

WSIZ, 2023

**Computer systems architecture**

Detailed list of examination questions

Dmitry Zaitsev

1. Historical prototypes of computer
2. Generations of computers
3. Computer application domains
4. Von Neumann computer architecture
5. Von Neumann principles of computing
6. Common bus architecture of computer, basic elements
7. Basic elements and scheme of processor
8. Processor pipeline
9. Processor instructions set, basic instruction types
10. Processor working cycle
11. Formats of processor instructions
12. Addressing modes of operands within processor instructions
13. Stack of processor
14. Communication with input/output devices: ports and interrupts
15. Hierarchy of computer memory
16. Machine language and assembler
17. Cross-platforms for embedded applications
18. Intel x86 Architecture: registers
19. Intel x86 Architecture: status flags
20. Intel x86-64 instruction format
21. Intel x86: data movement instruction
22. Intel x86: arithmetic instructions
23. Intel x86: suffices of instructions
24. Intel x86 assembler: input-output implementation, using libc functions
25. Intel x86 : bitwise logic instructions
26. Intel x86 : shift and rotate instructions
27. Intel x86 : conditional and unconditional jump instructions
28. Assembler pattern for implementation of branching “if-then-else”
29. Assembler pattern for implementation of loop “while”
30. Addressing modes of x86: register and direct memory addressing
31. Addressing modes of x86: immediate and indirect memory addressing
32. Addressing modes for work with array, base and index registers
33. Concept of stack, stack segment, top of stack and base of stack registers]
34. Functions of stack in modern programming languages

35. Intel x86 stack: instructions push and pop
36. Intel x86 call-return procedure, instructions, types of calls
37. Basic principles and techniques for modular software design
38. Interrupt procedure and interrupt handling components
39. Concept of hardware driver, principles of drivers design
40. Call of C function from assembler: basic commands to compile, link, and run code
41. Call of C function from assembler: agreements on passage of parameters
42. Allocation and addressing of local variables in language C style
43. Implementation of recursive functions in assembler language using language C style
44. Code optimization principles, levels of code optimization in language C
45. Standard for Floating-Point Arithmetic IEEE 754
46. Procedure of floating point data conversion according to IEEE 754
47. An overview of former X87 Floating-Point Unit architecture
48. SIMD –data parallel architecture
49. Streaming SIMD Extensions floating-point architecture
50. Streaming SIMD Extensions floating-point instructions
51. Hierarchical design of computer memory: registers, cache, main memory, external memory
52. Procedures of data caching, cache work scheme
53. Multicore architecture and multilayer cache
54. Interaction of memory cache layers
55. Multi-core architecture of processor
56. Concept of benchmark for hardware design, examples of benchmarks
57. Software implementation of benchmarks for C and assembler code
58. Segments of memory and special segment and memory control registers of processor
59. Concept of virtual memory and its hardware support with Memory Management Unit
60. Translation of virtual memory addresses, page tables: page directory and entry point
61. Hardware support of OS page swapping procedures
62. I/O device specification in computer architecture: ports and interrupts
63. Ports of I/O device: assigned range, principles of interaction
64. Basic formats of X86 In and Out instructions
65. Memory-Mapped and Port types of I/O implementation
66. Basic approaches to I/O: poll of readiness or interrupt on readiness
67. I/O ports classification: state port, data port, command port
68. Basic entry points of an I/O device driver
69. Typical architecture of an I/O device driver
70. I/O interrupt handling algorithm of driver
71. Compare basics of Block Device Driver and Character Device Driver work
72. Hard drive structure and basic commands
73. Busses and interfaces of computer
74. Concept of combinatorial logic circuit and basic stages of its design
75. Basis of Boolean functions and standard forms of Boolean algebra
76. Minimization of Boolean functions

77. Logical gates, their truth functions and graphical notation
78. Classification of basic combinatorial logic circuits
79. Hardware description language Verilog and its features
80. Syntax of Verilog module specification, input and output parameters
81. Design of full adder of 1-bit variables in Verilog
82. Hierarchical design of digital circuits in Verilog
83. Hierarchical design of 4-bits sequential adder in Verilog
84. Hardware testing in the process of its design using Verilog
85. Advantages and disadvantages of sequential implementation of digital hardware
86. Basic combinatorial circuits and their brief specification
87. Multiplexers design in Verilog
88. Demultiplexers design in Verilog
89. Digital circuits design using multiplexers/demultiplexors
90. Encoders design in Verilog
91. Decoders design in Verilog
92. Digital circuits design using encoders/decoders
93. A concept of a state machine working in discrete time
94. Mealy and Moore machines
95. Representation of state machines: state diagram and tabular
96. Mutual transformations of Mealy and Moore machines
97. An example of state machine for elevator control
98. Procedure of state machine minimization
99. Equivalence and n-equivalence of state machine states
100. Elementary state machines – flip-flops D and RS
101. Elementary state machines – flip-flops T and JK
102. Procedure of state machine (automata) synthesis
103. General layout of sequential logic
104. Chocolate vending machine specification and binary encoding
105. Concept of Sequential Logic circuit and its design with Verilog
106. Representation of numbers, arithmetic and logic expressions in Verilog
107. Flip-flop Verilog specification using statement *always*
108. Detailed specification of Verilog statement *always*, event *posedge*
109. Blocking and non-blocking assignments in Verilog
110. Examples of flip-flop and counter implementation in Verilog
111. Direct implementation of state machine in Verilog
112. Example of direct implementation of chocolate vending machine in Verilog
113. Implementation of state machine in Verilog using binary encoding
114. Example of chocolate vending machine implementation in Verilog using binary encoding
115. CPU structure, basic components of CPU
116. CPU Control unit functions
117. Hardwired Control Unit and Microprogrammable control unit of CPU
118. Microprogrammable CPU control unit scheme

- 119. Single cycle processor layout, basic components and their connections
- 120. Tracing instruction execution on CPU scheme: memory copy instruction
- 121. Tracing instruction execution on CPU scheme: binary operation instruction
- 122. Tracing instruction execution on CPU scheme: conditional branching instruction
- 123. CPU control unit decoder table a key to processor instructions execution
- 124. Special features of High Performance Computers and Supercomputers
- 125. Units to measure computing, standard benchmarks, Top 500 list
- 126. Generalized High Performance Computer architecture, typical scheme
- 127. Overview of supercomputer Fugaku architecture
- 128. High Performance Computer programming technology
- 129. Utilizing multi-core shared memory architecture with Open Multi-Processing
- 130. Architecture of Graphical Processing Units (GPU)
- 131. Massively-parallel computations on Graphical Processing Units (GPU) with CUDA
- 132. Utilizing distributed architecture with Message Passing Interface (MPI)
- 133. Disadvantages of traditional computer architecture
- 134. Basic directions of embedded control systems implementation
- 135. Micro-processor based embedded control architecture
- 136. Digital-analog and analog-digital converters for embedded applications
- 137. Sensors and actuators for embedded applications
- 138. Instrumental software for microcontroller embedded applications design
- 139. Field Programmable Gate Arrays (FPGA) based embedded architecture
- 140. Artificial Intelligence powered embedded control architecture
- 141. NVIDIA Jetson series of intellectual embedded control devices architecture and design tools
- 142. An example of autonomous vehicle driving with NVIDIA Jetson Nano and JetBot

\*- underlined question means beginning of the next lecture topics.

---