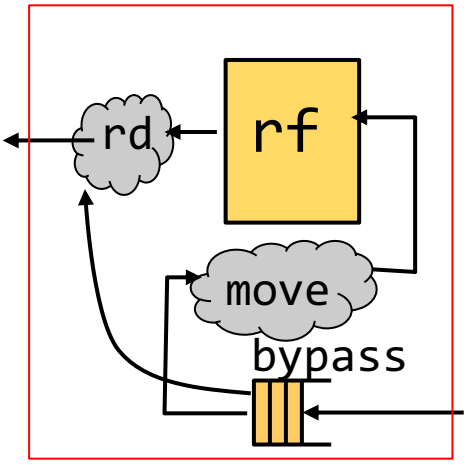
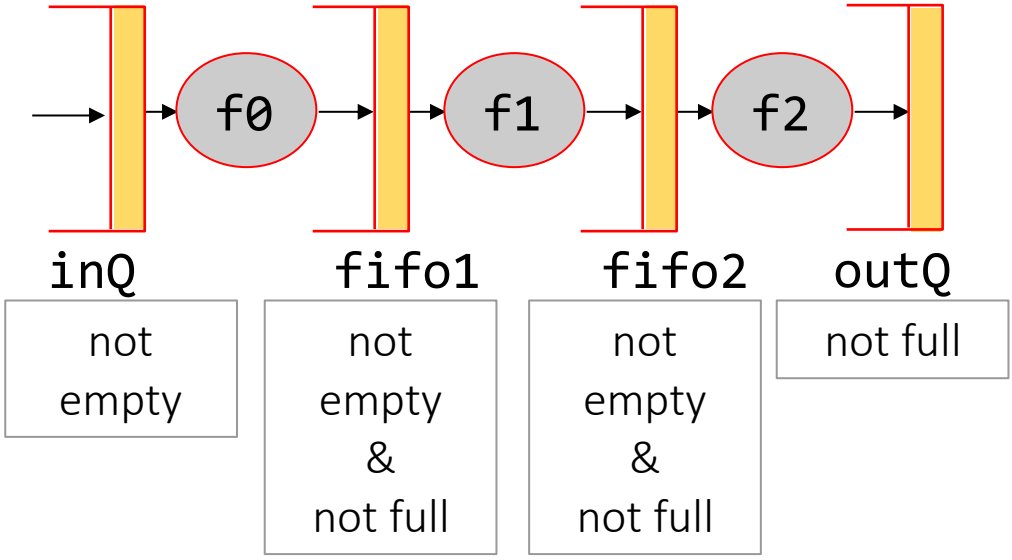
# 6.004 Recitation Problems
# L21 – Implementing Pipelines

**Problem 1:** Complete the design below. (Note: Sfifo = Searchable Fifo)

```
module mkBypassRFile(BypassRFile);
  RFile    rf <- mkRFile;
  SFifo#(1, RIdxData#(RIndx, Data))
                bypass <- mkBypassSFifo;
  rule move;




  method Action wr(RIndx rindx, Data data);




  method Data rd(RIndx rindx) =




endmodule

typedef struct {RIndx index; Data data}
RIdxData deriving (Bits);
```

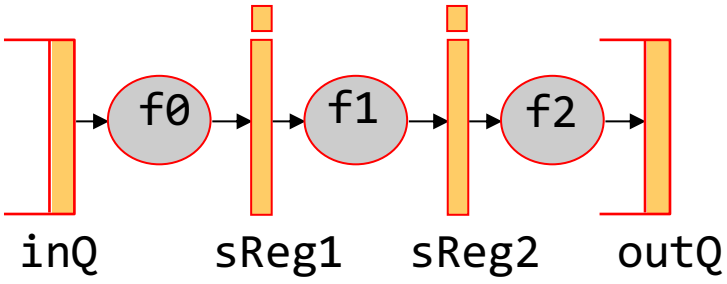


**Problem 2**: Can any pipeline stages fire concurrently if the FIFOs do not permit concurrent enq and deq?



**Problem 3**: When is this rule enabled?

```
rule sync-pipeline;
 if (inQ.notEmpty)
  begin sReg1 <= f0(inQ.first);
inQ.deq;
        sReg1v <= Valid end
  else  sReg1v <= Invalid;
  sReg2 <= f1(sReg1); sReg2v <=
sReg1v;
 if (sReg2v == Valid)
outQ.enq(f2(sReg2));
endrule
```




inQ sReg1v sReg2v outQ

| | | | |
|---|---|---|---|
| NE | V | V | NF |
| NE | V | V | F |
| NE | V | I | NF |
| NE | V | I | F |
| NE | I | V | NF |
| NE | I | V | F |
| NE | I | I | NF |
| NE | I | I | F |

Yes

inQ sReg1v sReg2v outQ

| | | | |
|---|---|---|---|
| E | V | V | NF |
| E | V | V | F |
| E | V | I | NF |
| E | V | I | F |
| E | I | V | NF |
| E | I | V | F |
| E | I | I | NF |
| E | I | I | F |

Yes

# Limitations of registers

- Using the register primitive no *communication* can take place in the same clock cycle between
  - two methods or
  - two rules or
  - a rule and a method

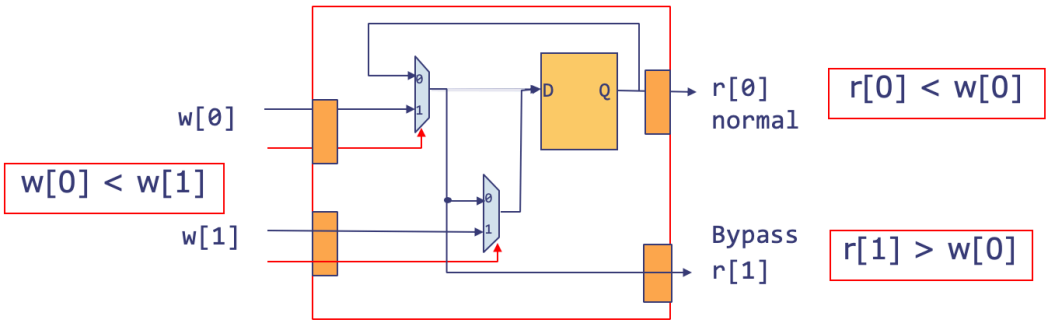
EHRs to the rescue …

Ephemeral History Register (EHR): a primitive element to remedy this problem



# Ephemeral History Register (EHR)

Dan Rosenband [MEMOCODE'04]




- r[1] returns:
  - the current state if w[0] *is not enabled*
  - the value being written if w[0] *is enabled*
- w[1] has higher priority than w[0]