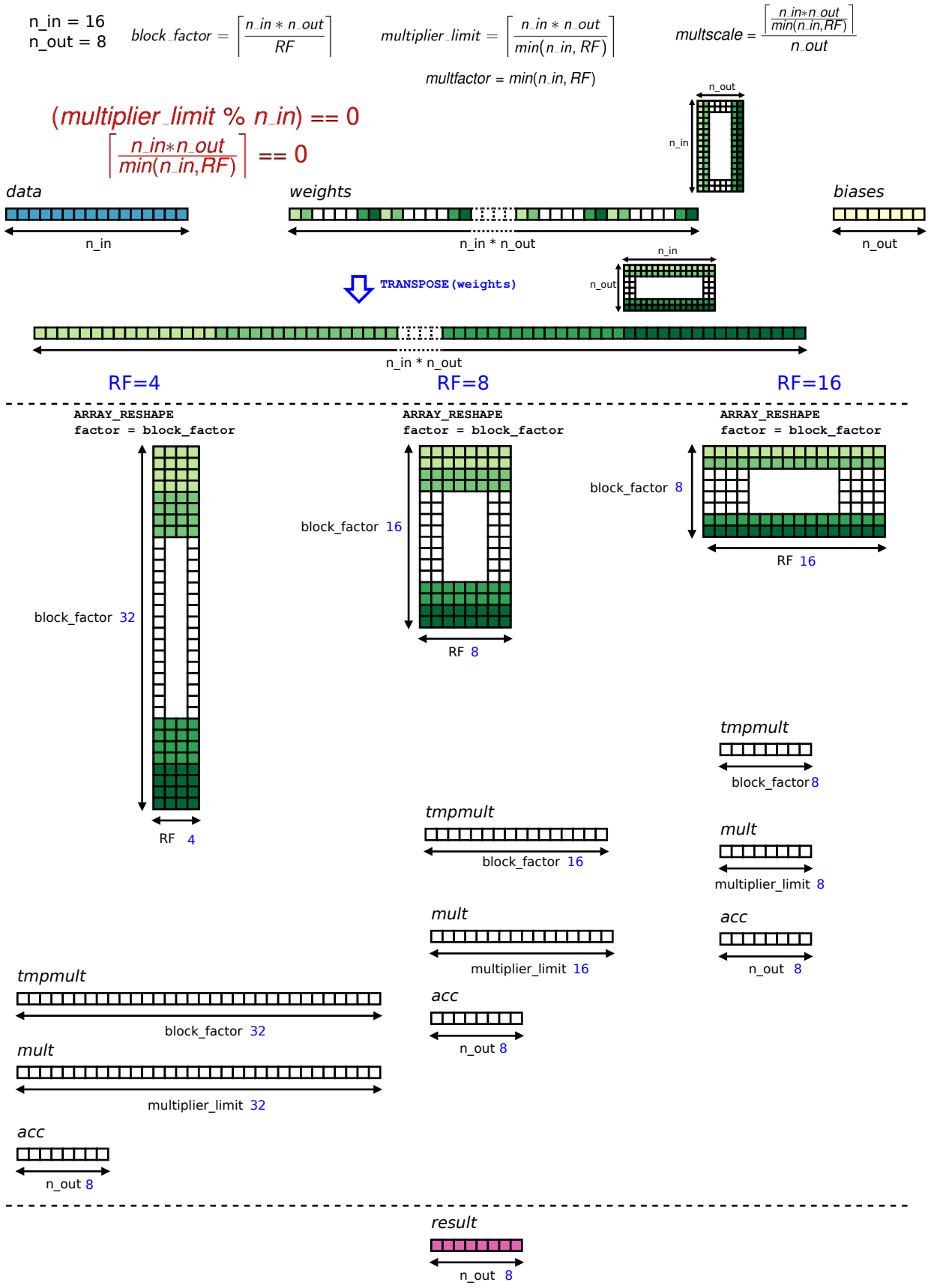
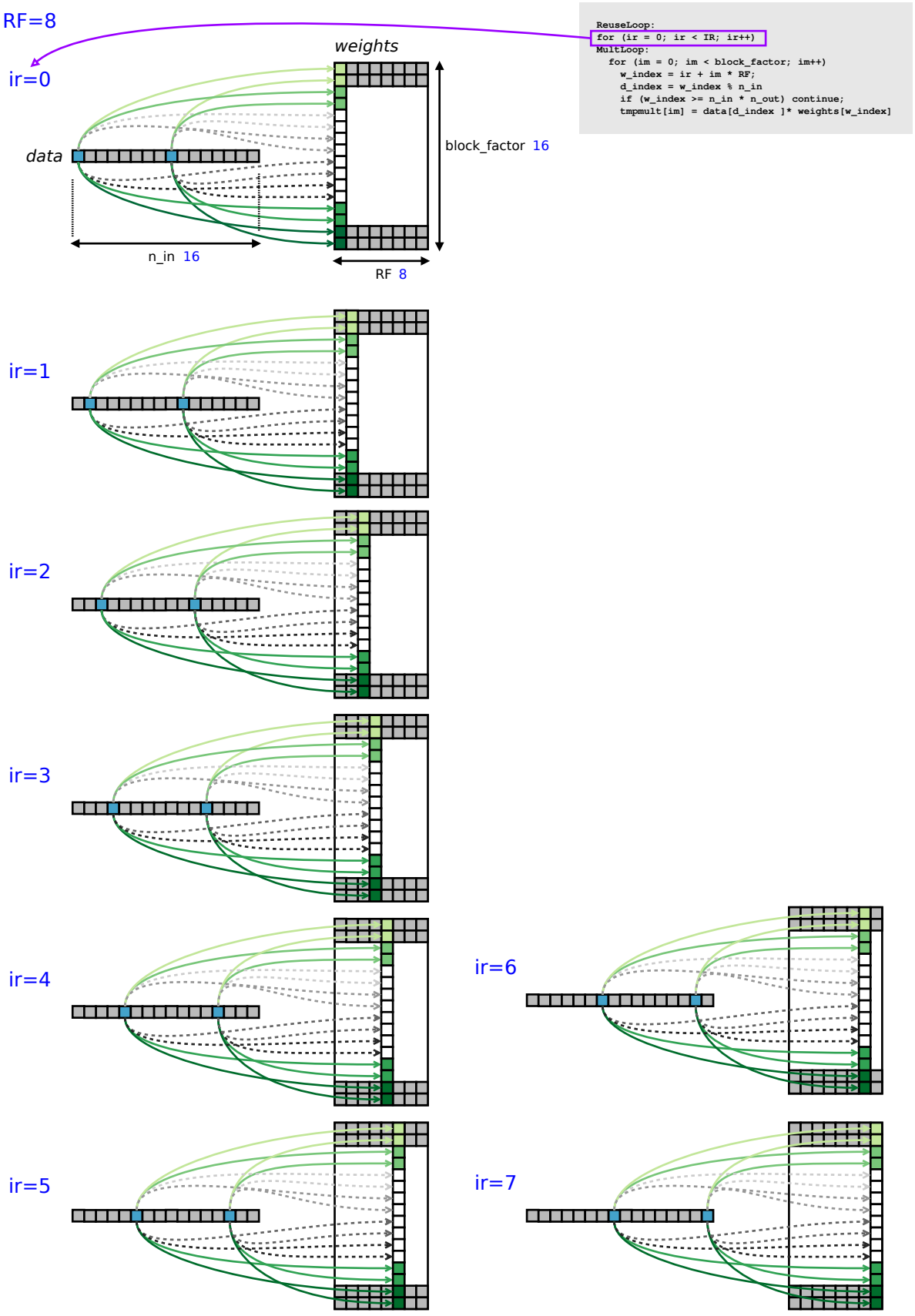
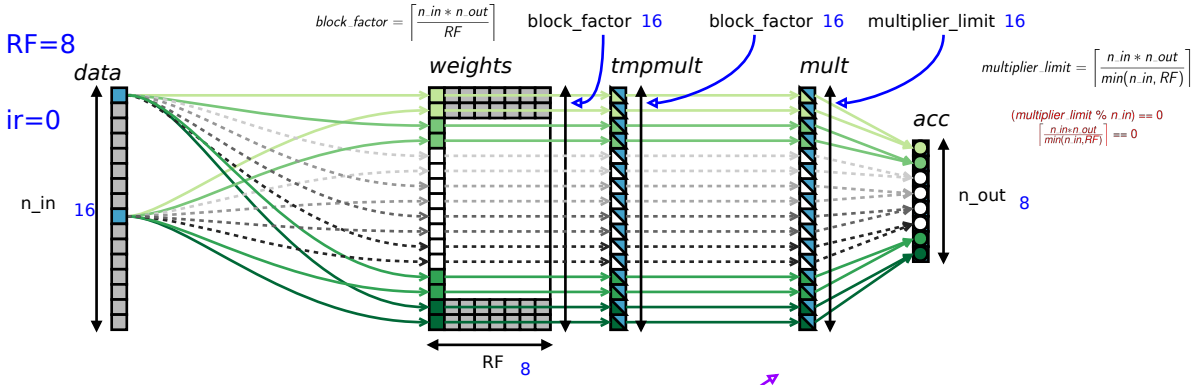
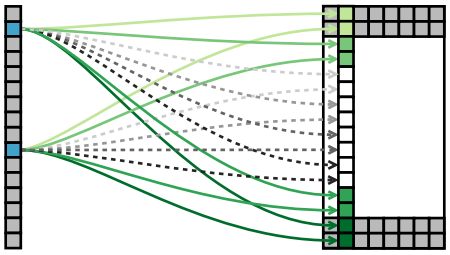
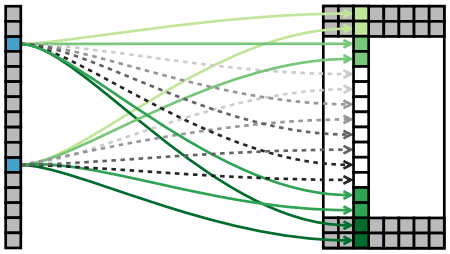
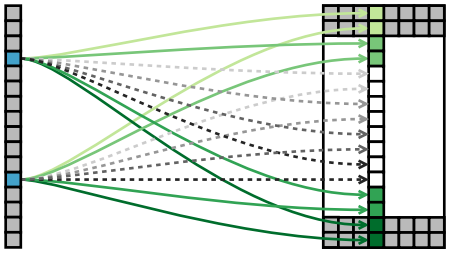

n_in = 16
n_out = 8
block_factor = ⌈n_in * n_out / RF⌉
multiplier_limit = ⌈n_in * n_out / min(n_in, RF)⌉
multscale = ⌈n_in*n_out / min(n_in,RF)⌉ / n_out
multfactor = min(n_in, RF)
(multiplier_limit % n_in) == 0
⌈n_in*n_out / min(n_in,RF)⌉ == 0
n_out
n_in
data
n_in
weights
n_in * n_out
biases
n_out
TRANSPOSE(weights)
n_in
n_out
n_in * n_out
RF=4
RF=8
RF=16
ARRAY_RESHAPE
factor = block_factor
block_factor 32
RF 4
ARRAY_RESHAPE
factor = block_factor
block_factor 16
RF 8
ARRAY_RESHAPE
factor = block_factor
block_factor 8
RF 16
tmpmult
block_factor 32
mult
multiplier_limit 32
acc
n_out 8
tmpmult
block_factor 16
mult
multiplier_limit 16
acc
n_out 8
tmpmult
block_factor 8
mult
multiplier_limit 8
acc
n_out 8
result
n_out 8

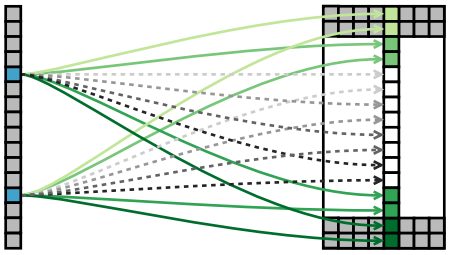

RF=8
ReuseLoop:
for (ir = 0; ir < IR; ir++)
MultLoop:
for (im = 0; im < block_factor; im++)
w_index = ir + im * RF;
d_index = w_index % n_in
if (w_index >= n_in * n_out) continue;
tmpmult[im] = data[d_index]* weights[w_index]
weights
ir=0
data
block_factor 16
n_in 16
RF 8
ir=1
ir=2
ir=3
ir=4
ir=5
ir=6
ir=7

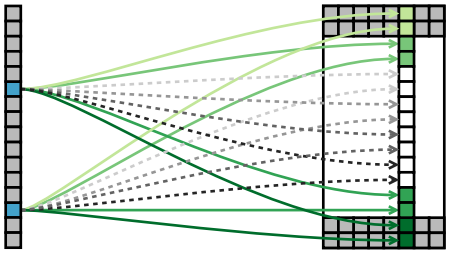


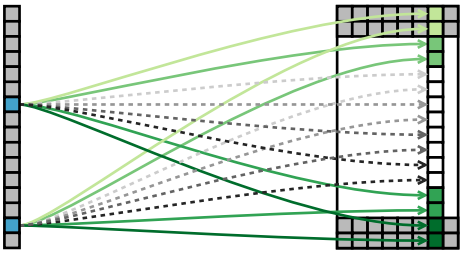


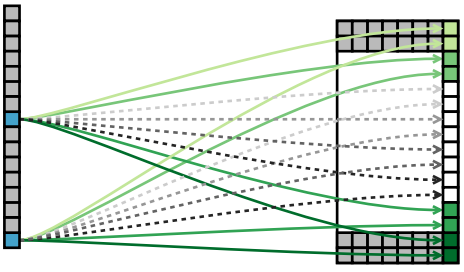




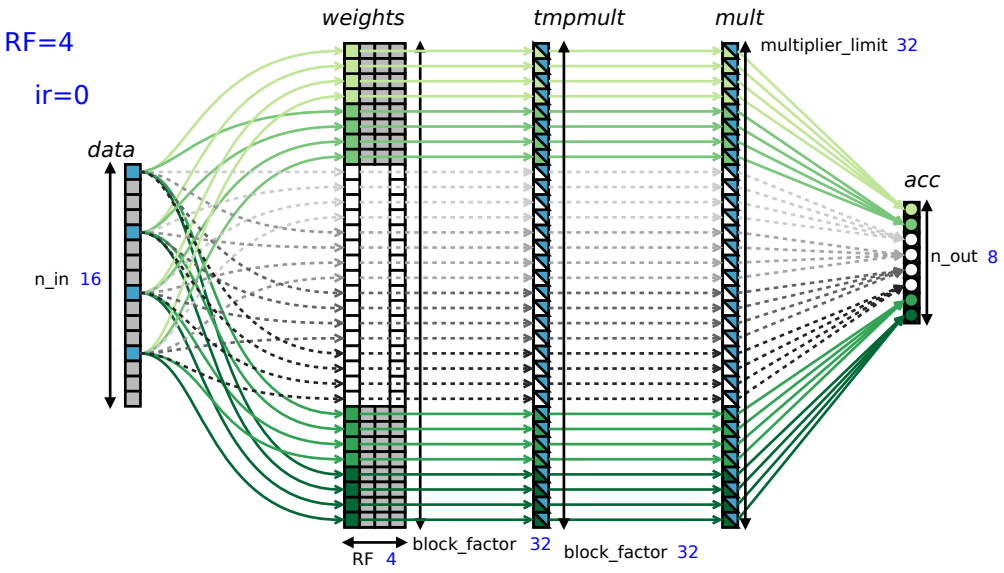


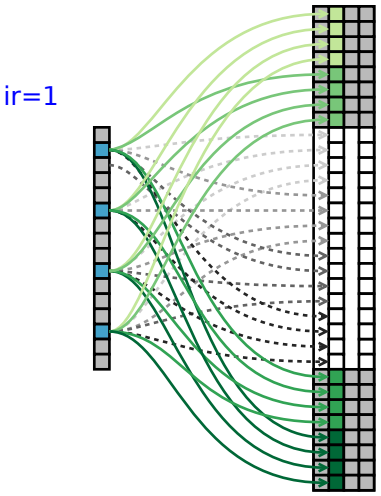




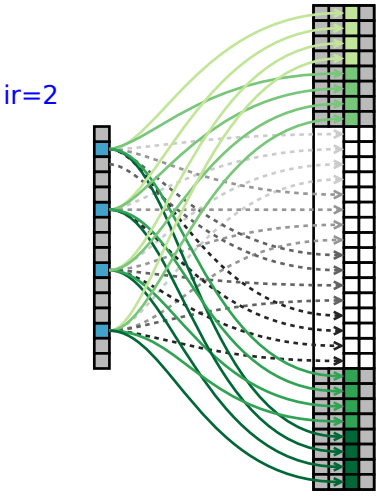


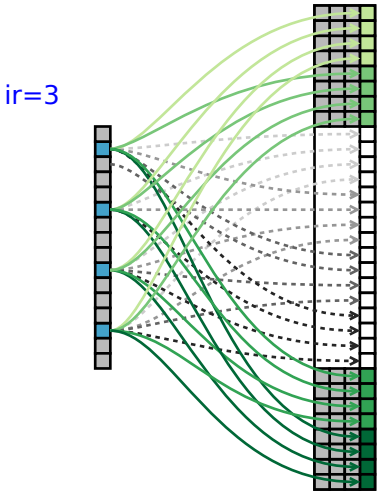

```
ReuseLoop:
for (ir = 0; ir < RF; ir++)
MultLoop:
  for (im = 0; im < block_factor; im++)
    w_index = ir + im * RF;
    d_index = w_index % n_in
    if (w_index >= n_in * n_out) continue;
    tmpmult[im] = data[d_index ]* weights[w_index]

ResetMul:
  for (im = 0; im < multiplier_limit; im++)
    mult[im] =0;

AccumLoop1:
  for (im = 0; im < block_factor; im++)
    w_index = ir + im * RF;
    out_index = w_index / multfactor;
    if (out_index >= multiplier_limit) continue;
    mult[out_index] += tmpmult[im];

AccumLoop2:
  for (im = 0; im < multiplier_limit; im++)
    out_index = im / multscale;
    acc[out_index] += mult[im];
```

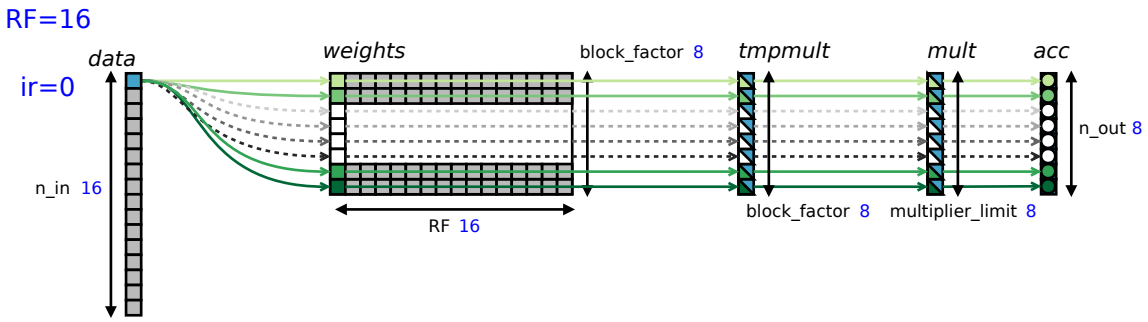





```
ReuseLoop:
for (ir = 0; ir < RF; ir++)
MultLoop:
  for (im = 0; im < block_factor; im++)
    w_index = ir + im * RF;
    d_index = w_index % n_in
    if (w_index >= n_in * n_out) continue;
    tmpmult[im] = data[d_index ]* weights[w_index]

ResetMul:
  for (im = 0; im < multiplier_limit; im++)
    mult[im] =0;

AccumLoop1:
  for (im = 0; im < block_factor; im++)
    w_index = ir + im * RF;
    out_index = w_index / multfactor;
    if (out_index >= multiplier_limit) continue;
    mult[out_index] += tmpmult[im];

AccumLoop2:
  for (im = 0; im < multiplier_limit; im++)
    out_index = im / multscale;
    acc[out_index] += mult[im];
```








```
ReuseLoop:
for (ir = 0; ir < RF; ir++)
MultLoop:
  for (im = 0; im < block_factor; im++)
    w_index = ir + im * RF;
    d_index = w_index % n_in
    if (w_index >= n_in * n_out) continue;
    tmpmult[im] = data[d_index ]* weights[w_index]

ResetMul:
  for (im = 0; im < multiplier_limit; im++)
    mult[im] =0;

AccumLoop1:
  for (im = 0; im < block_factor; im++)
    w_index = ir + im * RF;
    out_index = w_index / multfactor;
    if (out_index >= multiplier_limit) continue;
    mult[out_index] += tmpmult[im];

AccumLoop2:
  for (im = 0; im < multiplier_limit; im++)
    out_index = im / multscale;
    acc[out_index] += mult[im];
```

RF=1

ir=0