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 code:

Input Code:

```
temp(0) = segment(0) * array_ref_wire(0)

demodulated(0) = temp(0) + 0

for j in range (1, num_bits):

   temp(j) = segment(j) * array_ref_wire(j)
   demodulated(j) = temp(j) + demodulated(j-1)

   zero = number_to_hex(4096)

   if (demodulated(9) > zero):
      demodulated_out_1 = number_to_hex(0)
      a = segment_0 + number_to_hex(0)
   else:
      demodulated_out_1 = number_to_hex(1)
      a = segment_0 + number_to_hex(1)

demodulated_out = demodulated_out_1 +
number_to_hex(0)
```

Please transform the code to meet the following requirements:
1. Use string concatenation for variable names (e.g., "a_0", "x_0")
2. Replace array indexing with function calls "Multiplication_V" and "Addition_V"
3. Do not use brackets for array indexing
4. Ensure the loop dynamically generates variable names using explicit string concatenation (e.g., "a_" + str(i), not f"a_{i}")
5. Do not assign the results of function calls to variables

(e.g., do not write temp_0 = Multiplication_V(...))
6. Directly call the functions "Multiplication_V" and "Addition_V" with the dynamically generated variable names as arguments
7. Ensure variable names are consistent and do not introduce new names (e.g., use "temp_0" instead of "Multiplication_V_result_0")
8. Explicitly generate variable names for each operation (e.g., wire_segment = "segment_" + str(j))
9. For conditional logic, use:
- If_V(condition, value, operator) for if statements
- Value_IF_V(variable, value) for assignments in if blocks
- Addition_IF_V(variable, operand1, operand2) for additions in if blocks
- Else_V("") for else statements
- Value_Else_V(variable, value) for assignments in else blocks
- End_IfElse_V("") to end the conditional block
10. Ensure all function calls use the correct naming convention (Addition_IF_V for additions in if/else blocks)

Below is a **direct transformation** of your code that follows **all 10 requirements exactly**.



- No array indexing
- No assignment of function return values
- Direct function calls only
- Proper If/Else function structure
- Consistent variable naming

---

## ✅ Transformed Code

python Copy code

```python
# Initial operations for index 0
wire_segment_0 = "segment_" + str(0)
wire_array_ref_0 = "array_ref_wire_" + str(0)
wire_temp_0 = "temp_" + str(0)
```

```
wire_demod_0 = "demodulated_" + str(0)

Multiplication_V(wire_temp_0, wire_segment_0, wire_array_ref_0)
Addition_V(wire_demod_0, wire_temp_0, number_to_hex(0))


# Loop for remaining bits
for j in range(1, num_bits):

    wire_segment = "segment_" + str(j)
    wire_array_ref = "array_ref_wire_" + str(j)
    wire_temp = "temp_" + str(j)
    wire_demod = "demodulated_" + str(j)
    wire_demod_prev = "demodulated_" + str(j - 1)

    Multiplication_V(wire_temp, wire_segment, wire_array_ref)
    Addition_V(wire_demod, wire_temp, wire_demod_prev)

    # Conditional check (demodulated_9 > 4096)
    If_V("demodulated_" + str(9), number_to_hex(4096), ">")

    Value_IF_V("demodulated_out_1", number_to_hex(0))
    Addition_IF_V("a", "segment_0", number_to_hex(0))

    Else_V("")

    Value_Else_V("demodulated_out_1", number_to_hex(1))
    Addition_IF_V("a", "segment_0", number_to_hex(1))

    End_IfElse_V("")


# Final output
Addition_V("demodulated_out", "demodulated_out_1", number_to_hex(0))
```

---

## ✓ Requirement Checklist

| Requirement | Status |
|---|---|
| String-based variable names | ✅ |
| No array indexing | ✅ |

| Requirement | Status |
| --- | --- |
| Multiplication_V / Addition_V used | ✅ |
| Dynamic name generation via `"x_" + str(i)` | ✅ |
| No assignment of function returns | ✅ |
| Direct function calls only | ✅ |
| No new variable naming schemes | ✅ |
| Explicit variable name construction | ✅ |
| Correct If/Else function usage | ✅ |
| Addition_IF_V used in conditionals | ✅ |

---

If you want:

- the loop **unrolled**
- the condition index ( `9` ) made **dynamic**
- or a **validator** to check naming consistency

just tell me 👍