- This exam is a closed book, closed notes, closed laptops and smartphones, etc. exam
- All that you can use is either a pen or pencil
- The last few pages provide formula, which you may be helpful
- Do not spend too much time on any one question.

**Honor Code statement :** I pledge that this submission is solely my work. I pledge that I have not provided help to anyone. I pledge that I have not received help from anyone.

**Signature** ____________________________________________

<table>
<thead>
<tr>
<th>Section</th>
<th>Question Type</th>
<th>Question Number(s)</th>
<th>Points Possible</th>
<th>Points Earned</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>True/False</td>
<td>1-5</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>II</td>
<td>Multiple Choice</td>
<td>6, 7</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>III</td>
<td>Semaphores</td>
<td>8</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>IV</td>
<td>Logic</td>
<td>9, 10</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>V</td>
<td>Short Answer</td>
<td>11-15</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>VI</td>
<td>Multiple Answers</td>
<td>16</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total without curve</td>
<td></td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

**Curve Adjustment**

| VII     | Extra Credit         | 17, 18             | 5               |               |

**Total Exam Score**

---

**I. True/False. Instructions : Circle either True or False. No partial credit. 3 points each.**

1. True / False A **data hazard** arises when there are not enough functional units of the same step of a pipeline to permit multi-threading.

2. True / False When a semaphores is created, it **must** be **initialized** to 0 (zero).

3. True / False Every loop in a program **should** be parallelized if there are idle CPUs waiting

4. True / False In the context of concurrent programs, **partial correctness** is achieved when every loop and procedure call terminates

5. True / False Two processes are **independent** when the write set of each is disjoint from the read and write sets of the other.
II. Multiple Choice
Instructions: Circle the ONE letter to specify the BEST answer choice from among those listed. No partial credit. 8 points each.

6. If the below for loop were parallelized, how many different ways could the parallel version of the program print out the values of i? Assume print(i) is an atomic action.

```c
for (i=0; i<3; i++){
    print(i);
}
```

A. 1  
B. 2  
C. 4  
D. 6  
E. 12 
F. None of the above

7. For a pipeline with depth 4, with stage 1 latency of 2ns, stage 2 latency of 2ns, stage 3 latency of 4ns and stage 4 latency of 2ns, which of the following statements is true.

A. If the pipeline is empty, upon processing a single instruction, the observed pipeline latency will be 12ns for that one instruction.  
B. Assuming no stalls, after a large number of instructions have been executed, the observed pipeline throughput will be 2 instructions every 1 ns  
C. If the pipeline is empty, and 2 instructions are processed, one after the other, then the second instruction’s latency for the entire pipeline will be 12 ns  
D. None of the above
III. Semaphores. 14 points. Partial credit given.

8. The pseudocode below contains three functions, calcA() and calcB(), each with 2 instructions, and main(). All instructions inside calcA and calcB are atomic. Assume that a Semaphore class is available, which has a constructor Semaphore(int semaphoreValue), and functions increment() and decrement(). Variables x and y are global variables (ie, shared among all functions) saved in shared memory. Declare and use semaphores wherever needed, so that when the main method runs to completion, the output of the program is:

Values of x and y are 7 and 12

You can ONLY declare and use semaphores. You cannot update existing variables’ values, insert new code other than semaphores, etc.

```javascript
function calcA(){
    x = x + 1;
    y = 12;
}

function calcB(){
    y = x + 1;
    y = y + 1;
}

main(){
    x = 6; y = 7;
    co
    calcA();
    calcB();
    oc
    print("Values of x and y are ", x, "and", y);
}
```
IV. Logic. Points as indicated. Partial credit given.

9. (5 points) Prove that \( \neg(A \implies B) \iff A \land \neg B \)

10. (10 points) Prove the following bidirectional **without** using a truth table:

\[
(A \land \neg B) \lor B \iff A \lor B
\]

Use as many of the proof lines as are needed. If you need a sub-proof, indent it. Justify each line of your proof

<table>
<thead>
<tr>
<th>Proof Lines</th>
<th>Justification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
V. Short Answer. Provide a concise answer to each question. Partial credit. 6 points each.

11. Are either of the arms in the below concurrent program at-most-once? Why or why not. Explain.

```java
int x = 44, y=712;
co x = x + 1; // y++; oc
```

Your answer: __________________________________________

________________________________________________________________________

________________________________________________________________________

12. Assume a 5 step pipeline with the following stage latencies

<table>
<thead>
<tr>
<th>Stage</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage_1</td>
<td>5ns</td>
</tr>
<tr>
<td>Stage_2</td>
<td>6ns</td>
</tr>
<tr>
<td>Stage_3</td>
<td>8ns</td>
</tr>
<tr>
<td>Stage_4</td>
<td>3ns</td>
</tr>
<tr>
<td>Stage_5</td>
<td>4ns</td>
</tr>
</tbody>
</table>

Under these conditions, what will be the pipeline’s throughput when the 12,435th of 20,000 instructions is executing, and what will be the pipeline’s latency, also for instruction 12,435? Do not assume any stalls other than those that might be imposed by the stage latencies listed above.

Pipeline throughput: ____________________________________________

Pipeline latency: _______________________________________________

13. What cache reuse ratio is needed so that a program that uses cache is 2 times better (performance gain) than a program that does not use cache? Assume cache access time of 3ns and memory access time of 24ns.

Your Answer: ____________________________________________
14. Assume the below code statements $S_1$ and $S_2$, and that $x$ and $y$ are both initially 8. Both $x$ and $y$ are shared variables.

$S_1$: $x = x + 1;$  
$S_2$: $y = y - x;$

Under these conditions what are the possible final values of $x$ and $y$ when program P1, below, executes to completion?

P1 : co <S1;> // <S2;> oc

Possible value(s) of $x$: ____________________________________________

Possible value(s) of $y$: ____________________________________________

15. Define the term **concurrent**

Your Answer : _______________________________________________________

_______________________________________________________________

_______________________________________________________________

_______________________________________________________________

VI. Multiple Answers. Select ALL letters that are correct answers for the question. One, two or more, or even all choices may constitute a full correct answer. Partial credit is given. 10 points each.

16. Which of the following is/are true about the syntax and functionality of the sample await statement in the below box?

< await (p > 0) s = s + 10 >

A. $s = s + 10$ is the delay condition  
B. The < > brackets specify an atomic action  
C. $s = s + 10$ begins executing when $p = 0$  
D. It is possible that this await statement implements the behavior of an infinite loop  
E. $p > 0$ is guaranteed to be false when execution of $s = s + 10$ ends
VII. Extra credit. Partial credit not given.

Two extra credit questions will appear on the real exam.

VIII. Formula, Axioms, etc.

Cache/Main memory gain

\[ G(\tau, \beta) = \tau / (\beta + \tau (1-\beta)) \]
\[ T_c = T_m / \tau \]

Logical Axioms

\[
\begin{align*}
A \land (B \lor C) & \equiv (A \land B) \lor (A \land C) \\
A \lor (B \land C) & \equiv (A \lor B) \land (A \lor C) \\
\neg \neg A & \equiv A \\
A \lor \neg A & \equiv true \\
A \land true & \equiv A \\
A \land false & \equiv false \\
A \land \neg A & \equiv false \\
A \Rightarrow true & \equiv true \\
A \Rightarrow false & \equiv \neg A \\
true \Rightarrow A & \equiv A \\
false \Rightarrow A & \equiv true \\
A \Rightarrow A & \equiv true \\
A \Rightarrow B & \equiv \neg A \lor B \\
A \Rightarrow B & \equiv \neg B \Rightarrow \neg A \\
\neg (A \Rightarrow B) & \equiv A \land \neg B \\
A \land (A \lor B) & \equiv A \\
A \lor (A \land B) & \equiv A \\
A \land (\neg A \lor B) & \equiv A \lor B \\
A \lor (\neg A \land B) & \equiv A \lor B \\
\neg (A \land B) & \equiv \neg A \lor \neg B \\
\neg (A \lor B) & \equiv \neg A \land \neg B
\end{align*}
\]

Logical Inference Rules

- Modus Ponens
  \[
  A \Rightarrow B, A \frac{}{B}
  \]
- Modus Tollens
  \[
  A \Rightarrow B, \neg B \frac{}{\neg A}
  \]
- Conjunction
  \[
  A, B \frac{}{A \land B}
  \]
- Simplification
  \[
  A \land B \frac{}{A}
  \]
- Addition
  \[
  A \frac{}{A \lor B}
  \]
- Disjunctive syllogism
  \[
  A \lor B, \neg A \frac{}{B}
  \]
- Hypothetical syllogism
  \[
  A \Rightarrow B, B \Rightarrow C \frac{}{A \Rightarrow C}
  \]
- Constructive dilemma
  \[
  A \lor B, A \Rightarrow C, B \Rightarrow D \frac{}{C \lor D}
  \]
- Destructive dilemma
  \[
  \neg C \lor \neg D, A \Rightarrow C, B \Rightarrow D \frac{}{\neg A \lor \neg B}
  \]