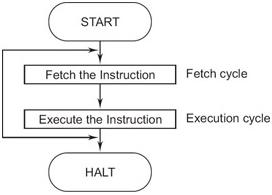
**P9 - Illustrate the use of the different processor registers in the fetch-execute cycle**

**Introduction**

In this report, I will illustrate the different types of registers that are used within a fetch-execute cycle. Then, I will an example for each one.

**Fetch-execute cycle**

Fetch-execute cycle is also known as an instruction cycle. This cycle requires instruction from its memory to work. In addition, it is carried out. This is continuous from once the computer is on until the computer is shut. The Central Processing Unit (CPU) repeats it. The image below shows how the cycle works. It gets the instruction from wherever it is, and it executes it.

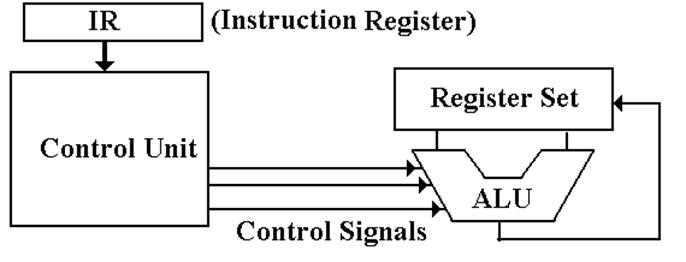


**The general-purpose register**

General-purpose register have several functions for the process. As this is a ‘general’ register, it has more than one job. It could save data as well. Arithmetic operations is one of them, when the converting a binary code. It could be plus, multiplication, division, or subtraction. All of these are important for the register. Each register is different, and they have a different job as well.

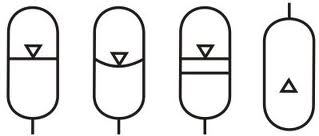
**The instruction register**

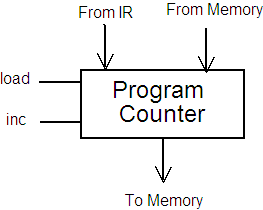
Also abbreviated as IR, this is part of the instruction cycle. Once the instruction has been fetched, it is loaded within the register to execute. Once it is stored within the instruction register, it decodes the information that it has been received. This means that it calculates it and converts it to binary, so that is does the execution can take place. IR is also part of the CPU’s control unit, and it has the same job for the instruction cycle. The image above shows how the instruction cycle works, and the image below shows how the IR register works.



**The accumulator**

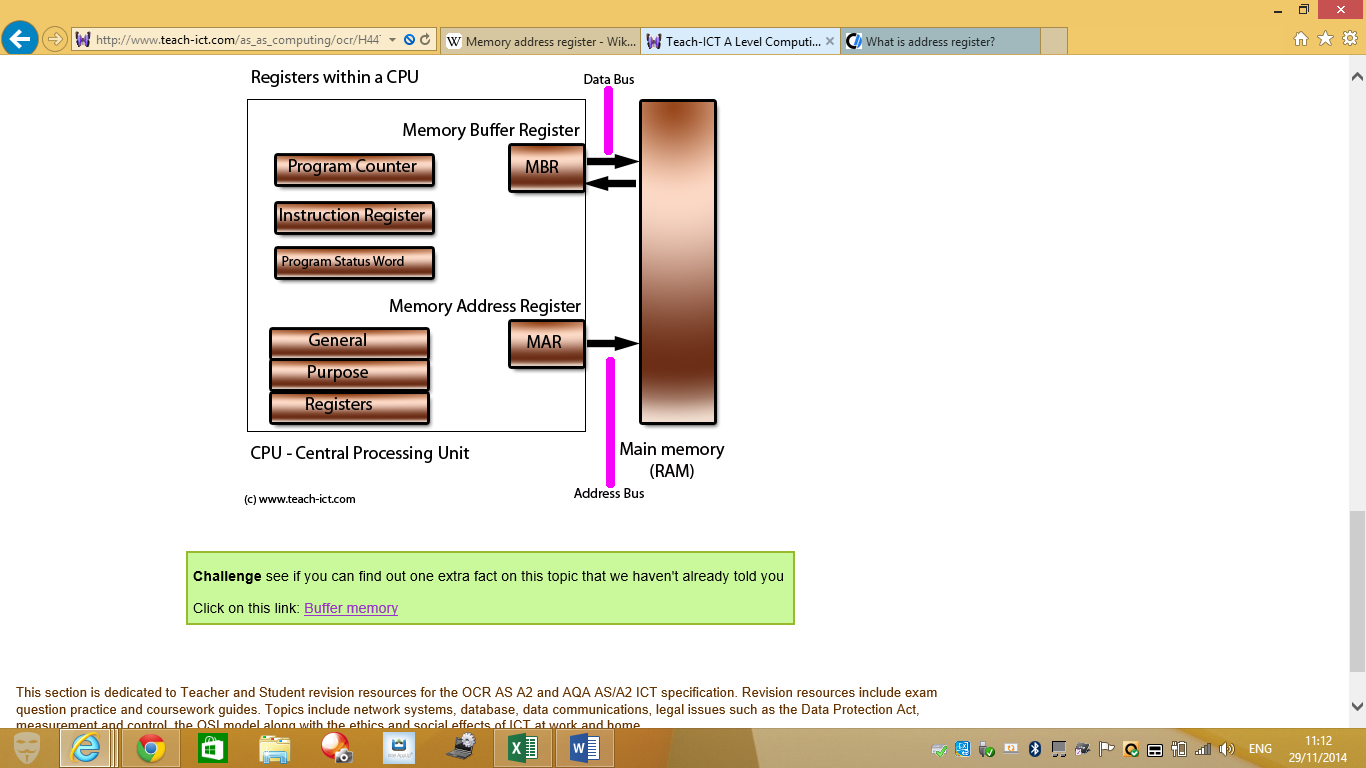
The accumulator makes all the calculations. These calculations includes subtraction, addition, multiplication, and subtraction. Without this register, you have to do it manually. Manually, as in you have to do it yourself, which is long. For example, if a document says ‘Tokyo’ the conversion would be 00010100 00101001 00100101 00110011 00101001. These are many codes, but each letter, has different zeros and ones for it. If it has to be added, it does it for you. This symbol represents an accumulator. Each one is different, as they are many types of them, e.g. hydraulic accumulator.

[](http://www.google.co.uk/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=&url=http://www.e4training.com/D8.html&ei=3FJ4VK2ZBKjQ7Abn7IDoBw&bvm=bv.80642063,d.ZGU&psig=AFQjCNF0ST_pXbHqM0bUTK1GKnctxTKnGA&ust=1417258105258411)



**The program counter**

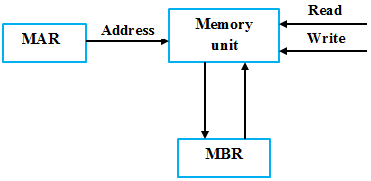
The program counter, also known as instruction pointer, is where it fetches the instruction that it would be executed, and it points to where it will go next. This is important for any computer, because it makes the process much better. It relates to the fetch-cycle, because the process of it is to fetch and execute. This gets it to organise it and tells it where to go.

**Memory Address Register**

Looking at the diagram below, MAR gets the data from the CPU and it holds the information for it to be executed, or stored to the memory unit. It keeps doing that, and that is the only job for it. This image shows the registers that are used within the CPU. The MAR gets ready to store it within the main memory (RAM). It has to communicate with the CPU and memory for its job to be done.

**Memory Buffer Register**

Memory buffer register (MBR) is a register that stores data from the access point and to the access point. It stores it in the memory address register. It works by the content sent to the MBR, from there it goes to the location, and before it, it must be processed in the ALU (all of the calculations). As you can see in the diagram below, it shows that it stores in the memory unit. It goes from the MAR and then to the MBR. The data is stored within the memory unit, and it goes to all of the others before it stores temporarily.



**Reference**

<http://en.wikipedia.org/wiki/Instruction_cycle>

<http://www.worldofcomputing.net/processor/general-purpose-registers.html>

<http://en.wikipedia.org/wiki/Instruction_register>

<http://en.wikipedia.org/wiki/Accumulator_(computing)>

<http://en.wikipedia.org/wiki/Program_counter>

<http://en.wikipedia.org/wiki/Memory_address_register>

<http://en.wikipedia.org/wiki/Memory_buffer_register>

<http://www.worldofcomputing.net/processor/general-purpose-registers.html>