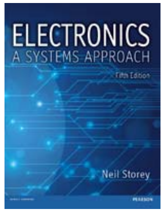
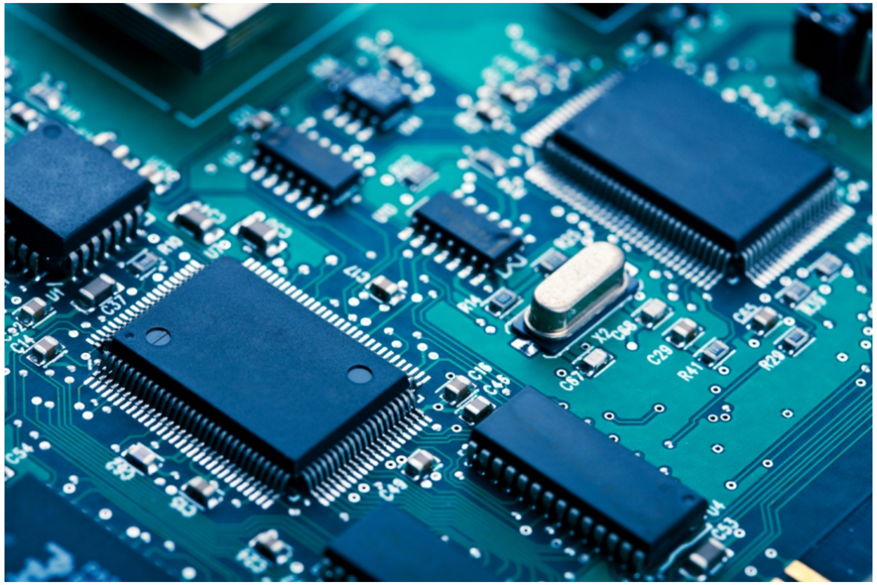


Chapter 26

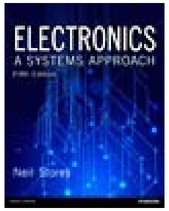
Last time: Digital devices



- Introduction
- Gate characteristics
- Logic families
- TTL
- CMOS
- Interfacing
- Noise and EMC in digital systems

Key points

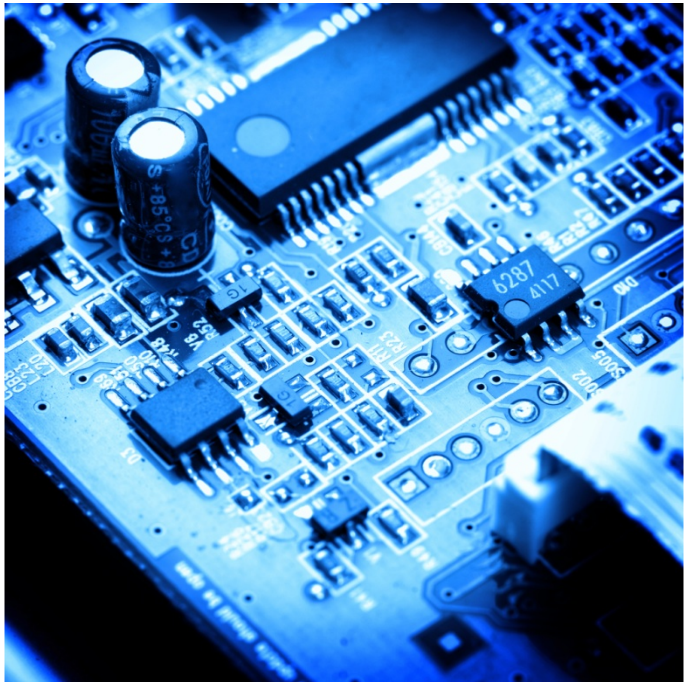
- Physical gates are not ideal components
- Logic gates are manufactured in a range of logic families
- The ability of a gate to ignore noise is its ‘noise immunity’
- Both MOSFETs and bipolar transistors are used in gates
- All logic gates exhibit a propagation delay
- The most widely used logic families are TTL and CMOS.
- Both TTL and CMOS gates are produced in a range of versions, each optimised for a particular characteristic
- Interface circuitry may be needed to link devices of different families
- Noise and EMC issues must be considered during design



Chapter 27

Implementing digital systems

- Introduction
- Array logic
- Microprocessors
- System-on-a-chip devices
- Selecting an implementation method



Video 27B

- **Programmable logic array (PLA)**
  - – has an array of inverters, AND gates and OR gates
  - – can implement any logic function (given limits on numbers of inputs and outputs)

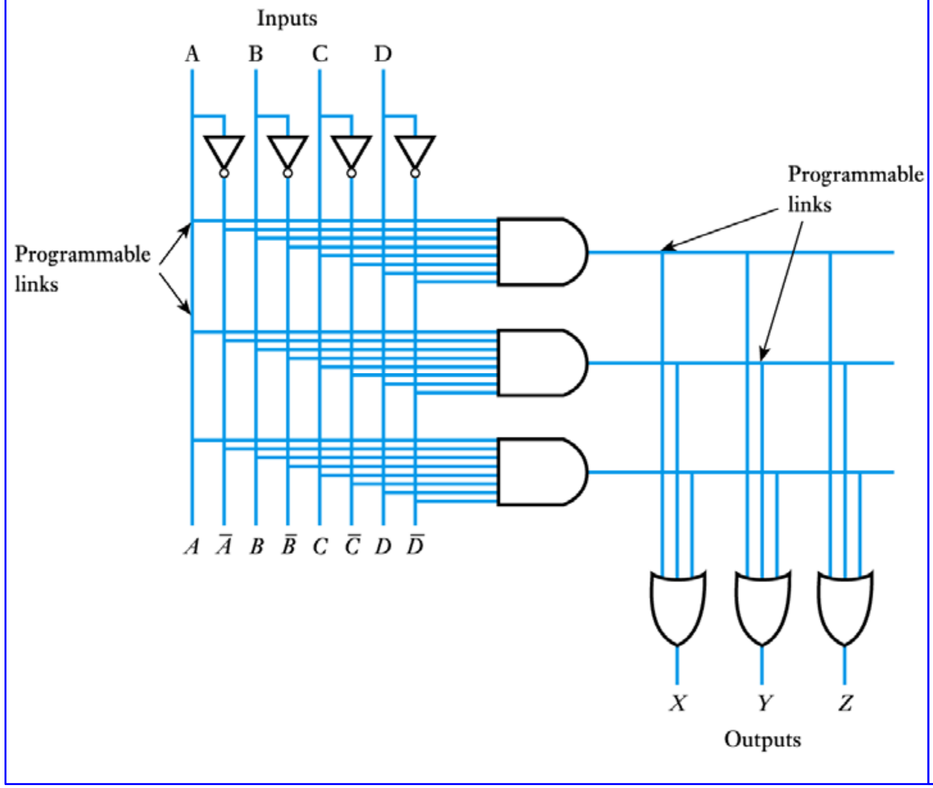
Example: consider a system with four inputs *A*, *B*, *C* and *D* and three output *X*, *Y* and *Z*, where

$$X = \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}CD$$

$$Y = \overline{A}\,\overline{B}CD + ABC\overline{D}$$

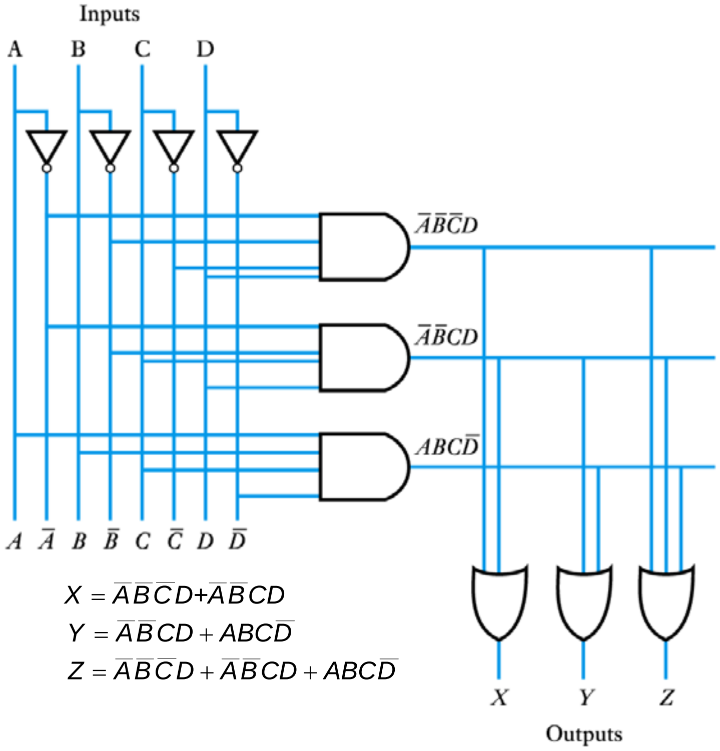
$$Z = \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}CD + ABC\overline{D}$$

— the structure of a simple PLA



Inputs
A
B
C
D
Programmable links
Programmable links
A $\overline{A}$ B $\overline{B}$ C $\overline{C}$ D $\overline{D}$
X
Y
Z
Outputs

—Programmed by blowing fusible links



Inputs
A
B
C
D
$\overline{A}\overline{B}\overline{C}D$
$\overline{A}\overline{B}CD$
$ABC\overline{D}$
A $\overline{A}$ B $\overline{B}$ C $\overline{C}$ D $\overline{D}$
X
Y
Z
Outputs

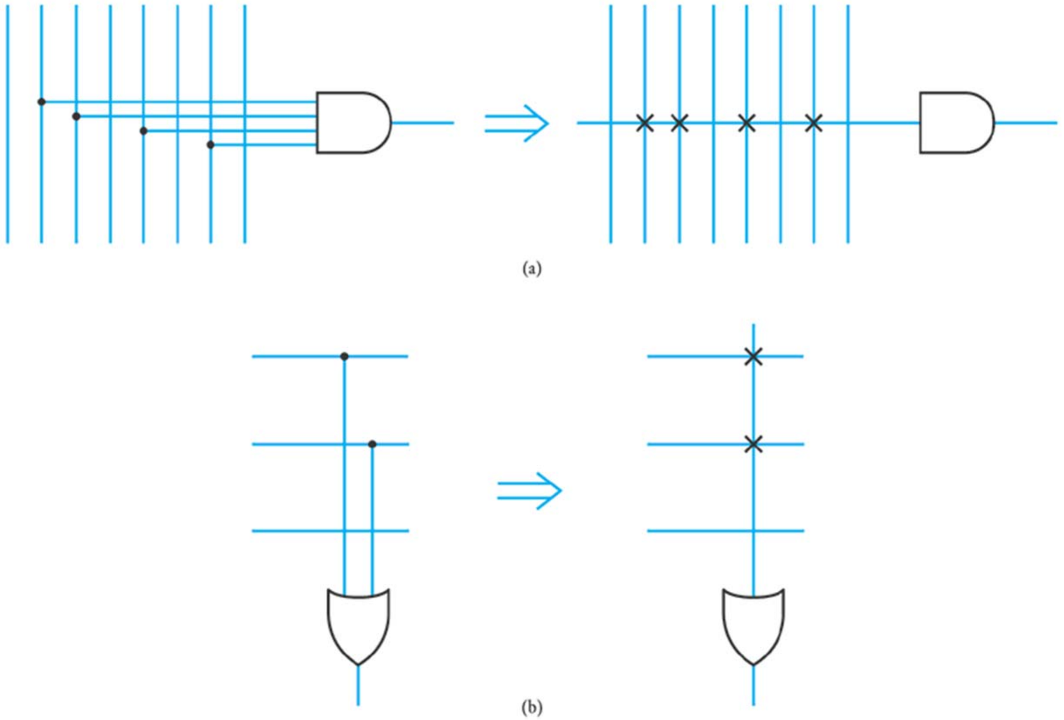
$$X = \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}CD$$

$$Y = \overline{A}\,\overline{B}CD + ABC\overline{D}$$

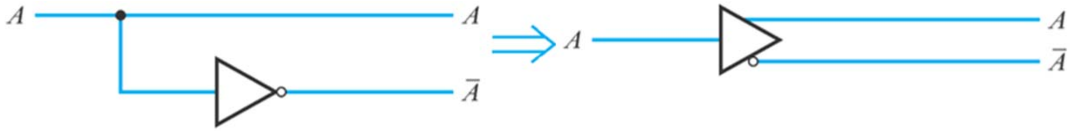
$$Z = \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}CD + ABC\overline{D}$$

Neil Storey, *Electronics*: *A Systems Approach,* 5th Edition © Pearson Education Limited 2013

– logic array symbolic notation

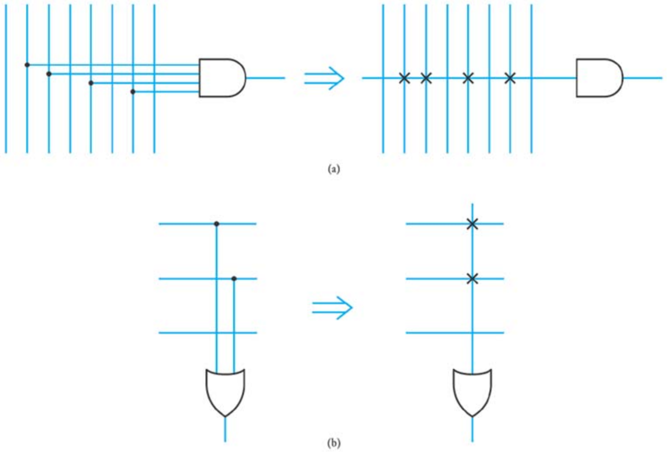

(a)

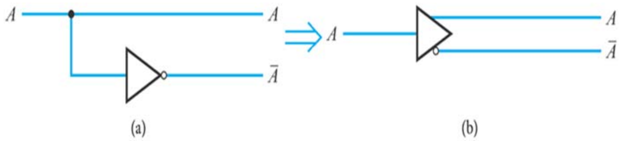
(b)

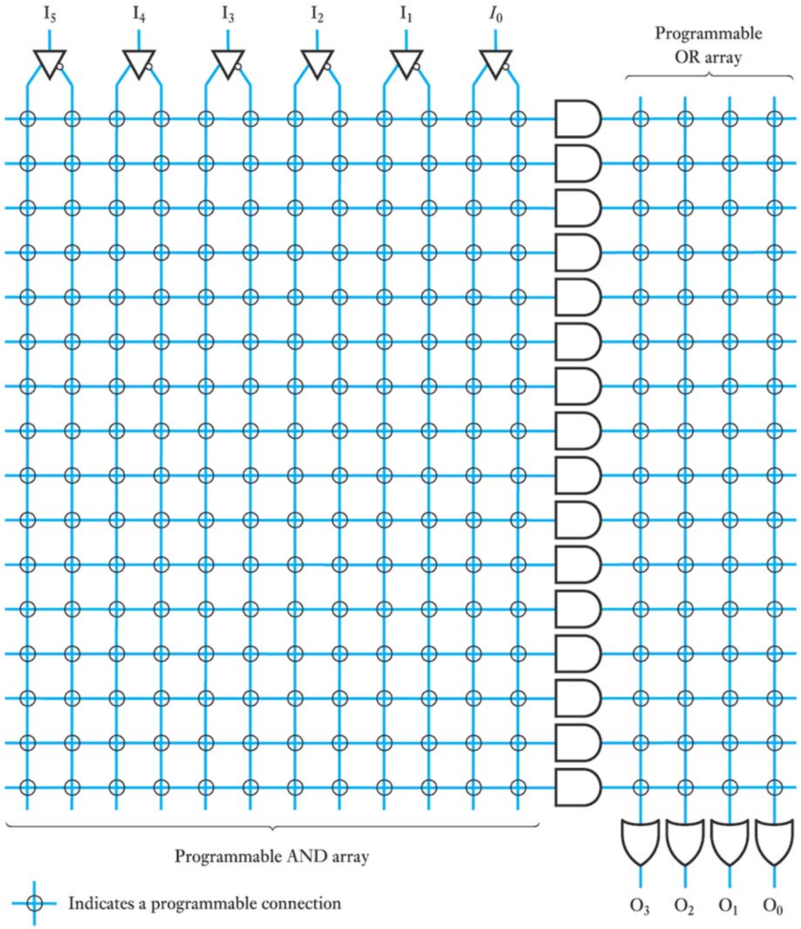


(a) (b)

- a PLA with 6 inputs, 4 outputs and 16 product terms
- where

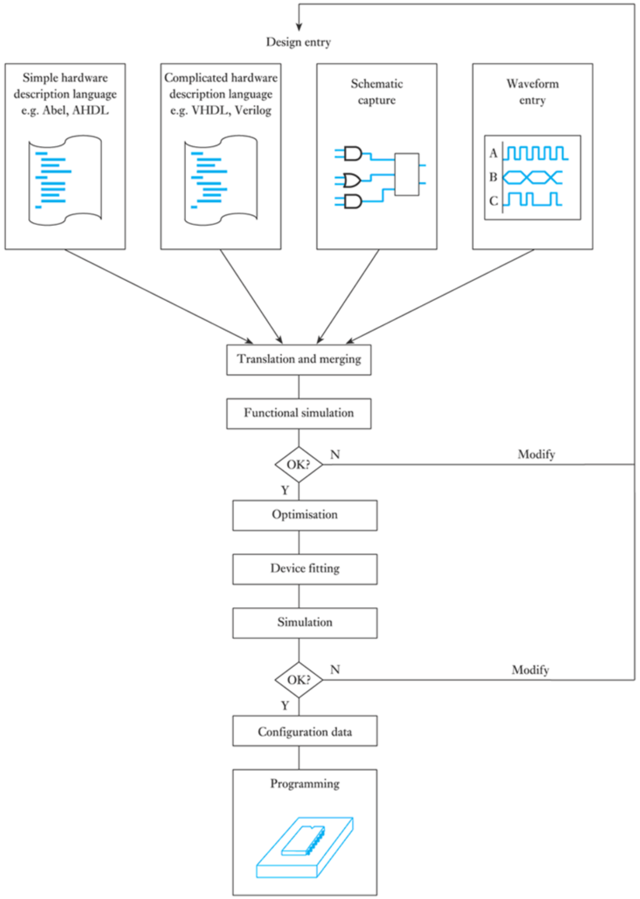






- **Programmable read-only memory (PROM)**
  - – similar structure to a PLA but has only 1 programmable array (NANDS as in the PAL) but fixed OR output array
  - – the pattern written into the OR array determines the outputs produced for each combination of inputs
  - – can be used to implement logic functions or to store **data** – when storing data the inputs are the **address**

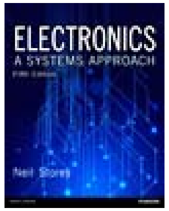
- **Programming tools for array logic**
  - automated tools are used in almost all cases
  - often make use of **hardware description languages**
  - fuse maps are passed to a **programmer** to configure the device



- **Custom and semi-custom ICs**
  - Some equipment may contain a **custom IC** designed by the manufacturer
  - Others may have **semi-custom** or **application specific integrated circuits** (**ASIC**)
    - produced by combining a number of standard cells (such as registers, counters, input/output circuitry and memory)
    - much less costly than a complete design from scratch
- PAL Easy to program and very adaptable
- Very difficult to find out what they do if you haven’t done the programming!

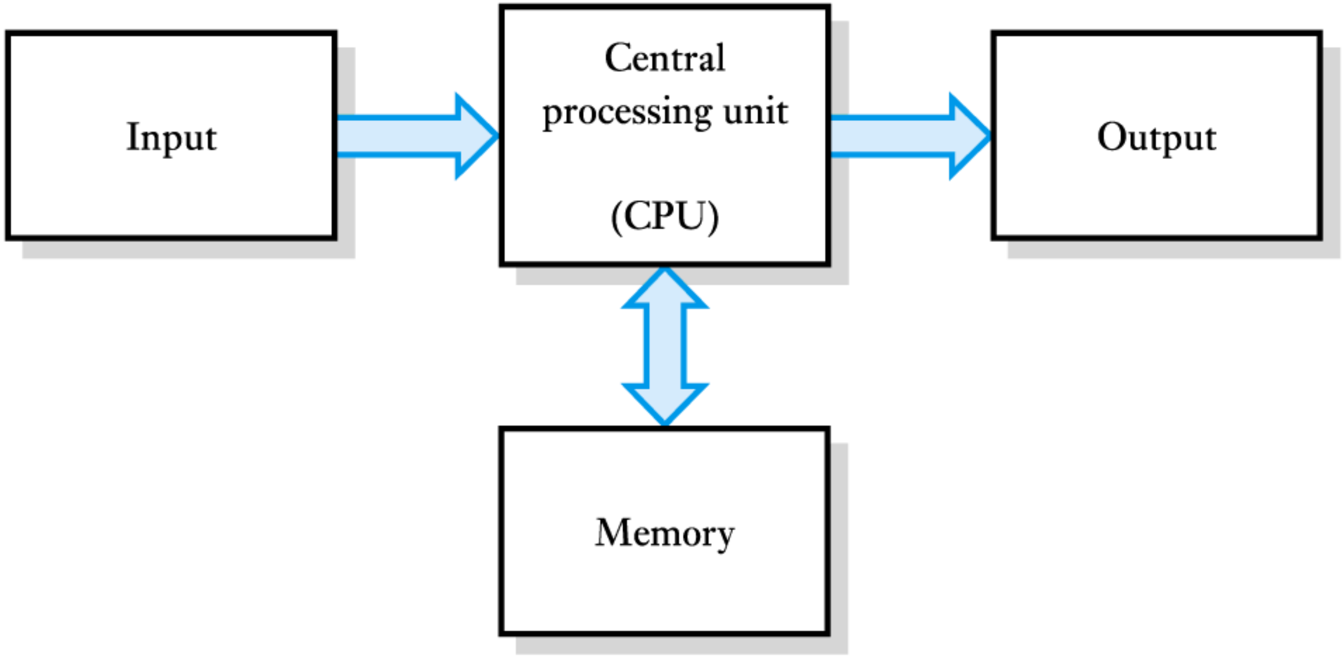
Microprocessors

Video 27E

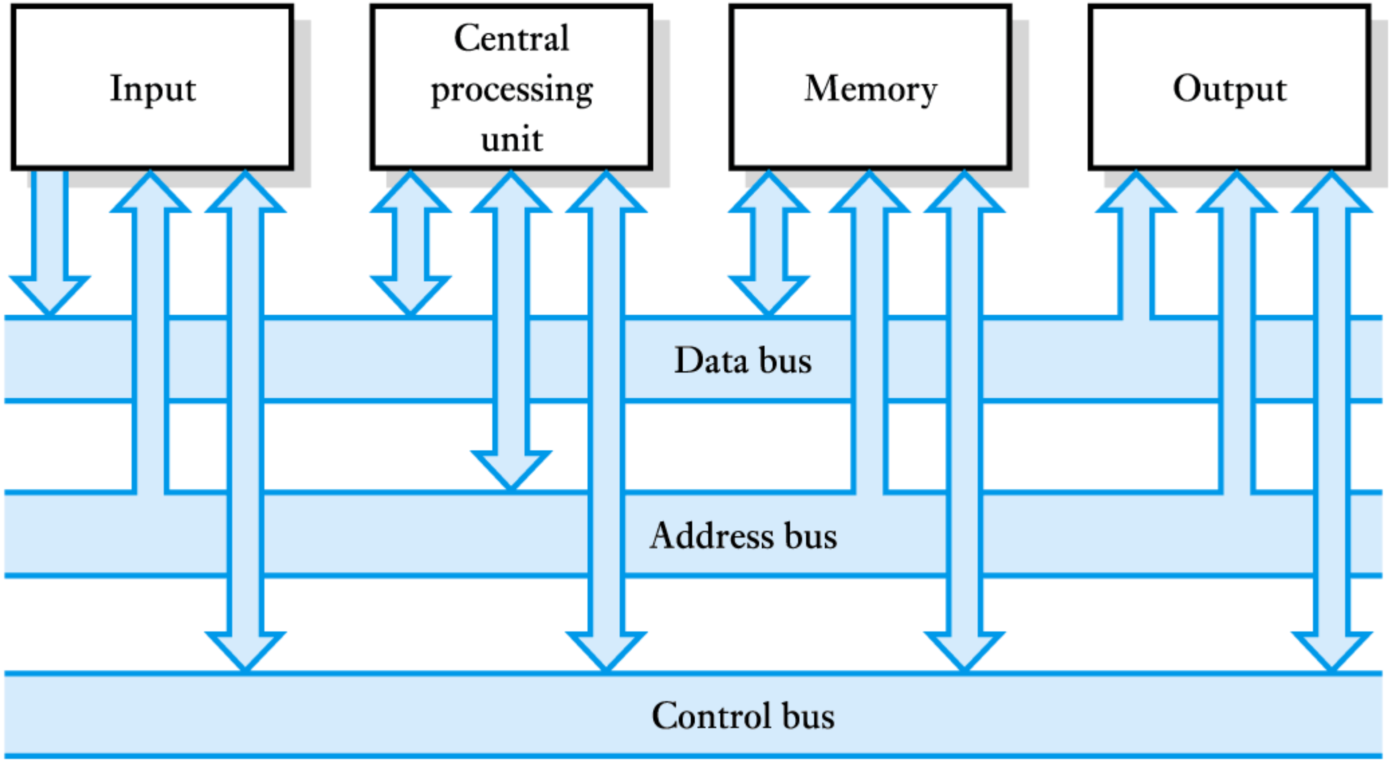


27.3

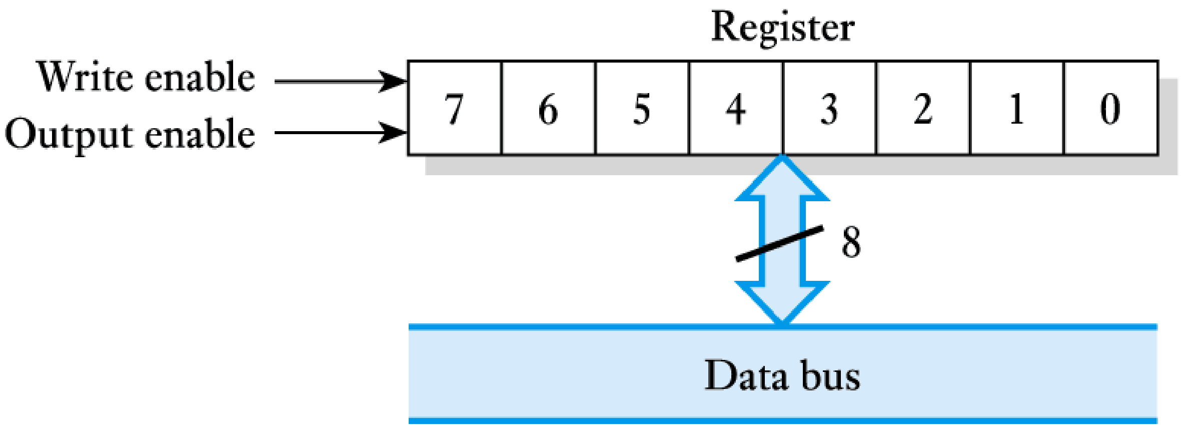
- **A microcomputer system**
  - the CPU takes the form of a **microprocessor**



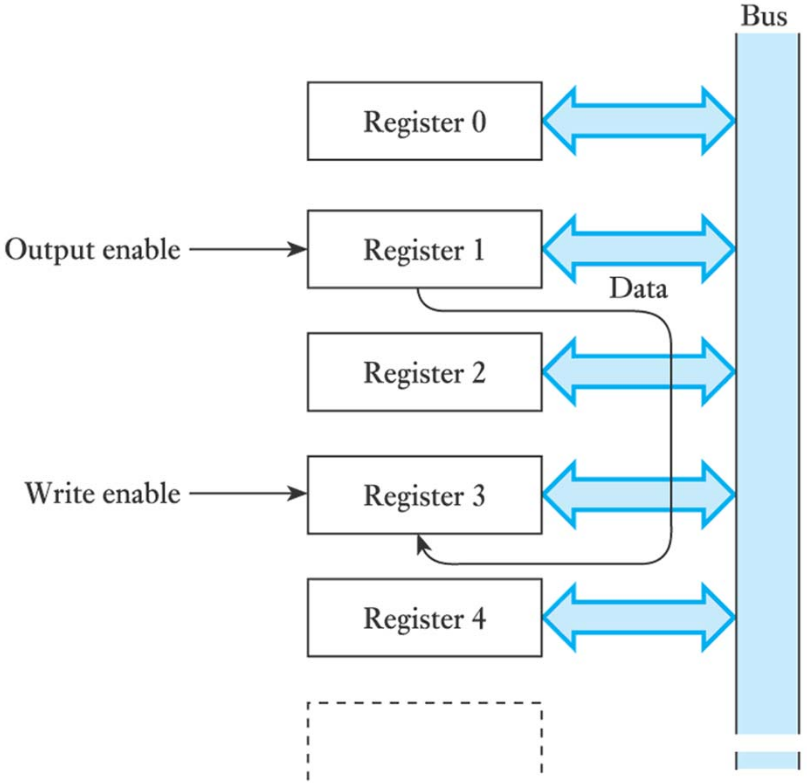
- **Communication within the microcomputer**



- **Registers**
  - – fundamental building blocks within computers
  - – can be constructed using D flip-flops
  - – some are used for storage, others for input/output



- **Communications between registers**
  - achieved by enabling the output of one register and the input of another
  - as all the registers are connected by the same data bus, only one piece of information can be transmitted at any time



- **Random access memory (RAM)**
  - this is *read-write* memory
  - *write* describes the process of storing information
  - *read* describes the process of retrieval
  - RAM is **volatile** in nature
  - several forms:
    - static RAM – uses circuitry similar to a bistable
    - dynamic RAM – uses charge on capacitors, needs refreshing
  - battery backup can be used to provide **non-volatility**

- **Read-only memory (ROM)**
  - this can be read from, but not written to
  - is inherently non-volatile (useful for programs, etc)
  - many forms available
    - some are programmed by the manufacturer (such as **masked programmed** devices)
    - others are user programmable (such as **EPROM**, and **EEPROM**)
  - memory such as EEPROM can be written to (programmed) as well as read, but it is *not* RAM
    - it can only be programmed relatively slowly

Microcomputer Programming

- Programming can use a number of techniques:
- Machine code programming
  - Coding directly in a machine readable form – very inefficient and hardly ever used.
- Assembly code programming
  - Allows direct control of the processors functions while being easier to perform than machine code techniques. Sometimes used where very small or very fast routines are required.
- High-level language programming
  - The most widely used and efficient method. Uses languages such as BASIC, C, C++, C#, Java, Pascal or Python.

Input/Output

Video 27F

- **Interface**
  - Make the signals produced by the sensor compatible with those of the computer system.
  - Make signals produced at the computer output suitable to drive the actuators directly.
- The operations performed by the interface referred to as **signal conditioning**.

Signal conditioning

- Signal conditioning includes:
  - Amplification (Chapter 14)
  - Filtering to remove noise (Chapter 8)
  - Isolation (Optical isolator Chapter 26.8.3)
  - Analogue-to-digital and/or digital-to-analogue conversion (Chapter 28)

Input/output

- The nature of the input/output section varies tremendously with the application.

- Input/output may use parallel or serial techniques.

- **Parallel I/O** often uses input/output *registers* which appear as simple *memory locations* to the processor

- **Serial I/O** can use a range of techniques including both synchronous and asynchronous methods.

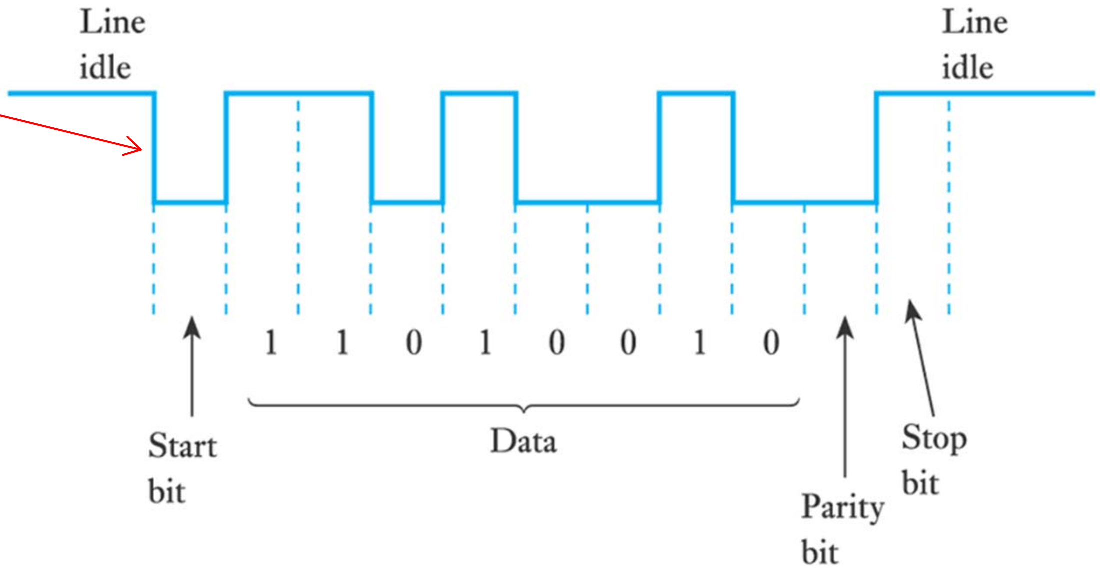
Serial I/O

- **Asynchronous serial communications**
  - – Sender and receiver have (accurate) independent clocks
  - – Clocks at the same frequency
  - – The structure of an asynchronous word

At transition receiver starts its clock

Samples at same bit-rate at sender

Stop bit monitored for polarity



Neil Storey, *Electronics*: *A Systems Approach,* 5th Edition © Pearson Education Limited 2013

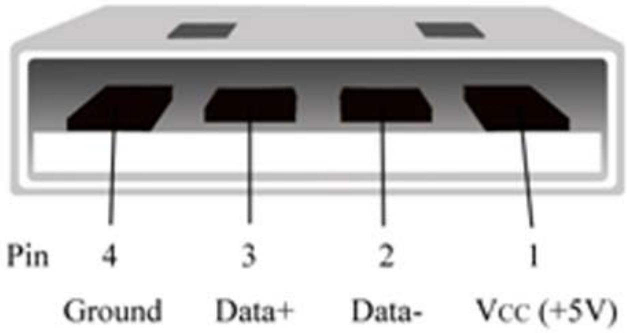
Serial I/O

- **Synchronous serial communications**
  - Synchronization field sent to receiver
    - A bit pattern or specific synchronization word
  - Receiver clock derived from sync-word/words

For complete definition of data structure, if you need it, see: TN_116_USB Data Structure.pdf on It's learning.

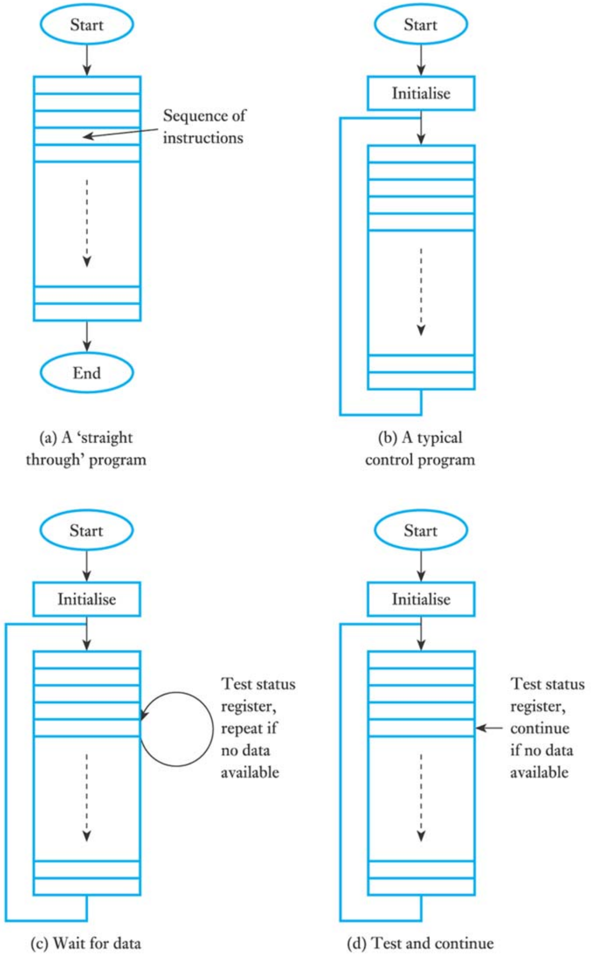
Serial communications standards

– One of the most important standards is the **Universal Serial Bus** or **USB**



- USB data is sent in packets Least Significant Bit (LSB) first.
- There are 4 main USB packet types :Token, Data, Handshake and Start of Frame.
- Each packet is constructed from different field types, namely SYNC, PID, Address, Data, Endpoint, CRC and EOP.
- The packets are then bundled into frames to create a USB message.

- **Programmed controlled input/output**
  - – polling of I/O devices


Start
Sequence of instructions
End
Start
Initialise
Start
Initialise
Test status register, repeat if no data available
Start
Initialise
Test status register, continue if no data available

(a) A 'straight through' program

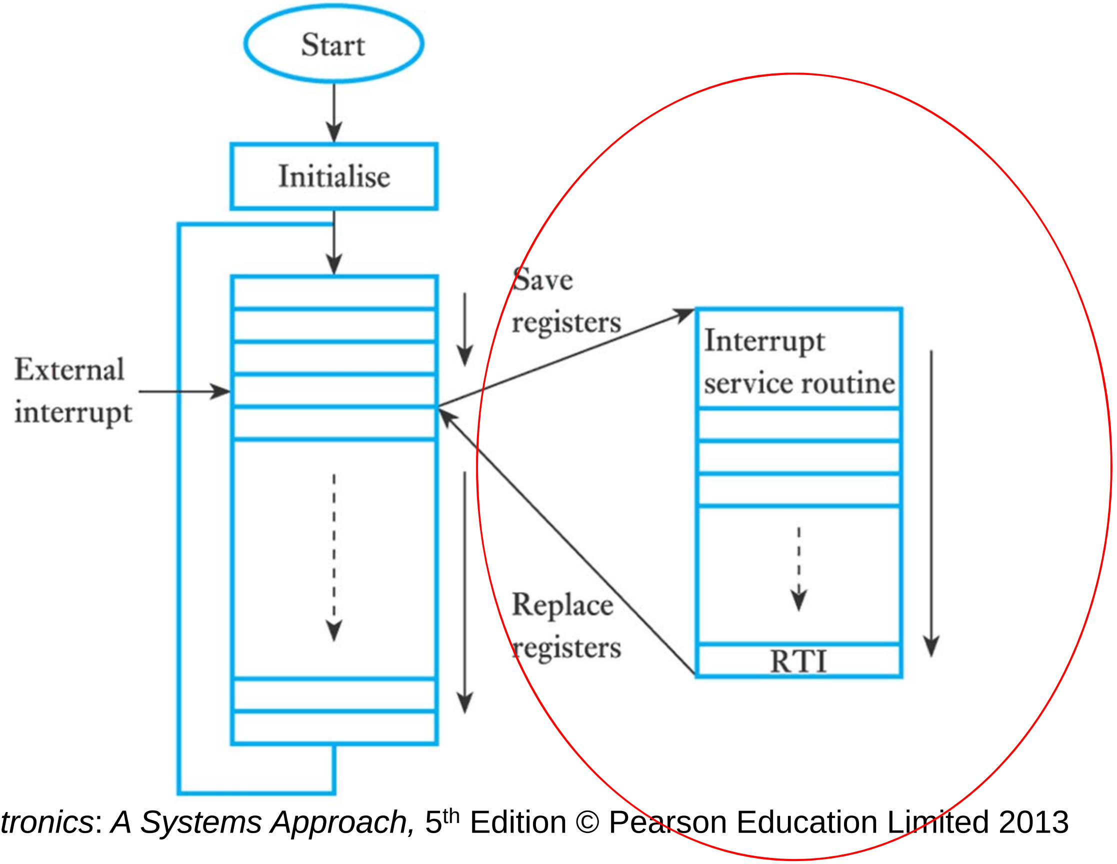
(b) A typical control program

(c) Wait for data

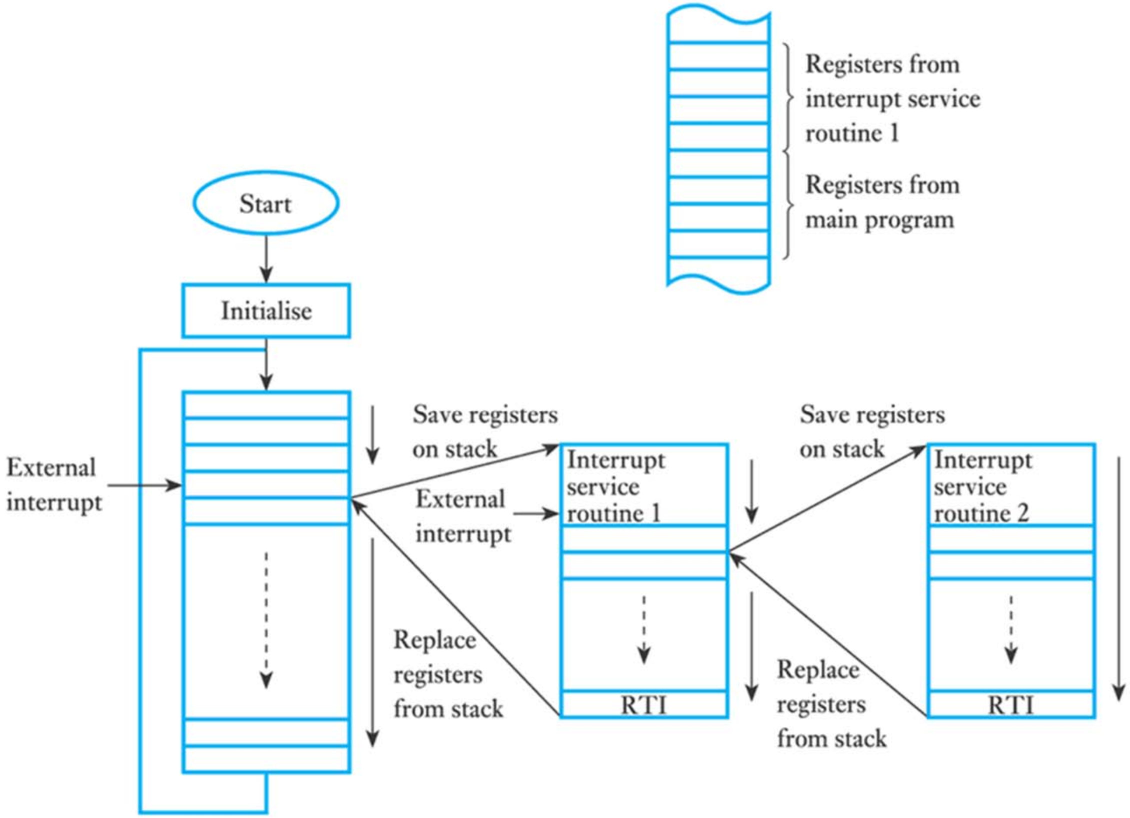
(d) Test and continue

Video 27G

- **Interrupts**
  - the interrupt mechanism



– multiple interrupt handling with a stack

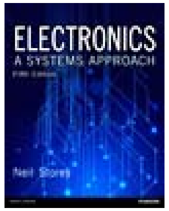


Neil Storey, *Electronics*: *A Systems Approach,* 5th Edition © Pearson Education Limited 2013

I/O techniques

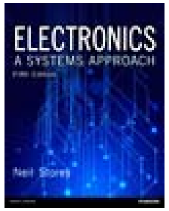
- Program controlled
  - Simplest to implement and test
- Interrupt-driven
  - Fast response and less processor time than polling
  - Hard to test as it is asynchronous
- Direct memory access.
  - Device accesses memory directly
  - Little impact on processor time, but
  - Extra interface hardware and expensive
  - used for large transfers (disk drives etc.)

- **Single-chip microcomputer**
  - – a single device providing the processor, memory and input/output sections of a microcomputer within a single integrated circuit
  - – many types available, ranging from very simple to very complex devices
  - – of particular interest are the PIC family of devices
    - uses a dual-bus Harvard architecture
    - use a RISC instruction set
    - extends from 6 pin devices with a few hundred bytes of memory to devices with more than 80 pins and 128 kbytes of memory



System-on-a-chip (SOC) devices

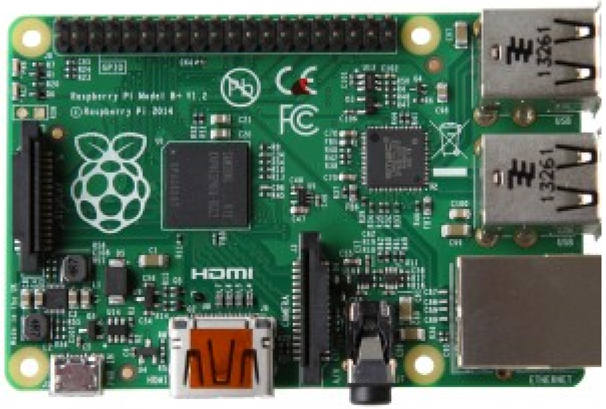
- Single-chip microcomputers contain all the elements of a computer within a single device, but will invariably need a range of additional components in order to produce a complete *system*
- SOC devices incorporate additional elements such as
  - Memory
  - Analogue interfaces
  - Communications interfaces
  - Timing elements
  - Power components



Selecting an implementation method

- The implementation method will depend on the complexity of the required functionality
  - applications requiring just a handful of gates might use **CMOS or TTL devices**
  - slightly more complex applications will often make use of **array logic**
  - complex digital applications will probably use either complex programmable devices (such as **CPLDs or FPGAs**) or a **microprocessor** (or a **SOC** or **PLC**)

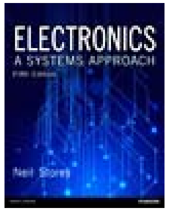
Something to consider for simple dedicated machines



- Raspberry Pi or Arduino
- Credit card computer with video/TV, keyboard, USB and ethernet interface.
- Developed for education/hobby (cheap)
- Many plug-in resources available for interface
- While developed for fun an learning, a surprisingly sophisticated computer with a suite of interfaces available for control and sampling.
- http://www.raspberrypi.org/
- https://www.arduino.cc/

Further Study

Video 27H



Further Study

- The Further Study section at the end of Chapter 27 considers the operation of a microcomputer-based controller in an automatic washing machine.
- Your task is to decide how the device should control the speed of the motor.
- Decide which of the input/output techniques discussed earlier is most appropriate for this task and watch the video to assess your ideas.

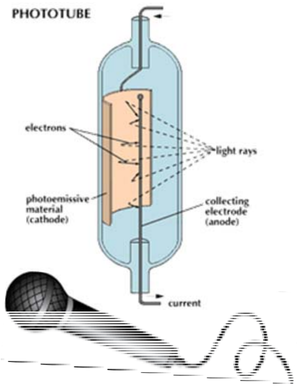
Key points

- Available complexity doubles every couple of years
- Array logic integrates large numbers of gates within a single package that is then configured for a particular application
- Complex digital systems can also be implemented using a microcomputer
- A programmable logic controller is a self-contained microcomputer that is optimised for industrial control
- The implementation method used will depend on the complexity of the required system

Typical “measurement”

Physical Quantity → Sensor → Electrical Signal Voltage or Current → Signal Processing Filtering, Amplification, Digitization → Computer Storage

Example:
pressure
temperature
charge
radiation intensity
sound intensity


PHOTOTUBE
electrons
light rays
photoemissive material (cathode)
collecting electrode (anode)
current

Electrical Noise → Electrical Signal Voltage or Current

1000 hPa Pressure ↔ What have we done to the signal in the process ↔ Record 42

Where are we?

- Have covered

Circuit components

Noise reduction

Filtering

How these can interact with sensor output

Amplification

Counting or Digitizing (next)

Capturing digital word

registers/latches

Interface with bus