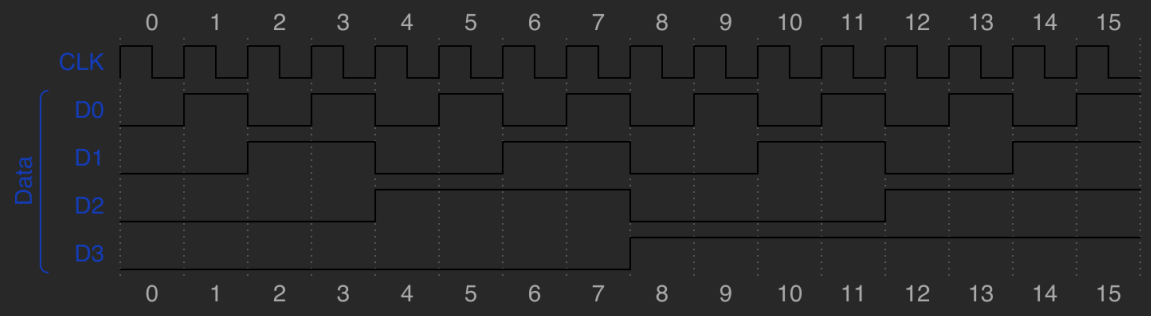
Kamarina Hewett

HW#2 : code

```
##1) A script that converts a binary number to a decimal number.
def bintodec(n):
   result = int(n, 2)

   return result

#############################################################################

##2) A script that converts a decimal number to a binary number.
def dectobin(n):
   n1 = bin(n)
   if n >= 15:
      n3 = n1.replace('b','').replace('0','',1)
   else:
      n3 = n1.replace('b', '')

   return n3

#############################################################################

##3) A script that converts a hexadecimal number to a decimal number.
def hextodec(n):
   Ref = {'0': 0, '1': 1, '2': 2, '3': 3, '4': 4, '5': 5, '6': 6, '7': 7, '8':
8, '9': 9,
          'A':10, 'B':11, 'C':12, 'D':13, 'E':14, 'F':15}
   n = n.upper()
   d = 0
   s = len(n)-1
   for num in n:
       d = d+Ref[num]*16**s
       s = s-1

   return d

#############################################################################

##4) A script that converts a decimal number to a hexadecimal number.
def dectohex(n):
   n = int(n)
   n1= hex(n)

   return n1

#############################################################################
```

```
##5) And successfully run the WaveDROM script (in Python3) on Moodle.
# import wavedrom
# signals = """
# {
#     "signal":
#         [
#             { "name": "CLK", "wave": "p..............."},
#             [ "Data",
#                 { "name": "D0", "wave": "lhlhlhlhlhlhlhlh"},
#                 { "name": "D1", "wave": "l.h.l.h.l.h.l.h."},
#                 { "name": "D2", "wave": "l...h...l...h..."},
#                 { "name": "D3", "wave": "l.......h......."},
#             ]
#         ],
#     "head":
#         {
#             "tock":0
#         },
#     "foot":
#         {
#             "tock":0
#         }
# }"""
# svg = wavedrom.render(signals)
# svg.saveas("timingExample6.svg")'
```

(output of wavedrom below)




```
#########################################################################
#########################################################################
```

```
#2s complement
def complements(n):
    onescomp = ''
    for i in n:
        if i == '0':
            onescomp += '1'
            continue
        elif i == '1':
            onescomp += '0'
            continue
    c = 1
    x = int(onescomp,2)
    h = x+1
    binh = bin(h)
    if h >= 6:
        twoscomp = binh.replace('b','').replace('0','',1)
    else:
        twoscomp = binh.replace('b', '')

    return (f'1s complement = {onescomp}, '
            f'2s complement = {twoscomp}')
```

Hw#3 : answers

```
#6 (a-h):
bintodec('1110')
    Out[3]: 14
bintodec('1010')
    Out[4]: 10
bintodec('11100')
    Out[5]: 28
bintodec('10000')
    Out[6]: 16
bintodec('10101')
    Out[7]: 21
bintodec('11101')
    Out[8]: 29
bintodec('10111')
    Out[9]: 23
bintodec('11111')
    Out[10]: 31
-----------------------------------------------------------------------
#13 (a-h):
dectobin(15)
    Out[3]: '1111'
dectobin(21)
    Out[4]: '10101'
```

```
dectobin(28)
   Out[5]: '11100'
dectobin(34)
   Out[6]: '100010'
dectobin(40)
   Out[7]: '101000'
dectobin(59)
   Out[8]: '111011'
dectobin(65)
   Out[9]: '1000001'
dectobin(73)
   Out[10]: '1001001'
----------------------------------------------------------------------
#22 (a-h)
complements('10')
   Out[3]: '1s complement = 01, 2s complement = 010'
complements('111')
   Out[4]: '1s complement = 000, 2s complement = 01'
complements('1001')
   Out[5]: '1s complement = 0110, 2s complement = 111'
complements('1101')
   Out[6]: '1s complement = 0010, 2s complement = 011'
complements('11100')
   Out[7]: '1s complement = 00011, 2s complement = 0100'
complements('10011')
   Out[8]: '1s complement = 01100, 2s complement = 1101'
complements('10110000')
   Out[9]: '1s complement = 01001111, 2s complement = 1010000'
complements('00111101')
   Out[10]: '1s complement = 11000010, 2s complement = 11000011'
----------------------------------------------------------------------
#39 (a-h)
hextodec('23')
   Out[4]: 35
hextodec('92')
   Out[5]: 146
hextodec('1a')
   Out[6]: 26
hextodec('8d')
   Out[7]: 141
hextodec('f3')
   Out[8]: 243
hextodec('eb')
   Out[9]: 235
hextodec('5c2')
   Out[10]: 1474
hextodec('700')
   Out[11]: 1792
----------------------------------------------------------------------
```

```
#40 (a-h)
dectohex('8')
   Out[3]: '0x8'
dectohex('14')
   Out[4]: '0xE'
dectohex('33')
   Out[5]: '0x21'
dectohex('52')
   Out[6]: '0x34'
dectohex('284')
   Out[7]: '0x11C'
dectohex('2890')
   Out[8]: '0xB4A'
dectohex('4019')
   Out[9]: '0xFB3'
dectohex('6500')
   Out[10]: '0x1964'
```