```c
/*
 Explore the operation of the OR, AND, Exclusive OR and NOT bitwise operations (|, & ^ and ~)
*/

#include "Bitwise_logic_header.h"

int main (void){

char comp;
char digits[8];
unsigned char lfsr;
char BWop;                                      //bit wise operation and complement (i.e. swap ones and zeros)
char PRN_counter = 0;

setup_HW;
for(int m = 0; m <= 7; m++)digits[m] = 0;

if(watch_dog_reset==0){String_to_PC_Basic("\r\n    Examining the operation of the\r\n\
    OR, XOR, NAND functions and their complements. (AK to continue)\r\n");
waitforkeypress_Basic();

    String_to_PC_Basic("\r\n   Select OP ('x' to change it):\
\r\n  |   for   OR\
\r\n  ^   for   XOR\
\r\n  &   for   AND\
\r\n  ~|    for   NOR\
\r\n  ~^    for   NXOR\
\r\n  ~&    for   NAND\
\r\n\r\n");}
BWop = waitforkeypress_Basic();
if (BWop == '~')
{Char_to_PC_Basic('N');comp = 1; BWop = waitforkeypress_Basic();}else comp = 0;      //detect complement operator
if ((BWop != '|') && (BWop != '^') && (BWop != '&'))
SW_reset;                                                                           //reset if duff char was sent

switch(BWop){
  case '|': String_to_PC_Basic("OR"); break;
  case '^': String_to_PC_Basic("XOR"); break;
  case '&': String_to_PC_Basic("AND"); break;}

do{
digits[0] = PRN_8bit_GEN();
digits[1] = PRN_8bit_GEN();                                                         //Second random number
digits[2] =  Op(digits[0] , digits[1], comp, BWop);                                 //Process the numbers
lfsr = digits[1];
I2C_Tx_BWops(digits);
}
while (waitforkeypress_Basic() !='x');                                              //Press 'x' to escape
{String_to_PC_Basic("\tAnother OP\r\n"); SW_reset;}}


/*****************************************************************************/
unsigned char Op(unsigned char A, unsigned char B, char comp, char BWOp)
{char result=0;
switch (BWOp){
case '|': result = A | B; break;
case '^': result = A ^ B; break;
case '&': result = A & B; break;}
if (comp == 1) result = ~result;
return result;}

/*****************************************************************/
```