Partial credit rule: Must show your intermediate steps clearly for partial credit!

1. Fill in the blanks with the best answer of at most three words:

(a) Certain instructions that can run only in operating system routines and not in user programs are called _________________________ mode instructions.

(b) In the _________________________________ register-saving convention, all variables that are live in the caller are saved on the stack before each procedure call.

(c) In ISAs lacking program status word registers, compare instructions usually store the branch condition in a ____________________________.

(d) In most instruction formats, integer and floating point registers are distinguished by different _____________________________(register encodings/instruction opcodes/mode bits/program contexts).

(e) 64-bit floating point arithmetic is often called _______________________________ precision arithmetic.

(f) Most modern ISAs use ________________________________ addressing for branch targets of branch and jump instructions.

(g) In _______________________________ addressing, the full address of the memory location accessed is contained in a register.

(h) An optimized compilation of a while loop executes ________________________ branch or jump instruction(s) per iteration.

(i) When the data and status of I/O devices are stored in certain addresses of the memory address space, this approach to I/O is called _______________________________ I/O.

(j) The _______________________________ method is the most-often-used I/O method in embedded devices.

(k) When the return PC value for a procedure call cannot be kept in a register any longer, it is usually stored in ____________________________ (global.Stack/heap) memory.
2. For each subpart (i) to (x) below, circle all correct answers from among the four given - note that more than one answer may be correct!

(2 points * 8 = 16 points)

(i) Features that modern ISAs aim to deliver include

(a) a predictable runtime independent of implementation.

(b) fixed length instructions.

(c) instructions that can be used automatically by compilers.

(d) compatibility of the latest software for the ISA on old implementations of the ISA.

(ii) By looking at the instruction format specification for a processor, one can tell

(a) the actual size of memory available on a particular implementation.

(b) the range of branch displacements allowed.

(c) the number of pipeline stages in the computer.

(d) the maximum size of immediate constants allowed in instructions.

(iii) Modern compilers for 32-bit RISC architectures

(a) load a 32-bit constant memory address into a register in one instruction.

(b) need to know about whether branch delay slots are present to generate correct code.

(c) usually use the direct addressing mode for accesses to array elements.

(d) need to know the length of data hazard stalls for generating correct code.

(iv) I/O

(a) requires new instructions when implemented using special hardware registers.

(b) requires new instructions when implemented using memory-mapped I/O.

(c) requires interrupts when the I/O method is DMA.

(d) requires interrupts when the I/O method is programmed I/O with busy waiting.
(v) Interrupt-driven I/O without DMA

(a) can be used for both input and output.

(b) transfers data at the granularity of cache lines.

(c) allows computation during the time when an output device is being written to.

(d) requires less hardware support than I/O with DMA.

(vi) Regarding caller vs. callee saves,

(a) these are choices of who saves registers upon procedure calls and returns.

(b) which is better depends upon the particular program.

(c) both of these choices are available for interrupts.

(d) the callee-saves mechanism saves all registers that the callee reads.

(vii) A trap

(a) is an exceptional condition generated by the program itself.

(b) can only be invoked by a system call.

(c) is often called a synchronous interrupt.

(d) is a more efficient way of checking for errors than software checks.

(viii) In recursive procedures

(a) the size of each stack frame is dependent upon its recursive call depth so far.

(b) multiple copies of the procedure’s local variables are stored in different recursive frames.

(c) interrupts are not allowed during when a recursive procedure is executing.

(d) the maximum height of the stack may be unknown at compile-time.
3. Consider the following C code fragment where a and c are 32-bit integers:

```c
if (a < 0) {
    c = 1;
} else {
    c = 2;
}
```

When compiling the above fragment to DLX assembly, assume that variable a is available in register R1 at the beginning of the fragment, and that c is to be stored in register R2. Do not overwrite register R1 in your DLX code below. (5 + 3 = 8 points)

(a) Write DLX assembly code for the above fragment. No registers other than R0, R1 and R2 may be used, keeping in mind that R0 is fixed to zero. Use symbolic labels as needed.

(b) Write DLX assembly code for the C fragment above that uses a new conditional assignment instruction called CAI but no branches or jumps. It takes the form ‘CAI Rx, Ry, const’ which has the semantic “if (Ry) then Rx ← const”. In other words register Rx is set to the immediate integer constant ‘const’, but only if Ry is 1; otherwise Rx is unchanged. Besides registers R0, R1, and R2 to be used as above, you are also allowed to use another register R3.

4. A DLX compiler wants to perform a conditional branch of the form ‘BEQZ Rx, long_disp’, where `long_disp` is a 26-bit pc-relative displacement constant. Unfortunately in DLX, conditional branch displacements are restricted to 16 bits, so this is not directly possible. Write a sequence of DLX instructions that effectively perform a 26-bit pc-relative branch to `long_disp`. (Hint: use the J unconditional jump instruction as part of your sequence. It allows for displacements up to 26 bits.) (3 points)
5. Consider an instruction set for a hypothetical computer with 32-bit instructions and 32 registers. Assume that instructions come in three types A, B, and C:

Type A: 40 instructions with 3 register operands
Type B: 6 instructions with 2 register and 1 $x$-bit immediate operands.
Type C: 35 instructions with 2 register operands.

In all questions below, please solve for $x$ in different cases. You do not need to specify the exact format you choose in each part. 

\[(2 + 3 + 2 = 7 \text{ points})\]

(a) If the opcode field of all instructions is of equal length, then what is the maximum value of $x$ possible in such a format?

(b) If expanding opcodes are used, then what is the maximum possible value of $x$ among all possible encodings of the instruction set?

(c) Suppose for pipeline efficiency it is imperative that expanding opcodes be used only in the following way: first, a fixed-length field distinguishes the three instruction types, and then the remaining bits contain more opcode bits to distinguish instructions within a type. In this type of encoding, what is the maximum value of $x$?
6. A computer executes two programs at different times. Program A performs a significant amount of computation, but produces no output. Program B is exactly the same as program A except that it also outputs 50 words to an output device using interrupt-driven I/O. Each output sequence composes of the following four steps. (i) Wait in a loop until ‘done’ interrupts from all previous outputs have been received and handled. (Runtime is 2 cycles if wait loop exits immediately). (ii) Write the output data and status words for the current output (5 cycles). (iii) Complete output on device and generate an ‘done’ interrupt (1000 cycles); (iv) Handle each ‘done’ interrupt (80 cycles). Notice that step (iii) can be overlapped with computation in program B when such computation is available. Ignore any bus contention between output and computation.

\[3 + 2 + 2 = 7 \text{ points}\]

(a) If each word is output separately and one ‘done’ interrupt generated per word, how many more cycles does program B take compared to program A? Assume that successive words of output are separated by more than 1000 cycles of computation.

(b) Repeat part (a) if successive words of output are separated by only 300 cycles of computation.

(c) If a single 50-word DMA transfer is used for output instead of word-by-word transfers, how many cycles slower is program B than program A? Costs for DMA are 2, 11, 1000+N and 80 cycles respectively for steps (i), (ii), (iii) and (iv). N is the number of words transferred. Assume that program B does more than 1000 cycles of computation after its DMA output.
7. Consider the following recursive procedure in C that computes the well-known fibonacci function:

```c
int fib (int n) {
    if ((n == 1) || (n == 2)) { // The || operator is a Boolean OR
        return 1;
    }
    return fib(n-1) + fib(n-2);
}
```

(a) List the contents of the stack frame for procedure fib in any order.

(b) What is the maximum number of stack frames on the stack for a call to fib(10)?

8. Register R1 in a DLX machine contains the four bytes ‘XYZW’, where X is the most-significant byte, and W is the least-significant. Write a series of DLX instructions that composes the value ‘00YZ00’ in register R2, where 0 is a hexadecimal digit. **Do not** use any shift instructions. Do not use any registers other than R0, R1 and R2. Keep in mind that all immediate constants in DLX are 16 bits.

(3 points)