
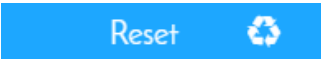
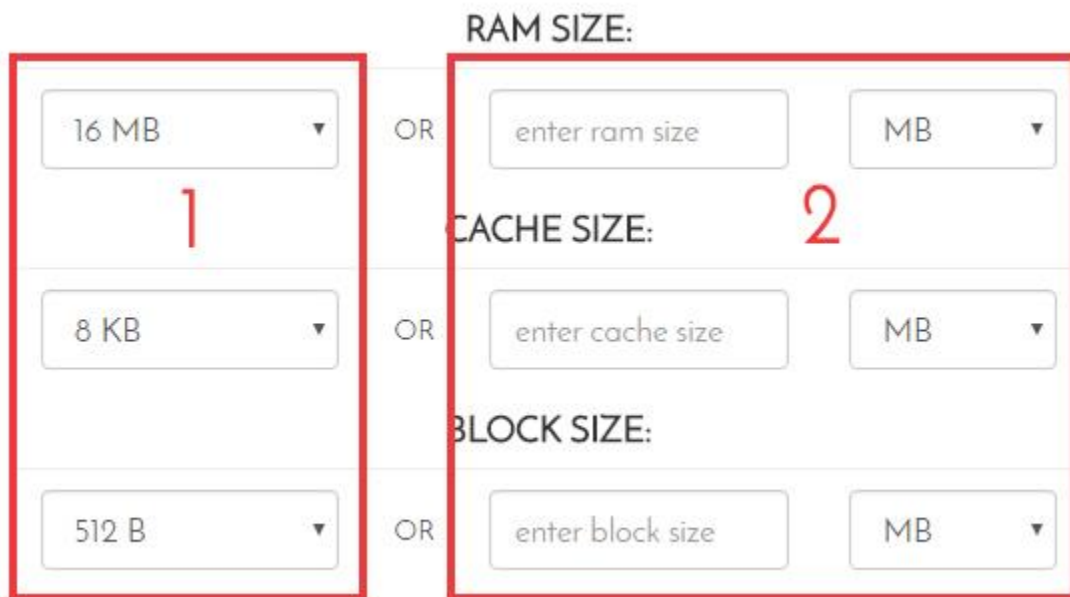


I.U.S.T Online Cache Simulator Helper

1. Start simulation with new Button 
2. Set the Sizes with 2 way (be careful to insert True Memory Size & ...):

1. from Defaults
2. Type Your Self (if you type something, Defaults won't be accepted. To use Defaults again clear inputs or press Reset Button )



RAM SIZE:

16 MB ▼

OR

enter ram size MB ▼

CACHE SIZE:

8 KB ▼

OR

enter cache size MB ▼

BLOCK SIZE:

512 B ▼

OR

enter block size MB ▼


3. Choose Mapping Options

METHOD:	Set ?-WAY:	Algorithm:
Set Associative ▼	8 way ▼	LRU ▼

4. Define Memory Addresses with 4 ways below:

1. Type Yourself into the textbox (not recommended)
2. Upload a File
3. Generate Random Addresses
4. Use Real Sequences we prepared for you (recommended)

The screenshot shows a web interface for defining memory addresses. At the top, there are two blue buttons: "Generate Random" (labeled with a red '3') and "Choose File" (labeled with a red '2'). Below these is a large text input area (labeled with a red '1') with the placeholder text "You Can Type Your Address into the Input". Underneath the input area is the text "or use Real Memory Addresses from Oberlin College Computer Science". At the bottom, there are three buttons labeled "art.trace", "mcf.trace", and "swim.trace" (labeled with a red '4'). A note at the very bottom states: "These traces were generated by a simulator of a RISC processor running three programs, art, mcf, and swim from the SPEC benchmarks".

4. Press submit  then you have to choice for simulation



First one is for step-by-step simulation, and second one for whole running.

There is an Example for one simulation:

Step-by-step (learned by Emanuele Viglianisi

(<https://github.com/emavgl>)):

Memory Access	Cache	Address
140027c10	Blocks	1400e8288
1400e8278	Block 0 [Tag: 1280] [LastAccess: 140027c10]	----- 24 b Address -----
1400e8280	Block 1	Tag: 11 b Index: 4 b Block Width: 9 b
1400e8311	Block 2	1010000000 0011 101000001
1400e8288	Block 3 [Tag: 1280] [LastAccess : 1400e8288]	INDEX = Identifies a blockposition in cache 3
1400e87d0	Block 4	BLOCK WIDTH = Identifies the bytes sequence inside the block
1400e8310	Block 5	TAG = Label associated with the position of the block
14000d750	Block 6	This memory block is already in cache - HIT
1400a1b68		

After whole running:

Direct Mapping #5

Memory Size: 2^{24}

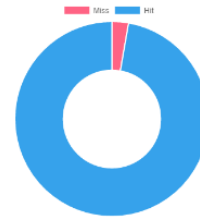
Cache Size: 2^{13}

Block Size: 2^9

Accesses: 833

Hits: 810 (97.24%)

Misses: 23 (2.76%)



5. You can do many simulations continuously and see the comparison like below:

