Calculation Variable Prinhead Resolution

Possible Resolutions

\(if(pixel_{offset} * \frac{dpi_{img}}{dpi_{ph}} = integer)\)

Possible in FPGA if \(integer = multiple_of(2)\)

Other Restrictions are (not considerated in the calculation)

- ErgoSoft Rip can only create images with a DPI of max \(2400dpi\)
- The Speed of the machine is limited. with \(360dpi\) default speed if \(840\frac{mm}{s}\). At \(180dpi\) default speed should be \(1680\frac{mm}{s}\), but in the Siemens PLC only 3 digits are used to set the print speed. Higher speeds are also not tested on the Digiround.

In [100]:

```
import math
def calc_possible_dpi(interline_gaps, dpi_ph, ph_type):
  possible_dpi = []   # dpi
  for dpi_img in range(1,3000): # dpi
    dpi_possible   = True
    for interlinegap in interlinegaps:
      mult_int    = int(interlinegap*(dpi_img/dpi_ph))
      mult_double = interlinegap*(dpi_img/dpi_ph)
      if (mult_double - mult_int) > 0:
        dpi_possible = False
    if dpi_possible:
      possible_dpi.append(dpi_img)

  mod_2 = []
  divide_possible = []
  decimation_value = []
  decimation_possible = []
  for dpi_img in possible_dpi:
    # check if value is mod 2
    if fmod((dpi_img/dpi_ph),(9.765625e-4)) == 0:
      mod_2.append(True)

    else:
      mod_2.append(False)

    # check if interlinegaps can be divided by the value
    divide_possible_temp = True
    for interlinegap in interlinegaps:
      if not fmod(interlinegap*(dpi_img/dpi_ph),1) == 0:
        divide_possible_temp = False
    divide_possible.append(divide_possible_temp)

    # Calc Decimation value
    fire_decimation_default = 32
    decimation_value.append((float(fire_decimation_default) * float(dpi_ph)/float(dpi_img)) - 1)
    decimation_possible_temp = False
    if decimation_value[-1] == int(decimation_value[-1]):
      decimation_possible_temp = True
    decimation_possible.append(decimation_possible_temp)



  # Print results
  print("Possible Resolutions of {}".format(ph_type))
  print("         |                |     For Jetmapping                         |      For Interpolator
")
  print("dpi_img  | dpi_img/dpi_ph | Possible Division | Possible Calculation | Decimation Register Value | Poss
ible")
  print("         |                | for FPGA          | for PH               |                           |
")
  print("---------+----------------+-------------------+----------------------+---------------------------+-----
----")
  for i in range(size(possible_dpi)):
    dpi_img = possible_dpi[i]
    if mod_2[i] and divide_possible[i] and decimation_possible[i]:
      print("--> {:4} | {:14} | {}              | {}                 | {:7.4}                   | {}  <--".forma
t(dpi_img,dpi_img/dpi_ph,mod_2[i], divide_possible[i], decimation_value[i], decimation_possible[i]))
```

```
    elif mod_2[i]:
      print("    {:4} | {:14} | {}               | {}                   | {:7.4}                     | {}".format(dpi
_img,dpi_img/dpi_ph,mod_2[i], divide_possible[i], decimation_value[i], decimation_possible[i]))
    else:
      print("    {:4} | {:14} | {}              | {}                   | {:7.4}                     | {}".format(dpi_
img,dpi_img/dpi_ph,mod_2[i], divide_possible[i], decimation_value[i], decimation_possible[i]))
```

KM1024i

In [101]:

```
interlinegaps = [12,28,40] # px
dpi_ph = 360.0             # dpi
calc_possible_dpi(interlinegaps, dpi_ph, "Konica Minolta KM1024i")
```

```
Possible Resolutions of Konica Minolta KM1024i
         |                |     For Jetmapping                       |     For Interpolator          |
dpi_img  | dpi_img/dpi_ph | Possible Division | Possible Calculation | Decimation Register Value | Possible
         |                | for FPGA          | for PH               |                           |
---------+----------------+-------------------+----------------------+---------------------------+---------
-->   90 |           0.25 | True              | True                 |   127.0                   | True  <--
-->  180 |            0.5 | True              | True                 |    63.0                   | True  <--
     270 |           0.75 | True              | True                 |   41.67                   | False
-->  360 |            1.0 | True              | True                 |    31.0                   | True  <--
     450 |           1.25 | True              | True                 |    24.6                   | False
     540 |            1.5 | True              | True                 |   20.33                   | False
     630 |           1.75 | True              | True                 |   17.29                   | False
-->  720 |            2.0 | True              | True                 |    15.0                   | True  <--
     810 |           2.25 | True              | True                 |   13.22                   | False
     900 |            2.5 | True              | True                 |    11.8                   | False
     990 |           2.75 | True              | True                 |   10.64                   | False
    1080 |            3.0 | True              | True                 |   9.667                   | False
    1170 |           3.25 | True              | True                 |   8.846                   | False
    1260 |            3.5 | True              | True                 |   8.143                   | False
    1350 |           3.75 | True              | True                 |   7.533                   | False
--> 1440 |            4.0 | True              | True                 |     7.0                   | True  <--
    1530 |           4.25 | True              | True                 |   6.529                   | False
    1620 |            4.5 | True              | True                 |   6.111                   | False
    1710 |           4.75 | True              | True                 |   5.737                   | False
    1800 |            5.0 | True              | True                 |     5.4                   | False
    1890 |           5.25 | True              | True                 |   5.095                   | False
    1980 |            5.5 | True              | True                 |   4.818                   | False
    2070 |           5.75 | True              | True                 |   4.565                   | False
    2160 |            6.0 | True              | True                 |   4.333                   | False
    2250 |           6.25 | True              | True                 |    4.12                   | False
    2340 |            6.5 | True              | True                 |   3.923                   | False
    2430 |           6.75 | True              | True                 |   3.741                   | False
    2520 |            7.0 | True              | True                 |   3.571                   | False
    2610 |           7.25 | True              | True                 |   3.414                   | False
    2700 |            7.5 | True              | True                 |   3.267                   | False
    2790 |           7.75 | True              | True                 |   3.129                   | False
--> 2880 |            8.0 | True              | True                 |     3.0                   | True  <--
    2970 |           8.25 | True              | True                 |   2.879                   | False
```

KM1024

In [102]:

```
interlinegaps = [20] # px
dpi_ph = 360.0       # dpi
calc_possible_dpi(interlinegaps, dpi_ph, "Konica Minolta KM1024")
```

```
Possible Resolutions of Konica Minolta KM1024
         |                |     For Jetmapping                       |     For Interpolator          |
dpi_img  | dpi_img/dpi_ph | Possible Division | Possible Calculation | Decimation Register Value | Possible
         |                | for FPGA          | for PH               |                           |
---------+----------------+-------------------+----------------------+---------------------------+---------
      18 |           0.05 | False             | True                 |   639.0                   | True
      36 |            0.1 | False             | True                 |   319.0                   | True
      54 |           0.15 | False             | True                 |   212.3                   | False
      72 |            0.2 | False             | True                 |   159.0                   | True
-->   90 |           0.25 | True              | True                 |   127.0                   | True  <--
     108 |            0.3 | False             | True                 |   105.7                   | False
     126 |           0.35 | False             | True                 |   90.43                   | False
     144 |            0.4 | False             | True                 |    79.0                   | True
     162 |           0.45 | False             | True                 |   70.11                   | False
-->  180 |            0.5 | True              | True                 |    63.0                   | True  <--
     198 |           0.55 | False             | True                 |   57.18                   | False
     216 |           0.6 | False            | True                |  52.33                  | False
     234 |          0.65 | False            | True                |  48.23                  | False
     252 |           0.7 | False            | True                |  44.71                  | False
     270 |          0.75 | True             | True                |  41.67                  | False
     288 |           0.8 | False            | True                |   39.0                  | True
     306 |          0.85 | False            | True                |  36.65                  | False
     324 |           0.9 | False            | True                |  34.56                  | False
     342 |          0.95 | False            | True                |  32.68                  | False
-->  360 |           1.0 | True             | True                |   31.0                  | True  <--
     378 |          1.05 | False            | True                |  29.48                  | False
     396 |           1.1 | False            | True                |  28.09                  | False
     414 |          1.15 | False            | True                |  26.83                  | False
     432 |           1.2 | False            | True                |  25.67                  | False
     450 |          1.25 | True             | True                |   24.6                  | False
     468 |           1.3 | False            | True                |  23.62                  | False
     486 |          1.35 | False            | True                |   22.7                  | False
     504 |           1.4 | False            | True                |  21.86                  | False
     522 |          1.45 | False            | True                |  21.07                  | False
     540 |           1.5 | True             | True                |  20.33                  | False
     558 |          1.55 | False            | True                |  19.65                  | False
     576 |           1.6 | False            | True                |   19.0                  | True
     594 |          1.65 | False            | True                |  18.39                  | False
     612 |           1.7 | False            | True                |  17.82                  | False
     630 |          1.75 | True             | True                |  17.29                  | False
     648 |           1.8 | False            | True                |  16.78                  | False
     666 |          1.85 | False            | True                |   16.3                  | False
     684 |           1.9 | False            | True                |  15.84                  | False
     702 |          1.95 | False            | True                |  15.41                  | False
-->  720 |           2.0 | True             | True                |   15.0                  | True  <--
     738 |          2.05 | False            | True                |  14.61                  | False
     756 |           2.1 | False            | True                |  14.24                  | False
     774 |          2.15 | False            | True                |  13.88                  | False
     792 |           2.2 | False            | True                |  13.55                  | False
     810 |          2.25 | True             | True                |  13.22                  | False
     828 |           2.3 | False            | True                |  12.91                  | False
     846 |          2.35 | False            | True                |  12.62                  | False
     864 |           2.4 | False            | True                |  12.33                  | False
     882 |          2.45 | False            | True                |  12.06                  | False
     900 |           2.5 | True             | True                |   11.8                  | False
     918 |          2.55 | False            | True                |  11.55                  | False
     936 |           2.6 | False            | True                |  11.31                  | False
     954 |          2.65 | False            | True                |  11.08                  | False
     972 |           2.7 | False            | True                |  10.85                  | False
     990 |          2.75 | True             | True                |  10.64                  | False
    1008 |           2.8 | False            | True                |  10.43                  | False
    1026 |          2.85 | False            | True                |  10.23                  | False
    1044 |           2.9 | False            | True                |  10.03                  | False
    1062 |          2.95 | False            | True                |  9.847                  | False
    1080 |           3.0 | True             | True                |  9.667                  | False
    1098 |          3.05 | False            | True                |  9.492                  | False
    1116 |           3.1 | False            | True                |  9.323                  | False
    1134 |          3.15 | False            | True                |  9.159                  | False
    1152 |           3.2 | False            | True                |    9.0                  | True
    1170 |          3.25 | True             | True                |  8.846                  | False
    1188 |           3.3 | False            | True                |  8.697                  | False
    1206 |          3.35 | False            | True                |  8.552                  | False
    1224 |           3.4 | False            | True                |  8.412                  | False
    1242 |          3.45 | False            | True                |  8.275                  | False
    1260 |           3.5 | True             | True                |  8.143                  | False
    1278 |          3.55 | False            | True                |  8.014                  | False
    1296 |           3.6 | False            | True                |  7.889                  | False
    1314 |          3.65 | False            | True                |  7.767                  | False
    1332 |           3.7 | False            | True                |  7.649                  | False
    1350 |          3.75 | True             | True                |  7.533                  | False
    1368 |           3.8 | False            | True                |  7.421                  | False
    1386 |          3.85 | False            | True                |  7.312                  | False
    1404 |           3.9 | False            | True                |  7.205                  | False
    1422 |          3.95 | False            | True                |  7.101                  | False
--> 1440 |           4.0 | True             | True                |    7.0                  | True  <--
    1458 |          4.05 | False            | True                |  6.901                  | False
    1476 |           4.1 | False            | True                |  6.805                  | False
    1494 |          4.15 | False            | True                |  6.711                  | False
    1512 |           4.2 | False            | True                |  6.619                  | False
    1530 |          4.25 | True             | True                |  6.529                  | False
    1548 |           4.3 | False            | True                |  6.442                  | False
    1566 |          4.35 | False            | True                |  6.356                  | False
    1584 |           4.4 | False            | True                |  6.273                  | False
    1602 |          4.45 | False            | True                |  6.191                  | False
    1620 |           4.5 | True             | True                |  6.111                  | False
    1638 |          4.55 | False            | True                |  6.033                  | False
    1656 |             4.6 | False            | True                  |   5.957                     | False
    1674 |            4.65 | False            | True                  |   5.882                     | False
    1692 |             4.7 | False            | True                  |   5.809                     | False
    1710 |            4.75 | True             | True                  |   5.737                     | False
    1728 |             4.8 | False            | True                  |   5.667                     | False
    1746 |            4.85 | False            | True                  |   5.598                     | False
    1764 |             4.9 | False            | True                  |   5.531                     | False
    1782 |            4.95 | False            | True                  |   5.465                     | False
    1800 |             5.0 | True             | True                  |     5.4                     | False
    1818 |            5.05 | False            | True                  |   5.337                     | False
    1836 |             5.1 | False            | True                  |   5.275                     | False
    1854 |            5.15 | False            | True                  |   5.214                     | False
    1872 |             5.2 | False            | True                  |   5.154                     | False
    1890 |            5.25 | True             | True                  |   5.095                     | False
    1908 |             5.3 | False            | True                  |   5.038                     | False
    1926 |            5.35 | False            | True                  |   4.981                     | False
    1944 |             5.4 | False            | True                  |   4.926                     | False
    1962 |            5.45 | False            | True                  |   4.872                     | False
    1980 |             5.5 | True             | True                  |   4.818                     | False
    1998 |            5.55 | False            | True                  |   4.766                     | False
    2016 |             5.6 | False            | True                  |   4.714                     | False
    2034 |            5.65 | False            | True                  |   4.664                     | False
    2052 |             5.7 | False            | True                  |   4.614                     | False
    2070 |            5.75 | True             | True                  |   4.565                     | False
    2088 |             5.8 | False            | True                  |   4.517                     | False
    2106 |            5.85 | False            | True                  |    4.47                     | False
    2124 |             5.9 | False            | True                  |   4.424                     | False
    2142 |            5.95 | False            | True                  |   4.378                     | False
    2160 |             6.0 | True             | True                  |   4.333                     | False
    2178 |            6.05 | False            | True                  |   4.289                     | False
    2196 |             6.1 | False            | True                  |   4.246                     | False
    2214 |            6.15 | False            | True                  |   4.203                     | False
    2232 |             6.2 | False            | True                  |   4.161                     | False
    2250 |            6.25 | True             | True                  |    4.12                     | False
    2268 |             6.3 | False            | True                  |   4.079                     | False
    2286 |            6.35 | False            | True                  |   4.039                     | False
    2304 |             6.4 | False            | True                  |     4.0                     | True
    2322 |            6.45 | False            | True                  |   3.961                     | False
    2340 |             6.5 | True             | True                  |   3.923                     | False
    2358 |            6.55 | False            | True                  |   3.885                     | False
    2376 |             6.6 | False            | True                  |   3.848                     | False
    2394 |            6.65 | False            | True                  |   3.812                     | False
    2412 |             6.7 | False            | True                  |   3.776                     | False
    2430 |            6.75 | True             | True                  |   3.741                     | False
    2448 |             6.8 | False            | True                  |   3.706                     | False
    2466 |            6.85 | False            | True                  |   3.672                     | False
    2484 |             6.9 | False            | True                  |   3.638                     | False
    2502 |            6.95 | False            | True                  |   3.604                     | False
    2520 |             7.0 | True             | True                  |   3.571                     | False
    2538 |            7.05 | False            | True                  |   3.539                     | False
    2556 |             7.1 | False            | True                  |   3.507                     | False
    2574 |            7.15 | False            | True                  |   3.476                     | False
    2592 |             7.2 | False            | True                  |   3.444                     | False
    2610 |            7.25 | True             | True                  |   3.414                     | False
    2628 |             7.3 | False            | True                  |   3.384                     | False
    2646 |            7.35 | False            | True                  |   3.354                     | False
    2664 |             7.4 | False            | True                  |   3.324                     | False
    2682 |            7.45 | False            | True                  |   3.295                     | False
    2700 |             7.5 | True             | True                  |   3.267                     | False
    2718 |            7.55 | False            | True                  |   3.238                     | False
    2736 |             7.6 | False            | True                  |   3.211                     | False
    2754 |            7.65 | False            | True                  |   3.183                     | False
    2772 |             7.7 | False            | True                  |   3.156                     | False
    2790 |            7.75 | True             | True                  |   3.129                     | False
    2808 |             7.8 | False            | True                  |   3.103                     | False
    2826 |            7.85 | False            | True                  |   3.076                     | False
    2844 |             7.9 | False            | True                  |   3.051                     | False
    2862 |            7.95 | False            | True                  |   3.025                     | False
--> 2880 |             8.0 | True             | True                  |     3.0                     | True  <--
    2898 |            8.05 | False            | True                  |   2.975                     | False
    2916 |             8.1 | False            | True                  |   2.951                     | False
    2934 |            8.15 | False            | True                  |   2.926                     | False
    2952 |             8.2 | False            | True                  |   2.902                     | False
    2970 |            8.25 | True             | True                  |   2.879                     | False
    2988 |             8.3 | False            | True                  |   2.855                     | False
```

Ricoh Gen5

In [103]:

```
interlinegaps = [13,279,292] # px
dpi_ph = 600.0               # dpi
calc_possible_dpi(interlinegaps, dpi_ph, "Ricoh GEN5")
```

```
Possible Resolutions of Ricoh GEN5
         |                |     For Jetmapping                       |      For Interpolator
dpi_img  | dpi_img/dpi_ph | Possible Division | Possible Calculation | Decimation Register Value | Possible
         |                | for FPGA          | for PH               |                           |
---------+----------------+-------------------+----------------------+---------------------------+---------
-->  600 |            1.0 | True              | True                 |    31.0                   | True  <--
--> 1200 |            2.0 | True              | True                 |    15.0                   | True  <--
    1800 |            3.0 | True              | True                 |   9.667                   | False
--> 2400 |            4.0 | True              | True                 |     7.0                   | True  <--
```

Kyocera KJ4B

In [104]:

```
interlinegaps = [0,20,70,80,90,100,150,160,170,180,220,230,240,250,260,300,310,320,330,380,390,400,410,460,480] # px
dpi_ph = 600.0               # dpi
calc_possible_dpi(interlinegaps, dpi_ph, "Kyocera KJ4B")
```

```
Possible Resolutions of Kyocera KJ4B
         |                |     For Jetmapping                       |      For Interpolator
dpi_img  | dpi_img/dpi_ph | Possible Division | Possible Calculation | Decimation Register Value | Possible
         |                | for FPGA          | for PH               |                           |
---------+----------------+-------------------+----------------------+---------------------------+---------
      60 |            0.1 | False             | True                 |   319.0                   | True
     120 |            0.2 | False             | True                 |   159.0                   | True
     180 |            0.3 | False             | True                 |   105.7                   | False
     240 |            0.4 | False             | True                 |    79.0                   | True
-->  300 |            0.5 | True              | True                 |    63.0                   | True  <--
     360 |            0.6 | False             | True                 |   52.33                   | False
     480 |            0.8 | False             | True                 |    39.0                   | True
     540 |            0.9 | False             | True                 |   34.56                   | False
-->  600 |            1.0 | True              | True                 |    31.0                   | True  <--
     720 |            1.2 | False             | True                 |   25.67                   | False
     780 |            1.3 | False             | True                 |   23.62                   | False
     900 |            1.5 | True              | True                 |   20.33                   | False
     960 |            1.6 | False             | True                 |    19.0                   | True
    1020 |            1.7 | False             | True                 |   17.82                   | False
    1080 |            1.8 | False             | True                 |   16.78                   | False
    1140 |            1.9 | False             | True                 |   15.84                   | False
--> 1200 |            2.0 | True              | True                 |    15.0                   | True  <--
    1260 |            2.1 | False             | True                 |   14.24                   | False
    1440 |            2.4 | False             | True                 |   12.33                   | False
    1500 |            2.5 | True              | True                 |    11.8                   | False
    1560 |            2.6 | False             | True                 |   11.31                   | False
    1740 |            2.9 | False             | True                 |   10.03                   | False
    1800 |            3.0 | True              | True                 |   9.667                   | False
    1860 |            3.1 | False             | True                 |   9.323                   | False
    1920 |            3.2 | False             | True                 |     9.0                   | True
    1980 |            3.3 | False             | True                 |   8.697                   | False
    2040 |            3.4 | False             | True                 |   8.412                   | False
    2100 |            3.5 | True              | True                 |   8.143                   | False
    2160 |            3.6 | False             | True                 |   7.889                   | False
    2220 |            3.7 | False             | True                 |   7.649                   | False
    2280 |            3.8 | False             | True                 |   7.421                   | False
    2340 |            3.9 | False             | True                 |   7.205                   | False
--> 2400 |            4.0 | True              | True                 |     7.0                   | True  <--
    2520 |            4.2 | False             | True                 |   6.619                   | False
    2580 |            4.3 | False             | True                 |   6.442                   | False
    2700 |            4.5 | True              | True                 |   6.111                   | False
    2820 |            4.7 | False             | True                 |   5.809                   | False
    2880 |            4.8 | False             | True                 |   5.667                   | False
```

Register Calculation image_to_printhead_resolution

image_to_printhead_resolution content

```
[31:4] = integer part
[3]    = 1/2 part
[2]    = 1/4 part
[1]    = 1/8 part
```

```
    [0]    = 1/16 part
```

In [105]:

```
def calc_floatparts(val):
  # get integer part
  int_val = int(val)
  # get 1/16 val
  temp_val = val-int_val
  if( mod(temp_val,0.125) == 0.0625 ):
    sixteenth_val = 1
    temp_val = temp_val - mod(temp_val,0.125)
  else:
    sixteenth_val = 0
  # get 1/8 val
  if( mod(temp_val,0.25) == 0.125 ):
    eigth_val = 1
    temp_val = temp_val - mod(temp_val,0.25)
  else:
    eigth_val = 0
  # get 1/4 val
  if( mod(temp_val,0.5) == 0.25 ):
    quater_val = 1
    temp_val = temp_val - mod(temp_val,0.5)
  else:
    quater_val = 0
  # get 1/2 val
  if( mod(temp_val,1) == 0.5 ):
    half_val = 1
    temp_val = temp_val - mod(temp_val,1)
  else:
    half_val = 0
  # Check if we got all
  if temp_val == 0:
    print("Calulation correct")
  else:
    print("Calulation wrong")

  # Concat for getting hex value
  hex_val = int_val*16 + half_val*8+ quater_val*4 + eigth_val*2 + sixteenth_val

  print("value       = {}".format(val))
  print("hex value   = 0x{:08X}".format(hex_val))
  print("integerpart = {}".format(int_val))
  print("1/2 part    = {}".format(half_val))
  print("1/4 part    = {}".format(quater_val))
  print("1/8 part    = {}".format(eigth_val))
  print("1/16 part   = {}".format(sixteenth_val))
  print("")

calc_floatparts(1.0)
calc_floatparts(3.0 + 0.5 + 0.25 + 0.125 + 0.0625 + 0.03125)
calc_floatparts(8.0 + 0.5 + 0.25 + 0.125 + 0.0625)
calc_floatparts(63.0 + 0.5 + 0.25 + 0.125 + 0.0625) # max value
```

```
Calulation correct
value       = 1.0
hex value   = 0x00000010
integerpart = 1
1/2 part    = 0
1/4 part    = 0
1/8 part    = 0
1/16 part   = 0

Calulation wrong
value       = 3.96875
hex value   = 0x00000030
integerpart = 3
1/2 part    = 0
1/4 part    = 0
1/8 part    = 0
1/16 part   = 0

Calulation correct
value       = 8.9375
hex value   = 0x0000008F
integerpart = 8
1/2 part    = 1
```

```
1/4 part    = 1
1/8 part    = 1
1/16 part   = 1

Calulation correct
value       = 63.9375
hex value   = 0x000003FF
integerpart = 63
1/2 part    = 1
1/4 part    = 1
1/8 part    = 1
1/16 part   = 1
```

Pixel Pos Values

In [106]:

```
# vhdl function ported to python
def unsigned_num_bits(num):
  _nbits = 1
  _n = num
  while(_n > 1):
    _nbits = _nbits + 1
    _n     = _n / 2
  return _nbits

def calcPosXVal(maxVal_x, maxMult, xpos_BitNb):
  for i in range(int(round(maxMult))):
    val = maxVal_x * i

    print("MaxVal: {:4}  Multiplication: {:2} Result: {:5} NumberofBits(needed/available): ({:2}/{:2})".format(maxVal_x, i, val, unsigned_num_bits(val), xpos_BitNb))
```

KM1024i

In [107]:

```
calcPosXVal(12, 63.9375, 8)
```

```
MaxVal:   12  Multiplication:  0 Result:     0 NumberofBits(needed/available): ( 1/ 8)
MaxVal:   12  Multiplication:  1 Result:    12 NumberofBits(needed/available): ( 4/ 8)
MaxVal:   12  Multiplication:  2 Result:    24 NumberofBits(needed/available): ( 5/ 8)
MaxVal:   12  Multiplication:  3 Result:    36 NumberofBits(needed/available): ( 6/ 8)
MaxVal:   12  Multiplication:  4 Result:    48 NumberofBits(needed/available): ( 6/ 8)
MaxVal:   12  Multiplication:  5 Result:    60 NumberofBits(needed/available): ( 6/ 8)
MaxVal:   12  Multiplication:  6 Result:    72 NumberofBits(needed/available): ( 7/ 8)
MaxVal:   12  Multiplication:  7 Result:    84 NumberofBits(needed/available): ( 7/ 8)
MaxVal:   12  Multiplication:  8 Result:    96 NumberofBits(needed/available): ( 7/ 8)
MaxVal:   12  Multiplication:  9 Result:   108 NumberofBits(needed/available): ( 7/ 8)
MaxVal:   12  Multiplication: 10 Result:   120 NumberofBits(needed/available): ( 7/ 8)
MaxVal:   12  Multiplication: 11 Result:   132 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 12 Result:   144 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 13 Result:   156 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 14 Result:   168 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 15 Result:   180 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 16 Result:   192 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 17 Result:   204 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 18 Result:   216 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 19 Result:   228 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 20 Result:   240 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 21 Result:   252 NumberofBits(needed/available): ( 8/ 8)
MaxVal:   12  Multiplication: 22 Result:   264 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 23 Result:   276 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 24 Result:   288 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 25 Result:   300 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 26 Result:   312 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 27 Result:   324 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 28 Result:   336 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 29 Result:   348 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 30 Result:   360 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 31 Result:   372 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 32 Result:   384 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 33 Result:   396 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 34 Result:   408 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 35 Result:   420 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 36 Result:   432 NumberofBits(needed/available): ( 9/ 8)
```

```
MaxVal:   12  Multiplication: 37 Result:   444 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 38 Result:   456 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 39 Result:   468 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 40 Result:   480 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 41 Result:   492 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 42 Result:   504 NumberofBits(needed/available): ( 9/ 8)
MaxVal:   12  Multiplication: 43 Result:   516 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 44 Result:   528 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 45 Result:   540 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 46 Result:   552 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 47 Result:   564 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 48 Result:   576 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 49 Result:   588 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 50 Result:   600 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 51 Result:   612 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 52 Result:   624 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 53 Result:   636 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 54 Result:   648 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 55 Result:   660 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 56 Result:   672 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 57 Result:   684 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 58 Result:   696 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 59 Result:   708 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 60 Result:   720 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 61 Result:   732 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 62 Result:   744 NumberofBits(needed/available): (10/ 8)
MaxVal:   12  Multiplication: 63 Result:   756 NumberofBits(needed/available): (10/ 8)
```

KY KJ4B 40KHz

In [108]:

```
calcPosXVal(480, 63.9375, 12)
```

```
MaxVal:  480  Multiplication:  0 Result:     0 NumberofBits(needed/available): ( 1/12)
MaxVal:  480  Multiplication:  1 Result:   480 NumberofBits(needed/available): ( 9/12)
MaxVal:  480  Multiplication:  2 Result:   960 NumberofBits(needed/available): (10/12)
MaxVal:  480  Multiplication:  3 Result:  1440 NumberofBits(needed/available): (11/12)
MaxVal:  480  Multiplication:  4 Result:  1920 NumberofBits(needed/available): (11/12)
MaxVal:  480  Multiplication:  5 Result:  2400 NumberofBits(needed/available): (12/12)
MaxVal:  480  Multiplication:  6 Result:  2880 NumberofBits(needed/available): (12/12)
MaxVal:  480  Multiplication:  7 Result:  3360 NumberofBits(needed/available): (12/12)
MaxVal:  480  Multiplication:  8 Result:  3840 NumberofBits(needed/available): (12/12)
MaxVal:  480  Multiplication:  9 Result:  4320 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 10 Result:  4800 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 11 Result:  5280 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 12 Result:  5760 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 13 Result:  6240 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 14 Result:  6720 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 15 Result:  7200 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 16 Result:  7680 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 17 Result:  8160 NumberofBits(needed/available): (13/12)
MaxVal:  480  Multiplication: 18 Result:  8640 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 19 Result:  9120 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 20 Result:  9600 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 21 Result: 10080 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 22 Result: 10560 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 23 Result: 11040 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 24 Result: 11520 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 25 Result: 12000 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 26 Result: 12480 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 27 Result: 12960 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 28 Result: 13440 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 29 Result: 13920 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 30 Result: 14400 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 31 Result: 14880 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 32 Result: 15360 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 33 Result: 15840 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 34 Result: 16320 NumberofBits(needed/available): (14/12)
MaxVal:  480  Multiplication: 35 Result: 16800 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 36 Result: 17280 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 37 Result: 17760 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 38 Result: 18240 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 39 Result: 18720 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 40 Result: 19200 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 41 Result: 19680 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 42 Result: 20160 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 43 Result: 20640 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 44 Result: 21120 NumberofBits(needed/available): (15/12)
```

```
MaxVal:  480  Multiplication: 45 Result: 21600 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 46 Result: 22080 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 47 Result: 22560 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 48 Result: 23040 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 49 Result: 23520 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 50 Result: 24000 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 51 Result: 24480 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 52 Result: 24960 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 53 Result: 25440 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 54 Result: 25920 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 55 Result: 26400 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 56 Result: 26880 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 57 Result: 27360 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 58 Result: 27840 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 59 Result: 28320 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 60 Result: 28800 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 61 Result: 29280 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 62 Result: 29760 NumberofBits(needed/available): (15/12)
MaxVal:  480  Multiplication: 63 Result: 30240 NumberofBits(needed/available): (15/12)
```

Ricoh GEN5

In [109]:

```
calcPosXVal(292, 63.9375, 10)
```

```
MaxVal:  292  Multiplication:  0 Result:     0 NumberofBits(needed/available): ( 1/10)
MaxVal:  292  Multiplication:  1 Result:   292 NumberofBits(needed/available): ( 9/10)
MaxVal:  292  Multiplication:  2 Result:   584 NumberofBits(needed/available): (10/10)
MaxVal:  292  Multiplication:  3 Result:   876 NumberofBits(needed/available): (10/10)
MaxVal:  292  Multiplication:  4 Result:  1168 NumberofBits(needed/available): (11/10)
MaxVal:  292  Multiplication:  5 Result:  1460 NumberofBits(needed/available): (11/10)
MaxVal:  292  Multiplication:  6 Result:  1752 NumberofBits(needed/available): (11/10)
MaxVal:  292  Multiplication:  7 Result:  2044 NumberofBits(needed/available): (11/10)
MaxVal:  292  Multiplication:  8 Result:  2336 NumberofBits(needed/available): (12/10)
MaxVal:  292  Multiplication:  9 Result:  2628 NumberofBits(needed/available): (12/10)
MaxVal:  292  Multiplication: 10 Result:  2920 NumberofBits(needed/available): (12/10)
MaxVal:  292  Multiplication: 11 Result:  3212 NumberofBits(needed/available): (12/10)
MaxVal:  292  Multiplication: 12 Result:  3504 NumberofBits(needed/available): (12/10)
MaxVal:  292  Multiplication: 13 Result:  3796 NumberofBits(needed/available): (12/10)
MaxVal:  292  Multiplication: 14 Result:  4088 NumberofBits(needed/available): (12/10)
MaxVal:  292  Multiplication: 15 Result:  4380 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 16 Result:  4672 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 17 Result:  4964 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 18 Result:  5256 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 19 Result:  5548 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 20 Result:  5840 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 21 Result:  6132 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 22 Result:  6424 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 23 Result:  6716 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 24 Result:  7008 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 25 Result:  7300 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 26 Result:  7592 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 27 Result:  7884 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 28 Result:  8176 NumberofBits(needed/available): (13/10)
MaxVal:  292  Multiplication: 29 Result:  8468 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 30 Result:  8760 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 31 Result:  9052 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 32 Result:  9344 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 33 Result:  9636 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 34 Result:  9928 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 35 Result: 10220 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 36 Result: 10512 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 37 Result: 10804 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 38 Result: 11096 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 39 Result: 11388 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 40 Result: 11680 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 41 Result: 11972 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 42 Result: 12264 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 43 Result: 12556 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 44 Result: 12848 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 45 Result: 13140 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 46 Result: 13432 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 47 Result: 13724 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 48 Result: 14016 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 49 Result: 14308 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 50 Result: 14600 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 51 Result: 14892 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 52 Result: 15184 NumberofBits(needed/available): (14/10)
```

```
MaxVal:  292  Multiplication: 53 Result: 15476 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 54 Result: 15768 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 55 Result: 16060 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 56 Result: 16352 NumberofBits(needed/available): (14/10)
MaxVal:  292  Multiplication: 57 Result: 16644 NumberofBits(needed/available): (15/10)
MaxVal:  292  Multiplication: 58 Result: 16936 NumberofBits(needed/available): (15/10)
MaxVal:  292  Multiplication: 59 Result: 17228 NumberofBits(needed/available): (15/10)
MaxVal:  292  Multiplication: 60 Result: 17520 NumberofBits(needed/available): (15/10)
MaxVal:  292  Multiplication: 61 Result: 17812 NumberofBits(needed/available): (15/10)
MaxVal:  292  Multiplication: 62 Result: 18104 NumberofBits(needed/available): (15/10)
MaxVal:  292  Multiplication: 63 Result: 18396 NumberofBits(needed/available): (15/10)
```