Table of Contents

[Home](#) * [Engines](#) * [CPW-Engine](#) * **Attacks**

This page holds the functions responsible for detecting if a given square is attacked by a given player.

isAttacked

```
#include "stdafx.h"
#include "0x88_math.h"

int isAttacked( char byColor, S8 sq ) {
/* pawns */
if ( byColor == WHITE ) {
   if ( IS_SQ( sq + SE ) &&
        isPiece( WHITE, PAWN, sq + SE )
      )
      return 1;
   if ( IS_SQ( sq + SW ) &&
        isPiece( WHITE, PAWN, sq + SW )
      )
      return 1;
}
else {
   if ( IS_SQ( sq + NE ) &&
        isPiece( BLACK, PAWN, sq + NE )
      )
      return 1;
   if ( IS_SQ( sq + NW ) &&
        isPiece( BLACK, PAWN, sq + NW )
      )
      return 1;
}
```

```
/* knights */
if ( leaperAttack( byColor, sq, KNIGHT ) )
   return 1;

 /* kings */
if ( leaperAttack( byColor, sq, KING ) )
   return 1;

 /* straight line sliders */
    if ( straightAttack( byColor, sq, NORTH ) ||
         straightAttack( byColor, sq, SOUTH ) ||
         straightAttack( byColor, sq, EAST  ) ||
         straightAttack( byColor, sq, WEST  )
       )
       return 1;

 /* diagonal sliders */
    if ( diagAttack( byColor, sq, NE ) ||
         diagAttack( byColor, sq, SE ) ||
         diagAttack( byColor, sq, NW ) ||
         diagAttack( byColor, sq, SW )
       )
       return 1;

    return 0;
}
```

leaperAttack

```
int leaperAttack( char byColor, S8 sq, char byPiece ) {
    S8 nextSq;
    for ( int dir = 0; dir < 8; dir++ ) {
        nextSq = sq + vector[byPiece][dir];
        if ( IS_SQ(nextSq) &&
             isPiece( byColor, byPiece, nextSq )
           )
           return 1;
    }
    return 0;
}
```

straightAttack

```
int straightAttack(char byColor, S8 sq, int vect) {
    int nextSq = sq + vect;

    while ( IS_SQ(nextSq) ) {
       if (b.color[nextSq] != COLOR_EMPTY ) {
           if ( ( b.color[nextSq] == byColor ) &&
                ( b.pieces[nextSq] == ROOK || b.pieces[nextSq] ==
 QUEEN )
              )
              return 1;
           return 0;
          }
       nextSq = nextSq + vect;
    }
     return 0;
}
```

diagAttack

```
int diagAttack(int byColor, S8 sq, int vect) {
    int nextSq = sq + vect;

    while ( IS_SQ( nextSq ) ) {
       if (b.color[ nextSq ] != COLOR_EMPTY ) {
           if ( ( b.color[nextSq] == byColor ) &&
                ( b.pieces[nextSq] == BISHOP || b.pieces[nextSq] ==
 QUEEN )
              )
              return 1;
           return 0;
          }
       nextSq = nextSq + vect;
    }
    return 0;
}
```

Up one Level