

## Block RAM

Block RAM uses synchronous read, whose signals are asynchronously decoded from **opcode**. Synchronous opcode decoding sets up the sources for various registers. For example, **new\_T** indicates that **T** is to be loaded from 1 of 16 sources.

DROP followed by DUP:

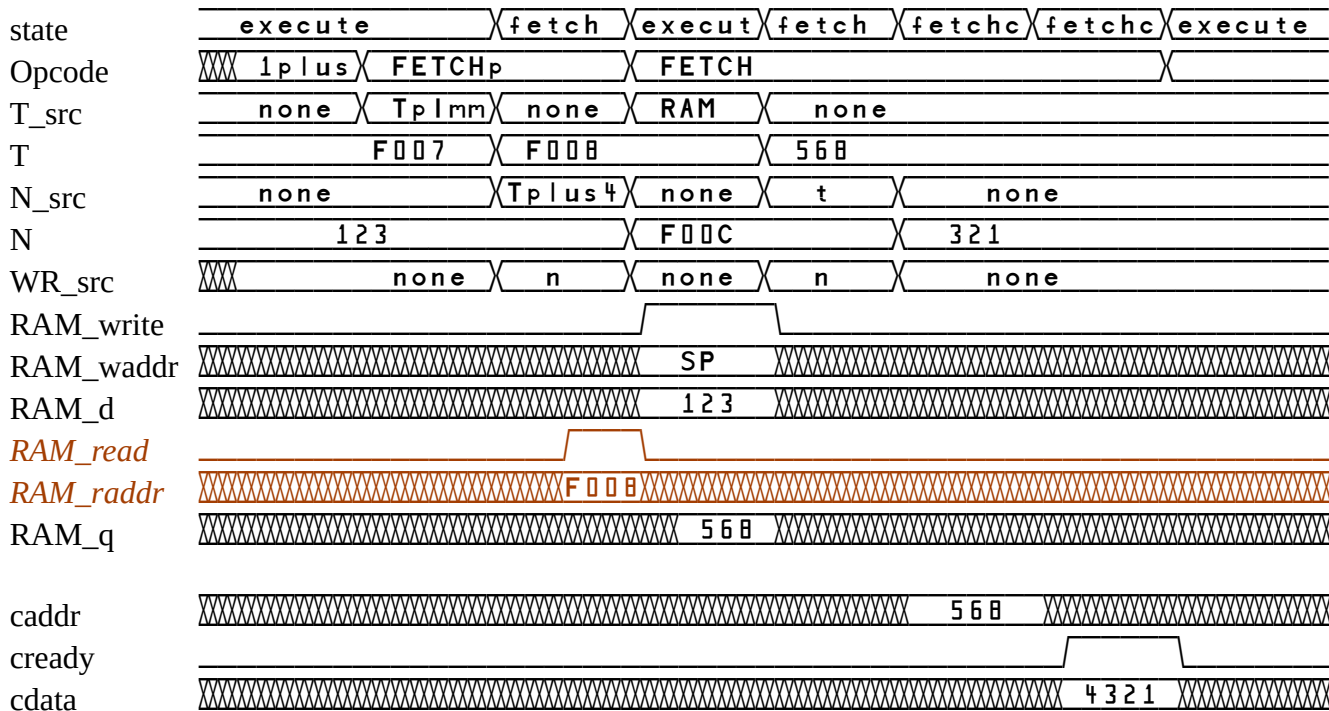
|           |        |       |      |      |      |
|-----------|--------|-------|------|------|------|
| Opcode    | 1 plus | SWAP  | plus | DUP  |      |
| T_src     | none   | TpImm | n    | Tpn  | none |
| T         | 5      | 6     | 123  | 129  |      |
| N_src     | none   | t     | RAM  | t    | none |
| N         | 123    | 6     | 321  | 129  |      |
| WR_src    |        | none  |      | n    | none |
| WR_dest   |        | none  |      | miSP | none |
| RAM_read  |        |       |      |      |      |
| RAM_raddr | SP     |       |      |      |      |
| RAM_q     | 321    |       |      |      |      |
| SPinc     |        |       |      |      |      |
| SP        | 40     | 41    | 40   |      |      |
| RAM_write |        |       |      |      |      |
| RAM_waddr | 40     |       |      |      |      |
| RAM_d     | 321    |       |      |      |      |

DUP followed by DROP: To prevent DROP's read from occurring before DUP's write, "WR\_src" must be "none" to allow decoding when a read is expected. If single-port RAM is used, decoding must also be held off while **RAM\_write** is high.

|           |        |       |        |          |
|-----------|--------|-------|--------|----------|
| Opcode    | 1 plus | DUP   | DROP   |          |
| T_src     | none   | TpImm | none   | n none   |
| T         | 5      | 6     |        |          |
| N_src     | none   | t     | none   | RAM none |
| N         | 123    | 6     | 123    |          |
| WR_src    |        | none  | n none |          |
| WR_dest   |        | none  | miSP   | none     |
| held      |        |       |        |          |
| RAM_write |        |       |        |          |
| RAM_waddr | 3F     |       |        |          |
| RAM_d     | 123    |       |        |          |
| RAM_read  |        |       |        |          |
| RAM_raddr | 40     | 3F    | 40     | 40       |
| RAM_q     | 123    |       |        |          |
| SPinc     |        |       |        |          |
| SP        | 40     | 3F    | 40     |          |

Fetch needs T before it can start a read.  $@+(a - a+4n)$ .

The fetch state checks T and fetches from ROM space (instead) if necessary.



## ROM

The ROM interface uses CADDR, CDATA, and CREADY. It's the CPU's interface to large ROM. When CADDR changes, CREADY drops until CDATA matches the new data. Usually it's a synchronous-read ROM. A dumb version would not look ahead:

