

## Original Program

```
for(i=0;i<=M;i++)
{
  r1 = in[i];
  r2 = r1*r0;
  out2[i] = r2;
}
```

## Process 1

```
for(i=0;i<=M;i++)
{
  r1 = in[i];
  push(r1Q, r1);
  r2 = r1*r0;
  push(r2Q, r2);
}
```

## Process 2

```
for(i=0;i<=M;i++)
{
  pop(r1Q, r1);
  pop(r2Q, r2);
  out2[i] = r2;
}
```

(a)

## Process 1

```
(0)r1 = in[i]
(0)push(r1Q, r1)
(0)r2 = r1*r0
(0)push(r2Q, r2)
(1)r1 = in[i]
(1)push(r1Q, r1)
(1)r2 = r1*r0
(1)push(r2Q, r2)
(2)r1 = in[i]
(2)push(r1Q, r1)
...
```

## Process 2

```
(0)pop(r1Q, r1)
(0)pop(r2Q, r2)
(0)out2[i] = r2
(1)pop(r1Q, r1)
(1)pop(r2Q, r2)
(1)out2[i] = r2
...
```

**No Reordering in Processes**

→ *DepD: Data dependency*

→ *DepF: Dependency due to FIFO size limit*

→ *DepS: Dependency due to static intra-process scheduling*

No cycle of dependence in the schedule (b)

## Process 1

```
(0)r1 = in[i]
(1)r1 = in[i]
(2)r1 = in[i] (0)r2 = r1*r0 (0)push(r1Q, r1)
(3)r1 = in[i] (1)r2 = r1*r0 (1)push(r1Q, r1)
(4)r1 = in[i] (2)r2 = r1*r0 (2)push(r1Q, r1)
(5)r1 = in[i] (3)r2 = r1*r0 (3)push(r1Q, r1) (0)push(r2Q, r2)
(6)r1 = in[i] (4)r2 = r1*r0 (4)push(r1Q, r1) (1)push(r2Q, r2)
...
```

## Process 2

```
(0)pop(r1Q, r1)
(0)pop(r2Q, r2)
(0)out2[i] = r2
(1)pop(r1Q, r1)
(1)pop(r2Q, r2)
(1)out2[i] = r2
...
```

**Loop Pipelining Applied in Process 1**

Cycle for Artificial Deadlock

```
(2)push(r1Q, r1) → (1)pop(r1Q, r1)
(0)push(r2Q, r2) ← (0)pop(r2Q, r2)
```

(c)