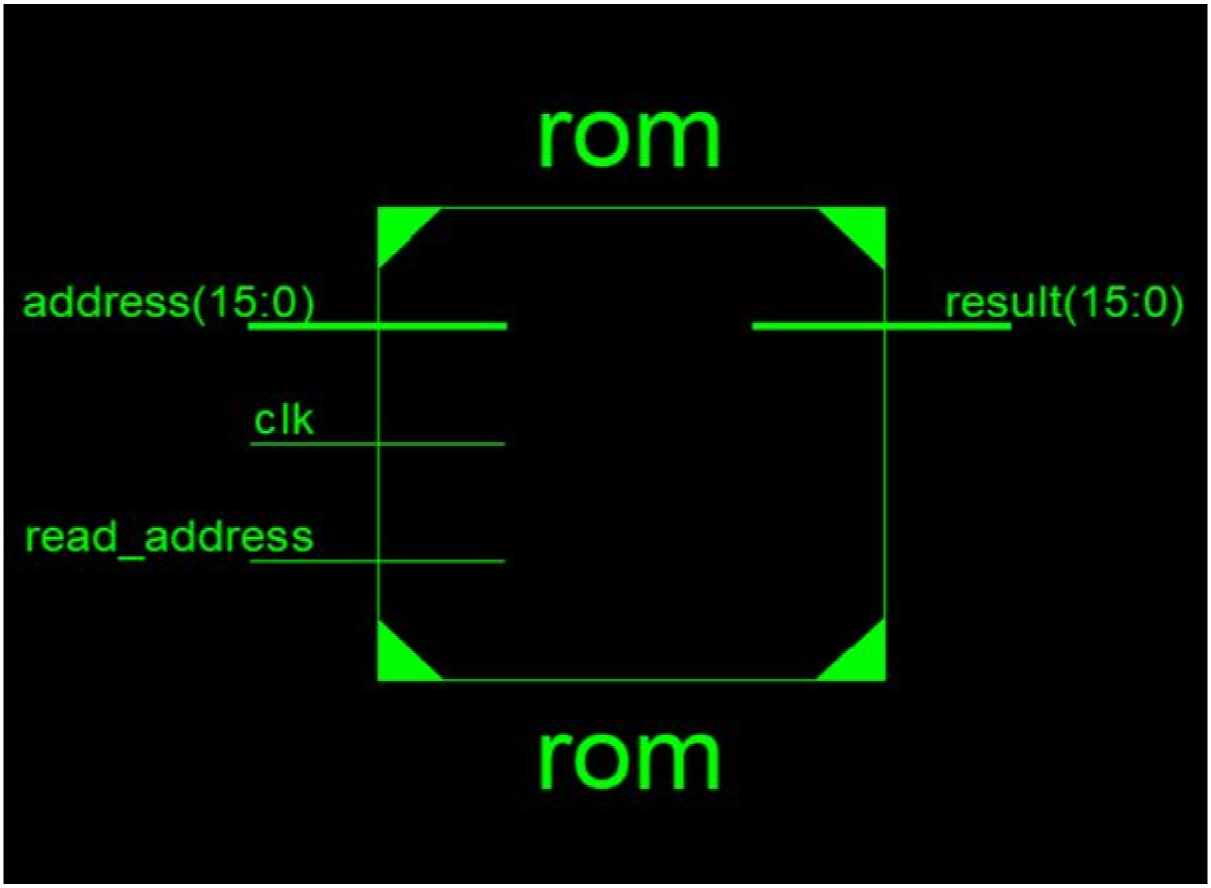
Memory Documentation

Computer Architecture Project

ROM:



Usage :

Input : Binary 16 bit address, clock , read_address which is basically an enable for reading the address if it is low the rom will get to the next address with each clock tic.

output : The 16 bit stored in passed address.

Rom Documentation :

The Rom is designed to go over the instructions pre-saved in the rom after each clock, which is the program code after compiling. To be able to use the loops and the if statements we needed to use the go-to method. So we added a read address signal which works as follows:

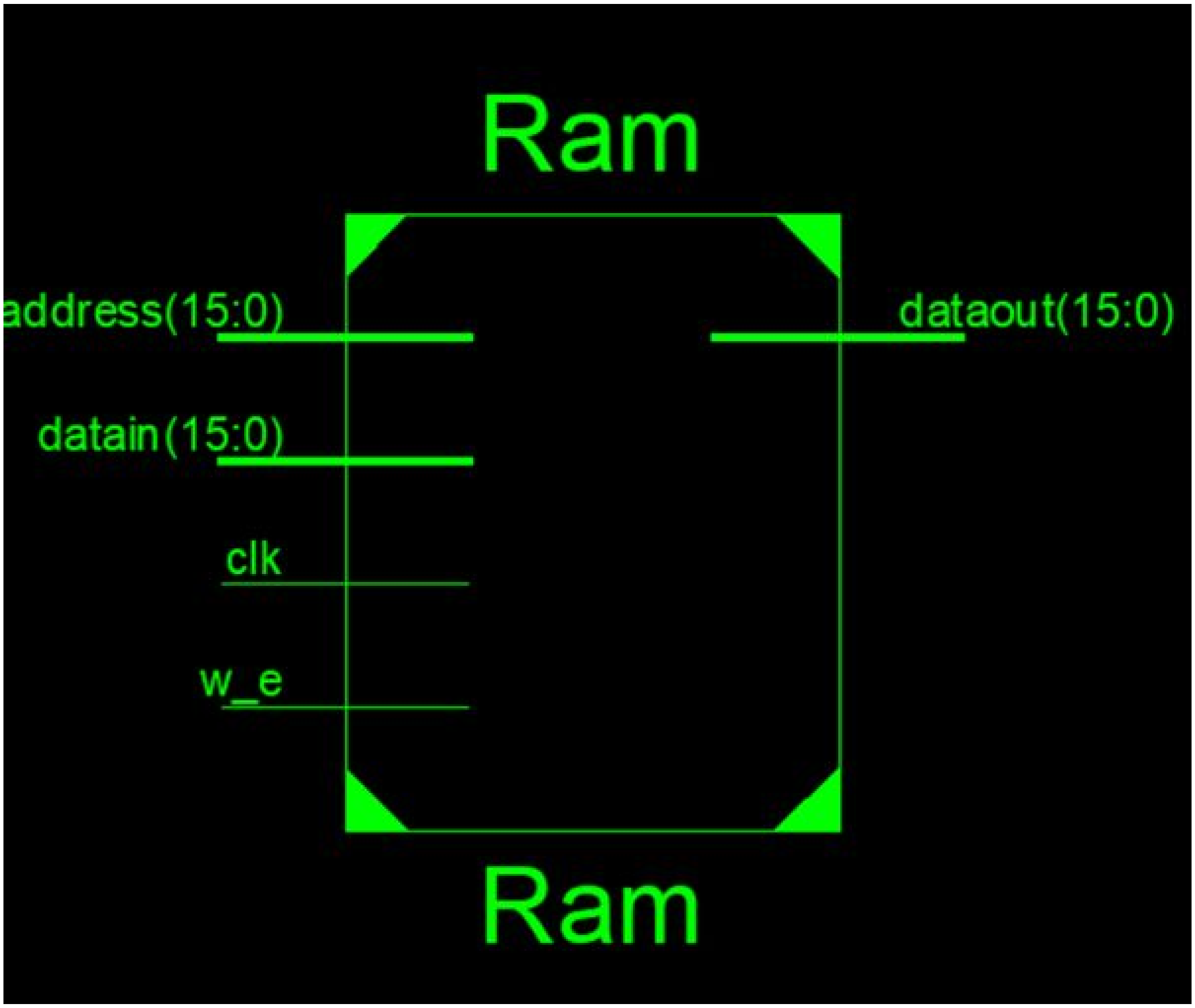
Read_address = 1 start counting from the given address where it gets to the next address after every clock tic.

Read_address = 0 continue counting from the last stored address.

If in the first clock tic it wasn't given an address it will start counting from address.

So that design made it possible to use the go-to (address) functionality.

RAM:



Usage :

Inputs:

w_e-(Write Enable): Type of operation where 1 = write 0 = read .

address: the address to be selected.

datain: The value to be stored only if w_e = 1.

Clock.

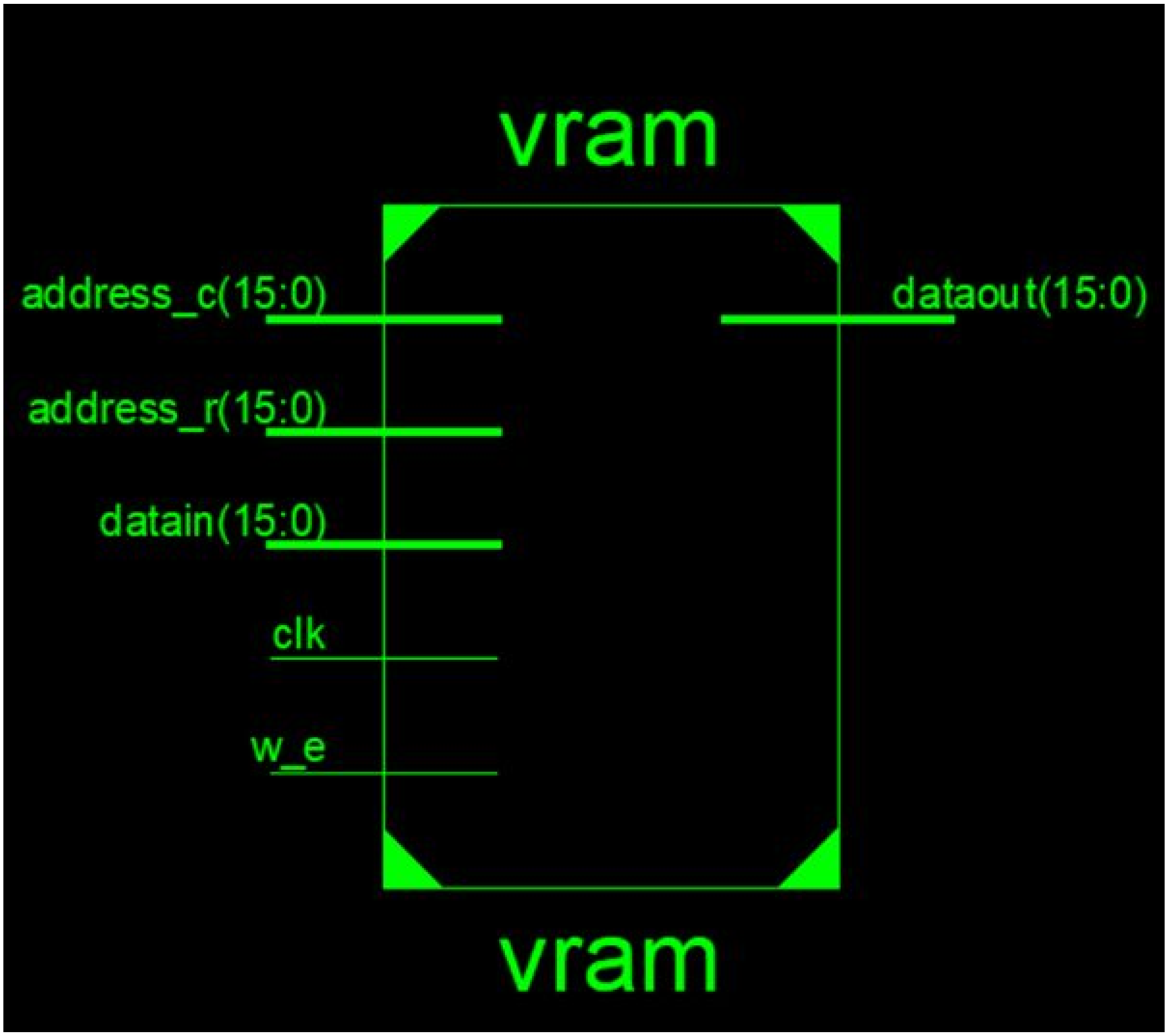
Output:

dataout : returns the word stored in the passed address only if w_e = 0 , if w_e =1 it returns the value passed to datain.

Ram Documentation:

The ram is very simple in terms of design. Where it was designed using block ram implementation with the desired size width and depth. It basically gets to the entered address and returns the stored word if the write enable is off if write enable was high it will save the given datain to the entered address.

Vram



Inputs:

- address_r, address_c which is row and column address.

- Datain which used whenever we need to store data in the vram

- Clock

- W_e which is a write enable to be able to select between the two operations read and write

Output:

- Dataout where the vram returns the value stored in the passed row, column address only if the write enable was 0, while if it was high the dataout will hold the value entered by the user through the datain .

Vram Documentation

Vram is a type of random access memory (RAM) used to store image data for a computer display. It works as a image buffer between the computer processor and the display. It was necessary for us to use as we wanted to display the game. So, in terms of design we had 2 options either we use a double port vram so the processor could still save the frame while the GPU reading it or using 2 Vrams. We chose having 2 Vrams where in each cycle one of them is used to store the new frame from the processor and the other is used by the GPU to display the stored frame in it, and in the next cycle they are switched so that the GPU can access the new frame.