```
C:\Users\usuario\Anaconda3\envs\tfq\python.exe C:/Users/usuario/qGAN/quantumGAN/performance_testing/
performance_get_output_generator.py
[0.23242188 0.13183594 0.53466797 0.10107422]
[0.21923828 0.13134766 0.56396484 0.08544922]
[0.22607422 0.13378906 0.55810547 0.08203125]
[0.22753906 0.13134766 0.56005859 0.08105469]
[0.23144531 0.13867188 0.54003906 0.08984375]
[0.23828125 0.13623047 0.54394531 0.08154297]
[0.23681641 0.14599609 0.52050781 0.09667969]
[0.22509766 0.14306641 0.54638672 0.08544922]
[0.22070312 0.1328125  0.5546875  0.09179688]
[0.22949219 0.13574219 0.54638672 0.08837891]
[0.21679688 0.14404297 0.54150391 0.09765625]
[0.23730469 0.15527344 0.52392578 0.08349609]
[0.23193359 0.14208984 0.54736328 0.07861328]
[0.23339844 0.13818359 0.54931641 0.07910156]
[0.23535156 0.13818359 0.54101562 0.08544922]
[0.22509766 0.13867188 0.55371094 0.08251953]
[0.22460938 0.13427734 0.54394531 0.09716797]
[0.22265625 0.11572266 0.57519531 0.08642578]
[0.21386719 0.14355469 0.55078125 0.09179688]
[0.22949219 0.14306641 0.53369141 0.09375   ]
[0.23876953 0.1328125  0.54052734 0.08789062]
[0.22412109 0.13720703 0.54589844 0.09277344]
[0.22998047 0.14599609 0.53955078 0.08447266]
[0.24267578 0.13623047 0.53515625 0.0859375 ]
[0.23193359 0.12939453 0.56103516 0.07763672]
[0.25       0.1484375  0.51171875 0.08984375]
[0.22021484 0.1328125  0.55712891 0.08984375]
[0.21435547 0.13623047 0.55859375 0.09082031]
[0.23388672 0.12939453 0.54785156 0.08886719]
[0.21582031 0.1328125  0.55957031 0.09179688]
[0.22705078 0.125      0.56884766 0.07910156]
[0.2265625  0.14306641 0.55517578 0.07519531]
[0.22802734 0.14111328 0.53955078 0.09130859]
[0.22363281 0.140625   0.54589844 0.08984375]
[0.21240234 0.13574219 0.56201172 0.08984375]
[0.23144531 0.12548828 0.55664062 0.08642578]
[0.22314453 0.13574219 0.56005859 0.08105469]
[0.21728516 0.14111328 0.546875   0.09472656]
[0.20703125 0.14111328 0.56005859 0.09179688]
[0.22851562 0.14355469 0.54248047 0.08544922]
[0.23144531 0.13818359 0.54931641 0.08105469]
[0.22509766 0.13769531 0.54492188 0.09228516]
[0.24511719 0.125      0.54882812 0.08105469]
[0.23095703 0.14648438 0.54248047 0.08007812]
[0.22167969 0.13623047 0.54394531 0.09814453]
[0.24169922 0.12792969 0.54638672 0.08398438]
[0.22705078 0.12792969 0.54785156 0.09716797]
[0.23291016 0.12451172 0.55761719 0.08496094]
[0.22265625 0.140625   0.5546875  0.08203125]
[0.21875    0.15185547 0.55126953 0.078125  ]
[0.23486328 0.13525391 0.54052734 0.08935547]
[0.22412109 0.16699219 0.52539062 0.08349609]
[0.22509766 0.13720703 0.54736328 0.09033203]
[0.21240234 0.14892578 0.54638672 0.09228516]
[0.24072266 0.14501953 0.53076172 0.08349609]
[0.22509766 0.13964844 0.54736328 0.08789062]
[0.24414062 0.13574219 0.53369141 0.08642578]
[0.22314453 0.14160156 0.56152344 0.07373047]
[0.24804688 0.13818359 0.53125    0.08251953]
[0.20996094 0.13818359 0.55761719 0.09423828]
[0.234375   0.12939453 0.55908203 0.07714844]
[0.21435547 0.140625   0.55322266 0.09179688]
[0.23486328 0.11621094 0.55859375 0.09033203]
[0.20751953 0.14111328 0.5625     0.08886719]
[0.22021484 0.14160156 0.5546875  0.08349609]
[0.24609375 0.13818359 0.52246094 0.09326172]
[0.21484375 0.14013672 0.5625     0.08251953]
[0.21240234 0.13476562 0.56298828 0.08984375]
[0.24804688 0.13867188 0.52148438 0.09179688]
[0.23632812 0.14111328 0.53613281 0.08642578]
[0.23583984 0.13818359 0.54589844 0.08007812]
[0.23486328 0.14501953 0.54199219 0.078125  ]
[0.23095703 0.13867188 0.5390625  0.09130859]
[0.22705078 0.13818359 0.54052734 0.09423828]
[0.21533203 0.1328125  0.5625     0.08935547]
[0.23144531 0.13330078 0.53613281 0.09912109]
[0.22509766 0.13964844 0.55029297 0.08496094]
[0.22558594 0.12939453 0.56201172 0.08300781]
[0.22167969 0.13720703 0.56347656 0.07763672]
[0.20654297 0.14013672 0.56103516 0.09228516]
[0.23535156 0.14501953 0.54980469 0.06982422]
[0.23876953 0.13037109 0.54638672 0.08447266]
[0.24121094 0.13623047 0.54296875 0.07958984]
[0.22949219 0.13427734 0.54394531 0.09228516]
[0.2265625  0.14453125 0.54199219 0.08691406]
[0.22167969 0.14160156 0.54833984 0.08837891]
[0.23339844 0.13818359 0.54052734 0.08789062]
```

```
 89 [0.25292969 0.12304688 0.52490234 0.09912109]
 90 [0.22558594 0.14208984 0.54101562 0.09130859]
 91 [0.25244141 0.12890625 0.53466797 0.08398438]
 92 [0.22753906 0.14404297 0.53857422 0.08984375]
 93 [0.22607422 0.13378906 0.53857422 0.1015625 ]
 94 [0.23242188 0.13769531 0.54785156 0.08203125]
 95 [0.21972656 0.12988281 0.57324219 0.07714844]
 96 [0.23095703 0.13232422 0.54345703 0.09326172]
 97 [0.22607422 0.14160156 0.55957031 0.07275391]
 98 [0.23876953 0.14599609 0.51904297 0.09619141]
 99 [0.22509766 0.15234375 0.53613281 0.08642578]
100 [0.22607422 0.14208984 0.53466797 0.09716797]
101 [0.22949219 0.14160156 0.53710938 0.09179688]
102 [0.21484375 0.15673828 0.53417969 0.09423828]
103 [0.21484375 0.14355469 0.55517578 0.08642578]
104 [0.22949219 0.14013672 0.53417969 0.09619141]
105 [0.21972656 0.12646484 0.56933594 0.08447266]
106 [0.21142578 0.14306641 0.55664062 0.08886719]
107 [0.21679688 0.13964844 0.55566406 0.08789062]
108 [0.20751953 0.15087891 0.55859375 0.08300781]
109 [0.22851562 0.14697266 0.53173828 0.09277344]
110 [0.22412109 0.13330078 0.55029297 0.09228516]
111 [0.22021484 0.13427734 0.55419922 0.09130859]
112 [0.23535156 0.14208984 0.53808594 0.08447266]
113 [0.234375   0.14404297 0.53857422 0.08300781]
114 [0.24023438 0.12646484 0.54541016 0.08789062]
115 [0.22607422 0.125      0.56738281 0.08154297]
116 [0.22509766 0.13769531 0.55371094 0.08349609]
117 [0.22509766 0.13134766 0.55712891 0.08642578]
118 [0.23681641 0.12841797 0.55322266 0.08154297]
119 [0.23535156 0.12939453 0.55126953 0.08398438]
120 [0.23291016 0.13916016 0.53857422 0.08935547]
121 [0.22509766 0.14160156 0.54492188 0.08837891]
122 [0.22998047 0.14794922 0.52734375 0.09472656]
123 [0.24365234 0.13330078 0.546875   0.07617188]
124 [0.21386719 0.14550781 0.55322266 0.08740234]
125 [0.23339844 0.12792969 0.55175781 0.08691406]
126 [0.23632812 0.12402344 0.54833984 0.09130859]
127 [0.22998047 0.13964844 0.55566406 0.07470703]
128 [0.21582031 0.13232422 0.55224609 0.09960938]
129 [0.23193359 0.13525391 0.54638672 0.08642578]
130 [0.25292969 0.13476562 0.53027344 0.08203125]
131 [0.22705078 0.13330078 0.54394531 0.09570312]
132 [0.2265625  0.13378906 0.55517578 0.08447266]
133 [0.234375   0.13818359 0.54492188 0.08251953]
134 [0.23046875 0.12548828 0.55029297 0.09375   ]
135 [0.23095703 0.15136719 0.53222656 0.08544922]
136 [0.25097656 0.13232422 0.53417969 0.08251953]
137 [0.24316406 0.13232422 0.53613281 0.08837891]
138 [0.22998047 0.13476562 0.54882812 0.08642578]
139 [0.23486328 0.13574219 0.53857422 0.09082031]
140 [0.22363281 0.14208984 0.54150391 0.09277344]
141 [0.22607422 0.12597656 0.56201172 0.0859375 ]
142 [0.22509766 0.13476562 0.54833984 0.09179688]
143 [0.21582031 0.14550781 0.55371094 0.08496094]
144 [0.22412109 0.13085938 0.55175781 0.09326172]
145 [0.23046875 0.14111328 0.54248047 0.0859375 ]
146 [0.22607422 0.14404297 0.54101562 0.08886719]
147 [0.21533203 0.13476562 0.56347656 0.08642578]
148 [0.22851562 0.13623047 0.54589844 0.08935547]
149 [0.22949219 0.14160156 0.54638672 0.08251953]
150 [0.21777344 0.13623047 0.55810547 0.08789062]
151 [0.21972656 0.13769531 0.5625     0.08007812]
152 [0.20605469 0.1484375  0.54443359 0.10107422]
153 [0.22998047 0.13916016 0.54492188 0.0859375 ]
154 [0.22216797 0.15039062 0.53466797 0.09277344]
155 [0.22607422 0.13330078 0.54638672 0.09423828]
156 [0.234375   0.14599609 0.53222656 0.08740234]
157 [0.21923828 0.13916016 0.55175781 0.08984375]
158 [0.21972656 0.1484375  0.546875   0.08496094]
159 [0.24462891 0.14111328 0.53417969 0.08007812]
160 [0.22412109 0.13525391 0.56005859 0.08056641]
161 [0.21337891 0.14941406 0.55566406 0.08154297]
162 [0.22070312 0.13427734 0.55810547 0.08691406]
163 [0.22021484 0.14257812 0.54345703 0.09375   ]
164 [0.22167969 0.13818359 0.55810547 0.08203125]
165 [0.21386719 0.13623047 0.56445312 0.08544922]
166 [0.22509766 0.14599609 0.53320312 0.09570312]
167 [0.23339844 0.140625   0.54003906 0.0859375 ]
168 [0.23730469 0.125      0.55859375 0.07910156]
169 [0.23974609 0.13574219 0.54101562 0.08349609]
170 [0.22558594 0.140625   0.55371094 0.08007812]
171 [0.23046875 0.13525391 0.54931641 0.08496094]
172 [0.23291016 0.140625   0.53564453 0.09082031]
173 [0.22949219 0.13769531 0.54638672 0.08642578]
174 [0.24121094 0.13818359 0.5390625  0.08154297]
175 [0.22363281 0.14257812 0.54248047 0.09130859]
176 [0.21630859 0.14404297 0.55615234 0.08349609]
177 [0.22558594 0.14160156 0.54296875 0.08984375]
```

```
[0.22558594 0.13818359 0.546875   0.08935547]
[0.22558594 0.14160156 0.52685547 0.10595703]
[0.22949219 0.13818359 0.546875   0.08544922]
[0.22900391 0.12841797 0.55224609 0.09033203]
[0.20214844 0.14306641 0.56005859 0.09472656]
[0.22021484 0.13085938 0.56738281 0.08154297]
[0.2421875  0.13232422 0.53710938 0.08837891]
[0.2265625  0.14892578 0.54052734 0.08398438]
[0.22265625 0.13964844 0.54833984 0.08935547]
[0.23046875 0.13964844 0.54150391 0.08837891]
[0.23681641 0.13818359 0.546875   0.078125  ]
[0.22802734 0.12402344 0.56152344 0.08642578]
[0.23095703 0.140625   0.5390625  0.08935547]
[0.23779297 0.14355469 0.54052734 0.078125  ]
[0.21777344 0.14746094 0.53857422 0.09619141]
[0.22314453 0.13671875 0.54541016 0.09472656]
[0.23974609 0.12646484 0.55615234 0.07763672]
[0.23291016 0.14208984 0.54052734 0.08447266]
[0.21289062 0.15039062 0.53857422 0.09814453]
[0.22167969 0.14306641 0.53466797 0.10058594]
[0.23242188 0.13378906 0.55029297 0.08349609]
[0.22460938 0.1328125  0.55566406 0.08691406]
[0.22460938 0.13769531 0.55371094 0.08398438]
[0.234375   0.13476562 0.53662109 0.09423828]
Timer unit: 1e-07 s

Total time: 15.6075 s
File: C:/Users/usuario/qGAN/quantumGAN/performance_testing/performance_get_output_generator.py
Function: get_output_V1 at line 34

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    34                                           def get_output_V1():
    35       101       3246.0     32.1      0.0      for noise in batch_noise:
    36       100       2217.0     22.2      0.0          real_keys = {"00", "10", "01", "11"}
    37
    38       100      93707.0    937.1      0.1          quantum = QuantumRegister(sum(num_qubits), name="q")
    39       100     241357.0   2413.6      0.2          qc = QuantumCircuit(sum(num_qubits))
    40
    41       100     191531.0   1915.3      0.1          init_dist = qiskit.QuantumCircuit(sum(num_qubits))
    42       100       3958.0     39.6      0.0          assert noise.shape[0] == sum(num_qubits)
    43
    44       300       4761.0     15.9      0.0          for num_qubit in range(sum(num_qubits)):
    45       200     230737.0   1153.7      0.1              init_dist.ry(noise[num_qubit], num_qubit)
    46
    47       100       4193.0     41.9      0.0          params = cast(np.ndarray, parameter_values)
    48
    49       100   12016547.0 120165.5      7.7          qc.append(construct_circuit(params), quantum)
    50       100    3831565.0  38315.7      2.5          final_circuit = qc.compose(init_dist, front=True)
    51       100     382781.0   3827.8      0.2          final_circuit.measure_all()
    52
    53       100    4096845.0  40968.4      2.6          simulator_1 = qiskit.Aer.get_backend("aer_simulator")
    54       100  125335069.0 1253350.7     80.3          final_circuit = qiskit.transpile(final_circuit, simulator_1)
    55       100    8681342.0  86813.4      5.6          result = simulator_1.run(final_circuit, shots=shots).result()
    56       100     254267.0   2542.7      0.2          counts = result.get_counts(final_circuit)
    57
    58       100       1594.0     15.9      0.0          try:
    59       100      30815.0    308.1      0.0              pixels = np.array([counts["00"], counts["10"], counts["01"], counts["11"]])
    60
    61                                                   except KeyError:
    62                                                       # dealing with the keys that qiskit doesn't include in the
    63                                                       # dictionary because they don't get any measurements
    64
    65                                                       keys = counts.keys()
    66                                                       missing_keys = real_keys.difference(keys)
    67                                                       # we use sets to get the missing keys
    68                                                       for key_missing in missing_keys:
    69                                                           counts[key_missing] = 0
    70
    71                                                       pixels = np.array([counts["00"], counts["10"], counts["01"], counts["11"]])
    72
    73       100      35345.0    353.4      0.0          pixels = pixels / shots
    74       100     633600.0   6336.0      0.4          print(pixels)

Total time: 12.8011 s
File: C:/Users/usuario/qGAN/quantumGAN/performance_testing/performance_get_output_generator.py
Function: get_output_V2 at line 77

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    77                                           def get_output_V2():
    78         1       2651.0   2651.0      0.0      simulator = qiskit.Aer.get_backend("aer_simulator")
    79       101       2754.0     27.3      0.0      for noise in batch_noise:
    80       100       1890.0     18.9      0.0          real_keys = {"00", "10", "01", "11"}
    81
    82       100      85365.0    853.6      0.1          quantum = QuantumRegister(sum(num_qubits), name="q")
    83       100     226124.0   2261.2      0.2          qc = QuantumCircuit(sum(num_qubits))
    84
    85       100     149032.0   1490.3      0.1         init_dist = qiskit.QuantumCircuit(sum(num_qubits))
    86       100       3195.0     31.9      0.0         assert noise.shape[0] == sum(num_qubits)
    87
    88       300       4841.0     16.1      0.0         for num_qubit in range(sum(num_qubits)):
    89       200     218317.0   1091.6      0.2             init_dist.ry(noise[num_qubit], num_qubit)
    90
    91       100       3702.0     37.0      0.0         params = cast(np.ndarray, parameter_values)
    92
    93       100   11584444.0 115844.4      9.0         qc.append(construct_circuit(params), quantum)
    94       100    3505527.0  35055.3      2.7         final_circuit = qc.compose(init_dist, front=True)
    95       100     330080.0   3300.8      0.3         final_circuit.measure_all()
    96
    97       100  102644991.0 1026449.9     80.2        final_circuit = qiskit.transpile(final_circuit, simulator)
    98       100    8299797.0  82998.0      6.5         result = simulator.run(final_circuit, shots=shots).result()
    99       100     208174.0   2081.7      0.2         counts = result.get_counts(final_circuit)
   100
   101       100       1897.0     19.0      0.0         try:
   102       100      27191.0    271.9      0.0             pixels = np.array([counts["00"], counts["10"], counts["01
"], counts["11"]])
   103
   104                                                  except KeyError:
   105                                                      # dealing with the keys that qiskit doesn't include in the
   106                                                      # dictionary because they don't get any measurements
   107
   108                                                      keys = counts.keys()
   109                                                      missing_keys = real_keys.difference(keys)
   110                                                      # we use sets to get the missing keys
   111                                                      for key_missing in missing_keys:
   112                                                          counts[key_missing] = 0
   113
   114                                                      pixels = np.array([counts["00"], counts["10"], counts["01
"], counts["11"]])
   115
   116       100      37688.0    376.9      0.0         pixels = pixels / shots
   117       100     673317.0   6733.2      0.5         print(pixels)


Process finished with exit code 0

```