тема 2. Архитектура и технология ПЛИС	13	4	4	8	5
Тема 3. Проектирование цифровых схем на Verilog	13	4	4	8	5
Тема 4. SoC FPGA и процессорные системы	13	4	4	8	5
Тема 5. Разработка пользовательских IP-блоков	13	4	4	8	5
Тема 6. Программное обеспечение для SoC FPGA	13	4	4	8	5
Тема 7. Профилирование и оптимизация	14	4	4	8	6
Тема 8. High-Level Synthesis	14	4	4	8	6
Аттестация	36				
КСР	2			2	
Итого	144	32	32	66	42

Содержание разделов и тем дисциплины

1. Введение в hardware/software co-design

Тема 1.1. Основы сопряженного проектирования

· Концепция hardware/software co-design: мотивация и история
· Гетерогенные вычислительные системы: CPU + FPGA, CPU + GPU, SoC
· Разделение функциональности между ПО и аппаратурой
· Критерии выбора: производительность, энергопотребление, гибкость, стоимость
· Методологии co-design: Y-chart, Ptolemy
· Design space exploration
· Примеры применения: обработка сигналов, криптография, нейросети, HPC

Тема 1.2. Эволюция вычислительных архитектур

· Специализированные ускорители vs программируемые
· Domain-specific architectures
· Сравнение платформ: FPGA, ASIC, GPU, NPU/TPU
· Место ПЛИС в ландшафте вычислительных систем

2. Архитектура и технология ПЛИС

Тема 2.1. Структура современных ПЛИС

· Логические блоки: LUT (Look-Up Tables), flip-flops, multiplexers
· Programmable interconnect: routing resources, switch matrix
· Конфигурационная память: SRAM-based vs flash-based
· DSP-блоки: умножители, MAC units, floating-point
· Блоки памяти: BRAM, UltraRAM, distributed RAM
· Высокоскоростные трансиверы: SerDes, GTH/GTY
· Clock management: PLL, MMCM, clock buffers
· Hard IP cores: PCIe, Ethernet, DDR controllers

Тема 2.2. Обзор платформ ПЛИС

· Xilinx/AMD: линейки 7-series, UltraScale, UltraScale+, Versal
· Intel/Altera: Stratix, Arria, Cyclone семейства
· Lattice и Microchip FPGA (обзор)
· SoC FPGA: Zynq-7000, Zynq UltraScale+ MPSoC, Intel SoC FPGA
· Adaptive compute acceleration platforms (ACAP)
· Сравнение характеристик: логические ресурсы, DSP, память, I/O

Тема 2.3. Инструменты разработки

· Xilinx Vivado: структура проекта, design flow
· Intel Quartus Prime: особенности и отличия
· Xilinx Vitis: unified software platform для HW и SW

· High-Level Synthesis (HLS): Vitis HLS, Intel HLS Compiler
· Симуляторы: ModelSim, Questa, Vivado Simulator
· IP integrator и block design

3. Проектирование цифровых схем на Verilog

Тема 3.1. Основы языка Verilog
· Синтаксис и структура модулей
· Типы данных: wire, reg, integer, real
· Операторы и выражения
· Комбинационная логика: assign, always @(*)
· Последовательная логика: always @(posedge clk)
· Блокирующее vs неблокирующее присваивание
· Параметризация модулей
· Generate конструкции

Тема 3.2. Проектирование синхронных схем
· Принципы синхронного дизайна
· Clock domain crossing и метастабильность
· Reset стратегии: синхронный vs асинхронный
· Конечные автоматы (FSM): Mealy и Moore
· Конвейеризация (pipelining): теория и практика
· Ретайминг для повышения частоты
· Управление задержками: timing constraints

Тема 3.3. Оптимизация ресурсов ПЛИС
· Resource utilization: LUT, FF, BRAM, DSP
· Area vs speed trade-offs
· Sharing ресурсов: multiplexing, resource sharing
· Использование DSP-блоков эффективно
· Memory inference: распределённая vs блочная
· Retiming и register balancing

Тема 3.4. Верификация HDL-кода
· Testbench структура и методология
· Behavioral simulation
· Timing simulation и gate-level simulation
· Assertion-based verification: SystemVerilog assertions
· Coverage metrics: code, functional, toggle
· Constrained random verification

4. SoC FPGA и процессорные системы

Тема 4.1. Архитектура Zynq SoC
· Processing System (PS): ARM Cortex-A cores
· Programmable Logic (PL): интеграция с FPGA fabric
· Интерфейсы PS-PL: AXI GP, AXI HP, AXI ACP
· Zynq UltraScale+ MPSoC: APU, RPU, GPU, PMU
· Boot process и конфигурация
· Memory map и address space

Тема 4.2. Шины и интерфейсы AXI
· AXI4 protocol: master, slave, channels
· AXI4-Lite для регистровых интерфейсов
· AXI4-Stream для потоковых данных
· AXI4-Full для высокопроизводительных передач
· Транзакции: read, write, burst

· Handshaking: ready/valid протокол
· AXI Interconnect: crossbar, SmartConnect
· DMA controllers и Scatter-Gather
Тема 4.3. IP Integration
· IP Catalog в Vivado: готовые IP-блоки
· IP Integrator и block design
· AXI GPIO, AXI Timer, AXI UART
· Создание пользовательских AXI IP
· IP packaging и реиспользование
· Address editor и memory mapping
5. Разработка пользовательских IP-блоков
Тема 5.1. Custom AXI IP creation
· Структура AXI IP: slave logic, user logic
· AXI slave interface: decode, read/write logic
· Register interface для управления
· Status и control регистры
· Interrupt generation и handling
· Packaging IP для реиспользования
Тема 5.2. Streaming и DMA IP
· AXI-Stream interfaces для данных
· FIFO buffers и flow control
· DMA engines: simple DMA, Scatter-Gather DMA
· Data movers и packet processors
· Zero-copy transfers
· Performance optimization
6. Программное обеспечение для SoC FPGA
Тема 6.1. Разработка ПО в Vitis
· Vitis unified IDE: platform, application, system
· Bare-metal vs Linux application
· Board Support Package (BSP)
· Device tree и hardware description
· Cross-compilation toolchain
· Debugging: JTAG, GDB, printf
Тема 6.2. Драйверы устройств
• Linux device driver model
• Character devices и /dev nodes
• Memory-mapped I/O в Linux
• UIO (Userspace I/O) драйверы
• Kernel drivers: probe, remove, ioctl
• DMA mapping и cache coherency
• Interrupt handling в драйверах
Тема 6.3. Взаимодействие SW с HW
· Register access: mmap, devmem
· Linux frameworks: GPIO, SPI, I2C
· Buffer management для DMA
· Synchronization: polling vs interrupts
· Latency и throughput optimization
Тема 6.4. Операционные системы для SoC FPGA
· Embedded Linux: buildroot, Yocto, PetaLinux
· RTOS: FreeRTOS, Zephyr на ARM cores

· Bare-metal applications
· Multi-OS: Linux на Cortex-A + RTOS на Cortex-R
· Hypervisors для FPGA SoC
7. Профилирование и оптимизация
Тема 7.1. Анализ производительности ПО
· Profiling tools: gprof, perf, Vitis Analyzer
· Hotspot identification
· Cache miss analysis
· Branch prediction и pipeline stalls
· Memory bandwidth bottlenecks
· Amdahl's law применительно к ускорению
· Тема 7.2. Выбор кандидатов для аппаратного ускорения (3 ч. лекции)
· Критерии выбора: compute intensity, parallelism, regularity
· Roofline model для анализа
· Compute-bound vs memory-bound алгоритмы
· Granularity ускорения: функция, loop, kernel
· Коммуникационные накладные расходы
· Break-even analysis: когда ускорение эффективно
Тема 7.3. Partitioning стратегии
· Функциональное разделение HW/SW
· Temporal vs spatial partitioning
· Co-processing models: offload, pipeline, feedback
· Data movement minimization
· Overlapping communication и computation
8. High-Level Synthesis
Тема 8.1. Основы HLS
• Концепция синтеза из C/C++
· Vitis HLS: workflow и инструменты
· Pragmas для оптимизации: pipeline, unroll, array_partition
· Интерфейсы: AXI, AXI-Stream, FIFO
· Latency, Throughput, Initiation Interval
· Resource estimates и scheduling
·
Тема 8.2. Оптимизация в HLS
· Loop optimizations: pipelining, unrolling, flattening
· Array partitioning для параллельного доступа
· Dataflow optimization для task-level parallelism
· Memory optimization: двухпортовая память, banking
· Arbitrary precision types: ap_int, ap_fixed
· Co-simulation и export RTL

4. Учебно-методическое обеспечение самостоятельной работы обучающихся

Самостоятельная работа обучающихся включает в себя подготовку к контрольным вопросам и заданиям для текущего контроля и промежуточной аттестации по итогам освоения дисциплины приведенным в п. 5.

Самостоятельная работа студентов при изучении дисциплины проводится с целью углубления знаний по дисциплине и предусматривает:

- повторение пройденного учебного материала, чтение рекомендованной литературы;
- выполнение общих и индивидуальных домашних заданий;
- работу с электронными источниками;
- подготовку к сдаче формы промежуточной аттестации.

Цель самостоятельной работы - формирование навыков непрерывного самообразования и профессионального совершенствования.

Самостоятельная работа способствует формированию аналитического и творческого мышления, совершенствует способы организации исследовательской деятельности, воспитывает целеустремленность, системность и последовательность в работе студентов, развивает у них навык завершать начатую работу.

Основные виды самостоятельной работы студентов:

- работа с основной и дополнительной литературой;
- изучение категориального аппарата дисциплины;
- самостоятельное изучение тем дисциплины;
- подготовка к экзамену;
- работа в библиотеке;
- изучение сайтов по темам дисциплины в сети Интернет.

Работа с основной и дополнительной литературой

Изучение рекомендованной литературы следует начинать с учебников и учебных пособий, затем переходить к научным монографиям и материалам периодических изданий. Работа с литературой предусматривает конспектирование наиболее актуальных и познавательных материалов. Это не только мобилизует внимание, но и способствует более глубокому осмыслению материала, его лучшему запоминанию, а также позволяет студентам проводить систематизацию и сравнительный анализ изучаемой информации. Таким образом, конспектирование – одна из основных форм самостоятельного труда, которая требует от студента активно работать с учебной литературой и не ограничиваться конспектом лекций. Студент должен уметь самостоятельно подбирать необходимую литературу для учебной и научной работы, уметь обращаться с предметными каталогами и библиографическим справочником библиотеки.

Изучение категориального аппарата дисциплины

Изучение и осмысление экономических категорий требует проработки лекционного материала, выполнения практических заданий, изучение словарей, энциклопедий, справочников. Индивидуальная самостоятельная работа студента направлена на овладение и грамотное применение экономической терминологии в области компьютерного моделирования.

Самостоятельное изучение тем дисциплины

Особое место отводится самостоятельной проработке студентами отдельных разделов и тем изучаемой дисциплины. Такой подход вырабатывает у студентов инициативу, стремление к увеличению объема знаний, умений и навыков, всестороннего овладения способами и приемами профессиональной деятельности.

Изучение вопросов определенной темы направлено на более глубокое усвоение основных категорий экономической теории, понимание экономических процессов, происходящих в обществе, совершенствование навыка анализа теоретического и эмпирического материала.

Подготовка к экзамену

Промежуточная аттестация студентов по дисциплине проходит в виде экзамена, предусматривающего оценку. Условием успешного прохождения промежуточной аттестации является систематическая работа студента в течение семестра. В этом случае подготовка к

экзамену является систематизацией всех полученных знаний по данной дисциплине. Рекомендуется внимательно изучить перечень вопросов к экзамену, а также использовать в процессе обучения программу, учебно-методический комплекс, другие методические материалы.
Желательно спланировать троекратный просмотр материала перед экзаменом. Во-первых, внимательное чтение с осмыслением, подчеркиванием и составлением краткого плана ответа. Во-вторых, повторная проработка наиболее сложных вопросов. В-третьих, быстрый просмотр материала или планов ответов для его систематизации в памяти.
Изучение сайтов по темам дисциплины в сети Интернет
Ресурсы Интернет являются одним из альтернативных источников быстрого поиска требуемой информации. Их использование возможно для получения основных и дополнительных сведений по изучаемым материалам. Необходимо помнить об оформлении ссылок на Интернет-источники.
Для повышения эффективности самостоятельной работы студентов преподавателю целесообразно использовать следующие виды деятельности:
- консультации,
- выдача заданий на самостоятельную работу,
- информационное обеспечение обучения,
- контроль качества самостоятельной работы студентов.

5. Фонд оценочных средств для текущего контроля успеваемости и промежуточной аттестации по дисциплине (модулю)

5.1 Типовые задания, необходимые для оценки результатов обучения при проведении текущего контроля успеваемости с указанием критериев их оценивания:

5.1.1 Типовые задания (оценочное средство - Тест) для оценки сформированности компетенции ПК-3:

- Что из перечисленного является основным критерием для переноса функции из программной части в аппаратную?
- A) Простота реализации
- B) Высокая вычислительная интенсивность и повторяемость
- C) Частое обращение к базе данных
- D) Малый объём кода
- Какой блок внутри ПЛИС предназначен для выполнения умножений с накоплением?
- A) LUT
- B) BRAM

- C) DSP-слайс
- D) PLL
- В Verilog какой тип данных используется для описания комбинационной логики внутри `always @(*)`?
- A) reg
- B) wire
- C) integer
- D) real
- Какой интерфейс SoC FPGA обеспечивает связь программируемой логики с процессорной подсистемой с низкой задержкой и возможностью burst-передач?
- A) I2C
- B) SPI
- C) AXI4
- D) JTAG
- Назовите два основных режима работы AXI4-интерфейса, используемых для обмена с пользовательскими IP.
- Что такое HLS (High-Level Synthesis)?
- A) Синтез аналоговых схем
- B) Преобразование C/C++ в RTL-описание
- C) Оптимизация размещения на кристалле
- D) Генерация кода для GPU
- Какой инструмент в Vivado позволяет в реальном времени наблюдать сигналы внутри ПЛИС без использования внешнего осциллографа?
- A) Logic Analyzer (ILA)
- B) Virtual I/O (VIO)
- C) High-Level Synthesis

- D) Timing Analyzer
- Какая прагма в HLS (Vitis/Vivado HLS) используется для указания, что цикл должен быть полностью развёрнут?
- A) #pragma HLS unroll
- B) #pragma HLS pipeline
- C) #pragma HLS array_partition
- D) #pragma HLS interface
- Какие факторы необходимо учитывать при выборе между ручным RTL-проектированием и HLS? (выберите два)
- A) Только стоимость лицензии
- B) Требуемая тактовая частота
- C) Сложность отладки и время разработки
- D) Количество триггеров в проекте
- В чём основное преимущество использования DMA в системах SoC FPGA?
- Какой тип конечного автомата в Verilog считается более безопасным для синтеза (менее подвержен гонкам)?
- A) Mealy
- B) Moore
- C) Одновременный
- D) Кодированный «one-hot»
- Как называется процесс анализа задержек распространения сигналов в ПЛИС для проверки соблюдения временных ограничений?
- Какая структура памяти в ПЛИС обычно используется для организации FIFO между процессором и IP-блоком?
- A) LUT-память
- B) BRAM
- C) Распределительные регистры

- D) Внешняя DRAM
- Какой язык используется в среде Vitis для описания аппаратных ускорителей в стиле OpenCL?
- Приведите пример задачи из естественных наук, которую эффективно ускорить на ПЛИС.

Критерии оценивания (оценочное средство - Тест)

Оценка	Критерии оценивания
превосходно	96-100% правильных ответов
отлично	81-95% правильных ответов
очень хорошо	76-80% правильных ответов
хорошо	61-75% правильных ответов
удовлетворительно	46-60% правильных ответов
неудовлетворительно	31-45% правильных ответов
плохо	30% и меньше правильных ответов

5.1.2 Типовые задания (оценочное средство - Практическое задание) для оценки сформированности компетенции ПК-3:

Задание 1. Проектирование на Verilog

Разработать модуль на Verilog, реализующий **скользящее среднее (moving average)** для потока данных шириной 8 бит (беззнаковое). Длина окна – 4 отсчёта. Модуль должен иметь:

- вход: clk, rst, data_in (8 bit), valid_in
- выход: data_out (8 bit), valid_out
- архитектура – конвейерная, использование регистрового сдвига.

Критерии: синтаксически корректный код, тестовый стенд (self-checking), сообщение о прохождении симуляции.

Задание 2. Создание IP-блока с AXI-Lite

Обернуть модуль из задания 1.1 в AXI-Lite интерфейс. Добавить управляющий регистр **scale** (для масштабирования результата). Реализовать на C тест (bare-metal), который записывает коэффициенты, отправляет тестовые данные в IP через AXI-Lite и читает результат.

Критерии: работающее взаимодействие, правильное чтение/запись регистров.

Задание 3. HLS прототип

Написать функцию на C++ для HLS (Vivado/Vitis), реализующую **матричное умножение 8×8** (фиксированная точка 16 бит). Использовать прагмы `#pragma HLS pipeline` и `#pragma HLS array_partition` для полного развёртывания внутренних циклов. Сгенерировать RTL, предоставить отчёт о ресурсах (LUT, BRAM, DSP) и достигнутой частоте.

Критерии: код HLS, скриншот summary отчёта.

Задание 4. Профилирование (симуляция)

Добавить в модуль из задания 1.1 счётчик тактов между valid_in и valid_out (задержка). Написать тест, подающий 100 случайных отсчётов, и вычислить среднюю задержку. Объяснить, почему она не всегда равна 4.

Критерии: наличие счётчика, корректный вывод статистики, объяснение (зависит от valid_in, начальная инициализация).

Критерии оценивания (оценочное средство - Практическое задание)

Оценка	Критерии оценивания
превосходно	Задание выполнено в полном объеме (все поставленные задачи решены), ответ логичен и обоснован, обучающийся отвечает четко и последовательно, показывает глубокое знание основного и дополнительного материала
отлично	Задание выполнено в полном объеме (все поставленные задачи решены), ответ логичен и обоснован, обучающийся отвечает четко и последовательно, показывает глубокое знание основного материала
очень хорошо	Задание выполнено в полном объеме (все поставленные задачи решены), ответ логичен и обоснован, обучающийся отвечает четко и последовательно, показывает глубокое знание материала, допущено не более 2 неточностей непринципиального характера
хорошо	Задание выполнено в полном объеме (все поставленные задачи решены), ответ логичен и обоснован, допущены неточности непринципиального характера, но обучающийся показывает систему знаний по теме своими ответами на поставленные вопросы
удовлетворительно	Задание выполнено не в полном объеме (решено более 50% поставленных задач), но обучающийся допускает ошибки, нарушена последовательность ответа, но в целом раскрывает содержание основного материала
неудовлетворительно	Задание выполнено не в полном объеме (решено менее 50% поставленных задач), обучающийся дает неверную информацию при ответе на поставленные задачи, допускает грубые ошибки при толковании материала, демонстрирует незнание основных терминов и понятий

Оценка	Критерии оценивания
плохо	Задание не выполнено, обучающийся демонстрирует полное незнание материала

5.2. Описание шкал оценивания результатов обучения по дисциплине при промежуточной аттестации

Шкала оценивания сформированности компетенций

Уровень сформированности компетенций (индикатора достижения компетенций)	**плохо**	**неудовлетворительно**	**удовлетворительно**	**хорошо**	**очень хорошо**	**отлично**	**превосходно**
	не зачтено		**зачтено**				
Знания	Отсутствие знаний теоретического материала. Невозможность оценить полноту знаний вследствие отказа обучающегося от ответа	Уровень знаний ниже минимальных требований. Имели место грубые ошибки	Минимально допустимый уровень знаний. Допущено много негрубых ошибок	Уровень знаний в объеме, соответствующем программе подготовки. Допущено несколько негрубых ошибок	Уровень знаний в объеме, соответствующем программе подготовки. Допущено несколько несущественных ошибок	Уровень знаний в объеме, соответствующем программе подготовки. Ошибок нет.	Уровень знаний в объеме, превышающем программу подготовки.
Умения	Отсутствие минимальных умений. Невозможность оценить наличие умений вследствие отказа обучающегося от ответа	При решении стандартных задач не продемонстрированы основные умения. Имели место грубые ошибки	Продемонстрированы основные умения. Решены типовые задачи с негрубыми ошибками. Выполнены все задания, но не в полном объеме	Продемонстрированы все основные умения. Решены все основные задачи с негрубыми ошибками. Выполнены все задания в полном объеме, но некоторые с недочетами	Продемонстрированы все основные умения. Решены все основные задачи. Выполнены все задания в полном объеме, но некоторые с недочетами.	Продемонстрированы все основные умения. Решены все основные задачи с отдельными несущественными недочетами, выполнены все задания в полном объеме	Продемонстрированы все основные умения. Решены все основные задачи. Выполнены все задания, в полном объеме без недочетов
Навыки	Отсутствие базовых навыков. Невозможность оценить наличие навыков вследствие отказа обучающегося от ответа	При решении стандартных задач не продемонстрированы базовые навыки. Имели место грубые ошибки	Имеется минимальный набор навыков для решения стандартных задач с некоторыми	Продемонстрированы базовые навыки при решении стандартных задач с некоторыми недочетами	Продемонстрированы базовые навыки при решении стандартных задач без ошибок и недочетов	Продемонстрированы навыки при решении нестандартных задач без ошибок и недочетов	Продемонстрирован творческий подход к решению нестандартных задач

			недочетами				

Шкала оценивания при промежуточной аттестации

Оценка		Уровень подготовки
зачтено	**превосходно**	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «превосходно», продемонстрированы знания, умения, владения по соответствующим компетенциям на уровне выше предусмотренного программой
	отлично	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «отлично».
	очень хорошо	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «очень хорошо»
	хорошо	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «хорошо».
	удовлетворительно	Все компетенции (части компетенций), на формирование которых направлена дисциплина, сформированы на уровне не ниже «удовлетворительно», при этом хотя бы одна компетенция сформирована на уровне «удовлетворительно»
не зачтено	**неудовлетворительно**	Хотя бы одна компетенция сформирована на уровне «неудовлетворительно».
	плохо	Хотя бы одна компетенция сформирована на уровне «плохо»

5.3 Типовые контрольные задания или иные материалы, необходимые для оценки результатов обучения на промежуточной аттестации с указанием критериев их оценивания:

5.3.1 Типовые задания (оценочное средство - Контрольные вопросы) для оценки сформированности компетенции ПК-3

1. Концепция hardware/software co-design: мотивация и история
2. Гетерогенные вычислительные системы: CPU + FPGA, CPU + GPU, SoC
3. Разделение функциональности между ПО и аппаратурой
4. Критерии выбора: производительность, энергопотребление, гибкость, стоимость
5. Методологии co-design: Y-chart, Ptolemy
6. Design space exploration
7. Примеры применения: обработка сигналов, криптография, нейросети, HPC
8. Специализированные ускорители vs программируемые
9. Domain-specific architectures
10. Сравнение платформ: FPGA, ASIC, GPU, NPU/TPU
11. Место ПЛИС в ландшафте вычислительных систем
12. Логические блоки: LUT (Look-Up Tables), flip-flops, multiplexers
13. Programmable interconnect: routing resources, switch matrix
14. Конфигурационная память: SRAM-based vs flash-based
15. DSP-блоки: умножители, MAC units, floating-point

16. Блоки памяти: BRAM, UltraRAM, distributed RAM
17. Высокоскоростные трансиверы: SerDes, GTH/GTY
18. Clock management: PLL, MMCM, clock buffers
19. Hard IP cores: PCIe, Ethernet, DDR controllers
20. Xilinx/AMD: линейки 7-series, UltraScale, UltraScale+, Versal
21. Intel/Altera: Stratix, Arria, Cyclone семейства
22. Lattice и Microchip FPGA (обзор)
23. SoC FPGA: Zynq-7000, Zynq UltraScale+ MPSoC, Intel SoC FPGA
24. Adaptive compute acceleration platforms (ACAP)
25. Сравнение характеристик: логические ресурсы, DSP, память, I/O
26. Xilinx Vivado: структура проекта, design flow
27. Intel Quartus Prime: особенности и отличия
28. Xilinx Vitis: unified software platform для HW и SW
29. High-Level Synthesis (HLS): Vitis HLS, Intel HLS Compiler
30. Симуляторы: ModelSim, Questa, Vivado Simulator
31. IP integrator и block design
32. Синтаксис и структура модулей
33. Типы данных: wire, reg, integer, real
34. Операторы и выражения
35. Комбинационная логика: assign, always @(*)
36. Последовательная логика: always @(posedge clk)
37. Блокирующее vs неблокирующее присваивание
38. Параметризация модулей
39. Generate конструкции
40. Принципы синхронного дизайна
41. Clock domain crossing и метастабильность
42. Reset стратегии: синхронный vs асинхронный
43. Конечные автоматы (FSM): Mealy и Moore
44. Конвейеризация (pipelining): теория и практика
45. Ретайминг для повышения частоты
46. Управление задержками: timing constraints
47. Resource utilization: LUT, FF, BRAM, DSP
48. Area vs speed trade-offs
49. Sharing ресурсов: multiplexing, resource sharing
50. Использование DSP-блоков эффективно
51. Memory inference: распределённая vs блочная
52. Retiming и register balancing
53. Testbench структура и методология
54. Behavioral simulation
55. Timing simulation и gate-level simulation
56. Assertion-based verification: SystemVerilog assertions
57. Coverage metrics: code, functional, toggle
58. Constrained random verification

Processing System (PS): ARM Cortex-A cores

59. Programmable Logic (PL): интеграция с FPGA fabric
60. Интерфейсы PS-PL: AXI GP, AXI HP, AXI ACP

61. Zynq UltraScale+ MPSoC: APU, RPU, GPU, PMU
62. Boot process и конфигурация
63. Memory map и address space
64. AXI4 protocol: master, slave, channels
65. AXI4-Lite для регистровых интерфейсов
66. AXI4-Stream для потоковых данных
67. AXI4-Full для высокопроизводительных передач
68. Транзакции: read, write, burst
69. Handshaking: ready/valid протокол
70. AXI Interconnect: crossbar, SmartConnect
71. DMA controllers и Scatter-Gather
72. IP Catalog в Vivado: готовые IP-блоки
73. IP Integrator и block design
74. AXI GPIO, AXI Timer, AXI UART
75. Создание пользовательских AXI IP
76. IP packaging и реиспользование
77. Address editor и memory mapping
78. Структура AXI IP: slave logic, user logic
79. AXI slave interface: decode, read/write logic
80. Register interface для управления
81. Status и control регистры
82. Interrupt generation и handling
83. Packaging IP для реиспользования
84. AXI-Stream interfaces для данных
85. FIFO buffers и flow control
86. DMA engines: simple DMA, Scatter-Gather DMA
87. Data movers и packet processors
88. Zero-copy transfers
89. Performance optimization
90. Vitis unified IDE: platform, application, system
91. Bare-metal vs Linux application
92. Board Support Package (BSP)
93. Device tree и hardware description
94. Cross-compilation toolchain
95. Debugging: JTAG, GDB, printf
96. Linux device driver model
97. Character devices и /dev nodes
98. Memory-mapped I/O в Linux
99. UIO (Userspace I/O) драйверы
100. Kernel drivers: probe, remove, ioctl
101. DMA mapping и cache coherency
102. Interrupt handling в драйверах
103. Register access: mmap, devmem
104. Linux frameworks: GPIO, SPI, I2C
105. Buffer management для DMA
106. Synchronization: polling vs interrupts

107. Latency и throughput optimization
108. Embedded Linux: buildroot, Yocto, PetaLinux
109. RTOS: FreeRTOS, Zephyr на ARM cores
110. Bare-metal applications
111. Multi-OS: Linux на Cortex-A + RTOS на Cortex-R
112. Hypervisors для FPGA SoC
113. Profiling tools: gprof, perf, Vitis Analyzer
114. Hotspot identification
115. Cache miss analysis
116. Branch prediction и pipeline stalls
117. Memory bandwidth bottlenecks
118. Amdahl's law применительно к ускорению
119. Тема 7.2. Выбор кандидатов для аппаратного ускорения (3 ч. лекции)
120. Критерии выбора: compute intensity, parallelism, regularity
121. Roofline model для анализа
122. Compute-bound vs memory-bound алгоритмы
123. Granularity ускорения: функция, loop, kernel
124. Коммуникационные накладные расходы
125. Break-even analysis: когда ускорение эффективно
126. Функциональное разделение HW/SW
127. Temporal vs spatial partitioning
128. Co-processing models: offload, pipeline, feedback
129. Data movement minimization
130. Overlapping communication и computation
131. Концепция синтеза из C/C++
132. Vitis HLS: workflow и инструменты
133. Pragmas для оптимизации: pipeline, unroll, array_partition
134. Интерфейсы: AXI, AXI-Stream, FIFO
135. Latency, Throughput, Initiation Interval
136. Resource estimates и scheduling
137. Loop optimizations: pipelining, unrolling, flattening
138. Array partitioning для параллельного доступа
139. Dataflow optimization для task-level parallelism
140. Memory optimization: двухпортовая память, banking
141. Arbitrary precision types: ap_int, ap_fixed
142. Co-simulation и export RTL

Критерии оценивания (оценочное средство - Контрольные вопросы)

Оценка	Критерии оценивания
превосходно	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «превосходно»
отлично	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «отлично», при этом хотя бы

Оценка	Критерии оценивания
	одна компетенция сформирована на уровне «отлично»
очень хорошо	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «очень хорошо», при этом хотя бы одна компетенция сформирована на уровне «очень хорошо»
хорошо	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «хорошо», при этом хотя бы одна компетенция сформирована на уровне «хорошо»
удовлетворительно	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «удовлетворительно», при этом хотя бы одна компетенция сформирована на уровне «удовлетворительно»
неудовлетворительно	Хотя бы одна часть компетенции сформирована на уровне «неудовлетворительно», ни одна из компетенций не сформирована на уровне «плохо»
плохо	Хотя бы одна часть компетенции сформирована на уровне «плохо»

5.3.2 Типовые задания (оценочное средство - Практическое задание) для оценки сформированности компетенции ПК-3

Цель: проверка умения оптимизировать, проводить HW/SW кодизайн и адаптировать математические модели.

Задание 1. Оптимизация узкого места

Дан Verilog-код конвейерного FIR-фильтра 8-го порядка, работающего на 50 МГц. После синтеза выяснилось, что максимальная частота – 45 МГц из-за длинной комбинационной цепи в сумматоре. Предложите **три** способа увеличения частоты без изменения архитектуры фильтра (только модификация RTL). Реализуйте один из них, покажите результат временного анализа.

Критерии: обоснование, готовый код, отчёт STA (WNS, TNS).

Задание 2. Интеграция ускорителя в SoC с Linux

Разработать IP-блок для **побайтового XOR** буфера длиной 1 Кбайт (шина AXI4-Stream на входе и выходе). Написать драйвер Linux (символьное устройство) и пользовательское приложение на C, которое:

- загружает битовый файл ПЛИС,
- передаёт через DMA произвольный буфер в IP,
- принимает результат и сравнивает с эталоном.

Критерии: работоспособность на целевой плате (Zynq или аналогичной), предоставление Makefile, инструкции по сборке.

Задание 3. Оптимизация с HLS для конкретной модели

Дана математическая модель (код на C):
y[i] = α * x[i] + (1-α) * y[i-1] (экспоненциальное сглаживание).
Реализовать HLS-версию с параметризуемым α (тип float или фиксированная точка). Добиться:

- пропускной способности не менее 1 отсчёта в такт (II=1) для потока данных,
- использования не более 2 блоков BRAM и 20 DSP.

Предоставить код, отчёты о ресурсах и инициализации (II=1, pipeline).

Критерии: достигнутые метрики, объяснение выбора типа данных.

Задание 4. Профилирование и распределение HW/SW

Дан приложение на ARM Cortex-A для обработки видеопотока 1080p: этапы – YUV → RGB, гауссово размытие (3×3), детектор границ Собеля, сжатие JPEG. Время выполнения: 150 мс (YUV → RGB – 20 мс, размытие – 60 мс, Собель – 50 мс, JPEG – 20 мс).
Требуется: предложить распределение между процессором и ПЛИС, чтобы достичь частоты 60 кадров/с (16.6 мс на кадр). Обосновать выбор, написать фрагменты кода вызова аппаратных ускорителей для двух самых тяжёлых этапов (используя псевдо-API).

Критерии: выбор этапов для ускорения (размытие + Собель), оценка достижимого ускорения, код передачи данных через DMA.

Задание 5. Создание математической модели с аппаратной акселерацией

Реализовать на языке Python модель **клеточного автомата (игра «Жизнь»)** на сетке 512×512. Показать, что программная реализация слишком медленная (более 0.5 с на поколение).
Задание: переписать ядро обновления состояния на C++/HLS, сгенерировать RTL и оценить теоретическую скорость на ПЛИС (ресурсы, тактовая частота, число тактов на одно поколение). Предоставить сравнительную таблицу: CPU (Python/NumPy), CPU (C++), FPGA.

Критерии оценивания (оценочное средство - Практическое задание)

Оценка	Критерии оценивания
превосходно	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «превосходно»
отлично	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «отлично», при этом хотя бы одна компетенция сформирована на уровне «отлично»
очень хорошо	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «очень хорошо», при этом

Оценка	Критерии оценивания
	хотя бы одна компетенция сформирована на уровне «очень хорошо»
хорошо	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «хорошо», при этом хотя бы одна компетенция сформирована на уровне «хорошо»
удовлетворительно	Вся компетенция (части компетенции), на формирование которой направлена дисциплина, сформированы на уровне не ниже «удовлетворительно», при этом хотя бы одна компетенция сформирована на уровне «удовлетворительно»
неудовлетворительно	Хотя бы одна часть компетенции сформирована на уровне «неудовлетворительно», ни одна из компетенций не сформирована на уровне «плохо»
плохо	Хотя бы одна часть компетенции сформирована на уровне «плохо»

6. Учебно-методическое и информационное обеспечение дисциплины (модуля)

Основная литература:

1. Бойков К. А. Системное проектирование цифровых аудиоустройств и радиоприложений на программируемых логических интегральных схемах: Практикум / Бойков К. А.,Печенкин С. М. - Москва : РТУ МИРЭА, 2025. - 63 с. - Библиогр.: доступна в карточке книги, на сайте ЭБС Лань. - Книга из коллекции РТУ МИРЭА - Инженерно-технические науки. - СЭБ. - ISBN 978-5-7339-2671-1. - Текст : электронный., https://e-lib.unn.ru/MegaPro/UserEntry?Action=FindDocs&ids=1003575&idb=0.
2. Микропроцессорные устройства. Проектирование и моделирование элементов микропроцессорных устройств на программируемых логических интегральных схемах : лабораторный практикум. Ч. 1. Микропроцессорные устройства. Проектирование и моделирование элементов микропроцессорных устройств на программируемых логических интегральных схемах Ч. 1: лабораторный практикум. - Санкт-Петербург : СПбГУТ им. М.А. Бонч-Бруевича, 2023. - 117 с. - Книга из коллекции СПбГУТ им. М.А. Бонч-Бруевича - Инженерно-технические науки. - Текст : электронный., https://e-lib.unn.ru/MegaPro/UserEntry?Action=FindDocs&ids=918579&idb=0.

Дополнительная литература:

1. Неелова О. Л. Архитектура вычислительных систем. Проектирование элементов вычислительных систем на программируемых логических интегральных схемах: практикум / Неелова О. Л. - Санкт-Петербург : СПбГУТ им. М.А. Бонч-Бруевича, 2022. - 39 с. - Книга из коллекции СПбГУТ им. М.А. Бонч-Бруевича - Информатика. - Текст : электронный., https://e-lib.unn.ru/MegaPro/UserEntry?Action=FindDocs&ids=829853&idb=0.

Программное обеспечение и Интернет-ресурсы (в соответствии с содержанием дисциплины):

1. Операционная система Microsoft Windows
2. Пакет прикладных программ Microsoft Office
3. Правовая система «Консультант плюс»
4. Правовая система «Гарант».
5. Интернет браузеры (Mozilla Firefox, Google Chrome)

7. Материально-техническое обеспечение дисциплины (модуля)

Учебные аудитории для проведения учебных занятий, предусмотренных образовательной программой, оснащены мультимедийным оборудованием (проектор, экран), техническими средствами обучения, компьютерами.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечены доступом в электронную информационно-образовательную среду.

Программа составлена в соответствии с требованиями ОС ННГУ по направлению подготовки/специальности 02.03.02 - Фундаментальная информатика и информационные технологии.

Автор(ы): Мееров Иосиф Борисович, кандидат технических наук, доцент.

Заведующий кафедрой: Мееров Иосиф Борисович, кандидат технических наук.

Программа одобрена на заседании методической комиссии от 02.12.2024, протокол № 5.