[20 marks] Digital Logic.

a) Let 'X' be the ternary connective such that 'Xpqr' is equivalent to '(p + q) /\ (q <--> r)'. 'p <--> q' is true iff 'p' and 'q' have the same truth value. '+' is exclusive or. 'F' and 'T' are the constant functions.

Using {'X', 'T'}, synthesize: ~p |= =| X p T T ✓

Using {'X', '~', 'F'}, synthesize: p \/ q |= =| ___ X ~~No solution~~ ✗

b) 'M' is the ternary 'minority' connective. 'Mpqr' is true iff a minority of its arguments is true. '#' is the ternary 'majority' connective. Thus, #pqr |= =| ~Mpqr. 'F' is a constant function.

Using {'M'}, synthesize: ~p |= =| M ~~p p q~~ ✓

Using {'#', 'F'}, synthesize: p /\ q |= =| # p q F ✓

2. [20 marks] Amdahl's Law. <two-decimal-place accuracy for reals>

a) On a uniprocessor, moderately serial portion A of program P takes 15 s, while perfectly parallel portion B takes 975 s. On a parallel computer, A speeds up by a factor of 'alpha', while B speeds up by the number of processors. On a machine with 1,024 processors, P's parallel run time is 5.237862 723214 s. What is the value of 'alpha'? What is the run time of P on 2,048 processors?

~~alpha = 0.079763372~~

~~new run time = 5.251089145~~ s ✗

b) A parallel computer can be built with an arbitrary number 'n' of 4-GFs/s cores. Portion A of data-parallel program P does constant serial work (4 GFs) independently of the amount of parallel work in portion B (we will scale B to match the number of cores). With one core, B = 4 GFs. A and B each take 1 s. With two cores, B = 8 GFs. A and B each take 1 s (two cores do twice the parallel work in the same time). With 'n' cores, B = 4n GFs. What is the ratio of the work done in 2 s on an n-core machine to that done in 2 s on a 1-core machine? What is the performance of program P on an n-core machine?

~~ratio = 1+n:2~~ ~~3~~ ✓

~~performance = 4+4n~~ GFs/s -3

3. [20 marks] Pipeline Boxes and Latches, and Pipeline Actions

```
+----------------------------------------------------------------------+ P
|    <I-cache>             <Register file>                 <D-cache>   | r
|                                                                      | o
|                        <Control circuitry>                           | c
|              +-+             +-+             +-+             +-+     | e
|    <f-box>   | |   <d-box>   | |   <x-box>   | |   <m-box>   | |   <w-box>
|              +-+             +-+             +-+             +-+     | s
|              f/d             d/x             x/m             m/w     | o
+----------------------------------------------------------------------+ r
```

```
                          1  2  3  4  5  6  7  8  9  0  1  2
1 l.d    f2,8(r2)         f  d  xi m1 m2 w
2 l.d    f4,16(r2)           f  d  xi m1 m2 w
3 mul.d  f6,f2,f4               f  d  s  s  x1 x2 x3 x4 n  w
```

Pipeline actions are specified with (box name, line number).

a) Action (d,1) puts 8, r2 into a latch for immediate use by action (xi,1).

b) Action (xi,2) puts 16+r2 into a latch for immediate use by action (m1,2).

c) Action (m2,2) puts ~~16+r2~~ value@ 16+r2 ✓ into a latch for immediate use by action (w,2).

d) Action (m2,2) puts f4 ✓ into a latch for immediate use by action ~~(w,2)~~.

Hex table:

0	0000	4	0100	8	1000	c	1100
1	0001	5	0101	9	1001	d	1101
2	0010	6	0110	a	1010	e	1110
3	0011	7	0111	b	1011	f	1111

Hex flips:

0 - f	4 - b
1 - e	5 - a
2 - d	6 - 9
3 - c	7 - 8

Hex powers:

1, 16, 256, 4096

Hex naturals:

a	b	c	d	e	f
10	11	12	13	14	15

4. [20 marks] Instructions with Base Register and Offset.

a) Assume 16-bit registers, instructions, and memory addresses, and 8-bit immediates. Consider '1.d f6,-115(r2)'. Show the (4-hexit) hexadecimal representation of the 16-bit integer that will be added to 'r2'.

ans: F F 8 D

b) Assume 64-bit registers, instructions, and memory addresses, and 32-bit immediates. Consider 'bne r1,r2,loop', where 'loop' has the value -7,320. Show the (8-hexit) hexadecimal representation of the low-order 32 bits of the 64-bit integer that will be added to 'PC'.

ans: F F F F 1 B 4 0

5. [20 marks] Fractional-Number Formats.

A special-purpose computer has 53-bit registers. It uses an "unbiased" floating-point (UFP) system. This is IEEE floating point except i) there is no sign bit, ii) the exponent is 10-bit two's complement, iii) the fractional field is 43 bits, and iv) the two most-negative exponents have been removed from the exponent range. There is no exponent bias, and normalization is strictly enforced.

a) i) Write 4 5/28 as a (3+infinity)-bit non-normalized binary fixed-point number, using 'bar' for infinite repetition. ii) Now, write 4 5/28 as a (1+43)-bit normalized binary-fixed point number in blackboard notation, using both '<x:n>' for finite repetition and a scale factor. 28 = 4 * 7.

i) $100.00\overline{101}$

-2 ii) 1.0000<101:~~39~~>[2]

b) What is the gap i) between the smallest strictly positive UFP number and the next larger UFP number, and ii) between the largest UFP number and the next smaller UFP number? Use powers of two.

i) 2^{-508}

-7 ii) ~~2^{511}~~