**PRACTICAL 9**

**COMPUTER ORGANISATION AND ARCHITECTURE**

|  |  |
| --- | --- |
| **NAME: VARUN KHADAYATE** | **ROLL NO: A016** |
| **PROGRAM: BTECH SY** | **DIVISION: CSBS** |
| **BATCH: 1** | **DATE OF EXPERIMENT: 27/10/2020** |

**AIM**

**Understanding the behaviour of associative cache from working module and designing an associative cache for given parameters**

**THEORY**

**Design of Associative Cache:**

Cache memory is a small (in size) and very fast (zero wait state) memory that sits between the CPU and main memory. The notion of cache memory relies on the correlation properties observed in sequences of address references generated by CPU while executing a program (principle of locality). When a memory request is generated, the request is first presented to the cache memory, and if the cache cannot respond, the request is then presented to the main memory.

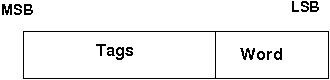
* **Hit:**  cache access finds data resident in the cache memory
* **Miss:**  cache access does not find data resident, so it forces to access the main memory.

Cache treats the main memory as a set of blocks. AS the cache size is much smaller than the main memory, so the number of cache lines is very less than the number of main memory blocks. So, a procedure is needed for mapping main memory blocks into cache lines. The cache mapping scheme affects cost and performance. There are three methods in block placement-

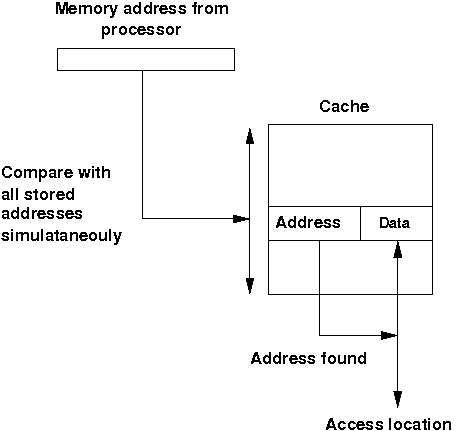
* **Direct Mapped Cache**
* **Fully Associative Mapped Cache**
* **Set Associative Mapped Cache**

**Associative Cache**

Any main memory block can map into any cache line. the main memory address is divided into two groups which are tags and word bits. Words are low-order bits and identify the location of a word within a block and tags are high-order bits that identify the block.



Block diagram of an associated cache:

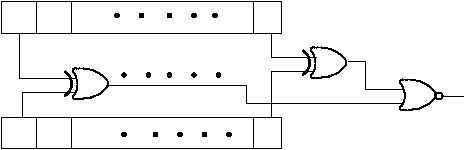


If a miss occurs CPU brings the block from the main memory to the cache, if there is no free block in the corresponding set it replaces a block and puts the new one. CPU uses different replacement policies to decide which block is to replace. The disadvantage of the associative cache is its high cost for implementing parallel tag comparison but suffer the most from thrashing due to the 'conflict misses' giving more miss penalty.

**Design issues:**

No replacement policy has been implemented in the experiment.

The comparator circuit through which tag is compared with specified bits of address:



**PROCEDURE**

**Design of Associative Cache:**

**Procedure to experiment with associative cache on the existing component 'Associative Cache' component in the 'other components' drawer in the simulator. This simulator supports 5-valued logic.**

1. Click on the 'Associative Cache' component(in the 'other components' drawer in the pallet) and then click on the position of the editor window where you want to add the component(no drag and drop, a simple click will serve the purpose), likewise add 15 Bit switches and 3 Bit Displays(from Display and Input drawer of the pallet, if it is not seen scroll down in the drawer)
2. The 'Associative Cache' component in the 'other components' drawer in the simulator supports both writings in the cache and the cache mapping. No replacement policy has been implemented. Initially, the cache is empty, the user must give inputs. the component contains 4 sets, each set has 5 bits, the leftmost bit is the valid bit, the next 2 bits are tags, the next bits are data bits, also it contains a one-dimensional array of memory with 2 bit to store the memory address, the user has to give this address input also. The cache reads all the data bits at a time so block offset is not required.
3. The pin configuration of the component can be seen whenever the mouse is hovered on any canned component of the palette or press the 'show pin config' button. Pin numbering starts from 1 and the bottom left corner (indicating with the circle) and increases anticlockwise.
4. For an 'Associative Cache' component pin configuration is:
   * pin-32= S (selects whether the user wants to perform cache write or cache mapping)
   * pin-31= R/W’A (selects whether the user wants to input the address or cache mapping)
   * pin-30=A1, pin-29=A0 (these 2 pins are used to give address input). A1 is the most significant bit and A0 is the least significant bit. As we are reading the whole word at a time, the whole address will be in the tag. So, A1 and A0 will be compared with the tag. A1 and A0 will select the corresponding set.
   * pin-28= R/W’D (selects whether the user wants to input in the set of cache or cache mapping)
   * pin-27= M1, pin-26=M0 (M1 is the most significant bit and M0 is the least significant bit). these two bits are used for cache write purpose, it selects the particular set of which user wants to give inputs to the valid bit, tag bits, and data bits.
   * pin-21= valid bit
   * pin-20= T1, pin-19=T0 (T1 is the most significant bit and T0 is the least significant bit). These are tag bits.
   * pin-18= D1, pin-17=D0 (D1 is the most significant bit and D0 is the least significant bit). These are data bits.
   * pin-14= Hit/Miss bit (if it gives 1 then hit otherwise miss)
   * pin-15= F1, pin-16=F0 (F1 is the most significant bit and F0 is the least significant bit). These are output data bits and will be given only when there is a hit.
5. **Essential pin configurations for writing in the cache: S=1, R/W'A=0, R/W'D=0**
6. **Essential pin configurations for cache mapping: S=0, R/W'A=1, R/W'D=1**
7. To connect any two components, select the Connection menu of Palette, and then click on the Source terminal and click on the target terminal. According to the circuit diagram connect all the components. After the connection is over click the selection tool in the pallet.
8. See the output, Bit switches are used to give input so that you can toggle its value with a double click and see the outputs with different inputs.

**CONCLUSION**

Hence, we can understand the behaviour of associative cache from working module and designing an associative cache for given parameters.