Solutions

C to LC3 Pointers: Examine the code segments below. What are the LC3 instructions generated by a C compiler for the C code below. The symbol table is shown below...

Symbol Table

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Offset</th>
<th>Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>Int</td>
<td>0</td>
<td>main</td>
</tr>
<tr>
<td>ptr</td>
<td>Int</td>
<td>-1</td>
<td>main</td>
</tr>
</tbody>
</table>

Recall: R5 is frame pointer (dynamic link), R6 is top of stack, R4 is pointer to static area.

\[ i = 4; \]

\[
\begin{align*}
\text{AND } R0, R0, #0 \\
\text{ADD } R0, R0, #5 \\
\text{STR } R0, R5, #0
\end{align*}
\]

\[ \text{ptr = } \&i; \]

\[
\begin{align*}
\text{ADD } R0, R5, #-0 &; \text{get address of } i \text{ into } R0 \\
\text{STR } R0, R5, # -1 &; \text{and set it equal to } ptr \\
&; \text{store address of } i \\
&; \text{local var } ptr
\end{align*}
\]

\[ \text{*ptr = *ptr +1;} \]

\[
\begin{align*}
\text{LDR } R1, R5, #-1 &; \text{get address stored at } ptr \\
\text{LDR } R0, R1, #0 &; \text{dereference } ptr \text{ to get value of } *ptr \text{ into } R1 \\
\text{ADD } R0, R0, #1 &; \text{add one to the value} \\
\text{STR } R0, R1, #0 &; \text{store result where the pointer points}
\end{align*}
\]
C to LC3 Arrays: Examine the code segments below. What are the LC3 instructions generated by a C compiler for the C code below. The symbol table is shown below. grid is array of size 10.

Symbol Table

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Type</th>
<th>Offset</th>
<th>Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>grid</td>
<td>int</td>
<td>-9</td>
<td>foo</td>
</tr>
<tr>
<td>x</td>
<td>int</td>
<td>-10</td>
<td>foo</td>
</tr>
<tr>
<td>ptr</td>
<td>int</td>
<td>-11</td>
<td>foo</td>
</tr>
<tr>
<td>i</td>
<td>int</td>
<td>-12</td>
<td>foo</td>
</tr>
</tbody>
</table>

Recall: R5 is frame pointer (dynamic link), R6 is top of stack, R4 is pointer to static area.

grid[6] = 5;
AND R0, R0, #0; set value of 5 in a register
ADD R0, R0, #5 ; R0 = 5
ADD R1, R5, #-9 ; get address of grid[0]
ADD R1, R1, #6 ; add 6 to it to get address @grid[6]
; to fetch grid[i] increment address
; of grid[0] by i
STR R0, R1, #0 ; store 5 into that address

x = grid[3] + 1;
ADD R1, R5, #-9 ; get address of grid[0]
ADD R1, R1, #3 ; add 3 to the address
LDR R0, R1, #0 ; fetch value at grid[3]
ADD R0, R0, #1 ; add 1 to it
STR R0, R5, # -10; store into x

grid[i] = x;
LDR R0, R5, #-10 ; get value of x into register R0
LDR R1, R5, #-12 ; get value of i into register R1
ADD R2, R5, #-9 ; get address grid[0] into R2
ADD R2, R2, R1 ; and add i to it to get @grid[i]
STR R0, R2, #0 ; store value of x into this
; address
ptr = grid;
ADD R0, R5, #-9 ; R0 = address of grid[0]
STR R0, R5, #-11 ; store R0 into ptr