

一.需要转换的几种情况

| 不带符号位的整数 | 2进制       | 10进制      | 16进制      |
|----------|-----------|-----------|-----------|
| 2进制      | –         | bin2dec() | bin2hex() |
| 10进制     | dec2bin() | –         | dec2hex() |
| 16进制     | hex2bin() | hex2dec() | –         |

带符号位，即2进制用补码表示。

| 带符号位的整数 | 2进制              | 10进制             | 16进制             |
|---------|------------------|------------------|------------------|
| 2进制     | –                | signed_bin2dec() | signed_bin2hex() |
| 10进制    | signed_dec2bin() | –                | signed_dec2hex() |
| 16进制    | signed_hex2bin() | signed_hex2dec() | –                |

说明：这里忽略了八进制，因为很少用到。

部分函数的实现效果（myBin2dec2hex为自编模块，其中未使用任何第三方库）：

```
1 import myBin2dec2hex
2
3 print(myBin2dec2hex.bin2dec('1010'))
4 print(myBin2dec2hex.hex2bin('0x45'))
5 print(myBin2dec2hex.dec2bin(12))
6 print('-----')
7 print(myBin2dec2hex.signed_bin2dec('1010'))
8 print(myBin2dec2hex.signed_dec2bin(12))
9 print(myBin2dec2hex.signed_dec2bin(-12, 10))
10 print(myBin2dec2hex.signed_bin2hex('101001010', 4))
11 print(myBin2dec2hex.signed_hex2bin('45F1', 18))
```

✓ 0.2s

10  
0b1000101  
0b1100  
-----  
-6  
0b01100  
0b1111110100  
0xff4a  
0b000100010111110001

二. Python 自带的进制转换函数

1.1 bin(整数)

输入整数，可带字符串，可在数字之间加任意下划线。

在数字之间加下划线是Python3的新特性，注意下划线只能加在数字之间，头尾都会报错，且不能出现连续的两个下划线\_，其它进制数也遵循这个规则，这也符合人们日常使用的习惯。

输出二进制字符串，且字符串前带有0b。

注意：1. bin()输入负数无法转换，只会加上一个负号；2. 输入小数会报错

```

File Edit Shell Debug Options Window Help
>>> bin(15)
'0b1111'
>>> bin(0)
'0b0'
>>> bin(-6)
'-0b110'
>>> bin(-_6)
Traceback (most recent call last):
  File "<pyshell#7>", line 1, in <module>
    bin(-_6)
NameError: name '_6' is not defined
>>> bin(+6_6)
'0b100010'
>>> bin(1.2)
Traceback (most recent call last):
  File "<pyshell#9>", line 1, in <module>
    bin(1.2)
TypeError: 'float' object cannot be interpreted as an integer
>>> bin(
(number, /)
Return the binary representation of an integer.
Ln: 19 Col: 8

```

## 1.2 int(2/10/16进制数/字符串, base = 2/10/16)

int有多种用法, 如下所示。

### 1.2.1 int(10进制数/字符串)

默认的base = 10可省略, 此用法与进制转换无关。

功能1: 将10进制整数字符串 (可带正负号, 可在数字之间加任意下划线\_) 转为整数, 小数字符串是不行的。

功能2: 对10进制数 (可带正负号) 取整, 小数部分会被舍弃掉。

```

File Edit Shell Debug Options Window Help
>>> int('10')
10
>>> int('+10')
10
>>> int('-10')
-10
>>> int(10)
10
>>> int(-10)
-10
>>> int(-10.8)
-10
>>> int(10.8)
10
>>> int(+10.8)
10
>>> int('+10.8')
Traceback (most recent call last):
  File "<pyshell#36>", line 1, in <module>
    int('+10.8')
ValueError: invalid literal for int() with base 10: '+10.8'
>>> int(
int([x]) -> integer
int(x, base=10) -> integer
Ln: 22 Col: 8

```

### 1.2.2 int(2进制字符串, base = 2)

发现: Python将0b1100视为一个数, 它和整数12是完全等价的, 和12一样可以带正负号。

当加上base = 2时, 第一个参数必须是字符串。即int(2进制字符串, base = 2)是固定用法。字符串带不带0b效果一样, 同样可带正负号, 可在数字之间加任意下划线\_。

```
My Research Folder
File Edit Shell Debug Options Window Help
>>> int(0b1100)
12
>>> int(-0b1100)
-12
>>> int('0b1100')
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    int('0b1100')
ValueError: invalid literal for int() with base 10: '0b1100'
>>> int('0b1100', base = 2)
12
>>> int('-0b1100', base = 2)
-12
>>> int('1100', base = 2)
12
>>> int('-1100', base = 2)
-12
>>> 0b1100
12
>>> int(0b1100, base = 2)
Traceback (most recent call last):
  File "<pyshell#8>", line 1, in <module>
    int(0b1100, base = 2)
TypeError: int() can't convert non-string with explicit base
>>> |
```

Ln: 25 Col: 4

### 1.2.3 int(16进制字符串, base = 16)

类比1.2.2, 16进制和2进制对于int()的用法是一样的。

Python将0xFF视为一个数, 完全等价于255, 可带正负号, 且字母不区分大小写。

当加上base = 16后, 第一个参数必须是字符串, 所以int(16进制字符串, base = 16)也是固定用法。字符串带不带0x效果一样, 不区分大小写, 也可带正负号, 可在数字之间加任意下划线\_。

```
IDLE Shell 3.9.5
File Edit Shell Debug Options Window Help
>>> int(-0xFF)
-255
>>> int(+0xff)
255
>>> int(0xff, base = 16)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    int(0xff, base = 16)
TypeError: int() can't convert non-string with explicit base
>>> int('-ff', base = 16)
-255
>>> int('+0xFF', base = 16)
255
>>>
```

Ln: 14 Col: 4

### 1.3 hex(整数)

hex()和bin()的使用完全一样, 输入整数, 可带正负号, 可在数字之间加任意下划线\_。输出16进制字符串, 带有0x的前缀。

注意: 1. 不能转换负数, 负号会保留。2. 不能输入小数, 会报错。

```

My Research Folder
File Edit Shell Debug Options Window Help
>>> hex(10)
'0xa'
>>> hex(255)
'0xff'
>>> hex(-12)
'-0xc'
>>> hex(-1_2)
'-0xc'
>>> hex(1.0)
Traceback (most recent call last):
  File "<pyshell#14>", line 1, in <module>
    hex(1.0)
TypeError: 'float' object cannot be interpreted as an integer
>>> hex(0xff)
'0xff'
>>> hex(0b11)
'0x3'
>>> |

```

### 三.不带符号位的自编函数实现进制转换

#### 3.1 bin2dec ()—— 不带符号位的2进制字符串 -> 10进制整数

```

1 def bin2dec(bin_str: str) -> int:
2     '''
3     函数功能: 不带符号位的2进制字符串 -> 10进制整数\n
4     输入: 2进制字符串, 可带正负号, 0b, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
5     输出: 10进制整数, 只保留负号, 正号不保留
6     '''
7     return int(bin_str.strip(), base = 2)

```

#### 3.2 bin2hex() —— 不带符号位的2进制字符串 -> 不带符号位的16进制字符串

```

1 def bin2hex(bin_str: str, hex_width :int = -1) -> str:
2     '''
3     函数功能: 不带符号位的2进制字符串 -> 不带符号位的16进制字符串\n
4     输入参数1: 2进制字符串, 可带正负号, 0b, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
5     输入参数2: 可选, 16进制字符串宽度, 若实际输出宽度>此参数, 警告并原样输出; 若实际输出宽度<=此参数, 高位补若干0\n
6     输出: 16进制字符串, 只保留负号, 正号不保留
7     '''
8     new_bin_str = bin_str.strip()
9     if (new_bin_str[0] == '+' or new_bin_str[0] == '-'): # 去除正负符号
10        new_bin_str = new_bin_str[1:]
11    if (new_bin_str[:2] == '0b'):
12        new_bin_str = new_bin_str[2:]
13    hex_str = hex(int(new_bin_str, base = 2))[2:]
14    if (hex_width == -1):
15        pass
16    elif (hex_width < len(hex_str)): # 位宽小于实际16进制数位数时
17        print('位宽参数' + str(hex_width) + ' < 2进制' + bin_str + '输出16进制' + '0x' + hex_str
18              + '实际位宽' + str(len(hex_str)) + ', 请修正位宽参数')
19    else:
20        hex_str = '0' * (hex_width - len(hex_str)) + hex_str # 扩展位补0
21    if (bin_str[0] == '-'):
22        return '-' + '0x' + hex_str
23    else:
24        return '0x' + hex_str

```

#### 3.3 dec2bin() —— 10进制整数/字符串 -> 不带符号位的2进制字符串

```

1 def dec2bin(dec_num: int, bin_width :int = -1) -> str:
2     '''
3     函数功能: 10进制整数/字符串 -> 不带符号位的2进制字符串\n
4     输入参数1: 10进制整数/字符串, 可带正负号, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
5     输入参数2: 可选, 2进制字符串宽度, 若实际输出宽度>此参数, 警告并原样输出; 若实际输出宽度<=此参数, 高位补若干0\n
6     输出: 16进制字符串, 只保留负号, 正号不保留
7     '''
8     input_dec_num = dec_num
9     if (type(dec_num) == str):
10        dec_num = int(dec_num.strip())
11

```

My Research Folder



```

14     bin_str = old_bin_str[3:]
15     else:
16         bin_str = old_bin_str[2:]
17     if (bin_width == -1):
18         pass
19     elif (bin_width < len(bin_str)):
20         print('位宽参数' + str(bin_width) + ' < 10进制' + str(input_dec_num) + '输出2进制' + old_bin_str
21             + '最小需要位宽' + str(len(bin_str)) + ', 请修正位宽参数')
22     else:
23         bin_str = '0' * (bin_width - len(bin_str)) + bin_str
24     if (old_bin_str[0] == '-'):
25         return '-0b' + bin_str
26     else:
27         return '0b' + bin_str

```

### 3.4 dec2hex() —— 10进制整数/字符串 -> 不带符号位的16进制字符串

```

1  def dec2hex(dec_num: int , hex_width: int = -1) -> str:
2      '''
3      函数功能: 10进制整数/字符串 -> 不带符号位的16进制字符串\n
4      输入参数1: 10进制整数/字符串, 可带正负号, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
5      输入参数2: 可选, 16进制字符串宽度, 若实际输出宽度>此参数, 警告并原样输出; 若实际输出宽度<=此参数, 高位补若干0\n
6      输出: 16进制字符串, 只保留负号, 正号不保留
7      '''
8      old_hex_str = bin2hex(dec2bin(dec_num))
9      if (old_hex_str[0] == '-'):
10         hex_str = old_hex_str[3:]
11     else:
12         hex_str = old_hex_str[2:]
13     if (hex_width == -1):
14         pass
15     elif (hex_width < len(hex_str)):
16         print('位宽参数' + str(hex_width) + ' < 10进制' + str(dec_num) + '输出16进制' + old_hex_str
17             + '实际位宽' + str(len(hex_str)) + ', 请修正位宽参数')
18     else:
19         hex_str = '0' * (hex_width - len(hex_str)) + hex_str
20     if (old_hex_str[0] == '-'):
21         return '-0x' + hex_str
22     else:
23         return '0x' + hex_str

```

### 3.5 hex2dec() —— 不带符号位的16进制字符串 -> 10进制整数

```

1  def hex2dec(hex_str: str) -> int:
2      '''
3      函数功能: 不带符号位的16进制字符串 -> 10进制整数\n
4      输入: 16进制字符串, 可带正负号, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
5      输出: 10进制整数, 只保留负号, 正号不保留
6      '''
7      return int(hex_str.strip(), base = 16)

```

### 3.6 hex2bin() —— 不带符号位的16进制字符串 -> 不带符号位的2进制字符串

```

1  def hex2bin(hex_str: str, bin_width = -1) -> str:
2      '''
3      函数功能: 不带符号位的16进制字符串 -> 不带符号位的2进制字符串\n
4      输入: 16进制字符串, 可带正负号, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
5      输入参数2: 可选, 2进制字符串宽度, 若实际输出宽度>此参数, 警告并原样输出; 若实际输出宽度<=此参数, 高位补若干0\n
6      输出: 2进制字符串, 只保留负号, 正号不保留
7      '''
8      old_bin_str = dec2bin(hex2dec(hex_str))
9      if (old_bin_str[0] == '-'):
10         bin_str = old_bin_str[3:]
11     else:
12         bin_str = old_bin_str[2:]
13     if (bin_width == -1):
14         pass
15     elif (bin_width < len(bin_str)):
16         print('位宽参数' + str(bin_width) + ' < 16进制' + hex_str + '输出2进制' + old_bin_str
17             + '实际位宽' + str(len(bin_str)) + ', 请修正位宽参数')
18     else:
19         bin_str = '0' * (bin_width - len(bin_str)) + bin_str
20     if (old_bin_str[0] == '-'):
21         return '-0b' + bin_str
22     else:
23         return '0b' + bin_str

```

My Research Folder



转换

#### 4.1 signed\_bin2dec ()—— 2进制补码字符串 -> 10进制整数

```

1 def signed_bin2dec(bin_str: str) -> int:
2     '''
3     函数功能: 2进制补码字符串 -> 10进制整数\n
4     输入: 2进制补码字符串, 不可带正负号, 前后可加任意个 \\n 和 空格, 数字间可加下划线\n
5     输出: 10进制整数, 只保留负号, 正号不保留
6     '''
7     bin_str = bin_str.strip()
8     if (bin_str[:2] == '0b'):
9         if (bin_str[2] == '_'):
10             bin_str = bin_str[3:]
11         else:
12             bin_str = bin_str[2:]
13     if (bin_str[0] == '_'):
14         int('输入 ' + bin_str + ' 不合法, 首字符不能是下划线 且 不允许出现连续两个下划线')
15     elif (bin_str[0] == '0'):
16         return int(bin_str, base = 2)
17     elif (bin_str[0] == '1'):
18         a = int(bin_str, base = 2) # 此语句可检查输入是否合法
19         bin_str = bin_str.replace('_', '')
20         return a - 2**len(bin_str)
21     else:
22         int('输入 ' + bin_str + ' 不合法, 必须为2进制补码, 不允许带正负号')

```

#### 4.2 signed\_bin2hex() —— 2进制补码字符串 -> 16进制补码字符串

```

1 def fourBin2OneHex(four_bin: str) -> str:
2     '''
3     函数功能: 4位2进制字符串 -> 1位16进制字符串\n
4     输入: 4位2进制字符串, 输入范围0000~1111\n
5     输出: 1位16进制字符串
6     '''
7     if (four_bin == '0000'):
8         return '0'
9     elif (four_bin == '0001'):
10        return '1'
11    elif (four_bin == '0010'):
12        return '2'
13    elif (four_bin == '0011'):
14        return '3'
15    elif (four_bin == '0100'):
16        return '4'
17    elif (four_bin == '0101'):
18        return '5'
19    elif (four_bin == '0110'):
20        return '6'
21    elif (four_bin == '0111'):
22        return '7'
23    elif (four_bin == '1000'):
24        return '8'
25    elif (four_bin == '1001'):
26        return '9'
27    elif (four_bin == '1010'):
28        return 'a'
29    elif (four_bin == '1011'):
30        return 'b'
31    elif (four_bin == '1100'):
32        return 'c'
33    elif (four_bin == '1101'):
34        return 'd'
35    elif (four_bin == '1110'):
36        return 'e'
37    elif (four_bin == '1111'):
38        return 'f'
39    else:
40        int('输入2进制字符串' + four_bin + '错误, 2进制只能包含0或1')
41
42 def signed_bin2hex(bin_str: str, hex_width: int = -1) -> str:
43     '''
44     函数功能: 2进制补码字符串 -> 16进制补码字符串\n
45     输入参数1: 2进制补码字符串, 不可带正负号, 前后可加任意个 \\n 和 空格, 数字间可加下划线\n
46     输入参数2: 可选, 16进制补码字符串宽度, 若实际输出宽度>此参数, 警告并原样输出; 若实际输出宽度<=此参数, 高位补若干符号位\n
47     输出: 16进制补码字符串
48     '''
49     input_bin_str = bin_str
50

```

My Research Folder



2进制字符串以0b开头

```

53     bin_str = bin_str[2:]
54     elif (bin_str[0] == '0' or bin_str[0] == '1'):
55         pass
56     else:
57         int('输入 ' + bin_str + ' 不合法, 输入必须为2进制补码, 不允许带正负号 且 首字符不能是下划线')
58     # 检查输入是否合法, 末尾字符不能是下划线 且 不能出现连续的两个下划线
59     if (bin_str[-1] == '_' or '___' in bin_str):
60         int('输入 ' + bin_str + ' 不合法, 末尾字符不能是下划线 且 不能出现连续的两个下划线')
61     else:
62         bin_str = bin_str.replace('_', '') # 输入合法则去除下划线
63     # 去掉2进制补码字符串前面多余的符号位, 保留两位
64     for i in range(len(bin_str)-1):
65         if (bin_str[i+1] == bin_str[0]):
66             if (i + 1 == len(bin_str)-1):
67                 bin_str = bin_str[i:]
68             else:
69                 continue
70         else:
71             bin_str = bin_str[i:]
72             break
73     if (len(bin_str) % 4 > 0): # 补符号位到位宽为4的倍数
74         bin_str = bin_str[0] * (4 - len(bin_str) % 4) + bin_str
75     hex_str = ''
76     for i in range(int(len(bin_str)/4)):
77         hex_str += fourBin2OneHex(bin_str[i*4 : i*4+4])
78     if (hex_width == -1):
79         pass
80     elif (hex_width < len(hex_str)):
81         print('位宽参数' + str(hex_width) + ' < 2进制补码' + input_bin_str + '输出16进制补码'
82               + '0x' + hex_str + '实际位宽' + str(len(hex_str)) + ', 请修正位宽参数')
83     else:
84         if (hex_str[0] in ['0', '1', '2', '3', '4', '5', '6', '7']):
85             hex_str = '0' * (hex_width - len(hex_str)) + hex_str
86         else:
87             hex_str = 'f' * (hex_width - len(hex_str)) + hex_str
88     return '0x' + hex_str

```

#### 4.3 signed\_dec2bin() —— 10进制数/字符串 -> 2进制补码字符串

```

1  def signed_dec2bin(dec_num: int, bin_width: int = -1) -> str:
2      '''
3      函数功能: 10进制数/字符串 -> 2进制补码字符串\n
4      输入参数1: 10进制数/字符串, 可带正负号, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
5      输入参数2: 可选, 2进制补码字符串宽度, 若实际输出宽度>此参数, 警告并原样输出; 若实际输出宽度<=此参数, 高位补若干符号位\n
6      输出: 2进制补码字符串
7      '''
8      dec_num_str = str(dec_num)
9      if (type(dec_num) == str):
10         dec_num = int(dec_num.strip())
11     if (dec_num == 0):
12         bin_str = '0'
13     elif (dec_num > 0):
14         bin_str = '0' + bin(dec_num)[2:] # 补符号位0
15     else:
16         for i in range(10000):
17             if (2**i + dec_num >= 0):
18                 bin_str = bin(2**(i+1) + dec_num)[2:] # 一个负数num的补码等于 (2**(i+1) + dec_num)
19                 break
20     if (bin_width == -1):
21         pass
22     elif (bin_width < len(bin_str)):
23         # 实际位宽大于设定位宽则报警告, 然后按实际位宽输出
24         print('位宽参数' + str(bin_width) + ' < 10进制' + dec_num_str + '输出2进制补码'
25               + '0b' + bin_str + '实际位宽' + str(len(bin_str)) + ', 请修正位宽参数')
26     else:
27         bin_str = bin_str[0] * (bin_width - len(bin_str)) + bin_str # 实际位宽小于设定位宽则补符号位
28     return '0b' + bin_str

```

#### 4.4 signed\_dec2hex() —— 10进制数/字符串 -> 16进制补码字符串

```

1  def signed_dec2hex(dec_num: int, hex_width = -1) -> str:
2      '''
3      函数功能: 10进制数/字符串 -> 16进制补码字符串\n
4      输入参数1: 10进制数/字符串, 可带正负号, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
5      输入参数2: 可选, 16进制补码字符串宽度, 若实际输出宽度>此参数, 警告并原样输出; 若实际输出宽度<=此参数, 高位补若干符号位\n
6      输出: 16进制补码字符串
7      '''

```

My Research Folder



gned\_dec2bin(dec\_num))[2:]

```

10     if (hex_width == -1):
11         pass
12     elif (hex_width < len(hex_str)):
13         print('位宽参数' + str(hex_width) + ' < 10进制' + str(dec_num) + '输出16进制补码' + '0x' +
14               hex_str + '实际位宽' + str(len(hex_str)) + ', 请修正位宽参数')
15     else:
16         if (hex_str[0] in ['0', '1', '2', '3', '4', '5', '6', '7']):
17             hex_str = '0' * (hex_width - len(hex_str)) + hex_str
18         else:
19             hex_str = 'f' * (hex_width - len(hex_str)) + hex_str
20     return '0x' + hex_str

```

#### 4.5 signed\_hex2bin() —— 16进制补码字符串 -> 2进制补码字符串

```

1  def oneHex2fourBin(one_hex: str) -> str:
2      '''
3      函数功能: 1位16进制字符串 -> 4位2进制字符串\n
4      输入: 1位16进制字符串, 输入范围0~9, a~f或A~F\n
5      输出: 4位2进制字符串
6      '''
7      if (one_hex == '0'):
8          return '0000'
9      elif (one_hex == '1'):
10         return '0001'
11     elif (one_hex == '2'):
12         return '0010'
13     elif (one_hex == '3'):
14         return '0011'
15     elif (one_hex == '4'):
16         return '0100'
17     elif (one_hex == '5'):
18         return '0101'
19     elif (one_hex == '6'):
20         return '0110'
21     elif (one_hex == '7'):
22         return '0111'
23     elif (one_hex == '8'):
24         return '1000'
25     elif (one_hex == '9'):
26         return '1001'
27     elif (one_hex == 'a' or one_hex == 'A'):
28         return '1010'
29     elif (one_hex == 'b' or one_hex == 'B'):
30         return '1011'
31     elif (one_hex == 'c' or one_hex == 'C'):
32         return '1100'
33     elif (one_hex == 'd' or one_hex == 'D'):
34         return '1101'
35     elif (one_hex == 'e' or one_hex == 'E'):
36         return '1110'
37     elif (one_hex == 'f' or one_hex == 'F'):
38         return '1111'
39     else:
40         int('输入16进制字符串' + one_hex + '错误, 16进制只能包含0~9, a~f或A~F')
41
42  def signed_hex2bin(hex_str: str, bin_width: int = -1) -> str:
43      '''
44      函数功能: 16进制补码字符串 -> 2进制补码字符串\n
45      输入参数1: 16进制补码字符串, 不可带正负号, 前后可加任意个 \n 和 空格, 数字间可加下划线\n
46      输入参数2: 可选, 2进制补码字符串宽度, 若实际输出宽度>此参数, 警告并原样输出; 若实际输出宽度<=此参数, 高位补若干符号位\n
47      输出: 2进制补码字符串
48      '''
49      input_hex_str = hex_str
50      hex_str = hex_str.strip()
51      # 检查输入是否合法, 不允许带正负号, 首尾不能是下划线, 不能出现连续两个下划线
52      if (hex_str[0] in ['+', '-', '_'] or hex_str[-1] == '_' or '_' in hex_str):
53         int('输入' + input_hex_str + '不合法, 必须为16进制补码, 不允许带正负号, '
54             + '首尾不能是下划线, 不能出现连续两个下划线')
55     elif (hex_str[:2] == '0x'):
56         hex_str = hex_str[2:]
57     hex_str = hex_str.replace('_', '') # 输入合法则去除下划线
58     bin_str = ''
59     for i in hex_str:
60         bin_str += oneHex2fourBin(i)
61     # 去掉2进制补码字符串前面多余的符号位, 保留两位
62     for i in range(len(bin_str)-1):
63         if (bin_str[i+1] == bin_str[0]):

```



My Research Folder

Q

n\_str)-1):  
tr[i:]

```
66         else:  
67             continue  
68     else:  
69         bin_str = bin_str[i:]  
70         break  
71     if (bin_str == '00'):  
72         bin_str = '0'  
73     if (bin_width == -1):  
74         pass  
75     elif (bin_width < len(bin_str)):  
76         # 实际位宽大于设定位宽则报警告, 然后按实际位宽输出  
77         print('位宽参数' + str(bin_width) + ' < 16进制补码' + input_hex_str + '输出2进制补码'  
78             + '0b' + bin_str + '实际位宽' + str(len(bin_str)) + ', 请修正位宽参数')  
79     else:  
80         bin_str = bin_str[0] * (bin_width - len(bin_str)) + bin_str # 实际位宽小于设定位宽则补符号位  
81     return '0b' + bin_str
```

#### 4.6 signed\_hex2dec() —— 16进制补码字符串 -> 10进制整数

```
1 def signed_hex2dec(hex_str: str) -> int:  
2     '''  
3     函数功能: 16进制补码字符串 -> 10进制整数\n  
4     输入: 16进制补码字符串, 不可带正负号, 前后可加任意个 \\n 和 空格, 数字间可加下划线\n  
5     输出: 10进制整数, 只保留负号, 正号不保留  
6     '''  
7     return signed_bin2dec(signed_hex2bin(hex_str))
```

#### 四. import自编模块

见我的另一篇博客, [Python如何导入自己编写的py文件](#)

#### 五. 进制转换模块及其测试jupyter notebook文档分享

我将上述所有转换函数放入bin2dec2hex.py文件中, 还有测试用的文档, 一起放入了我的码云开源仓库中, 需要的可以去以下链接自取:

<https://gitee.com/xuxiaokang/python-self-compiled-module.git>



徐晓康的博客

专注于智能硬件、FPGA、嵌入式知识工具分享

 微信公众号 >