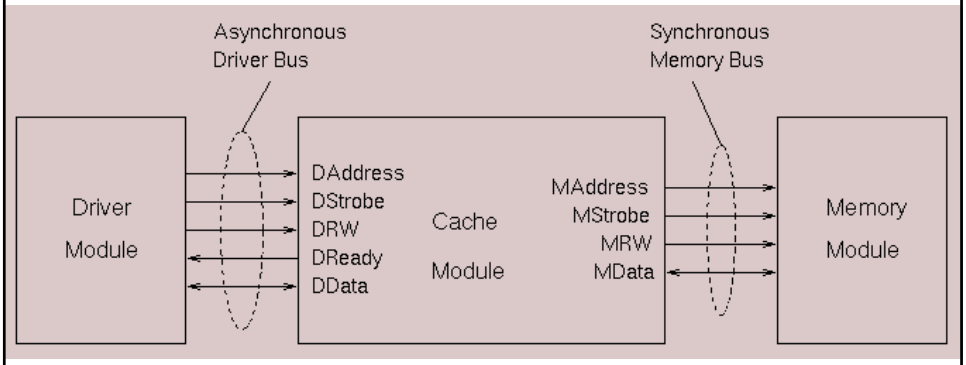
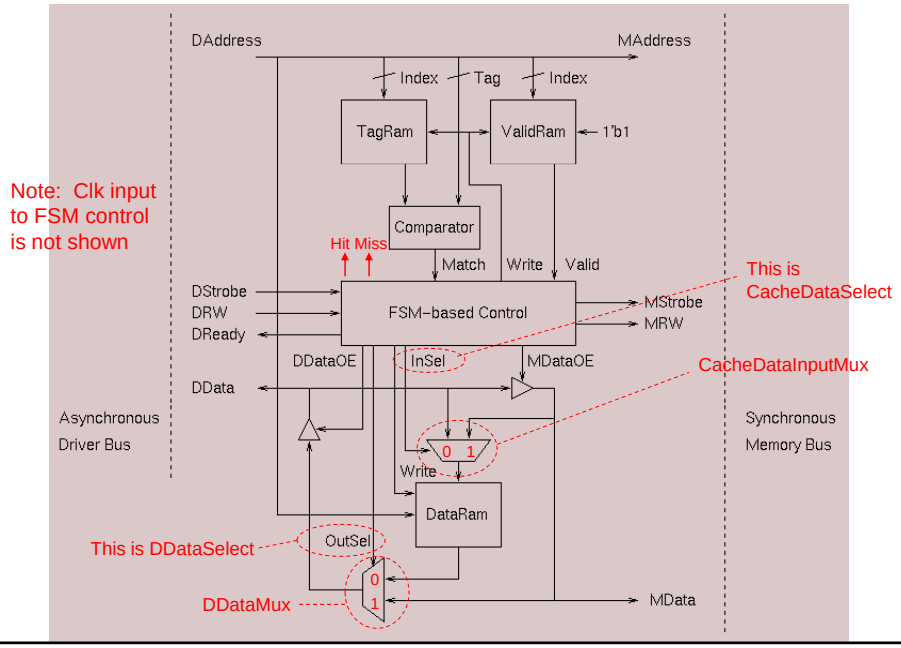
First Verilog Project (Cache Memory)

Some explanation of the direct mapped cache model

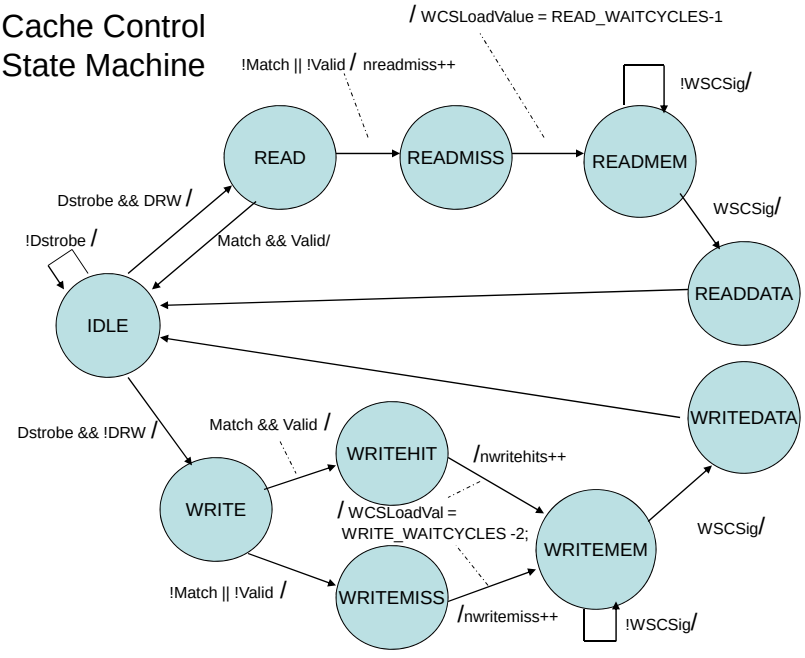
Overall Structure:



Cache Structure



Cache Control State Machine



Cache Controller--States

- IDLE: no memory access underway
- READ: Read access initiated by driver; Cache is checked during this state. If hit, access is satisfied from cache during this cycle and control returns to IDLE state at next transition. If miss, transition to READMISS state to initiate main memory access
- READMISS: Initiate memory access following a read miss. Wait state counter is loaded to time the wait for completion of the main memory access. Transition to READMEM State.
- READMEM: Main memory read in progress. Remain in this state until wait state counter expires then transition to READDATA state. (Main memory read requires READ_WAITCYCLES cycles to complete)
- READDATA: Data available from main memory read. Write this data into the cache line and use it to satisfy the original processor (driver) read request

Cache Controller States--Continued

- WRITE: Write access initiated by Driver. If cache is hit, transition to WRITEHIT state. If miss, transition to WRITEMISS state.
- WRITEHIT: Cache has been hit on a write operation. Complete write to cache and initiate write-through to main memory. Load wait state counter to time main memory access waiting period. Transition to WRITEMEM state.
- WRITEMISS: Cache has been missed on a write operation. Write to cache (cache load) and initiate write-through to main memory Load wait state timer to time main memory waiting period
- WRITEMEM: Main memory write in progress. Wait for expiration of wait state counter, then transition to WRITEDATA state.
- WRITEDATA: Last Cycle of Main memory write. Assert Ready signal to Driver to indicate completion of write.

Cache Control—Signals Asserted

- IDLE: none
- READ: DReadyEnable, DDataOE, Hit (if read hit)
- READMISS: Miss, WSCLoad, MStrobe, MRW, DDataOE
- READMEM: MRW, DDataOE
- READDATA: Ready, Write, MRW, CacheDataSelect, DDataSelect, DDataOE
- WRITE:
- WRITEHIT: Hit, WSCLoad, Write, MStrobe, MDataOE
- WRITEMISS: Miss, WSCLoad, Write, MStrobe, MDataOE
- WRITEMEM: MDataOE
- WRITEDATA: Ready, MDataOE

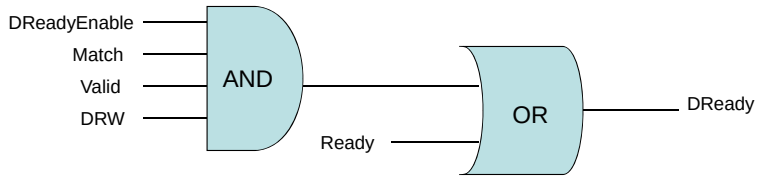
Note: Signals **Hit** and **Miss** are not shown on the diagrams or used in the implementation of the direct mapped cache. These signals may aid in the implementation of the set-associative cache

Explanation of the DReady Signal

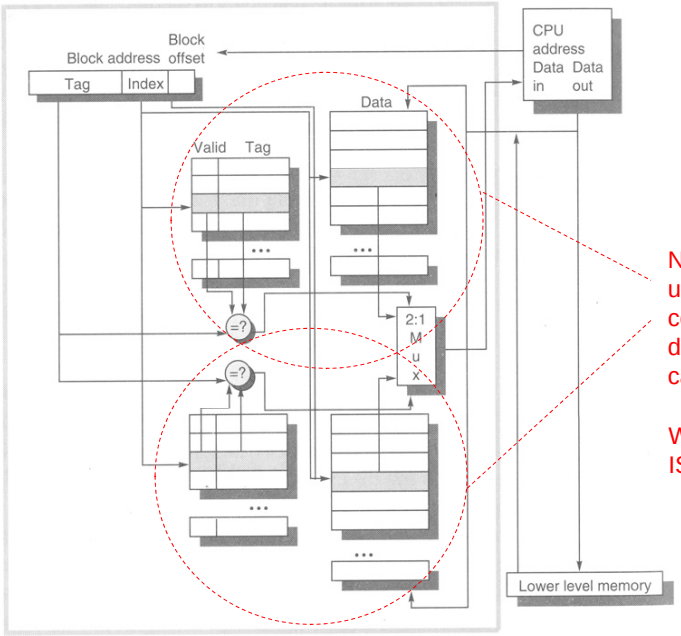
- In module CacheControl, the output DReady is controlled by a “continuous assignment” of the form:

```
wire DReady = (DReadyEnable && Match && Valid && DRW) || Ready;
```

- This is equivalent to:



Two-way set-associative cache



Note: Basically utilizes two copies of a direct mapped cache.

WHAT ELSE IS NEEDED?

Some Additional Pointers

- You should not need to mess with the driver module (driver.v)
- You should not need to mess with (or even understand the internals of) the main memory module (hashmem.v)
- You should not need to modify the cache controller (control.v) although you need do thoroughly understand the finite-state machine that it implements).