

A C code:

```
for (int i=0; i<n; i++) {  
    sum += a[i] * b[i];  
} return sum;
```

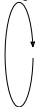
B C code with intrinsics:

```
for (int i=0; i<n; i+=svcntd()){  
    pg = svwhilelt_b64(i, n);  
    sva = svld1_f64(pg, &a[i]);  
    svb = svld1_f64(pg, &b[i]);  
    svsum = svla_f64(pg, sva, svb);  
}  
return svaddv_f64(svptrue_b64(), svsum);
```

C ARM SVE no intrinsics:

$x0 \leftarrow a[i]$ $x8 \leftarrow n$
 $x1 \leftarrow b[i]$ $d0 \leftarrow \text{sum}$

```
mov      x8, xzr  
whilelo  p0.d, xzr, x9  
fmov     d0, xzr  
loop:    ld1d    { z1.d }, p0/z, [x0, x8, lsl #3]  
         ld1d    { z2.d }, p0/z, [x1, x8, lsl #3]  
         incd    x8  
         fmul    z1.d, z1.d, z2.d  
         fadda   d0, p0, d0, z1.d  
         whilelo p0.d, x8, x9  
         b.first .loop
```



D ARM SVE with intrinsics:

$x0 \leftarrow a[i]$ $x8 \leftarrow n$
 $x1 \leftarrow b[i]$ $d0 \leftarrow \text{sum}$

```
mov      z0.d, #0  
loop:    whilelt  p0.d, w8, w2  
         sxtw     x10, w8  
         ld1d    { z1.d }, p0/z, [x0, x10, lsl #3]  
         ld1d    { z2.d }, p0/z, [x1, x10, lsl #3]  
         add     w8, w8, w9  
         cmp     w8, w2  
         fmla    z0.d, p0/m, z1.d, z2.d  
         b.lt    .loop  
         ptrue   p0.d  
         faddv   d0, p0, z0.d  
         ret
```

