Chapter 4 Instruction Sets and the Processor organizations

Lesson 01:

General-Purpose Register Organization

Objective

- Learn a GPRs based organization

Processor Organisation

$$\begin{tikzcd} & \text{Processor Organisation} \arrow[dl] \arrow[d] \\ \text{GPR based Organisation} & \text{Stack Based Organisation} \end{tikzcd}$$

GPR Means General Purpose register

GPR based organization difference with Stack-based organization

- Program assumes that there are no registers
- Stack based organization has the feature that the register file is invisible to the program
- The operations are done at the top level of an invisible stack of registers

GPR based Organisation

GPR Based organization in a Processor

- In GPR based organizations, the register file and the memory are completely independent
- Programs are responsible for moving data between these two types of storage as necessary

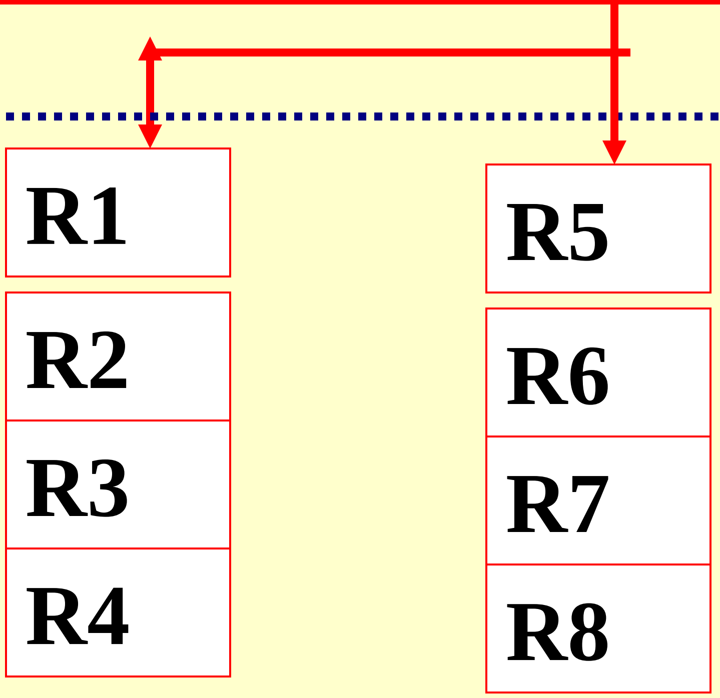
GPR Based organization in a Processor

- The processor register file— a random-access device where each register can be independently read and written by the processor

GPR

- A GPR in the register file is random access, which means that the general-purpose register file allows an instruction to access the registers in any order by specifying the number (also called the register ID) of the register to be accessed, much like the memory system allows the addresses in the memory to be accessed in any order

GPRs' Set (File)



Register Set (File)

- A programmer can choose which values to keep in the register file at any time and can use the value in the GPR(s) of the register file as the input to any instruction

GPR

- Significant difference between the general-purpose register file and stack— is that reading the contents of a general-purpose register does not change them, unlike popping values off of a stack
- Successive reads of a general-purpose register with no intervening writes will return the same result, while successive stack pops will return the contents of the stack in LIFO order

Special meanings to some of the registers

- For example, some processors hard-wire register 0 (r0) with the value 0 to make it easier to generate this common constant
- 80x96 processor r0, register at 0x0000 in memory
- ARM assigns r15 to Program Counter
- r14 to Link Register (for return to the Program on call)

General-purpose register organization

- Dominant in more recent years because of improvements in technology and the switch to high-level languages
- As memory capacities have increased and memory costs have decreased, the amount of space taken up by a program has become less important, making stack-based organization's advantage in instruction size less important

Compilers in a GPR based Processor

Compilers for GPR organizations

- Generally able to achieve better performance with a given number of general-purpose registers than they are on stack-based organizations with the same number of registers
- The compiler can choose which values to keep in register file and can choose any value as input to any instruction

Compilers for GPR organizations

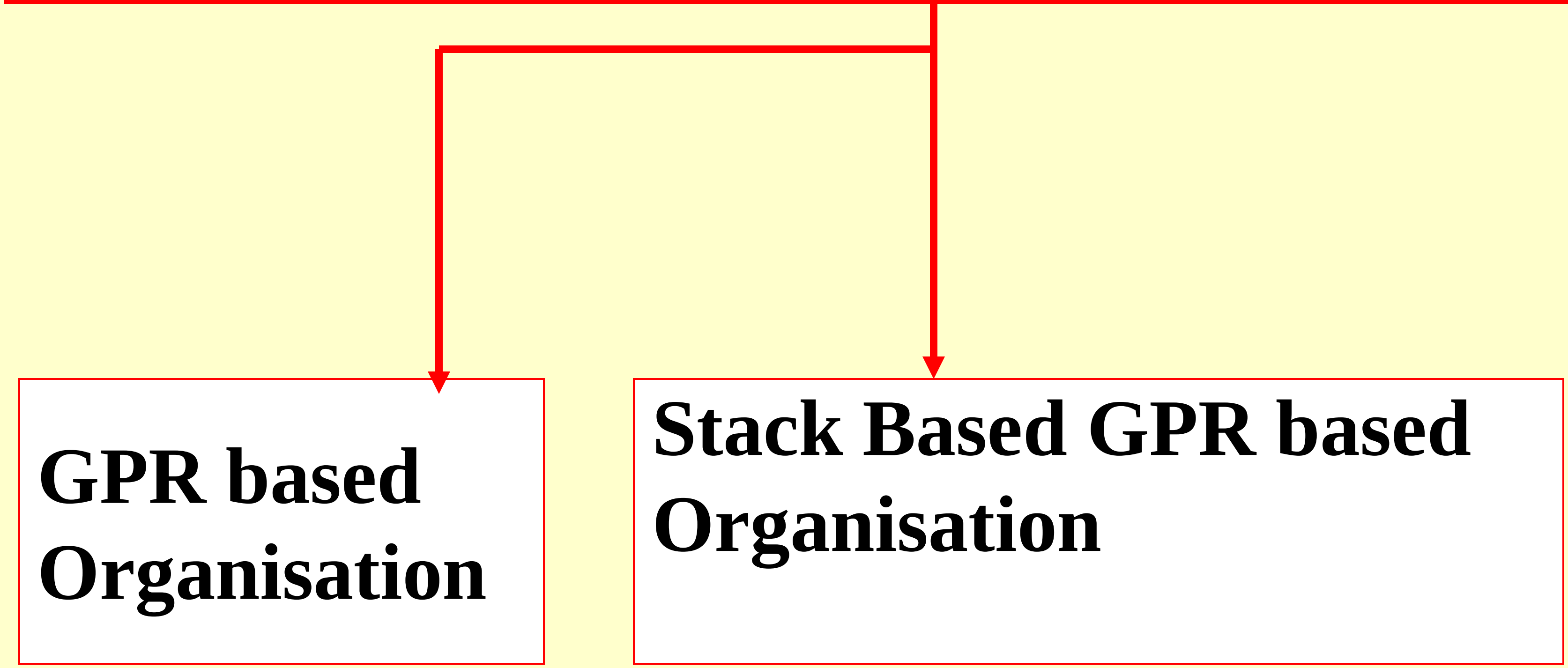
- Because of their performance advantages and the decreasing importance of code size, virtually all recent-workstation processors have been GPR architectures

Summary

We Learnt

- GPR based Organisation the register file and memory are completely independent
- Programs are responsible for moving data between these two types of storage as necessary and from one register to another

Processor Organisation



End of Lesson 01 on

General-Purpose Register Organization