```python
def fast_copy(source, count):
    destination = [None] * count
    i = 0
    n = count // 8
    remainder = count % 8

    if remainder >= 7:
        destination[i] = source[i]
        i += 1
    if remainder >= 6:
        destination[i] = source[i]
        i += 1
    if remainder >= 5:
        destination[i] = source[i]
        i += 1
    if remainder >= 4:
        destination[i] = source[i]
        i += 1
    if remainder >= 3:
        destination[i] = source[i]
        i += 1
    if remainder >= 2:
        destination[i] = source[i]
        i += 1
    if remainder >= 1:
        destination[i] = source[i]
        i += 1

    while n > 0:
        destination[i] = source[i]
        destination[i+1] = source[i+1]
        destination[i+2] = source[i+2]
        destination[i+3] = source[i+3]
        destination[i+4] = source[i+4]
        destination[i+5] = source[i+5]
        destination[i+6] = source[i+6]
        destination[i+7] = source[i+7]

        i += 8
        n -= 1

    return destination

source = [1,2,3,4,5,6,7,8,9,10]
result = fast_copy(source, len(source))
print(result)
```

Explanation:

Example:

```
for(i=0;i<count;i++)

dest[i]=src[i]
```

Each iteration checks a condition and costs time.

To overcome this loop unrolling optimization technique is used to reduce the overhead of loop control.

The fast_copy() copies elements from a source list to destination list using an optimized approach inspired by Duff’s device.

It reduces loop overhead by copying data in blocks of 8 elements and handling leftover elements separately