1. **What is a microprocessor?**

   A programmable IC that fetches, decodes, and executes instructions.

2. **Which was the first microprocessor?**

   Intel 4004, a 4-bit processor released in 1971.

3. **Why is a microprocessor called the CPU of a computer?**

   It controls and processes all system operations.

4. **How does the ALU work?**

   It performs arithmetic and logic operations.

5. **What is the role of the control unit?**

   It generates control signals to coordinate operations.

6. **Which buses connect CPU to memory?**

   Address, data, and control buses.

7. **Why is program counter needed?**

   It holds the next instruction address.

8. **How does a stack help?**

   It stores temporary data like return addresses.

9. **What are the flags in 8085?**

   Sign, Zero, Auxiliary carry, Parity, and Carry.

10. **What is instruction set?**

    A collection of machine-level instructions.

11. **Which addressing modes exist in 8085?**

    Immediate, direct, register, indirect, and implied.

12. **Why is assembly language useful?**

    It gives low-level hardware control with mnemonics.

13. **How does assembler work?**

    It translates mnemonics into machine code.

14. **What is opcode?**

    It specifies the operation to be performed.

15. **Which registers are in 8086?**

    General, segment, pointer, index, and flag registers.

16. **Why does 8086 use segmentation?**

    To access 1MB memory using 20-bit addressing.

17. **How many bits are in 8085's data bus?**

    It has an 8-bit data bus.

18. **What is pipelining?**

    Overlapping instruction execution phases.

19. **Why is instruction queue in 8086?**

    It prefetches instructions to speed execution.

20. **What is difference between RISC and CISC?**

    RISC has simple, fast instructions; CISC has complex ones.

21. **Which is faster: register or memory access?**

    Registers are much faster.

22. **What does instruction decoder do?**

    It interprets opcodes and signals operations.

23. **Why are interrupts used?**

    They handle urgent events without polling.

24. **How does maskable differ from non-maskable interrupt?**

    Maskable can be disabled; non-maskable cannot.

25. **Which is the highest priority interrupt in 8085?**

    TRAP, a non-maskable interrupt.

26. **Why are flags needed in ALU?**

    They show result conditions like zero or carry.

27. **How many address lines in 8085?**

    16 lines for 64KB memory.

28. **What is stack pointer?**

    A 16-bit register pointing to top of stack.

29. **Which memory is cache?**

    SRAM between CPU and RAM.

30. **Why is clock signal needed?**

    It synchronizes CPU operations.

31. **What is machine cycle?**

    Time for one fetch/read/write operation.

32. **Which is faster: SRAM or DRAM?**

    SRAM is faster but expensive.

33. **Why is ALE signal in 8085?**

    It separates multiplexed address/data lines.

34. **How many instructions in 8085?**

    74 instructions, 246 opcodes.

35. **What is difference between CALL and JUMP?**

    CALL saves return address; JUMP does not.

36. **Which instruction clears accumulator?**

    XRA A sets accumulator to 0.

37. **Why are macros used?**

    They replace repetitive code with one definition.

38. **How are subroutines defined?**

    Small reusable code blocks called by CALL.

39. **Which instruction checks zero flag?**

    JZ checks if result is zero.

40. **Why use SIM instruction?**

    To mask interrupts and set serial output.

41. **What do IN/OUT instructions do?**

    They transfer data between CPU and I/O ports.

42. **Why are multiplexed buses used?**

    To reduce chip pin count.

43. **What is DMA?**

    Direct device-to-memory data transfer.

44. **Which signal indicates read?**

    $\overline{RD}$ indicates memory or I/O read.

45. **What is T-state?**

    One clock period of operation.

46. **Which are segment registers in 8086?**

    CS, DS, SS, and ES.

47. **Which register pair points memory in 8085?**

    HL register pair.

48. **Why are logical instructions needed?**

    For AND, OR, XOR, shifts, complements.

49. **What is difference between immediate and direct addressing?**

    Immediate has value; direct has memory address.

50. **Which instruction halts CPU?**

    HLT stops execution.

51. **Why is segmentation beneficial?**

    Supports modular programming and large memory.

52. **How does microprocessor differ from microcontroller?**

    Processor needs peripherals; controller has them built-in.

53. **What is signed vs unsigned numbers?**

    Signed include negatives; unsigned only positives.

54. **Which instruction rotates accumulator?**

    RLC and RRC.

55. **Why are assembler directives used?**

    They guide assembler, not CPU.

56. **How are loops implemented?**

    With LOOP instruction and CX.

57. **What is near vs far jump?**

    Near changes IP only; far changes CS:IP.

58. **Why is CALL powerful?**

    It supports modular coding.

59. **Which instruction swaps data?**

    XCHG exchanges register contents.

60. **Why use pseudo-instructions?**

    For labels and variables.

61. **How does PUSH work?**

    SP decrements and data is stored.

62. **Which restores data from stack?**

    POP retrieves register content.

63. **Why segment overrides in 8086?**

    To access non-default segments.

64. **What are software vs hardware interrupts?**

    Software from instructions; hardware from signals.

65. **Which instruction generates software interrupt?**

    INT n.

66. **Why priority encoder in interrupts?**

    To resolve multiple requests.

67. **What is instruction queue size in 8086?**

    6 bytes.

68. **Which signal distinguishes memory vs I/O?**

    IO/$\overline{M}$.

69. **Why parity flag?**

    It checks even/odd bits.

70. **Which instruction complements accumulator?**

    CMA flips all bits.

71. **Why use shift instructions?**

    For fast multiply/divide by 2.

72. **What is SHR vs SAR?**

    SHR logical; SAR arithmetic.

73. **Why interrupts faster than polling?**

    They avoid constant checking.

74. **Which register is base pointer in 8086?**

    BP register.

75. **Why overlap segments in 8086?**

    For memory efficiency.

76. **What is trap flag?**

    It enables single-step debugging.

77. **Why conditional jumps?**

    They branch based on flags.

78. **Which instruction multiplies signed numbers?**

    IMUL.

79. **Why minimum and maximum mode in 8086?**

    For single or multiprocessor use.

80. **Which instruction divides unsigned numbers?**

    DIV.

81. **What is assembler pass-1?**

    It builds symbol table and checks syntax.

82. **What is pass-2?**

    It generates machine code.

83. **Why are labels used in assembly?**

    They mark instruction addresses.

84. **Which register holds instruction pointer?**

    IP register.

85. **What is difference between MOV and MVI?**

    MOV copies register/memory; MVI loads immediate data.

86. **Why segment:offset addressing?**

    To form 20-bit address.

87. **What are predefined interrupts in 8086?**

    INT 0–4 for divide, debug, NMI, etc.

88. **Which flag shows overflow?**

    Overflow flag (OF).

89. **Why conditional call instructions?**

    They call subroutines only if flag set.

90. **What is difference between assembler and linker?**

    Assembler makes object code; linker combines modules.

91. **Which instruction clears carry flag?**

    CLC.

92. **Why stack grows downward?**

    To avoid overlap with code/data segments.

93. **What is difference between ORG and EQU?**

    ORG sets location; EQU assigns constant.

94. **Which instruction is no-operation?**

    NOP.

95. **Why LOOP instruction useful?**

    It reduces jump overhead.

96. **What is role of CS in 8086?**

    Holds code segment base address.

97. **Which instruction loads effective address?**

    LEA.

98. **Why test instruction used?**

    It performs AND without changing operands.

99. **What is use of XLAT?**

    It translates byte from lookup table.

100. **Which instruction copies string?**

    MOVS copies bytes/words.

101. **Why direction flag important?**

    It decides string movement left/right.

102. **Which instruction scans string?**

    SCAS searches for byte/word.

103. **Why repeat prefixes used?**

    To repeat string instructions.

104. **What is difference between REP and REPE?**

    REP repeats always; REPE while equal.

105. **Which instruction compares strings?**

    CMPS compares two strings.

106. **Why use conditional assembly?**

To assemble code selectively.

107. **What are macros vs procedures?**

Macros expanded inline; procedures invoked.

108. **Why modular programming important?**

It improves readability and reuse.

109. **Which instruction returns from interrupt?**

IRET.

110. **What is difference between CLI and STI?**

CLI clears interrupts; STI enables them.

111. **Why are WAIT states used?**

To match slow memory speed.

112. **What is READY signal in 8086?**

It indicates peripheral readiness.

113. **Which pin indicates power supply in 8085?**

Vcc = +5V, Vss = GND.

114. **Why multiplex address/data lines?**

To save pins on IC.

115. **What is function of SID/SOD in 8085?**

Serial input/output lines.

116. **Which register holds memory pointer in 8086?**

SI and DI.

117. **Why is SS register needed?**

It points to stack segment.

118. **What is difference between JMP and LCALL?**

JMP changes control; LCALL saves return.

119. **Why PUSHF and POPF used?**

To save and restore flags.

120. **Which instruction shifts with carry?**

RCL/RCR.

121. **What is role of NOP?**

It wastes one cycle for timing.

122. **Why assembler generates listing file?**

For debugging and reference.

123. **What is use of END directive?**

It marks end of source program.

124. **What is microcode in processors?**

It is low-level control instructions stored in control memory.

125. **Which pipeline hazards occur in microprocessors?**

Structural, data, and control hazards.

126. **Why cache coherence is important?**

To keep multiple cache copies consistent.

127. **How does virtual memory help CPUs?**

It extends memory using disk storage.

128. **What is paging in memory systems?**

Dividing memory into fixed-size pages.

129. **Why segmentation faults occur?**

Due to invalid memory access.

130. **How does MMU work?**

It translates virtual to physical addresses.

131. **Which technique improves instruction throughput?**

Pipelining.

132. **Why branch prediction important?**

To reduce pipeline stalls.

133. **What is microprogrammed control?**

Control signals generated by microcode.

134. **Which addressing mode is fastest?**

Register addressing.

135. **Why is Harvard architecture used?**

It separates instruction and data buses.

136. **How does superscalar architecture improve performance?**

By executing multiple instructions per cycle.

137. **What is difference between macro and micro instruction?**

Macro is high-level assembly; micro is hardware-level control.

138. **Why stack overflow occurs?**

Due to exceeding allocated stack memory.

139. **How does interrupt vector table work?**

It holds addresses of ISRs.

140. **Which scheduling is used in pipelines?**

Dynamic instruction scheduling.

141. **Why out-of-order execution is useful?**

It avoids stalls by reordering instructions.

142. **What is register renaming?**

Technique to eliminate false dependencies.

143. **Why is CISC still used today?**

For code density and legacy support.

144. **How does instruction latency differ from throughput?**

Latency = time per instruction; throughput = instructions per unit time.

145. **What is speculative execution?**

Executing instructions before branch resolution.

146. **Which register stores flags in 8086?**

FLAGS register.

147. **Why is instruction pipelining limited?**

Due to hazards and dependencies.

148. **What is difference between control and status signals?**

Control directs operations; status shows current state.

149. **Why is DMA faster than CPU transfer?**

It bypasses CPU for memory access.

150. **How does vectored interrupt differ from non-vectored?**

Vectored has fixed ISR address; non-vectored needs external.

151. **What is shadow register?**

Backup registers for fast context switching.

152. **Why prefetch buffer used?**

To reduce memory access delay.

153. **What is pipeline flushing?**

Clearing pipeline on wrong branch.

154. **Which flag detects arithmetic overflow?**

OF flag.

155. **Why instruction length varies in CISC?**

For flexibility and complex operations.

156. **What is stack frame?**

Memory layout for procedure calls.

157. **Why call stack important?**

It tracks active subroutine calls.

158. **How does priority interrupt work?**

By assigning fixed priorities to requests.

159. **Which unit performs floating-point operations?**

FPU (Floating-Point Unit).

160. **Why is microprocessor clock doubled internally?**

To speed up instruction phases.

161. **How does pipeline stall resolved?**

Using forwarding or bubbles.

162. **What is hyper-threading?**

Simultaneous multi-threading for parallelism.

163. **Why instruction alignment important?**

Misalignment causes extra cycles.

164. **What is watchdog timer?**

A timer to detect system hangs.

165. **Why memory-mapped I/O used?**

To unify I/O and memory address space.

166. **Which unit fetches instructions in pipeline?**

Instruction Fetch Unit.

167. **Why cache associativity used?**

To reduce conflict misses.

168. **What is difference between write-through and write-back cache?**

Write-through updates memory immediately; write-back delays.

169. **Why hazard detection unit needed?**

To avoid pipeline errors.

170. **How does clock skew affect CPU?**

It causes synchronization issues.

171. **What is speculative branch execution?**

Executing predicted branch path.

172. **Why loop unrolling used in assembly?**

It reduces branch overhead.

173. **What is instruction window?**

A buffer holding decoded instructions.

174. **Why instruction set must be orthogonal?**

For simplicity and consistency.

175. **What is micro-operation?**

Small step implementing instruction execution.

176. **Why paging with TLB important?**

It speeds up address translation.

177. **What is bus arbitration?**

Process of deciding bus master.

178. **Why priority inversion problem occurs?**

When low-priority task blocks high-priority.

179. **How does cache miss penalty reduced?**

Using multi-level caches.

180. **What is difference between maskable and vectored interrupts?**

Maskable can be disabled; vectored has predefined addresses.

181. **Why is instruction pre-decoding done?**

To speed up later pipeline stages.

182. **What is barrel shifter?**

A circuit for fast shifting/rotation.

183. **Why instruction cycle differs from machine cycle?**

Instruction cycle is multiple machine cycles.

184. **How does Harvard differ from Von Neumann?**

Harvard separates memory; Von Neumann uses single.

185. **Why stack-based machines simpler?**

They avoid explicit register addressing.

186. **What is pipeline depth?**

Number of stages in pipeline.

187. **Why hazard forwarding used?**

To solve data hazards quickly.

188. **What is branch delay slot?**

Instruction executed after branch regardless.

189. **Why dynamic branch prediction better?**

It adapts to program behavior.

190. **What is bus cycle?**

One transfer operation over bus.

191. **Why instruction-level parallelism important?**

It improves CPU throughput.

192. **What is superscalar execution?**

Multiple pipelines execute instructions simultaneously.

193. **Why control word used in 8255?**

To configure modes of operation.

194. **What is tri-state buffer?**

It has high, low, and high-impedance states.

195. **Why instruction profiling done?**

To optimize performance hotspots.

196. **How does hardwired control differ from microprogrammed?**

Hardwired is fast but inflexible; microprogrammed is flexible.

197. **Why segmentation in memory beneficial?**

It supports modular programs.

198. **What is paging overhead?**

Extra time for address translation.

199. **Why wait states added?**

For slow peripherals to sync.

200. **What is bus contention?**

Conflict when devices drive same bus.

201. **Why stack pointer auto-decrements during PUSH?**

To store data at lower memory addresses as stack grows downward.

202. **How does microprocessor handle recursive calls?**

By using stack to save return addresses for each call.

203. **What is front-end and back-end of CPU pipeline?**

Front-end fetches and decodes; back-end executes and writes results.

204. **Why instruction reordering is done by CPU?**

To reduce stalls and improve execution efficiency.

205. **Which instructions are privileged in 8086?**

Those that control system resources, like CLI and STI.

206. **Why instruction cache is smaller than main memory?**

To provide faster access and reduce latency.

207. **What is context switch in microprocessor?**

Saving and loading registers to switch between tasks.

208. **Why memory interleaving improves speed?**

It allows parallel access to multiple memory banks.

209. **Which instruction sets include SIMD operations?**

MMX, SSE, AVX in modern Intel processors.

210. **Why does pipeline have hazards?**

Due to instruction dependencies and branching.

211. **How is branch prediction implemented?**

Using static (fixed) or dynamic (history-based) algorithms.

212. **What is reorder buffer in CPU?**

It stores instructions executed out-of-order for correct commit.

213. **Why speculative load used in microprocessors?**

To fetch data before confirmation, reducing stall cycles.

214. **Which instruction handles signed division in 8086?**

IDIV instruction divides signed numbers.

215. **Why multiple clock domains in CPU?**

To allow components to run at optimal frequencies.

216. **What is memory-mapped I/O advantage?**

Same instructions can access both memory and I/O.

217. **How does branch target buffer work?**

It caches addresses of recently taken branches.

218. **Why instruction decode stage critical?**

Errors here affect entire execution pipeline.

219. **What is dual-ported memory?**

Memory with two independent access ports for parallel read/write.

220. **Why instruction prefetch improves performance?**

It reduces fetch delays by loading instructions in advance.

221. **What is write buffer in CPU?**

Temporary storage for writes before updating main memory.

222. **Why is instruction scheduling important?**

To minimize pipeline stalls and maximize throughput.

223. **Which instruction sets support bit manipulation?**

8085/8086 includes ROL, ROR, SHL, SHR, BT, BTS, BTR.

224. **How does hardware loop counter work?**

Automatically decrements register and checks zero to control loops.

225. **Why CPU uses speculative execution?**

To utilize idle pipeline stages and improve throughput.

226. **What is memory alignment?**

Data stored at addresses divisible by its size for efficiency.

227. **Why cache line size matters?**

It determines how much contiguous memory is fetched per access.

228. **Which instruction saves flags on stack?**

PUSHF in 8086 saves all flag registers.

229. **Why instruction register important?**

It holds current instruction being executed.

230. **What is dynamic voltage and frequency scaling (DVFS)?**

Adjusts CPU speed and voltage for power efficiency.

231. **How does TLB improve virtual memory access?**

It caches recent virtual-to-physical address translations.

232. **Why branch misprediction costly?**

Pipeline must flush and refill, wasting cycles.

233. **Which instruction repeats string operation?**

REP, REPE, or REPNE.

234. **What is priority encoder?**

Selects highest-priority active input among multiple requests.

235. **Why stack grows downward in most processors?**

To avoid overlapping with code and data segments.

236. **How does CPU handle nested interrupts?**

By stacking return addresses and enabling priority-based servicing.

237. **Which instruction swaps bytes in a word?**

XCHG can swap register contents; BSF/BTC used in bit-level.

238. **Why control unit can be hardwired or microprogrammed?**

Hardwired is faster; microprogrammed is flexible.

239. **What is difference between synchronous and asynchronous interrupts?**

Synchronous occur with instruction execution; asynchronous by external events.

240. **How does instruction fusion improve performance?**

Combines multiple instructions into a single micro-op.

241. **Why CPU uses multi-level caches?**

To reduce latency and increase hit rate for memory accesses.

242. **What is branch folding?**

Skipping execution of branch when outcome is already known.

243. **Which instructions support atomic operations?**

XCHG, CMPXCHG, and LOCK-prefixed instructions.

244. **Why stack frame pointer used in debugging?**

It helps trace local variables and procedure call history.

245. **How do superscalar pipelines handle multiple instructions?**

By executing independent instructions simultaneously in parallel pipelines.