Week 1 Task:

1 Install & Sanity-Check the Toolchain

Question to ask AI tools

"I have downloaded riscv-toolchain-rv32imac-x86_64-ubuntu.tar.gz. How exactly do I unpack it, add it to PATH, and confirm the gcc, objdump, and gdb binaries work?"

What AI tool should return / what you should do

- Step-by-step tar -xzf … command.
- export PATH=$HOME/riscv/bin:$PATH lines for ~/.bashrc.
- Verification commands (riscv32-unknown-elf-gcc --version, etc.).

---

2 Compile "Hello, RISC-V"

Question

"Show me a minimal C 'hello world' that cross-compiles for RV32IMC and the exact gcc flags to produce an ELF."

Expected Method

- Copy the one-line printf("Hello…") program.
- Run riscv32-unknown-elf-gcc -march=rv32imc -mabi=ilp32 -o hello.elf hello.c.
- Use file hello.elf to confirm it's 32-bit RISC-V.

---

3 From C to Assembly

Question

"How do I generate the .s file and explain the prologue/epilogue of the main function?"

Method

- riscv32-unknown-elf-gcc -S -O0 hello.c generates hello.s.
- ChatGPT explains why you see addi sp,sp,-16, sw ra,12(sp)…

---

4 Hex Dump & Disassembly

Question

“Show me how to turn my ELF into a raw hex and to disassemble it with objdump. What do each columns mean?”

Method

- riscv32-unknown-elf-objdump -d hello.elf > hello.dump.
- riscv32-unknown-elf-objcopy -O ihex hello.elf hello.hex.
- Walk through one instruction field (address, opcode, mnemonic, operands).

---

5 ABI & Register Cheat-Sheet

Question

“List all 32 RV32 integer registers with their ABI names and typical calling-convention roles.”

Answer Outline

- Table mapping x0-x31 to zero, ra, sp, gp, tp, t0-t6, s0-s11, a0-a7.
- Calling convention summary: a0-a7 = args/returns, s-regs callee-saved, t-regs caller-saved.

---

6 Stepping with GDB

Question

“How do I start riscv32-unknown-elf-gdb on my ELF, set a breakpoint at main, step, and inspect registers?”

Method

- Run riscv32-unknown-elf-gdb hello.elf → target sim → break main → run.
- Use info reg a0 / disassemble to watch execution.

---

7 Running Under an Emulator

(needed if participants don't yet have real hardware)

Question

"Give me spike or QEMU commands to boot my bare-metal ELF and print to the 'UART' console."

Method

- spike --isa=rv32imc pk hello.elf or
- qemu-system-riscv32 -nographic -kernel hello.elf.
- Show UART output in terminal.

---

8 Exploring GCC Optimisation

Question

"Compile the same file with -O0 vs -O2. What differences appear in the assembly and why?"

Method

- Two .s listings side-by-side.
- ChatGPT explains dead-code elimination, register allocation, inlining.

---

9 Inline Assembly Basics

Question

"Write a C function that returns the cycle counter by reading CSR 0xC00 using inline asm; explain each constraint."

Method

- Provide static inline uint32_t rdcycle(void) { uint32_t c; asm volatile ("csrr %0, cycle" : "=r"(c)); return c; }.
- Break down "=r"(c) and volatile.

---

10 Memory-Mapped I/O Demo

Question

“Show a bare-metal C snippet to toggle a GPIO register located at 0x10012000. How do I prevent the compiler from optimising the store away?”

Method

- Use volatile uint32_t *gpio = (uint32_t*)0x10012000; *gpio = 0x1;.
- Discuss the volatile keyword and alignment.

---

11 Linker Script 101

Question

“Provide a minimal linker script that places .text at 0x00000000 and .data at 0x10000000 for RV32IMC.”

Method

- Skeleton SECTIONS { .text 0x0 : { *(.text*) } .data 0x10000000 : { *(.data*) } }.
- Explain why Flash vs SRAM addresses differ.

---

12 Start-up Code & crt0

Question

“What does crt0.S typically do in a bare-metal RISC-V program and where do I get one?”

Answer Highlights

- Sets up stack pointer, zeroes .bss, calls main, enters infinite loop.
- Point to device-specific examples or newlib.

---

13 Interrupt Primer

Question

“Demonstrate how to enable the machine-timer interrupt (MTIP) and write a simple handler in C/asm.”

Method

- Write to mtimecmp, set mie, poll mstatus.
- Use __attribute__((interrupt)) or naked ISR pattern.

---

14 rv32imac vs rv32imc – What's the "A"?

Question

"Explain the 'A' (atomic) extension in rv32imac. What instructions are added and why are they useful?"

Answer

- Introduces lr.w, sc.w, amoadd.w, etc.
- Needed for lock-free data structures, OS kernels.

---

15 Atomic Test Program

Question

"Provide a two-thread mutex example (pseudo-threads in main) using lr/sc on RV32."

Method

- Spin-lock implementation in C with inline asm fallback.

---

16 Using Newlib printf Without an OS

Question

"How do I retarget _write so that printf sends bytes to my memory-mapped UART?"

Answer Outline

- Implement _write(int fd, char* buf, int len) that loops over bytes to UART_TX.
- Re-link with -nostartfiles + custom syscalls.c.

---

17 Endianness & Struct Packing

Question

“Is RV32 little-endian by default? Show me how to verify byte ordering with a union trick in C.”

Method

- Union of uint32_t and uint8_t[4], store 0x01020304, print bytes.
