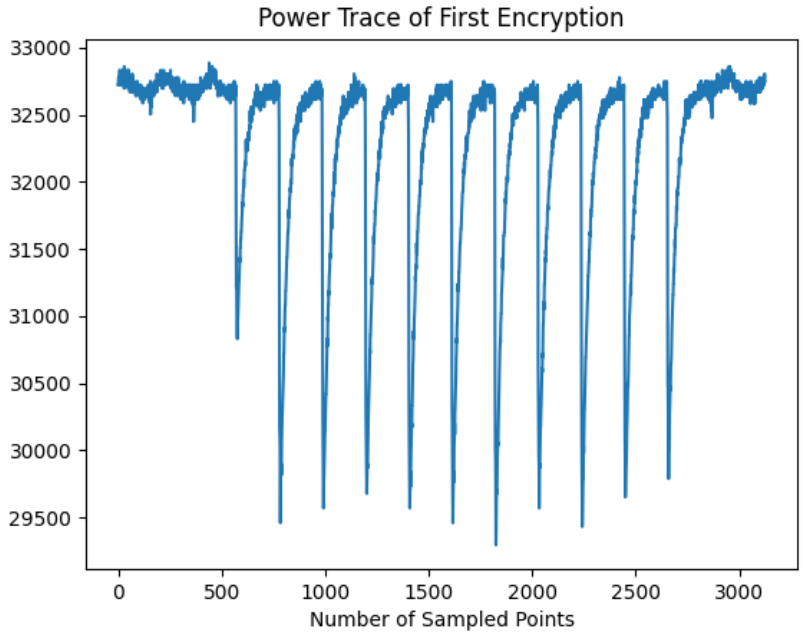
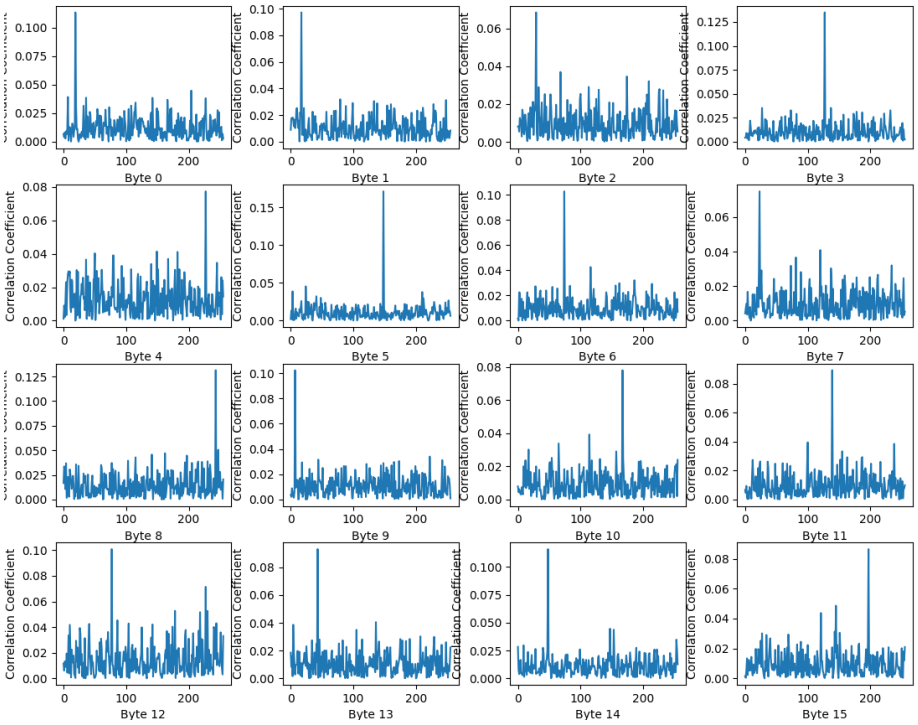
1 Plot the trace and Q1



Question 1: Besides 10 rounds of encryption, there is another "addRoundKey" before the first round, which triggers the updates of register content, leading to a lot of switching activities in the circuit. That is why there is one more dip in the power trace.

2 Plot all key bytes



3 The entire 16-byte last-round key in hex

Byte 0: 0x13
Byte 1: 0x11
Byte 2: 0x1d
Byte 3: 0x7f
Byte 4: 0xe3
Byte 5: 0x94
Byte 6: 0x4a
Byte 7: 0x17
Byte 8: 0xf3
Byte 9: 0x07
Byte 10: 0xa7
Byte 11: 0x8b
Byte 12: 0x4d
Byte 13: 0x2b
Byte 14: 0x30
Byte 15: 0xc5

4 Summary of experience

```
for j in range(16):
    i = indexmap[j]
    for g in range(256):
        HD_tmp = np.zeros(7000)
        for l in range(7000):
            interm = sbox_inv[ciphers[l][j] ^ g]
            HD_tmp[l] = hamming_weight(interm ^ ciphers[l][i])
        corr[j][g], _ = pearsonr(HD_tmp, traces[:, leak_point])
        corr[j][g] = abs(corr[j][g])

key = np.argmax(corr, axis=1)
```

The most important part is the code chunk above. The fundamental theory is that the signals that trigger updates in register content cause power fluctuation due to circuit activities. So if we know the key, we could have the similar power fluctuation at the leak point (circuit activity peak). For each byte of the 16-byte key, we guess 256 possible values. If we guess the correct value, the power would be very similar to the leak point in the power traces. We get the correct value for each byte by trying every possible value and pick the value that is most correlative to the power fluctuation at the leak point. We choose Hamming distance function for the power model, which means we use hamming distance function to indicate the power fluctuation.