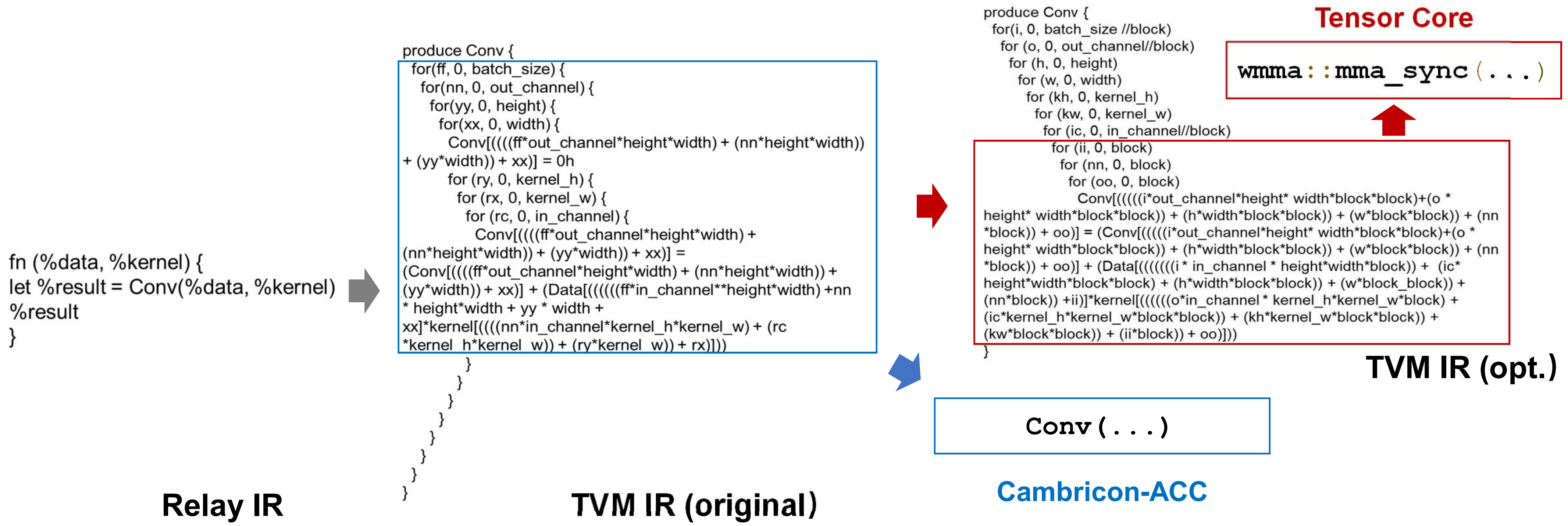

```
fn (%data, %kernel) {
let %result = Conv(%data, %kernel)
%result
}
```

```
produce Conv {
 for(ff, 0, batch_size) {
  for(nn, 0, out_channel) {
   for(yy, 0, height) {
    for(xx, 0, width) {
     Conv[((((ff*out_channel*height*width) + (nn*height*width))
+ (yy*width)) + xx)] = 0h
     for (ry, 0, kernel_h) {
      for (rx, 0, kernel_w) {
       for (rc, 0, in_channel) {
        Conv[((((ff*out_channel*height*width) +
(nn*height*width)) + (yy*width)) + xx)] =
(Conv[(((((ff*out_channel*height*width) + (nn*height*width)) +
(yy*width)) + xx)] + (Data[((((((ff*in_channel**height*width) +nn
* height*width + yy * width +
xx]*kernel[((((nn*in_channel*kernel_h*kernel_w) + (rc
*kernel_h*kernel_w)) + (ry*kernel_w)) + rx)]))
         }
        }
       }
      }
     }
    }
   }
}
```

```
produce Conv {
 for(i, 0, batch_size //block)
  for (o, 0, out_channel//block)
   for (h, 0, height)
    for (w, 0, width)
     for (kh, 0, kernel_h)
      for (kw, 0, kernel_w)
       for (ic, 0, in_channel//block)
        for (ii, 0, block)
         for (nn, 0, block)
          for (oo, 0, block)
           Conv[((((((i*out_channel*height* width*block*block)+(o *
height* width*block*block)) + (h*width*block*block)) + (w*block*block)) + (nn
*block)) + oo)] = (Conv[((((((i*out_channel*height* width*block*block)+(o *
height* width*block*block)) + (h*width*block*block)) + (w*block*block)) + (nn
*block)) + oo)] + (Data[(((((((i * in_channel * height*width*block)) + (ic*
height*width*block*block) + (h*width*block*block)) + (w*block_block)) +
(nn*block)) +ii)]*kernel[((((((o*in_channel * kernel_h*kernel_w*block) +
(ic*kernel_h*kernel_w*block*block)) + (kh*kernel_w*block*block)) +
(kw*block*block)) + (ii*block)) + oo)]))
}
```

```
wmma::mma_sync(...)
```

```
Conv(...)
```