1. (12 pts) What are three drawbacks for taking a single cycle to execute each instruction?

- A single cycle is as long as the slowest instruction
- It’s too memory intensive since memory and a data memory
- Modern processors have separate caches.

2. (6 pts) If you were to add a no-op instruction (the instruction performs no operation) to the MIPS ISA and considering the multi-cycle datapath, how many cycles would this instruction require?

\[ \text{NOP} = \text{AMD} \] 40, 1, 0, 0, 0, 0, 0, 0

Two cycles. One to load from memory. One to decode and set the select line so that no data passes through the system.

3. (5 pts) Which of the following statements are potential arguments for MIPS to have 32 general-purpose registers (GPRs) $s0$-$s31$ instead of 64 GPRs? Mark all that apply.

1) The ALU can be made smaller with 32 GPRs
2) The register file can be faster with 32 GPRs
3) The use of a 32-bit architecture implies the use of 32 GPRs
4) Encoding of R-type instructions with 64 GPRs would lead to very few opcodes available in the 32-bit instruction word
5) The control logic is simpler with 32 GPRs

4. (10 pts) The MIPS “jump and link” instruction implicitly stores the PC+4 value to a fixed register (R31). Discuss the implications, if any, if the ISA was changed such that any arbitrary register could be specified.

If register was specified then there would be less bits available for addresses, thus making your jump page smaller and would also mess up the idea of aligning the bits for the PC's address.
5. (7 pts) Which method, callee or caller saved registers, may result in fewer assembly instructions and why.

Using the callee saved register will result in fewer instructions. If the callee is using the temp regs (at 67), the caller must save them to the stack before doing call, otherwise only the callee has to save the save regs (at 86) to the stack. Only the callee uses the callee uses.

6. (10 pts) In MIPS ISA the source registers are fixed (Srs and Srt), while the destination register can be Srd (R-type) or Srt (I-type). Why is this approach preferable to keeping the destination register fixed and the source registers different?

R-Types require three arguments to the opcodes while I-Types require two and an immediate value from memory, so only only Rrd or Srt are ever distinct registers.

This is just into about the last format. You didn't answer the question.

7. (50 pts total) Consider the addition of a new instruction into the MIPS instruction set given the multi-cycle instruction execution implementation. This instruction, inc? Srd, Srt, is used to automatically increment the values stored in both registers Srd and Srt in a single instruction (i.e., Srd = Srd + 1 and Srt = Srt + 1).

Pretty sure I did this as IRS and BRT. I did account for writing to Rrd first.

a) (10 pts) Discuss why this instruction cannot be supported by the current multi-cycle datapath.

The output of the ALU is only 32-bits. This would require a 64-bit output and bus. Also the register file only has one 32-bit write port.

The controller would also need to be modified for the R-type instruction.?

Now?

Why?
b) (15 pts) If the instruction cannot be supported by the current multi-cycle datapath, discuss the architectural and control logic changes, at a high level, that are necessary to support this instruction.

- The \textit{R-type} state path would need to be modified for this special case (Main Controller).
- The bus width from the ALU to Register File would need to be increased to 64-bit, why?
- The register file will need an additional write port.
- The ALU needs to be modified to handle the new instruction.
- The ALU decoder needs to decode the new data in function field and Opcode.
d) (15 pts) How many cycles will this new instruction require? What will be done on each cycle?

3 cycles
- ① Decode, set proper select lines
- ② Set Register destination
- ③ Store data to register file

ALU ops currently take 4 cycles, so how is yours faster than that?

e) (5 pts) Does the clock cycle time need to change to support this new instruction?

No

f) (5 pts) What instruction type (I-, R-, J-type) would be most appropriate for this instruction and why?

R-Type. The instruction doesn't require any memory accesses. The data is already in the register file.