Appendix B-5 - Syllabus - Autonomous development

Contents

Competence field	Autonomous development
Curriculum designation	STM32 electronic system design and engineering application
Curriculum code	9061324010
Semester(s) in which the curriculum is taught	5th Semester
Person responsible for the curriculum	Professor Li Wenguo
Lecturer	Professor Li Wenguo, assistant Li Maolin and assistant Liu Xiongjie
Language	Chinese
The relationship between the curriculum and the major	STM32 Electronic System Design and Engineering Application is an elective professional development course for the Electronic Information Engineering major that combines theory with practice. It is an applied course integrating hardware technology, software technology, interface technology, and development tools. Through this course, students will be able to understand and master the theories and knowledge related to ARM embedded systems, and on this basis, proficiently use the software development environment and program debugging methods of the STM32 processor, grasp the programming methods for GPIO, interrupts, serial ports, timers, and other basic peripheral interfaces, and understand the methods for porting embedded real-time operating systems and developing multitasking applications on STM32. Ultimately, students will possess the capability to develop, design, analyze, and debug electronic systems based on STM32 embedded systems, thereby laying a solid technical foundation for future research training in "Intelligent Perception and Control," scientific competitions, product development, comprehensive graduation internships, and on-the-job internships.
Type of teaching, contact hours	Target students: Electronic Information Engineering major Teaching method: theoretical teaching + experiment Contact hours: 32 hours Including: Theoretical teaching: 16 hours Experimental/practical teaching: 16 hours Class size: Four classes with about 160 students
Workload	Total workload = 90 hours; Contact hours = 32 hours; Self-study hours = 58 hours;
Credit points	3.0
Requirements according to the examination regulations	Students who have an attendance rate of more than 2/3 and a homework completion rate of more than 2/3 are eligible for the test.
Prerequisite curriculum	Microcontroller principle and application/microcomputer principle and interface technology, C language program design

curriculum objectives /expected learning outcomes	**Learning outcomes:** The main task of this course is to enable students to master the principles, programming and application development of STM32 microcontroller, and have the ability to independently design and implement electronic systems based on STM32. The specific objectives include: **Knowledge:** 1. Deeply understand the basic principles and internal structure of STM32 microcontroller. Be familiar with various peripherals of STM32, such as GPIO, ADC, PWM, timer, etc. Master the working principle, configuration method and application scenario of these peripherals. 2. Learn to use the development environment and programming language of STM32, be familiar with its programming framework, and learn to master the writing and debugging of common peripherals and their drivers of STM32. 3. Understand the communication interface and protocol of STM32 with other peripheral devices, such as I2C, SPI, USART, etc., to lay a foundation for communication design in practical applications. **Skill:** 1. Be able to skillfully use mainstream embedded development tools such as Keil, master the C language programming method of STM32, and be able to compile, simulate and debug programs. 2. Be able to apply and practice peripheral devices based on STM32 peripheral modules, such as LED lamp control, temperature and humidity monitoring, motor drive, etc., and master the interface circuit design and program design methods of related peripheral devices. 3. Be able to carry out the overall design, hardware design and software design of the system scheme according to the requirements of the system, and have the initial ability of actual project development. **Ability:** 1. Through course study and practice, students can develop their ability to analyze and solve problems and propose effective solutions for practical problems. 2. Understand the basic principles, development mode and method of embedded system design, and cultivate systematic thinking. 3. Be able to play an innovative role, explore new applications and new methods of STM32 in electronic system design, and have initial innovation ability.
Contents	**Theoretical teaching (16 contact hours, 42 self-study hours)** Lecture 1: Introduction (2 contact hours, 1 self-study hour) 1. Overview of embedded systems, ARM architecture;

	2. Introduction to STM32 series microcontroller and development board; 3. Development and application of microcontroller; Lecture 2: Internal structure and working principle of STM32 (2 contact hours, 4 self-study hours) 1. Interpretation of STM32 chip resources; 2. Development environment setup, program download method; 3. New engineering templates (based on firmware library, based on registers). Lecture 3: Working principle of GPIO and register configuration (4 contact hours, 8 self-study hours) 1. The basic structure and working mode of GPIO; 2. GPIO related registers and library functions; 3. GPIO application examples (flashlight experiment based on library functions, registers and bit operations). Lecture 4: Working principle and application of STM32 system clock (2 contact hours, 3 self-study hours) 1. Clock system block diagram, the function and definition of clock configuration related registers; 2. SystemInit Clock system initialization function; 3. A method for calculating a register address from the name of a register address in the registry. Lecture 5: Interrupted system (2 contact hours, 5 self-study hours) 1. Interrupt basic concept, interrupt management method, interrupt priority setting; 2. I/O port mapping relationship of external interrupt in STM32, and setting of external interrupt library function; 3. The design idea and configuration process of external interruption; 4. Examples of interrupt applications. Lecture 6: STM32 timer (2 contact hours, 5 self-study hours) 1. Counter mode: upward counting, downward counting, upward and downward bidirectional counting mode; 2. The working process of the timer, and the concept of four registers in the timing unit; 3. Timer function setting and timer interrupt implementation steps; 4. Timer application examples. Lecture 7: Integrated design of electronic information system based on STM32 (2 contact hours, 16 self-study hours)

	1. This paper explains the basic process of designing practical cases of STM32 embedded system, including project topic selection, scheme design, hardware construction, software programming, debugging and testing, and summary report. 2. Select typical practical cases for analysis, such as the ten-thousand-year calendar, traffic lights, etc. Explain the design ideas, implementation process and key technical points of the cases. 3. Students are required to design an electronic product system based on STM32. The detailed design scheme including hardware circuit design and software program design is required to be formulated. And the project summary report including project background, design scheme, implementation process, test results, etc. is required to be written. **Experimental teaching (16 contact hours, 16 self-study hours)** Experiment 1: Running light and button input experiment. (4 contact hours, 4 self-study hours) Experimental Content: Familiarize with development environments such as Keil for program writing, compilation, downloading, and debugging, learn to use library functions to create new projects and write programs. Master the GPIO configuration methods of the STM32 microcontroller, including port, input or output mode configurations. Control the electrical level status of I/O ports through programming to achieve control of external devices. Experiment 2: External interruption experiment. (4 contact hours, 4 self-study hours) Experimental Content: GPIO pin initialization external interrupt initialization configuration of external interrupt (EXTI) lines setting interrupt trigger modes (such as rising edge falling edge or both edge triggering) and enabling the interrupt lines. Through NVIC configure interrupt priorities and enable interrupt channels. Write interrupt service routines perform program debugging and testing. Experiment 3: Timer interrupt experiment. (4 contact hours, 4 self-study hours) Experimental Content: Initialize the general-purpose timer of STM32 including setting the automatic reload value (ARR) and pre-scaling factor (PSC). Configure the interrupt type of the timer and enable the timer interrupt. Set the priority of the timer interrupt through the NVIC and enable the timer interrupt channel. Write the interrupt service function and perform program debugging and testing. Experiment 4: PWM output experiment (4 contact hours, 4 self-study hours)

	Experimental Content: Configure the timer of the STM32 microcontroller to generate PWM signals and control their frequency and duty cycle. Call the corresponding functions or configure registers to initialize the PWM output parameters of the timer, such as PWM mode, output polarity, output state, etc. Adjust the duty cycle of the PWM signal by modifying the value of the capture/comparison register (CCRx).
Study and examination requirements and forms of examination	1. Attendance rate (10%): Basic requirements for the course (no late arrival, no early departure, no absence without reason). 2. Assignment (30%): lab report. 3. Final examination (60%): Design an electronic product system based on STM32 and write a report.
Media employed	Multimedia computer, projector, laser pen, blackboard, chalk, Keil and ALIENTEK warship STM32 development board
Reading list	1. Textbooks [1] Zhang Yang et al. Atomic Teaching You to Play STM32 Library Functions[M], Beihang University, 2019. 2. Reference book [1] Liu Jun, Chen Bin et al. Example of STM32[M]. Beijing University of Aeronautics and Astronautics Press, 2021. [2] Zhang Yang, Liu Jun et al. STM32F4-Database Function Version[M]. Beijing University of Aeronautics and Astronautics Press, 2015.

Competence field	Autonomous development
Curriculum designation	Principles and Applications of Embedded Systems
Curriculum code	9061324130
Semester(s) in which the curriculum is taught	5th Semester
Person responsible for the curriculum	Lecturer Zhang Lincheng
Lecturer	Lecturer Zhang Lincheng, Professor Li Wenguo and Associate Professor Cui Zhi
Language	Chinese
The relationship between the curriculum and the major	The course "Principles and Applications of Embedded Systems" is an elective professional course platform for the Electronic Information Engineering major. This course focuses on ARM-based microprocessors and the real-time operating system Linux, emphasizing the basic structure and principles of embedded system hardware and fundamental software development techniques. Its objective is to enable students to learn the basics of embedded systems from both software and hardware application perspectives, understand the working principles and cutting-edge development trends of embedded systems, master their application theories and technologies, establish an overall concept of embedded system applications and development, become familiar with the hardware circuit principles and software development methods and processes of embedded systems, and gain a preliminary grasp of embedded system application design methods through the integration of theory and practice, thereby acquiring the initial capability for analyzing, applying, designing, and developing embedded systems.
Type of teaching, contact hours	Target students: Electronic Information Engineering major Teaching method: theoretical teaching + experiment Contact hours: 64 hours Including: Theoretical teaching: 48 hours Experimental/practical teaching: 16 hours Class size: four classes with about 160 students
Workload	Total workload = 150 hours; Contact hours = 64 hours; Self-study hours = 86 hours;
Credit points	5.0
Requirements according to the examination regulations	Only students who attend class with a attendance rate of more than 2/3 and complete their homework with a completion rate of more than 2/3 can take the exam.
Prerequisite curriculum	Computer fundamentals, microcontroller principles and applications, microcomputer principles and applications, embedded Linux

	operating system, C language programming.
curriculum objectives /expected learning outcomes	**Learning outcomes:** The main task of this course is to make students deeply understand the principles and applications of ARM embedded systems, and master the design and development methods of ARM embedded systems. The specific objectives include: **Knowledge:** Understand the basic structure fundamental principles and basic methods of ARM-based embedded systems including the basics of embedded systems ARM architecture and programming models ARM instruction set ARM application system hardware design introduction to Linux system Linux programming basics Bootloader Linux kernel transplantation etc. **Skills:** 1. Be able to use ARM embedded hardware platform and Ubuntu operating system and other software and hardware tools to complete the design, development and debugging of embedded application products. 2. Able to enable students to master embedded technology on the basis of application-oriented approach from product perspective grasp the interdisciplinary application of electronic technology computer technology software engineering and other technologies establish the overall thinking and process of product development apply theory to practice lay a technical foundation for students future employment. **Ability:** 1. Be able to understand and master the basic methods of development and design of the most popular or newly launched new processors and other related chips, software debugging tools, operating systems and application software. 2. Be able to design reasonable experimental steps according to the experimental scheme, be able to correctly use the relevant development board of ARM hardware platform, complete the relevant design and development of applications under Linux operating system platform, be able to debug and analyze the experimental results, and get reasonable and effective conclusions. 3. Have the ability to summarize, organize and express the purpose, principle, content, steps, results and analysis of experiments, and be able to write experimental reports and design documents for embedded systems and applications.
Contents	**Theoretical teaching (48 contact hours, 50 self-study hours)** Chapter 1: Introduction to embedded systems (4 contact hours, 4 self-study hours) 1. The concept, composition, characteristics and classification of embedded systems

	2. Applications of embedded systems 3. Development trends of embedded systems 4. Design and development methods of embedded systems; Chapter 2: ARM Architecture and Programming Model (6 contact hours, 7 self-study hours) 1. Introduction to ARM microprocessor structure and performance of each series of processors; 2. The operating mode, working mode and status of ARM processor 3. ARM register/storage organization; 4. ARM unusual. Chapter 3: ARM instruction system (4 contact hours, 4 self-study hours) 1. ARM instruction set version 2. ARM microprocessor instruction format 3. Addressing mode of ARM microprocessor instructions 4. ARM instruction classification. Chapter 4: ARM assembly program design (2 contact hours, 2 self-study hours) 1. ARM assembles pseudo-instructions and macro instructions 2. ARM ATPCS 3. ARM programming Chapter 5: ARM application system hardware design (8 contact hours, 9 self-study hours) 1. S3C2410X Introduction 2. Peripheral circuit design of development board 3. Development board interface circuit design 4. Other servo circuits Chapter 6: Introduction to Linux (6 contact hours, 6 self-study hours) 1. An introduction to Linux 2. Ubuntu System installation 3. Linux commands 4. Linux text editing Chapter 7: Basics of Linux Programming (8 contact hours , 8 self-study hours) 1. Cross-development environment establishment 2. Shell script 3. Makefile

	Chapter 8: Bootloader (2 contact hours, 2 self-study hours) 1. BootLoader Basics 2. ViVi 3. U-Boot Chapter 9: Linux Kernel Transplant (2 contact hours, 2 self-study hours) 1. The concept of Linux porting 2. Linux kernel and structure 3. Linux kernel porting Chapter 10: Embedded Linux driver development (2 contact hours, 2 self-study hours) 1. Basics of device drivers 2. Driver development example Chapter 11: Embedded Linux Application Development (4 contact hours, 4 self-study hours) 1. Network communication protocol 2. Linux network programming basics 3. Embedded WEB server 4. Introduction to embedded graphics systems 5. Embedded GUI design based on Qt/e 6. Introduction and examples of Qt development **Experimental teaching (16 contact hours, 36 self-study hours)** Experiment 1: Installation and use of Ubuntu. (4 contact hours, 8 self-study hours) Experimental Content: Able to independently install ubuntu such as through virtual machine or USB drive bootable installation media and perform partitioning and configuration; as well as basic Ubuntu operations such as familiarizing with the desktop environment file management terminal command usage system settings software management and network configuration etc. Experiment 2: Establishment of cross-development environment. (2 contact hours, 6 self-study hours) Experimental content: Based on the experimental one, configure the cross-compiling environment Ubuntu, familiarize the structure and connection of the experimental box, realize the interconnection between the host machine and the target machine, and use the cross-compiling compiler to compile the program and run on the experimental box. Experiment 3: Embedded development design. (2 contact hours, 6 self-study hours)

	Experimental content: According to the design requirements of embedded system, hardware selection and design, software architecture design, software coding and module testing, software and hardware integration and debugging, as well as the final system optimization, testing, deployment and maintenance steps. Experiment 4: Design of intelligent terminal for 2048 game based on Linux (8 contact hours, 16 self-study hours) Experimental Content: Based on the actual projects carried out, conduct project requirement analysis and formulate detailed design plans. Design and implement game logic, including core functions such as board initialization, block movement and merging, and new number generation; utilize a graphical user interface library to design a simple and clear user interface, achieving real-time display of game status and response to user input; finally, conduct thorough testing and debugging to ensure stable operation and complete functionality of the game. Through this experiment, students will gain a deeper understanding of the embedded Linux application development process and enhance their programming practical skills. Finally, write a project summary report including project background, design plan, implementation process, and test results.
Study and examination requirements and forms of examination	1. Attendance rate (10%): Basic requirements for the course (no late arrival, no early departure, no absence without reason). 2. Classroom interaction (5%): answering questions in class, etc 3. Work and experiment (25%): experiment report. 4. Final assessment (60%): final examination.
Media employed	Multimedia computer, projector, laser pen, blackboard, chalk, Linux operating system and Arm development board, ARM embedded comprehensive experimental box and so on
Reading list	1. Textbooks [1] ARM Embedded Technology Principles and Applications, edited by Chen Ze, Beijing University of Aeronautics and Astronautics Press, 2023,12th edition 2. Reference book [1] ARM Embedded Linux System Development-From Beginner to Master, edited by Li Yafeng et al. Tsinghua University Press [2] Embedded Linux System Development Technology-ARM based, Zhang Jikun/Zhang Xiaoquan, Peoples Posts and Telecommunications Press.

Competence field	Autonomous development
Curriculum designation	Robot development
Curriculum code	9061324 070
Semester(s) in which the curriculum is taught	6^{th} Semester
Person responsible for the curriculum	Lecturer Zhang Lincheng
Lecturer	Lecturer Zhang Licheng, Professor Li Wenguo and Assistant Professor Li Maolin
Language	Chinese
The relationship between the curriculum and the major	The course "Robot Development" is an elective course for the Electronic Information Engineering major. This course focuses on the fundamental mathematical foundations of robotics, principles of kinematics and dynamics, principles of robot control, robot trajectory planning, robot sensors, and robot programming. It emphasizes the basic structure, fundamental principles, and related development technologies of simple robots. The objective is to enable students to learn the basic knowledge of robots from both theoretical and practical perspectives, understand the working principles and cutting-edge trends of robots, master their application theories and technologies, establish an overall concept of robot application and development, become familiar with the methods and processes of robot development, and initially grasp the design methods of robot development through the integration of theory and practice, thereby acquiring preliminary capabilities in analyzing, applying, designing, and developing robot systems.
Type of teaching, contact hours	Target students: Electronic Information Engineering major Teaching method: theoretical teaching + experiment Contact hours: 48 hours Including: Theoretical teaching: 32 hours Experimental/practical teaching: 16 hours Class size: four classes with about 160 students
Workload	Total workload = 1 20 hours; Contact hours = 48 hours; Self-study hours = 72 hours;
Credit points	4.0
Requirements according to the examination regulations	Only students who attend class with a attendance rate of more than 2/3 and complete their homework with a completion rate of more than 2/3 can take the exam.
Prerequisite curriculum	University physics, advanced mathematics, automatic control principles, sensors, computer language programming.
curriculum objectives	**Learning outcomes:**

/expected learning outcomes	The main task of this course is for students to learn the basic knowledge of robots from both theoretical and practical perspectives understand the working principles and cutting-edge development trends of robots master their application theories and technologies establish an overall concept of robot application and development be familiar with the methods and processes of robot development initially grasp the design methods of robot development from the combination of theory and practice and preliminarily possess the ability to analyze apply and design develop robot systems. **Knowledge:** Understand the basic structure, basic principles and basic methods of robots. It includes the basic mathematical knowledge of robots, robotics kinematics, robotics dynamics, control of force and position of robots, robotics sensors, trajectory planning and programming of robots. **Skill:** 1. Be able to understand and master the most popular or newly launched new robots and related technologies, related software debugging tools, and basic methods of development and design; be able to propose overall solutions based on complex engineering problems, and design robot units or processes that meet specific requirements. 2. Able to design reasonable experimental procedures according to the experimental plan, able to correctly use the relevant development boards of robots, complete the design and development of simple robots through professional and mature robot development platforms, able to debug, analyze experimental results, and draw reasonable and effective conclusions; possess the ability to summarize, organize, and express the purpose, principles, content, procedures, results, and analysis of experiments in writing, and be capable of writing experimental reports and design documents for embedded systems and applications. **Ability:** Able to use robot hardware and software development platforms and related tools to complete the design, development, and debugging tests of robot application products, enabling students to master robot-related technologies and focus on application-centered approaches from a product perspective, grasp the interdisciplinary applications of electronic technology, computer technology, software engineering, etc., establish an overall mindset and process for product development, apply theory to practice, and lay a technical foundation for future employment.
Contents	**Theoretical teaching (32 hours contact, 48 hours self-study hours)** Chapter 1 Introduction to robots (2 contact hours, 3 self-study hours)

	1. Basic knowledge of robots; 2. Related research fields of robots; 3. Composition and characteristics of robots. Chapter 2 Mathematical foundation (6 contact hours, 9 self-study hours) 1. Posture and coordinate system description; 2. Coordinate mapping; 3. Homogeneous coordinate transformation; 4. Object transformation and transformation equation; 5. General rotary transformation. Chapter 3 Robot kinematics (6 contact hours, 9 self-study hours) 1. Overview of robot kinematics; 2. Representation of robot motion equation; 3. Solving the motion equation of the robot; 4. Analysis of robot motion examples; 5. Robot jacobi matrix calculation. Chapter 4 Robot Dynamics(4 contact hours, 6 self-study hours) 1. Lagrange equation; 2. Newton-Euler equation; 3. Dynamics equation of the manipulator; 4. Dynamic characteristics of robots; 5. Static characteristics of the robot. Chapter 5 Control of robot position and force (4 contact hours, 6 self-study hours) 1. Overview of robot control and transmission; 2. position control; 3. Force and position mixed control; 4. Decompose motion control. Chapter 6 Robot Sensors (4 contact hours, 6 self-study hours) 1. Overview of robot sensors; 2. Internal sensors; 3. External sensors; 4. Robot vision device. Chapter 7 Robot trajectory planning (4 contact hours, 6 self-study hours) 1. Overview of robot trajectory planning; 2. Interpolated calculation of joint trajectory; 3. Descartes path trajectory planning; 4. Real-time generation of planned trajectory; Chapter 8 Robot Programming (2 contact hours, 3 self-study hours) 1. Programming requirements and language types; 2. Language system and basic functions; 3. Common programming languages; 4. Offline programming of robots;

	Experimental teaching (16 contact hours, 24 self-study hours) Experiment project 1: Simulation of robot coordinate system transformation. (4 contact hours, 6 self-study hours) Experimental content: robot coordinate system translation, rotation and conforming transformation are realized through matlab simulation. Experiment project 2: Representation and solution of robot kinematic equations. (8 contact hours, 12 self-study hours) Experimental Content: 1) Focus on learning how to use the Link function to establish the DH parameter table of links; 2) Focus on learning how to use the Seria lLink function to establish a serial robot model; 3) Learn to use teach for robot modeling and teaching; 4) Learn to use the jtraj function for robot trajectory planning; 5) Learn to use the fkine function for robot forward kinematics simulation; 6) Learn to use the ikine function for solving robot inverse kinematics Experiment project 3: Robot trajectory planning (4 contact hours, 6 self-study hours) Experimental content: 1) Familiarize with the polynomial interpolation method of joint space trajectory; 2) Interpolating calculation of joint space trajectory-cubic polynomial interpolation and linear interpolation with parabolic transition 3) Program the trajectory planning according to the requirements of joint space trajectory.
Study and examination requirements and forms of examination	1. Attendance rate (10%): Basic requirements of the course (no late arrival, no early departure, no absence without reason). 2. Classroom interaction (5%): answering questions in class, etc 3. Work and experiment (25%): experiment report. 4. Final assessment (60%): final examination.
Media employed	Multimedia computer, projector, laser pen, blackboard, chalk, matlab and so on
Reading list	1. Textbooks Course materials: "Robotics" edited by Cai Zixing, Tsinghua University Press, 2022. 2. References Robotics Technology and Application. Chen Ken. Tsinghua University Press, 2006.

Competence field	Autonomous development
Curriculum designation	FPGA Principles and Applications
Curriculum code	9061324020
Semester(s) in which the curriculum is taught	6th Semester
Person responsible for the curriculum	Professor Li JiaSheng
Lecturer	Professor Li JiaSheng and Associate Professor Deng Yaqi
Language	Chinese
The relationship between the curriculum and the major	FPGA Principles and Applications is an elective professional course for the Electronic Information Engineering major that combines theory with practice. This course introduces knowledge about FPGA hardware technology, software technology, interface technology, and development tools. Through the study of this course, students will master the structure of FPGA, the basic methods and syntax of VHDL programming, understand and master the design methods of FPGA application systems, cultivate their ability to program based on FPGA hardware platforms, possess the basic qualities and capabilities required for FPGA-based development and research work, and lay a solid foundation for future use or design of FPGA application systems.
Type of teaching, contact hours	Target students: Electronic Information Engineering major Teaching method: theoretical teaching + experiment Contact hours: 48 hours Including: Theoretical teaching: 32 hours Experimental/practical teaching: 16 hours Class size: Four classes with about 160 students
Workload	Total workload = 120 hours; Contact hours = 48 hours; Self-study hours = 72 hours;
Credit points	4.0
Requirements according to the examination regulations	Only students who attend class with a attendance rate of more than 2/3 and complete their homework with a completion rate of more than 2/3 can take the exam.
Prerequisite curriculum	Advanced mathematics, circuit analysis, analog electronic technology, digital electronic technology
curriculum objectives /expected learning outcomes	**Learning outcomes:** The main task of this course is to make students deeply understand the principle and application of FPGA, and master the design and development method of FPGA system. The specific objectives include: **Knowledge:**

	1. Understand the structure and principle of FPGA; 2. Master FPGA chip design and development skills; 3. Understand FPGA control system and interface technology, master application system configuration and interface technology; 4. Be able to comprehensively use engineering knowledge, VHDL and other hardware languages to compare and evaluate FPGA design schemes. **Skill:** 1. Master programming knowledge and system application program design; 2. Be able to propose systematic solutions for specific engineering problems, and determine design objectives, technical requirements, development cycle and process, etc. **Ability:** 1. Master the use of professional software to simulate, analyze and computer-aided design methods for F PGA control system; 2. Conduct simple product development.
Contents	**Theoretical teaching (32 contact hours, 56 self-study hours)** Chapter 1: Overview of Programmable Logic Devices (2 contact hours, 2 self-study hours) 1. Basic structure and circuit representation of programmable logic devices; 2. Classification of PLD. Chapter 2: Large-scale programmable logic devices CPLD/FPGA (2 contact hours, 4 self-study hours) 1. Principle of complex programmable logic device CPLD structure; 2. On-site programmable gate array FPGA structure principle; 3. Introduction, programming and configuration of PLD products. Chapter 3: Introduction to QuarusII design software (2 contact hours, 4 self-study hours) 1. Introduction of software functions, design input, project compilation and matching, project simulation and timing analysis, device programming download; 2. Introduction to commonly used design input methods: schematic design input method, text design input (VHDL) method introduction, waveform input method introduction, hierarchical design input method introduction; 3. Basic applications: project design input, project compilation and adaptation, project function simulation and timing analysis, reassignment and positioning of pins, device download programming and hardware implementation. Chapter 4: Hardware Description Language VHDL (6 contact hours, 12 self-study hours) 1. The basic structure of VHDL language;

	2. Basic knowledge of VHDL; 3. The main descriptive statements of VHDL: sequential statements, parallel statements; 4. Subroutines, packages, libraries and configurations; 5. An example of VHDL design. Chapter 5: Basic VHDL Descriptive Statements (8 contact hours, 16 self-study hours) 1. Sequential statements: used to implement the algorithm description of the model; 2. Parallel statement: it is used to indicate the connection between the algorithm descriptions of each module. Chapter 6: Subroutines and packages (2 contact hours, 2 self-study hours) 1. subprogram; 2. software package. Chapter 7: VHDL description of commonly used circuits (8 contact hours, 16 self-study hours) 1. VHDL language for combinational logic circuit design; 2. VHDL language for sequential logic circuit design 3. VHDL language is used to design the memory Chapter 8: VHDL Design Application Examples (2 contact hours, 4 self-study hours) 1. Correctly use VHDL to develop and design 8-bit adder; 2. Correctly use VHDL to design PWM signal generator; 3. Correctly use VHDL to develop and design traffic light signal controller. **Experimental teaching (16 contact hours, 16 self-study hours)** Project 1: Introduction to simple logic circuit design (flow light, buzzer). (2 contact hours, 2 self-study hours) Experimental content: use Quartus II to design half adder with VHDL, conduct waveform simulation, pin allocation, and download to the experimental equipment for logical function verification. Project 2: 2 Selectors (2 contact hours, 2 self-study hours) Experimental content: Use VHDL text design method in Quartus II integrated environment to design 2 select 1 multiplexer, conduct waveform simulation, pin allocation and download to the experimental equipment for logical function test. Project 3: D flip-flop design (2 contact hours, 2 self-study hours) Experimental content: Use the VHDL text design method in the Quartus II integrated environment to design a simple sequential circuit —— D flip-flop, conduct waveform simulation and analysis according to the working characteristics of D flip-flop, pin allocation

	and download to the experimental equipment for functional testing. Project 4: Design of 1-bit binary full adder (using circuit diagram input method) (2 contact hours, 2 self-study hours) Experimental content: use the circuit diagram input method under the Quartus II environment to design the full adder, conduct waveform simulation, pin allocation and download to the experimental equipment for logical function verification. Project 5: 4-bit adder counter (2 contact hours, 2 self-study hours) Experimental content: Use the VHDL text design method in the Quartus II integrated environment to design a 4-bit adder counter, conduct waveform simulation and analysis, pin allocation, and download it to the experimental equipment for functional testing. Project 6: Design of a two-digit decimal counter enabled by a clock (circuit diagram input method) (4 contact hours, 4 self-study hours) Experimental content: use the circuit diagram input method under the Quartus II environment to enable a two-bit decimal counter with a clock, conduct waveform simulation, pin allocation and download to the experimental equipment for logical function verification. Project 7: Design of 8-bit hexadecimal digital frequency counter (2 contact hours, 2 self-study hours) Experimental content: According to the actual engineering problems, design, select and demonstrate the scheme, correctly design the control program, use the circuit diagram input method in Quartus II environment to digital frequency counter with sixteen digits, conduct waveform simulation, pin allocation, and download it to the experimental equipment for logical function verification. Project 8: Design of PWM signal generator (2 contact hours, 2 self-study hours) Experimental Content: Design and debug a pulse-counting digital modulation signal generator, this signal generator consists of two identical self-loading adder counters LCNT8, the high/low level pulse widths of its output signals can be controlled by two sets of 8-bit preset numbers; use the EDA experimental development system for hardware verification.
Study and examination requirements and forms of examination	1、Attendance rate (10%): Basic requirements of the course (no late arrival, no early departure, no absence without reason). 2、Assignment (30%): assignment, lab report. 3、Final assessment (60%): final examination.
Media employed	Multimedia computer, projector, laser pen, blackboard, chalk, Keil and Proteus software, microcontroller experiment box

Reading list	1. Course materials: Zhang Wenai. EDA Technology and FPGA Application Design[M]. Electronic Industry Press, 2023. 2. References: Tan HuiSheng. EDA Technology and Application [M]. Xi an University of Electronic Science and Technology Press, 2022. Dong Haiqing, Chen Hong, Tang Min. Fundamentals of Programmable Logic Devices [M]. Tsinghua University Press, 2018. Zhou Shuge. FPGA_CPLD System Design and Application Development[M]. Electronic Industry Press, 2021. 3. Teaching websites: http://www.intel.com.cn/