In [ ]:
```
#We encode the faulty datasets from decimal to binary to analyse the data
num_pins = 10
def encode_binary(num, n):
    return bin(num)[2:].rjust(n, '0')


f1 = open("partd_dataset_fault1.txt", 'r')
f2 = open("partd_dataset_fault2.txt", 'r')

op1 = open("partd_encoded_fault1.txt", 'w')
op2 = open("partd_encoded_fault2.txt", 'w')

lines1 = f1.readlines()
num_lines1 = len(lines1)
lines2 = f2.readlines()
num_lines2 = len(lines2)

for i in range(num_lines1):
    encoded_num = encode_binary(int(lines1[i]), num_pins)
    op1.write(encoded_num + '\n')

for i in range(num_lines2):
    encoded_num = encode_binary(int(lines2[i]), num_pins)
    op2.write(encoded_num + '\n')
```

In [ ]:
```
#Checking if one of the bits is always stuck at a constant value, indicat
def checkStaticBit(f, num_pins):
    data = f.read().split()
    bits_info = []
    for _ in range(10): #because there are 10 bits, we want to store valu
        bits_info.append([])
    for i in range(len(bits_info)):
        for d in data:
            bits_info[i].append(int(d[i]))


    for i in range(len(bits_info)):
        #print(bits_info[i])
        if all(bits_info[i]):
            print(f'pin {num_pins - i} has a faulty connection. It only s

        elif not(any(bits_info[i])):
            print(f'pin {num_pins - i} has a faulty connection. It only s
```

In [ ]:
```
f1e = open("partd_encoded_fault1.txt", 'r')
checkStaticBit(f1e, 10)
```

```
pin 9 has a faulty connection. It only shows value of 1.
```

Trying something with f2e. trying to see the gray equivalent too see if the problem is because one contact pin touches the next before the other contact pin does, which is the exact problem that graycodes solve.

In [ ]:
```
def bintogray(bin_num): #bin num is a str
    graynum = bin_num[0]
    for i in range(1, len(bin_num)):
        graynum += str(int(bin_num[i-1]) ^ int(bin_num[i]))
```

```
    return graynum


def checkSomething(f, num_pins):
    data = f.read().split()
    graydata = list(map(bintogray, data))
    return graydata
```

In [ ]:
```
f2e = open("partd_encoded_fault2.txt", 'r')
filey = open('partd_fault2_graycode.txt', 'w')

newdata = checkSomething(f2e, 10)
for line in newdata:
    filey.write(line + '\n')

fileyread = open('partd_fault2_graycode.txt', 'r')
checkStaticBit(fileyread, 10)
```

This does not seem to be correct. I dont think it has anything to do with that. We will try another approach. I think the error is due to loose contact. So the bit switched from 1 to 0 or vice versa rather quickly. To find this in the higher order bits, we can assume that the max change from one reading to the next is 100. Now we can search for all cases where this is not the case, and we can check which of the higher order bits has changed. We can flip this bit and draw the graphs again.