```python
def print_board(title, board):
    print(f"{title}:")
    for row in board:
        print(' '.join(row))
    print()

def winner(b):
    lines = b + [list(c) for c in zip(*b)] + [[b[i][i] for i in range(3)], [b[i][2-i] for i in range(3)]]
    for line in lines:
        if line == ['x']*3: return 'x'
        if line == ['o']*3: return 'o'
    return None

def is_full(b):
    return all(cell != '.' for row in b for cell in row)

def minimax(b, is_max):
    w = winner(b)
    if w == 'x': return 1
    if w == 'o': return -1
    if is_full(b): return 0

    best = -2 if is_max else 2
    for i in range(3):
        for j in range(3):
            if b[i][j] == '.':
                b[i][j] = 'x' if is_max else 'o'
                score = minimax(b, not is_max)
                b[i][j] = '.'
                best = max(best, score) if is_max else min(best, score)
    return best

def best_move(b):
    move, best = None, -2
    for i in range(3):
        for j in range(3):
            if b[i][j] == '.':
                b[i][j] = 'x'
                score = minimax(b, False)
                b[i][j] = '.'
                if score > best:
                    best = score
                    move = (i, j)
    return move

board = [['x', 'o', 'x'],
         ['o', 'x', '.'],
         ['.', 'o', 'x']]

print_board("Current Board", board)
move = best_move(board)
print("Best Move:", move, "\n")
board[move[0]][move[1]] = 'x'
print_board("Board after best move", board)
```

Output

```
Current Board:
x o x
o x .
. o x

Best Move: (1, 2)

Board after best move:
x o x
o x x
. o x


=== Code Execution Successful ===
```