1. Implement the logic equation below using only 3 Input NOR gates. (5 pt.)

\[ Y = \overline{(A \cdot B)} \cdot (D + E) \quad ; \quad A.H, B.L, D.L, E.L, Y.H \]

No Key -0.5
2 Input NORs -1
Missing inverter -1 ea.
Extra Inverter -0.5 or -1 depending on situation

Ground error -0.5
Floating -1

2. For the circuit below, derive Y.L. DO NOT SIMPLIFY! (5 pt.)

\[ Y.L = \overline{(A \cdot B)} \cdot (\overline{C+D}) + E = (\overline{A+B}) \cdot (C+D) + E = (A\oplus B)(C+D) \oplus E \]
3A. For the simple CPU in Appendix A and the following ROM contents, fill out the Cycle Table below for instructions stored in ROM. **Assume all registers are initially reset to zero.** Use 'X' to indicate a “don’t care” condition and show all answers in HEX. (9 pts.)

<table>
<thead>
<tr>
<th>Cycle</th>
<th>State</th>
<th>Input3:0</th>
<th>IR</th>
<th>PC</th>
<th>Reg. A3:0</th>
<th>MSA1:0</th>
<th>MSC2:0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (reset)</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x\times x</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x\times x</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>x\times x</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>x\times x</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>x\times x</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>x\times x</td>
</tr>
<tr>
<td>7</td>
<td>0\times 2</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>x\times x</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>11</td>
<td>110</td>
</tr>
</tbody>
</table>

3B. After the instructions above are executed, what should the contents of Register A:30 be in the 9th cycle? 3 HEX (1 pt.)

4. In your simple CPU used in Lab #9, what **HARDWARE modifications** and **new HARDWARE** is required to add the new instruction below? (4 pt.)

STAA Addr ;store register A to memory specified by the 8 bit address operand “Addr”

IR2:0 → IR3:0

PC3:0 → PC7:0

MAR or Temp reg to hold operand (address)

Addr mux to select between PC and MAR

Tri-states (buffers) on output bus to connect to input bus

New control signals: R/W, MAR-LD, Addr Sel
5. See Appendix B to write the required program below. Write your code in the left column first and then wrap around to the right column for extra lines/space. (8 pt.)

\[-127 + 127 = 0 \quad \text{constant} = 0x7F\]

\[\text{Ldaa } \#0; \quad \text{clear sum}\]
\[\text{Staa } 0x4080\]
\[\text{Ldab } \#0x7F \quad \text{constant}\]
\[\text{Ldy } \#0x4080; \quad \text{j input ptr}\]

**Top:**
- \[\text{Ldaa } 0;y \quad \text{last value?}\]
- \[\text{BEQ } \text{Done}\]
- \[\text{Sum } \text{ba}, \quad j = 127?\]
- \[\text{BNE } \text{skip}\]

\[\text{Ldd } 0x4080; \quad \text{sum}\]
\[\text{INCA } \quad \text{j sum}+1\]
\[\text{Staa } 0x4080\]
\[\text{Skip} +1 \text{NY}\]
\[+1 \text{BNE } \text{Top}\]

**Done:**
- \[\text{BEQ } \text{Done}\]

6. See the attached G-CPU program in Appendix C. Assuming that the code was placed in ROM starting at address 0x0. Answer the questions below:

**6A.** Based on the program execution, what size SRAM must exist in the G-CPU memory map?

RAM size: $C000 - FFFF$ 16K x 8 (1 pt.)

**6B.** What is the -RAM_CE Logic Equation? $-\text{RAM}_{-CE} = A_{15} \times A_{14}$ or $= A_{15} \times A_{14} \times D_{5}\_L$ (1 pt.)

**6C.** If the clock is 50 MHz, how many seconds does it take to execute the STAB 0xFFFF instruction? (1 pt.)

STAB 0xFFFF Execution Time = $\frac{5}{\sqrt{50} \times 10^6} = 10^{-7}$ seconds 0.1 μsec

**6D.** Assuming that the loop for the code in Appendix C is executed twice, show all SRAM addresses/data modified by the loop. (3 pt.)

<table>
<thead>
<tr>
<th>ADDRESS (Hex)</th>
<th>DATA (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C000$</td>
<td>FF +0.5 +0.5</td>
</tr>
<tr>
<td>$C001$</td>
<td>$03 \leftarrow FF$ 2nd Pass</td>
</tr>
<tr>
<td>$C002$</td>
<td>$64 + 0.5$ +0.5</td>
</tr>
</tbody>
</table>

Page 3

Page Score =
6E. Fill out the cycle diagram below for execution of the \textbf{LDA A 5,Y} and \textbf{STAB 0,X} instructions in the \textbf{SECOND PASS} of the loop. Show \textbf{ALL VALUES IN HEX}.

<table>
<thead>
<tr>
<th>Cycle</th>
<th>State</th>
<th>Addr Bus</th>
<th>Data Bus</th>
<th>Reg Driving Addr Bus</th>
<th>Device Driving Data Bus</th>
<th>IR</th>
<th>PC</th>
<th>X Reg</th>
<th>Y Reg</th>
<th>R/-W</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0x12</td>
<td>OD</td>
<td>PC</td>
<td>Rom</td>
<td>OF</td>
<td>12</td>
<td>0011</td>
<td>0001</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>0012</td>
<td>OD</td>
<td>PC</td>
<td>Rom</td>
<td>OD</td>
<td>12</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1E</td>
<td>0013</td>
<td>05</td>
<td>PC</td>
<td>Rom</td>
<td>OD</td>
<td>13</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1F</td>
<td>0006</td>
<td>04</td>
<td>Y BLK</td>
<td>Rom</td>
<td>OD</td>
<td>14</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0014</td>
<td>12</td>
<td>PC</td>
<td>Rom</td>
<td>OD</td>
<td>14</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0014</td>
<td>12.5</td>
<td>PC</td>
<td>Rom</td>
<td>12</td>
<td>14</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>28</td>
<td>0015</td>
<td>00</td>
<td>PC</td>
<td>Rom</td>
<td>12</td>
<td>15</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>29</td>
<td>C001</td>
<td>FF</td>
<td>X BLK</td>
<td>CPU</td>
<td>12</td>
<td>16</td>
<td>C001</td>
<td>0001</td>
<td>0</td>
</tr>
</tbody>
</table>

7. \textbf{Extra Credit}. How many times does the loop run in \textbf{HEX}? (1 pt.)

\begin{tikzpicture}
  \node (node1) at (0,0) {32};
  \node (node2) at (1,0) {Hex};
  \node (node3) at (0,-1) {+1};
  \node (node4) at (1,-1) {+1 0.5 0.5 +1}
  \node (node5) at (1.5,-1) {each FF 64}
  \node (node6) at (2.5,-1.5) {each}
\end{tikzpicture}
Appendix B. Programming Problem #4

Write a G-CPU assembly program to sum the number of \(-127\) values stored in RAM starting at address 0x4081 and \textit{terminating with a zero} (last value in the array). The final "SUM" should be stored at 0x4080 and should be used as temporary storage for the SUM during your program execution. Use the Y register as your input pointer and register B to hold the required constant.

Assume the following new instructions are available:

- \texttt{DECA}, decrement Register A (A-1 => A),
- \texttt{INCA}, increment Register A (A+1 => A).

Appendix C. G-CPU Code for Problems 5 - 10:

```
ORG 0x0 ;this tells the assembler to start the program at addr zero
LDAB #0xFE  0,1
STAB 0xFFFF 2,3,4
LDAB #100   5,6
STAB 0xFFFF 7,8,9
LDY #0x0   A, B, C
LDX #0xC000 D, E, F

Top:
LDAB 3,Y 2 1st pass y=0
LDAA 5,Y 3 2nd pass x=1

\*\* STAB 0,X 1 1st pass x=0000 = 0
STAA 1,X 1
INY
INX
LDAB 0xFFFF
LDAA 0xFFFF
SUM_BA
STAA 0xFFFF
BNE Top

End1:
BEQ END1
```

![Diagram](image.png)