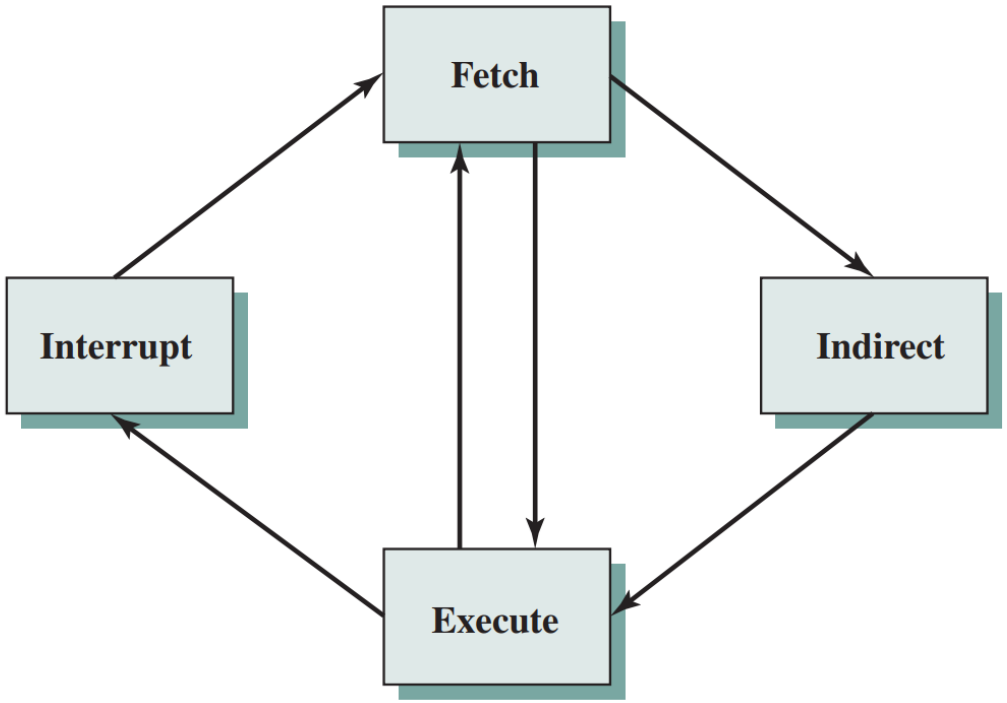
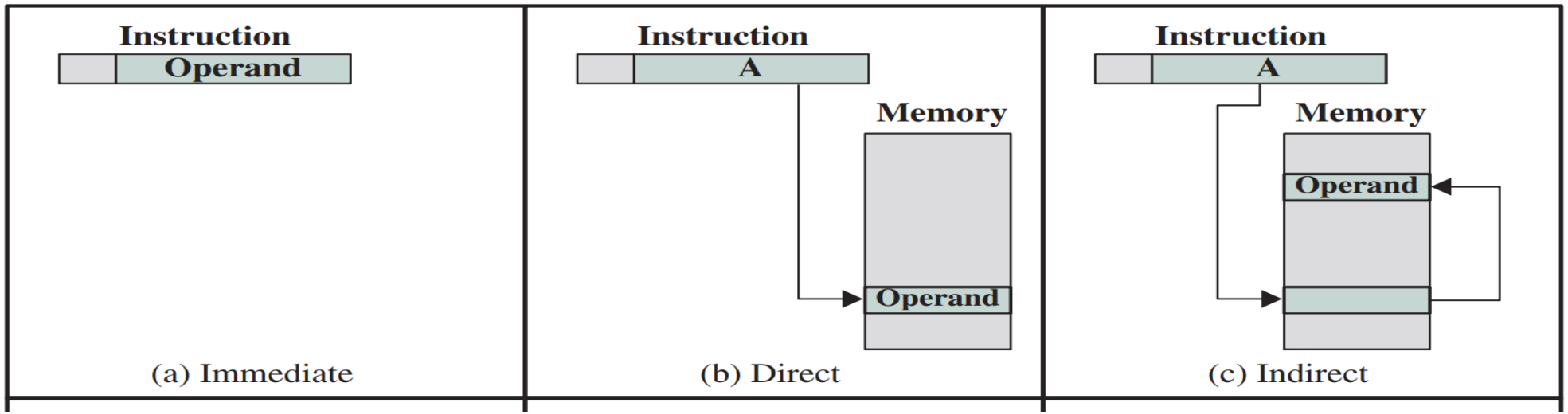


Figure 14.4 The Instruction Cycle


Instruction
Operand
(a) Immediate
Instruction
A
Memory
Operand
(b) Direct
Instruction
A
Memory
Operand
(c) Indirect

During the *fetch cycle*, an instruction is read from memory. Figure 14.6 shows the flow of data during this cycle. The PC contains the address of the next instruction to be fetched. This address is moved to the MAR and placed on the address bus. The control unit requests a memory read, and the result is placed on the data bus and copied into the MBR and then moved to the IR. Meanwhile, the PC is incremented by 1, preparatory for the next fetch.

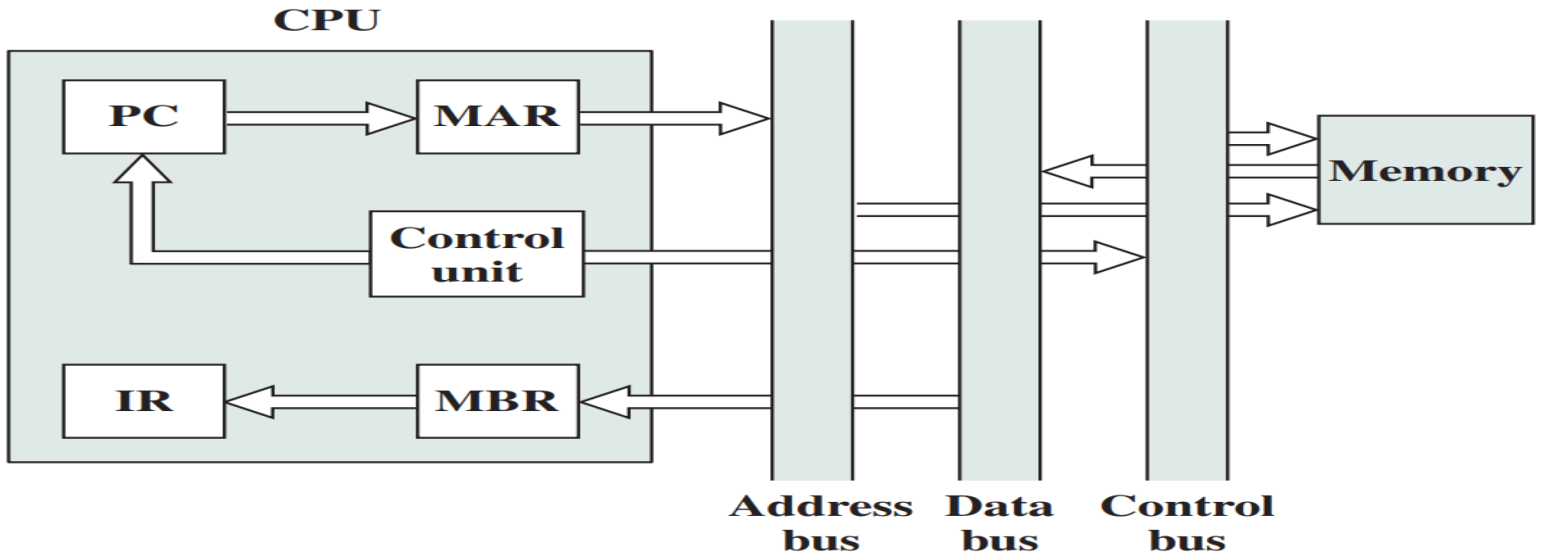


Figure 14.6 Data Flow, Fetch Cycle

indirect cycle is performed. As shown in Figure 14.7, this is a simple cycle. The right-most *N* bits of the MBR, which contain the address reference, are transferred to the MAR. Then the control unit requests a memory read, to get the desired address of the operand into the MBR.

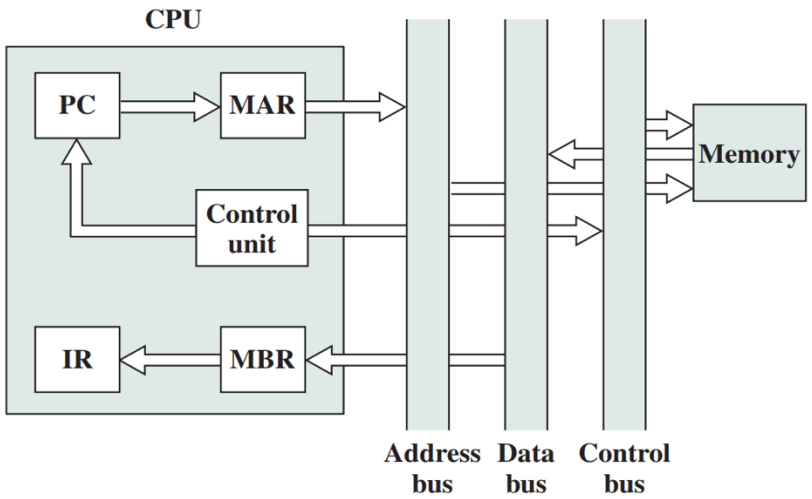


Figure 14.6 Data Flow, Fetch Cycle

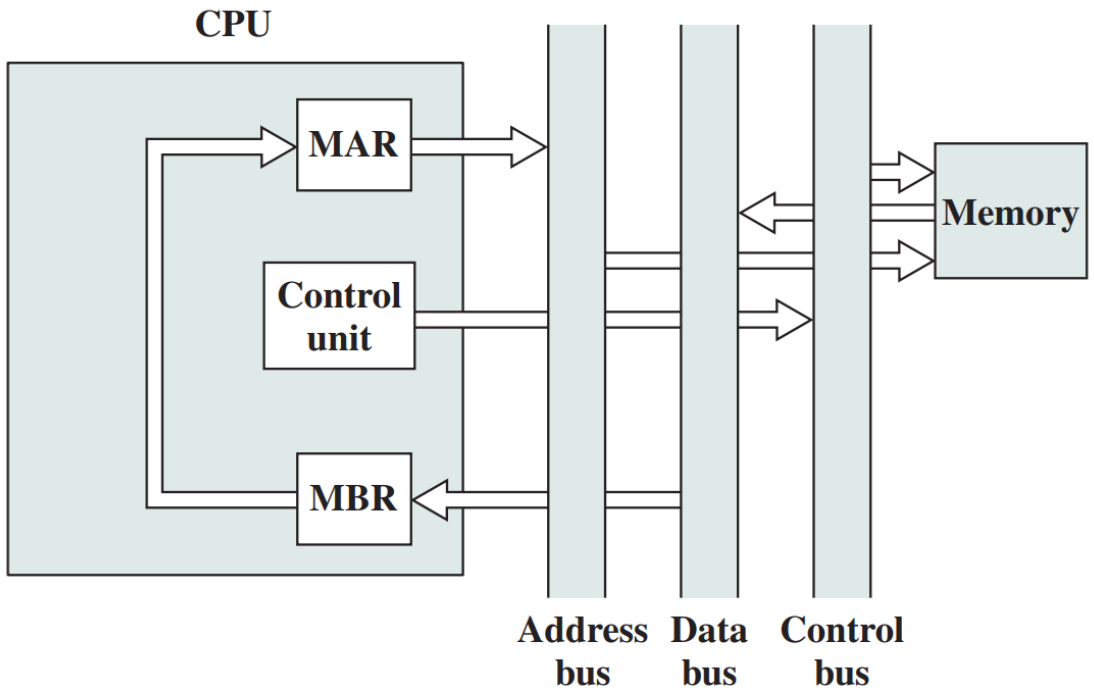
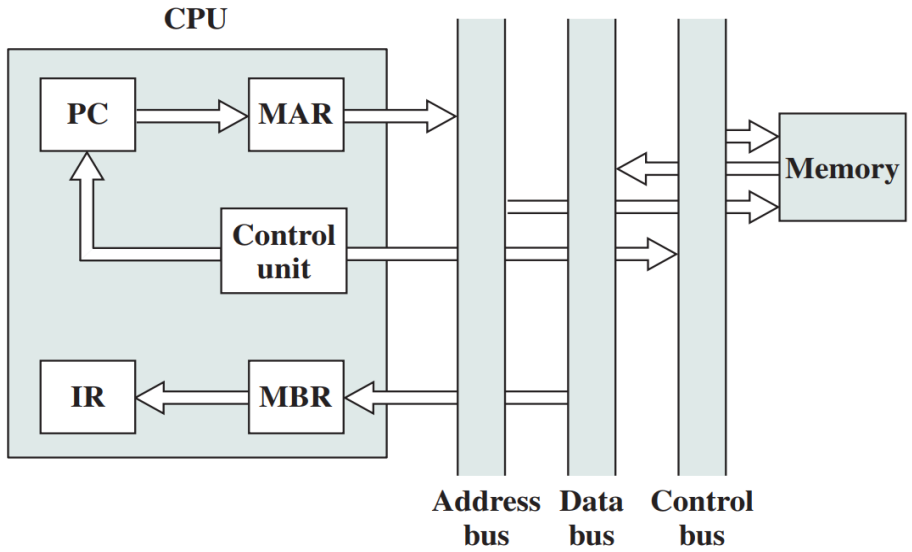


Figure 14.7 Data Flow, Indirect Cycle

Like the fetch and indirect cycles, the *interrupt cycle* is simple and predictable (Figure 14.8). The current contents of the PC must be saved so that the processor can resume normal activity after the interrupt. Thus, the contents of the PC are transferred to the MBR to be written into memory. The special memory location reserved for this purpose is loaded into the MAR from the control unit. It might, for example, be a stack pointer. The PC is loaded with the address of the interrupt routine. As a result, the next instruction cycle will begin by fetching the appropriate instruction.



MBR = Memory buffer register
MAR = Memory address register
IR = Instruction register
PC = Program counter

Figure 14.6 Data Flow, Fetch Cycle

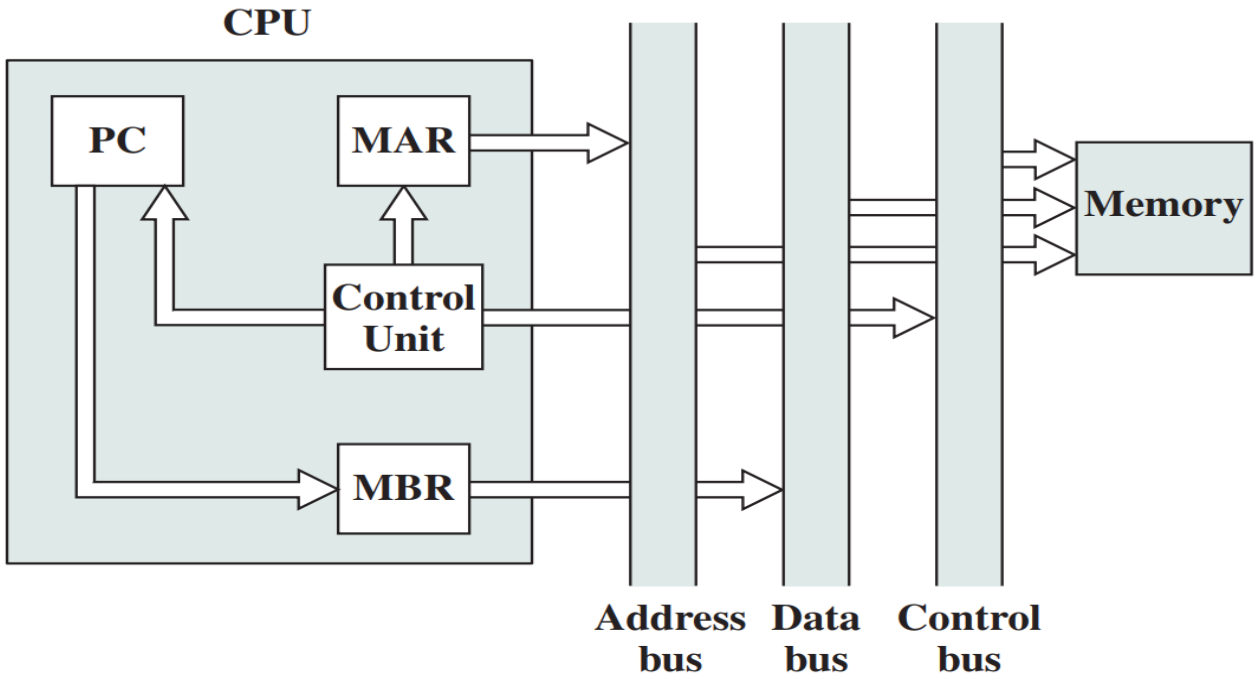


Figure 14.8 Data Flow, Interrupt Cycle