Data Cache Tag Mechanism

Designer: Todor Arnaudov

Date: 04.03.2008

Table of Contents

1. Functionality

This block is an example of logical implementation of the tag-mechanism of a 4-way set associative copyback data cache. However, it must be explicitly stated, that the system does not work with appropriate timing for a real cache design, i.e. one result at every clock. The design is done following standard methodology of splitting the logic in pipelines, in order to shorten the length of gates that the signal passes in one clock, instead of achieving extreme timing performance.

The cache consists of 1024 tags (256 x 4), has line width of 8 words. It maintains dirty indication and least recently used (LRU) replacement policy, and is able to react on a microcontroller transaction in 6 clocks, which is what makes the design not suitable for practical use.

1.1. Input/Output Signals

Signal	Width	Direction	Description
addr	22	Input	Address in main memory
rd_wr_b	1	Input	Read (1) or Write (0) transaction
valid	1	Input	Valid address
cb_addr	22	Output	Copyback address
cb_dirty	8	Output	Copyback dirty indications
cb_valid	8	Input	Copyback valid

Table 1. Input/Output Signals

1.2. Registers

Registers	Size(bits)	Description
rdwr_t0, rdwr_t1, rdwr_t2, rdwr_t3, rdwr_t4	4 x 1	Rd_wr signal pipeline
valid_t0, valid_t1, valid_t2, valid_t3, valid_t4	4 x 1	Valid signal pipeline
index_t0, index_t1, index_t2, index_t3, index_t4	4 x 8	Pipeline for the 8 LSB of the input ad dress, treated as addresses in the 4 cache sets.
in_ad1_t0, in_ad2_t1, in_ad3_t2, in_ad4_t3	4 x 11	Pipeline for the part of the input address which is saved as a tag on miss
in_line1_t0, in_line2_t1, in_line3_t2, in_line4_t3, in_line5_t4	5 x 3	Pipeline which updates the dirty indication.
Set0_t3, Set1_t3, Set2_t3, Set3_t3	4 x 22	Output from the cache set memories. Contains a Valid flag (1 bit), a Recentness indication (2 bits), Tag (11 bits) and Dirty indication (8 bits)
tag_0_t4, tag_1_t4, tag_2_t4, tag_3_t4	4 x 22	Bypass registers
dirty_0_t4, dirty_1_t4, dirty_2_t4, dirty_3_t4	4 x 8	Dirty indication bypass registers.
wh_0_t4, wh_1_t4, wh_2_t4, wh_3_t4, wh_4_t4	4 x 1	Write_Hit signal bypass regsters
Hit_0_t4, Hit_1_t4, Hit_2_t4, Hit_3_t4	4 x 2	Hit index bypass registers

Table 2. Registers

1.3. Memory

Name	Size(bits)	Description
Set0, Set1, Set2, Set3	4 x 512 x 22	Tag memory.

Table 3. Memory

1.4. Behaviour

Since the cache line is 8 words long, the value of 3 LSB bits of the address define the address of the word in the line, which in the cache is represented by dirty indications.
The next 8 LSB represent the *index* – the address of the tag in the memories of the sets.
The 11 MSB are the value of the *tag*.

The cache has a **hit**, when the 11 MSB of the address match with the tag value of the address in any of the four sets memories.

The cache has a **miss**, when the 11 MSB of the address do not match with the tag value of the address in all of the four sets memories.

On miss - the input address is sent to the index of one of the cache sets.

If there is a set, which has an empty place for the index address – it is not previously written after reset and thus is not valid – then the invalid address is filled with the 11 MSB of the input address, its valid flag is set to 1, the dirty indication is set to the priority encoded value of the last 3 MSB of the input address and the recentness field is set to 3 (most recent).

In case all four sets have valid tags, then the input address must overwrite the value in one of the sets, which first is sent through the outputs: cb_addr, cb_valid, cb_dirty. Selection of the set where the tag will be replaced is done using Least Recently Used policy and Copyback mechanism. LRU policy is implemented by attaching recentness value for each index in all four sets. The recentness value represents a queue of the order in which the sets were accessed.

On each access to an index of a set, either on miss or on hit, its recentness is set to the maximum: 3, while the recentness values of the other sets are determined by the following algorithm: the sets which have recentness value smaller than the current value of the set which is accessed, remain the same, because their order do not change. On the other hand, the sets which have bigger value are decreased with one.

Copyback, known also as Write Back mechanism is one of the two ways to connect the cache with the secondary memory. The other way is using Write Through mechanism, which means on each access to the cache, either on miss or on hit, the current value of the tag to be sent back to the upper level of the memory. In Copyback design, this is done only when the tag is “dirty” and it must be replaced; then the full address of the line which it represents, and the dirty indication, are copied back to the secondary memory.

On hit - the recentness value and the dirty indication of the hit tag is updated.

2. Subsystems

The system is divided in 3 subsystems: Tag Logic, Set Memories and LRU Logic. Please, see Appendix 1. for detailed schematics of all blocks. Some of the schematics are omitted here, because they are too big.

2.1. Tag Logic and Set Memories

Tag Logic is the main subsystem. It receives the input and sends the output signals. Tag logic is tightly coupled with Set Memories and uses LRU Logic block to update the tag values and eventually to copy back tag to the output.

Because of the use of memories and in order to split long paths of gates, several pipelines are used.
There are 5-step input pipelines for rd_wr_b and valid signals. The 22-bit address input is splitted in 3 5-step pipelines: one 8-bit for the index, one 3-bit for the address in the line, which is decoded to dirty indication, and one 11-bit which contains the field of the address which is matched with the tags in the sets.

2.1.1. Reading tags from set memories

The value of the index, rd_wr_b and valid are taken from the second step in the pipeline and used in parallel, respectively as Read Address and for deciding the value of Read Enable in all four sets.

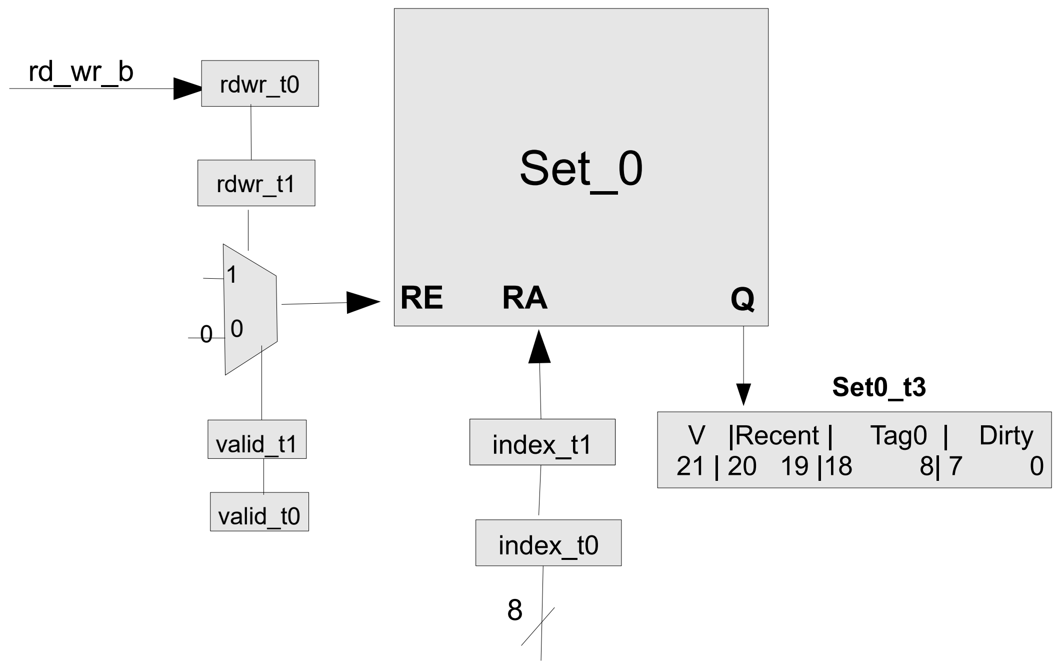

Figure . Simplified schematics of the reading from the tag sets

2.1.2. Routing tag fields and setting hit and miss flags

The value read from the memory is 22-bit long and consists of the following fields:

V	Recent	Tag0	Dirty
21	20 19	18 8	7 0

Validity field (V) value is sent to two directions – to a bypass register which delays it for the final clock cycle, and to the logic for calculating the values of Read_Hit, Read_Miss, Write_Miss, Write_Hit and the Hit_Index. Hit_Index is the number of the set which is matched with the input address in case of hit. It is calculated using 4 comparators and 4 ANDs – one for each of the sets which find is there any hit. LRU block prevents cases where more than one set match an input address, thus a coder is used to compute Hit_Index value which passes through a bypass register before going to the LRU block.

The outcomes of the 4 hit tests pass through a 4-input OR, then Hit and Miss flags are set and passed through bypass registers before going to the LRU block.

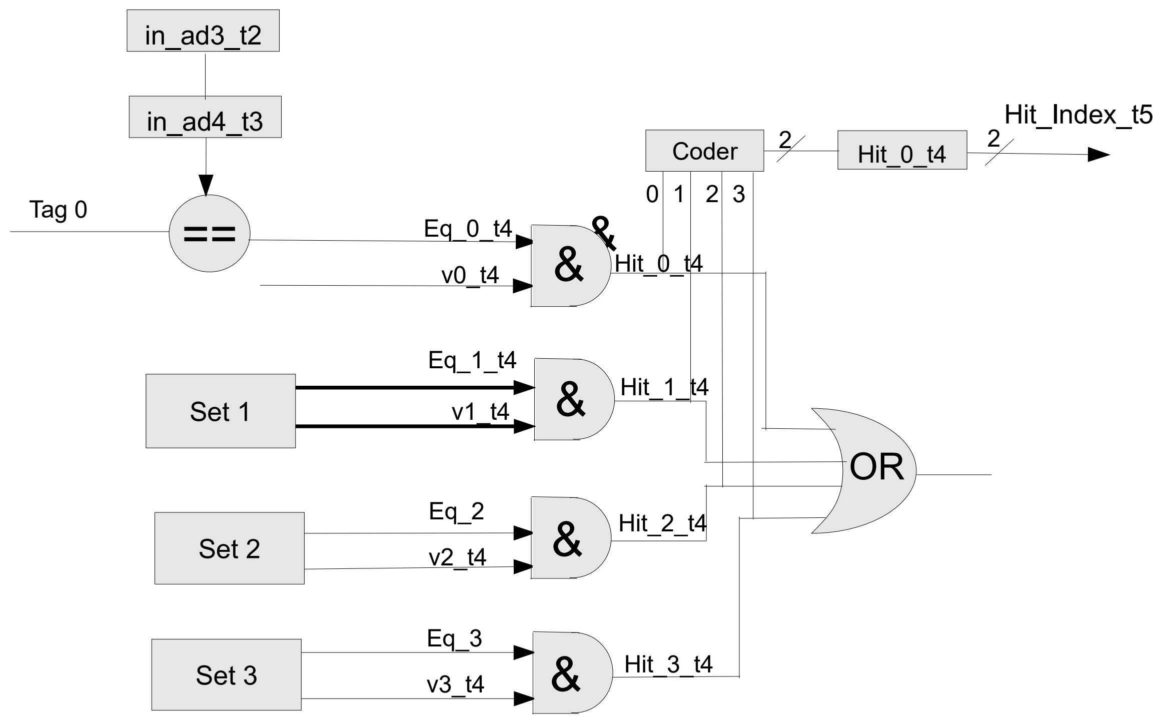

Figure . Calculating Hit_Index value

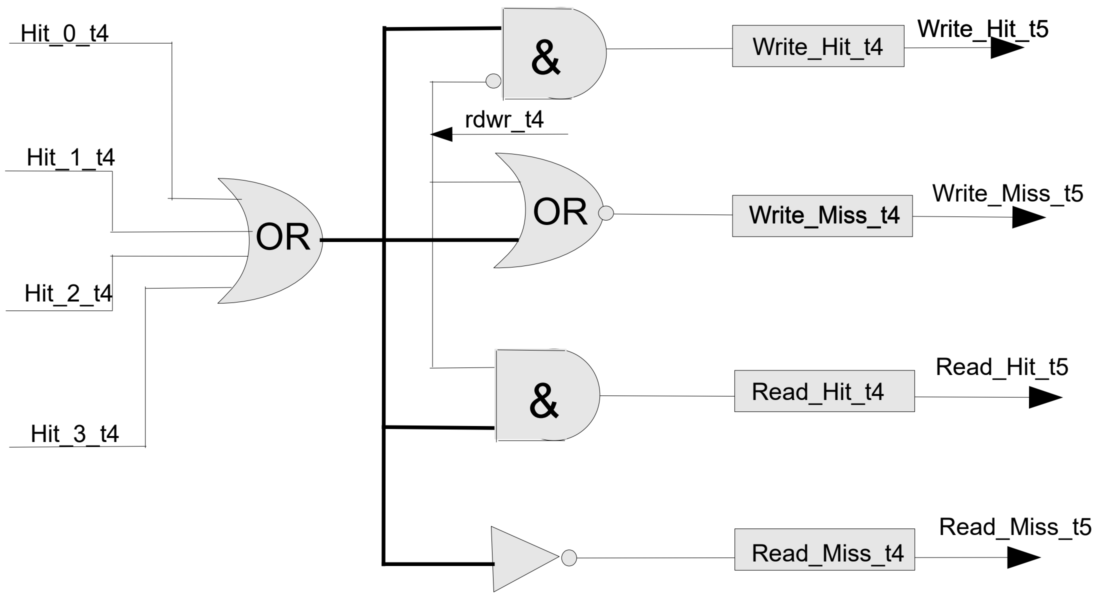

Figure . Setting Hit and Miss flags.

2.1.3. Writing updated values to Set memories

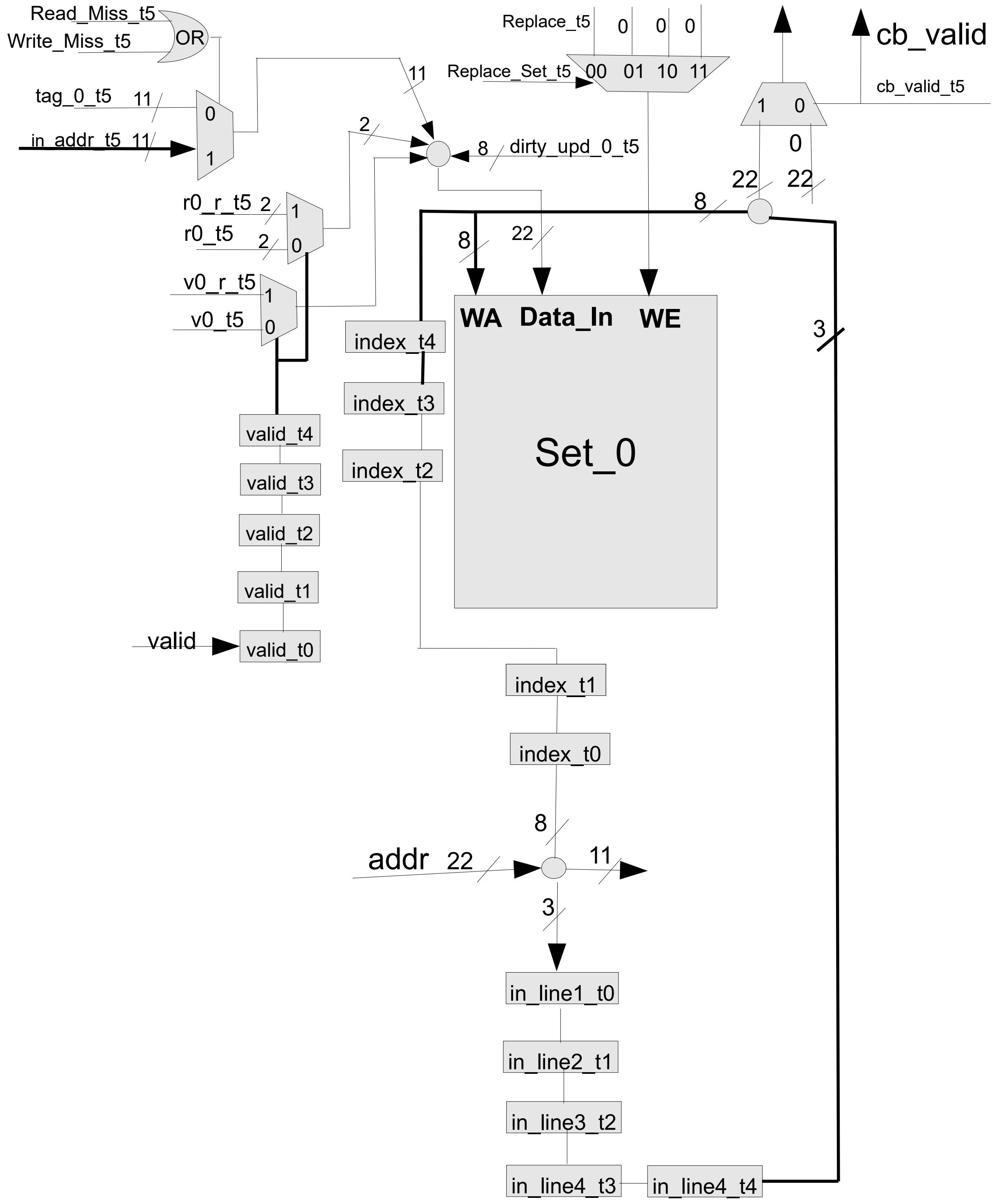

Figure . Writing updated values to set memories

Decision to update the sets is made using two control signals from the LRU block – Replace_t5 and Replace_Set_t5. There are four 4-input multiplexers which set the respective Write Enable signal to 1 only if the set to be replaced matches the number of the set, and if the flag to do replacement is set. Writing address is the value of the 5-th step of Index pipeline.

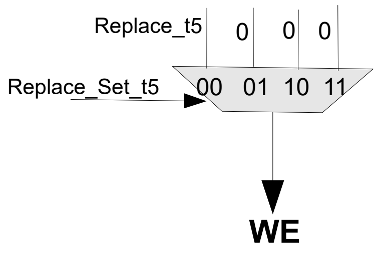

Figure . Writing Enable multiplexer

Data_In contains a 22-bit vector, consisting of the old or a new 11-bit tag, 1-bit validity flag, 2-bit recentness flag and 8-bit dirty indication. Selection between delayed values of the latest read valid flag and recentness field, and the updated values by the LRU block - v0_r_t5 and r0_r_t5 for the first set, - is done using two 2-input multiplexers, controlled by the delayed value of the valid input signal.

Another multiplexer, controlled by an OR between Read_Miss and Write_Miss flags, controls whether the tag field will be updated with the incoming address from the last step of a pipeline or it will be rewritten with its current value, taken from the respective *tag_N_t4* bypass register.

The last 8-bits in the vector – dirty_upd_t5 - come from the logic for calculating the dirty indications.

2.1.4. Calculating dirty indications

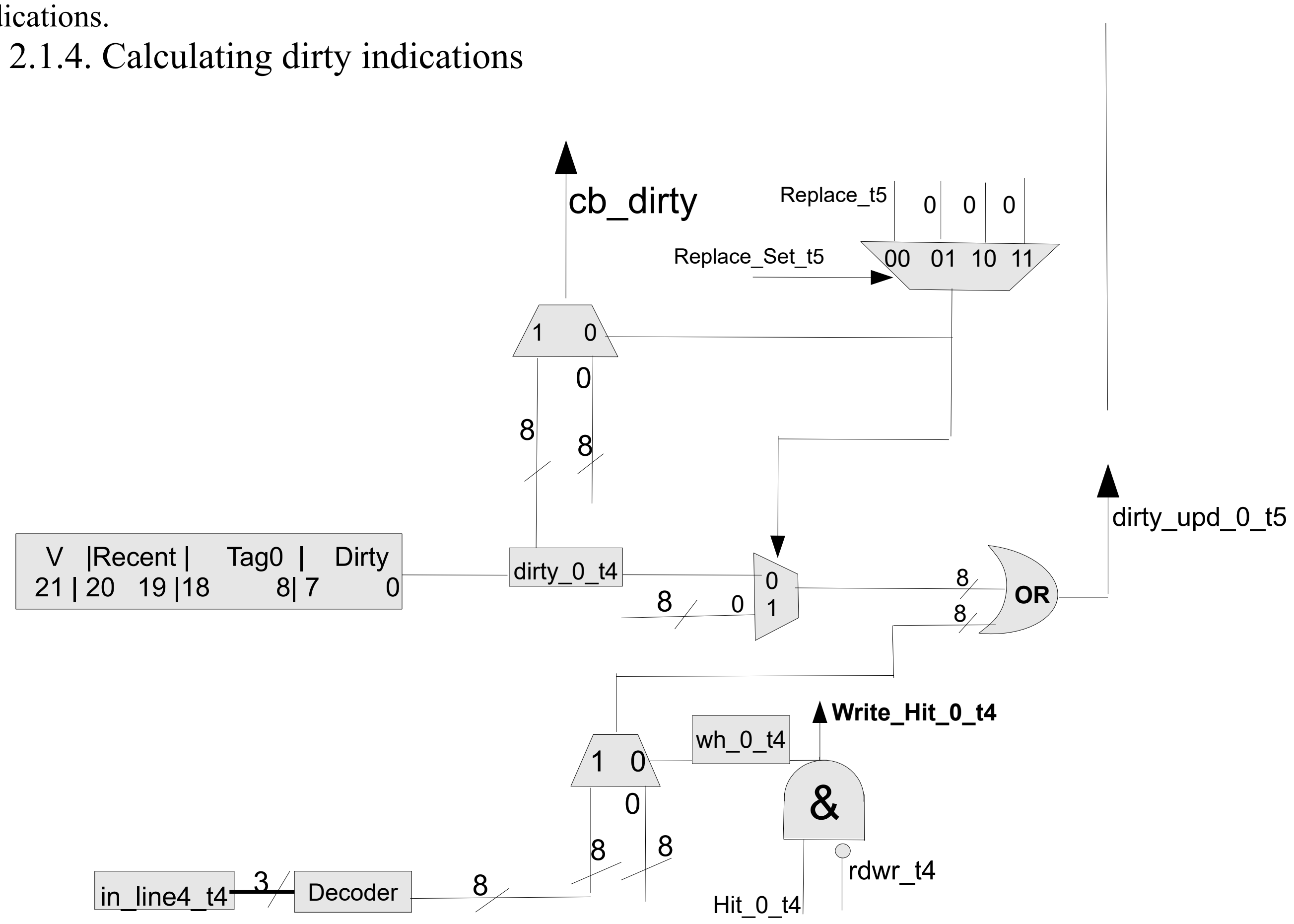

Figure . Calculating dirty indications

If there is a write hit, then dirty indication – a decoded value of the 3 LSB of the address - is OR-ed with the current value of the dirty indication of the respective tag. Otherwise – on write miss, read hit or read miss, - dirty indication is just overwritten with the incoming decoded 3 LSB of the address.

2.1.5. Copying back to upper level memory

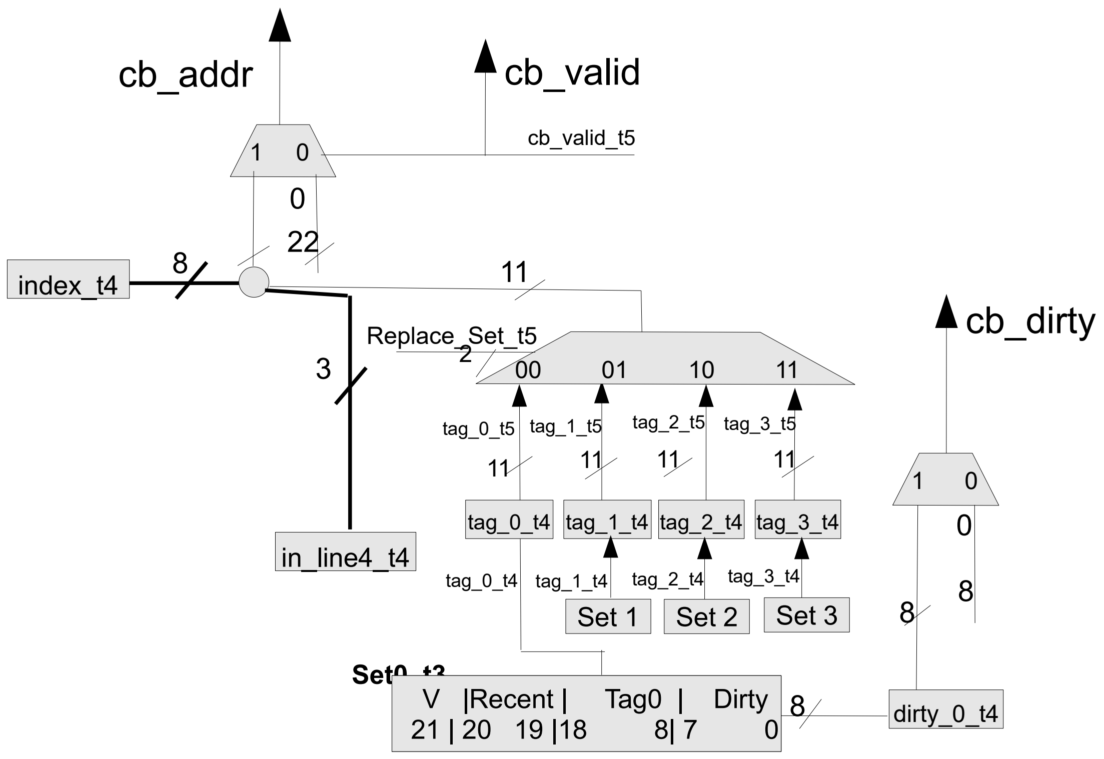

Figure . Copying back

A 4-input 11-bit multiplexer with 2-bit control input selects the set from which the tag is selected to be overwritten, based on the value of the Replace_Set_t5 signal, generated by the LRU block.

The address for the cb_addr is reconstructed by the delayed value of the index and the position of the from the index_t4 register and

which Copy back is performed when the LRU block sets Replace_t5.
\If there is a write hit, then dirty indication – a decoded value of the 3 LSB of the address - is OR-ed with the current value of the dirty indication of the respective tag. Otherwise – on write miss, read hit or read miss, - dirty indication is just overwritten with the incoming decoded 3 LSB of the address.

2.2 LRU Logic

LRU Logic block is a combinational logic, without any registers, which updates the validity and recentness values of the tags and selects the least recently used tag to be replaced, when all sets are full.

2.2. Tag Logic

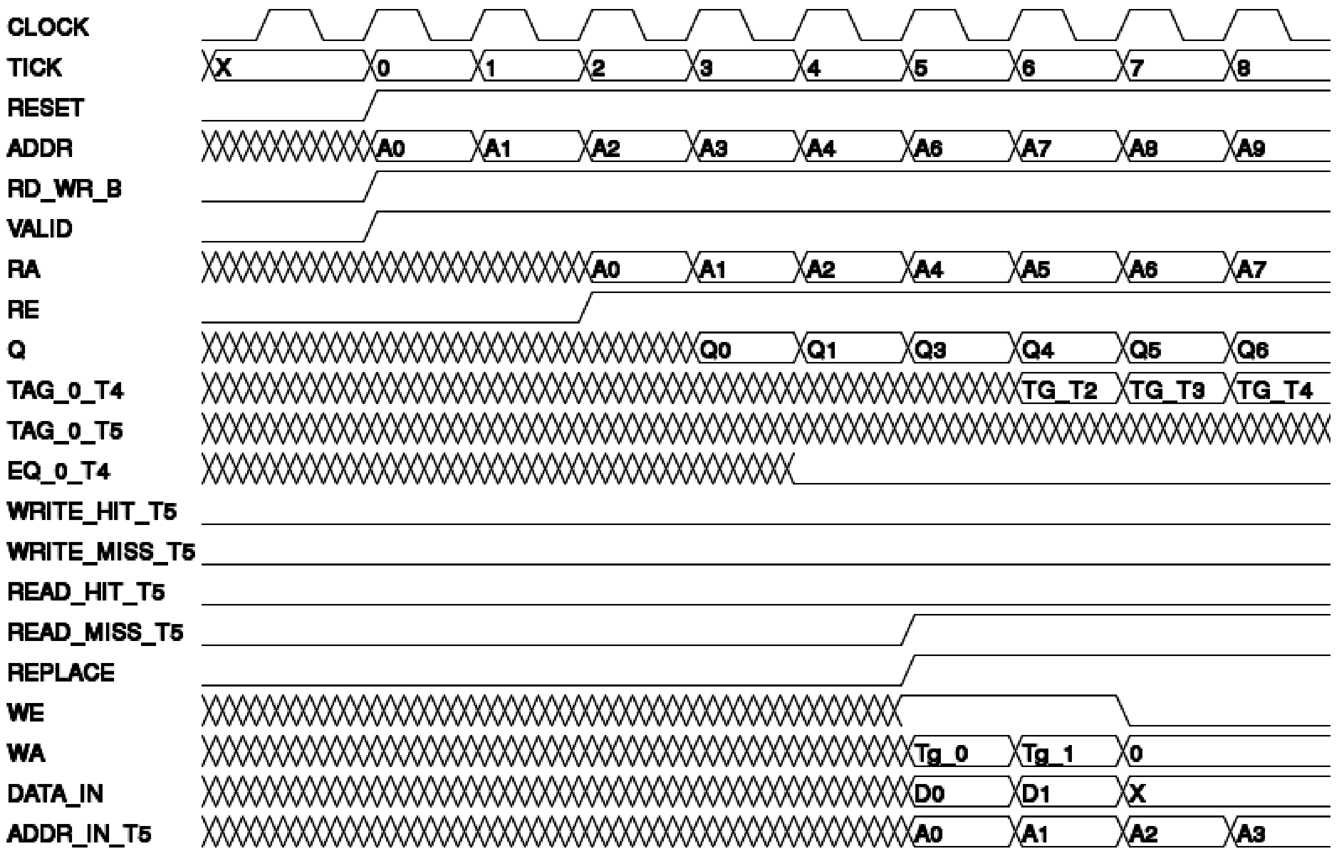

Figure . Tag Logic Timing Diagram

2.3 LRU Logic Interface

2.4 Sets Interface

2.5 LRU Logic

(...)

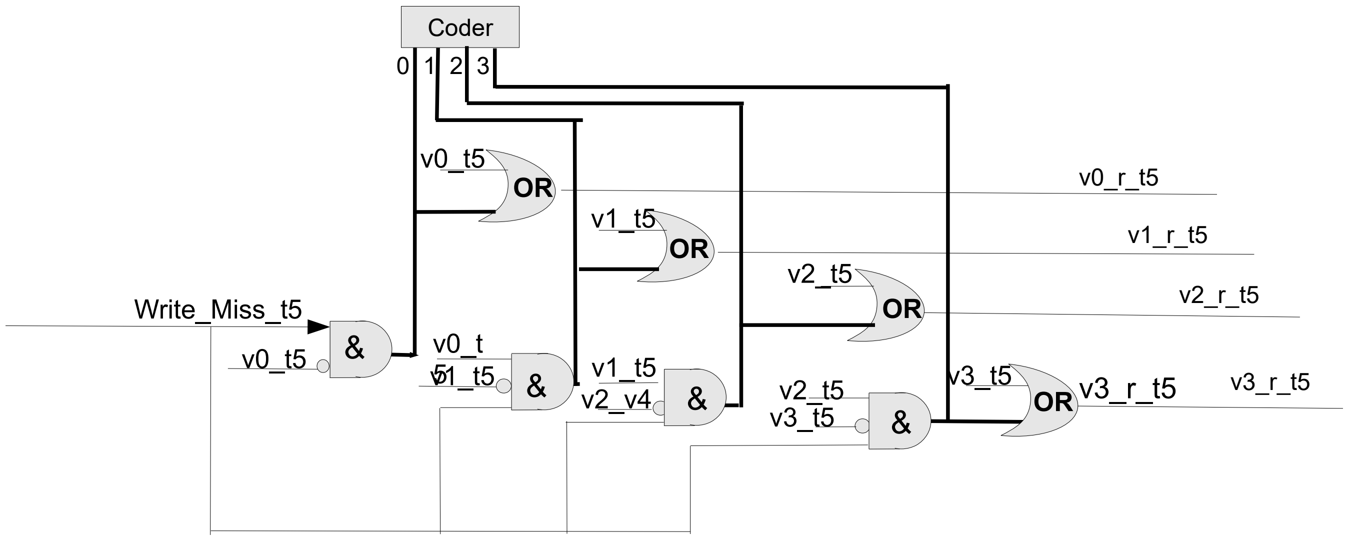

Figure . Set validity block

This block sets the valid bits from the first set to the last, when there is a row of misses at the same index and there still are free sets.