# Control Design.

-CA

## Introduction:

The datapath is a network of functional and storage units capable of performing certain operations on data words. The purpose of the control unit is to issue control signals to the datapath. These control signals enter the datapath at "control points", where they select the functions to be performed at specific times and route the data through the appropriate parts of the datapath unit.

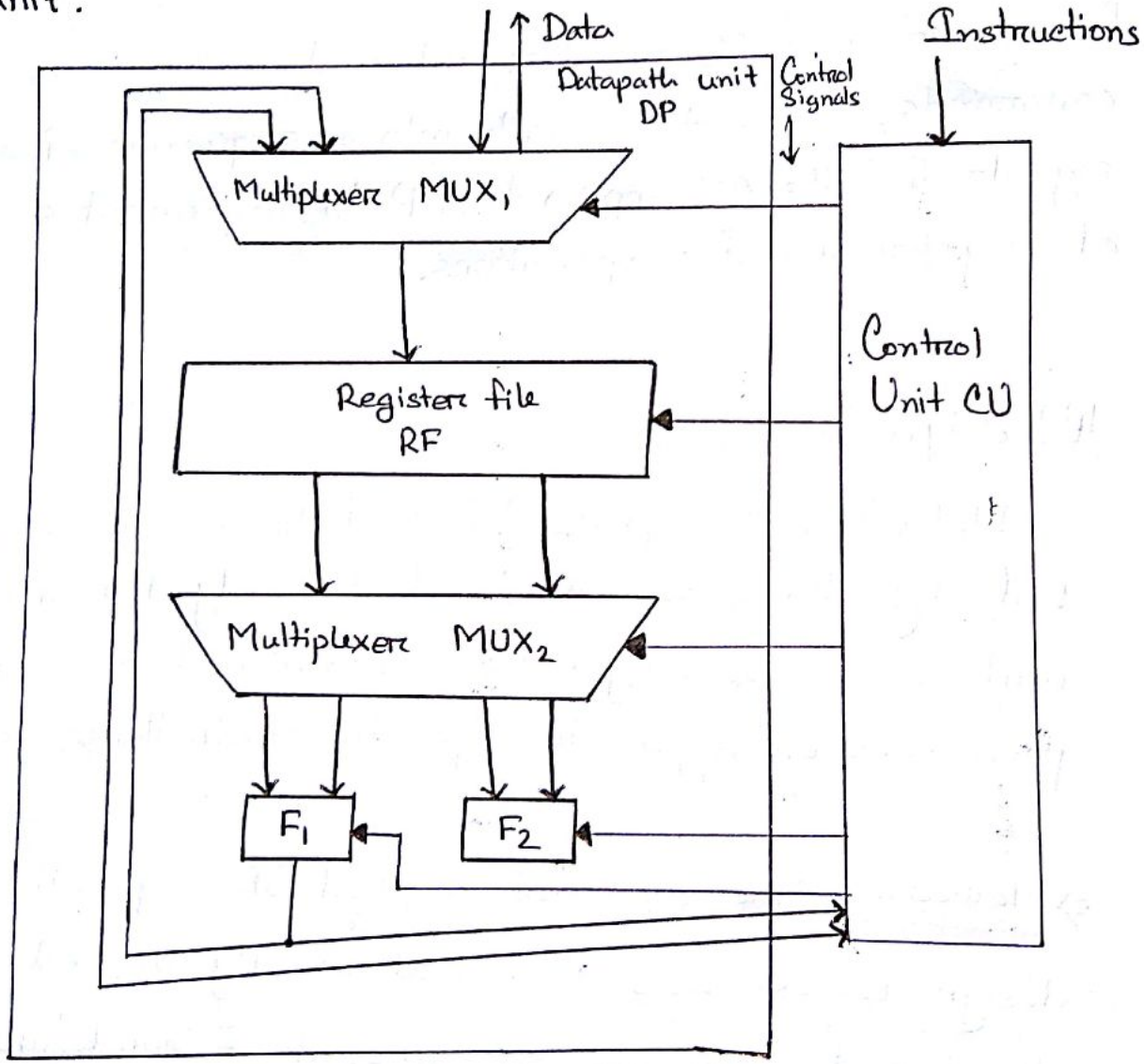


Fig: Processor composed of a datapath unit DP and a control unit CU.

A CPU's datapath contains to perform "arithmetic" and "logical" operations on words such as fixed-point or floating-point numbers. The internal structure of the datapath circuit DP of a small microprocessor is depicted in figure. It contains a register file RF for temporary storage of operands, two functional unit $F_1$ and $F_2$ responsible for data processing and multiplexers to allow the data to be steered through DP.

The control unit CU receives external instructions or commands, which it converts into a sequence of control signals that the CU applies to DP to implement a sequence of register-transfer operations.

## Design Methods:

Methods are representative of those used in practice, but by themselves are suitable only for small control units such as might be encountered in simple RISC processors or application-specific controllers, given below:

⇒ Method 1: The classical method of sequential circuit design, which was discussed briefly in 2nd chapter. It attempts to minimize the amount of hardware in particular, by using only $\lceil \log_2 p \rceil$ flip-flops to realize a p-state circuit.

Method 2: An approach that uses one flip-flop per state and is known as the "one-hot method". While expensive in terms of flip-flops, this method simplifies CU design and debugging.

## State table:

The behavior required of a control unit, like that of any finite-state machine, can be represented by a "State table".

State tables for a finite-state machine:

a) Mearly type.
b) Moore type.

## GCD Algorithm:

We use a variant of Euclid's algorithm

```
gcd(in:X,Y; out:Z);
    register XR, YR, TEMPR;
    XR := X;        (Input the data)
    YR := Y;
    while XR>0 do begin
        if XR≤YR then begin    (Swap XR and YR)
            TEMRR := YR;
            YR := XR;
            XR := TEMPR; end
            XR := XR-YR;       (Subtract YR from XR)

    end
    Z := YR;    (Output the result)
end gcd;
```

## Classical Method:

The major steps of the classical design method are as follows:

1. Construct a P-row state table that defines the desired input-output behavior.

2. Select the minimum number p of D-type flip-flops and assign a p-bit binary code to each state.

3. Design a combinational circuit C that generates the primary output signals $\{Z_i\}$ and the secondary outputs $\{D_i\}$ that must be applied to the flip-flops.

## One-hot Method:

While the classical design method minimizes a control unit's memory elements, its effect on the amount of combinational logic C is less obvious. Furthermore, control units designed by this technique tend to have a complicated, "random" structure, which makes design debugging and subsequent maintenance of the circuit difficult. An alternative approach that simplifies the design process and gives C a regular and predictable, is the "one-hot method", so called because its binary state assignment always contains a single 1- the "hot" bit - while all the remaining bits are 0.

## Design of a DMA Controller:

This problem, which is adapted from is representative of control units that link several interacting systems - in this case, main memory and a set of IO devices. The target machine is the control part of a four-channel DMA controller of the kind found in the IO subsystem of most computers. It is a six-state Moore-type machine with four input and five output signals, which are identified as follows:

Inputs: IOREQ - Any of four data-transfer signals.
CONT - Continue (indicates pending, unprocessed requests)
MACK - Memory transfer acknowledgement
PBGINT - Processor bus grant (indicates availability of data transfer bus)

Output:

CE - Count enable (bookkeeping function)
CMREQ - Channel Memory request
CNTLD - Counter load (bookkeeping function)
RLD - Register load ( " " )
PBREQ - Processor bus request for control of data-transfer bus.