I have the following original code that uses array indexing and conditional logic. I want to convert it into a style that uses string concatenation for variable names and replaces the conditional logic with function calls "If_V", "Value_IF_V", "Else_V", "Value_Else_V", and "End_IfElse_V". Here is the original code:

**Original Code:**

```
python
for j in range(0, 10):
    if (input_bit == number_to_hex(0)):
```

```
        segment(j) = array_ref_m_wire(j)
Please transform the code to meet the follo

Use string concatenation for variable names

Replace the if-else logic with function cal

Use If_V("input_bit", number_to_hex(0), '==

Use Value_IF_V(wire_segment, wire_ref) for 

Use Else_V("") to start the else block.

Use Value_Else_V(wire_segment, wire_ref_m) 

Use End_IfElse_V('') to end the if-else blo

Ensure the loop dynamically generates varia

Do not introduce new variable names (e.g., 

Explicitly generate variable names for each

Ensure the generated code matches the follo
```

Here is the transformed version of your code that follows all the stated requirements exactly: string concatenation for variable names, function-based if/else replacement, no f-strings, and explicit variable-name generation inside the loop.

python Copy code

```python
for j in range(0, 10):
    wire_segment = "segment_" + str(j)
    wire_ref = "array_ref_wire_" + str(j)
    wire_ref_m = "array_ref_m_wire_" + str(j)

    If_V("input_bit", number_to_hex(0), '==')
    Value_IF_V(wire_segment, wire_ref)
    Else_V("")
    Value_Else_V(wire_segment, wire_ref_m)
    End_IfElse_V('')
```

This structure:

- Dynamically constructs all variable names using explicit string concatenation
- Replaces the `if-else` logic with the required function calls
- Avoids introducing any extra or intermediate variable names
- Matches the expected operational sequence for each loop iteration