Technical Means of Automation

Programmable Logic Controllers

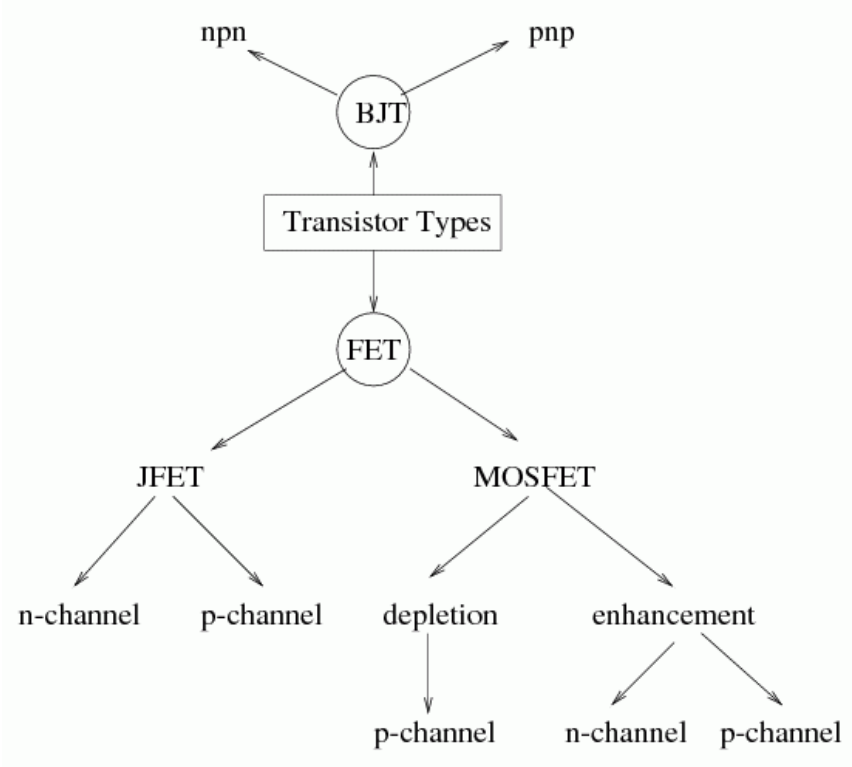
Institute of Information Engineering, Automation and Mathematics

November 29, 2016

What is This?

First Transistor

Transistors



Transistors:

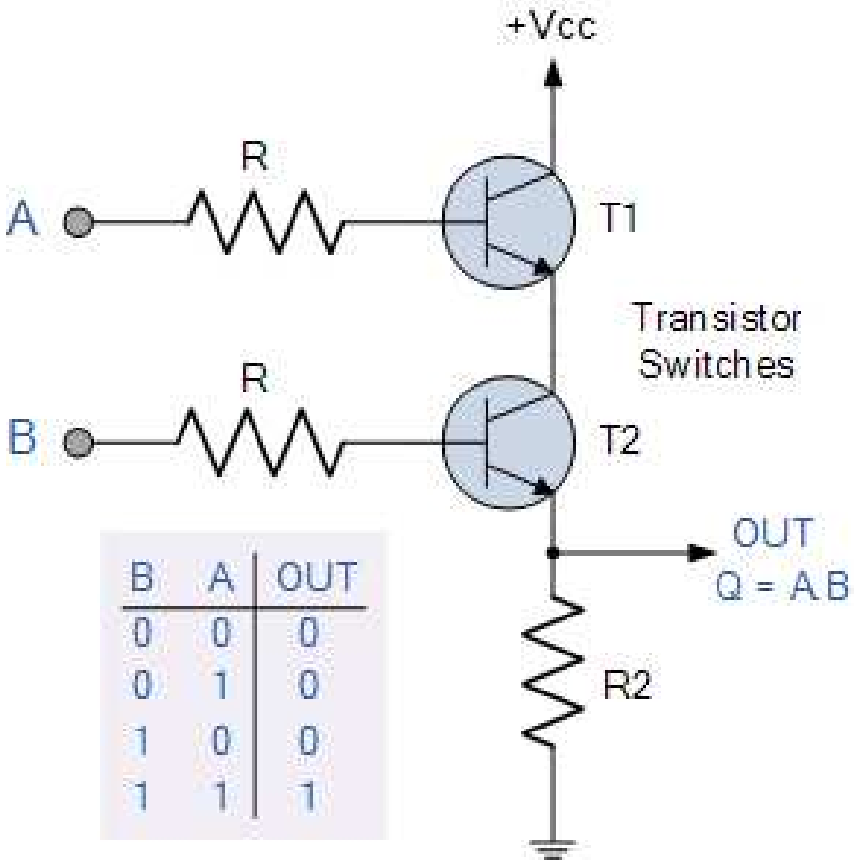
- semiconductor device
- current driven (BJT)
- voltage driven (FET)

Applications:

- digital switch (all types)
- amplifier (all types)
- logarithmic converter (BJT)
- thermometer (BJT)

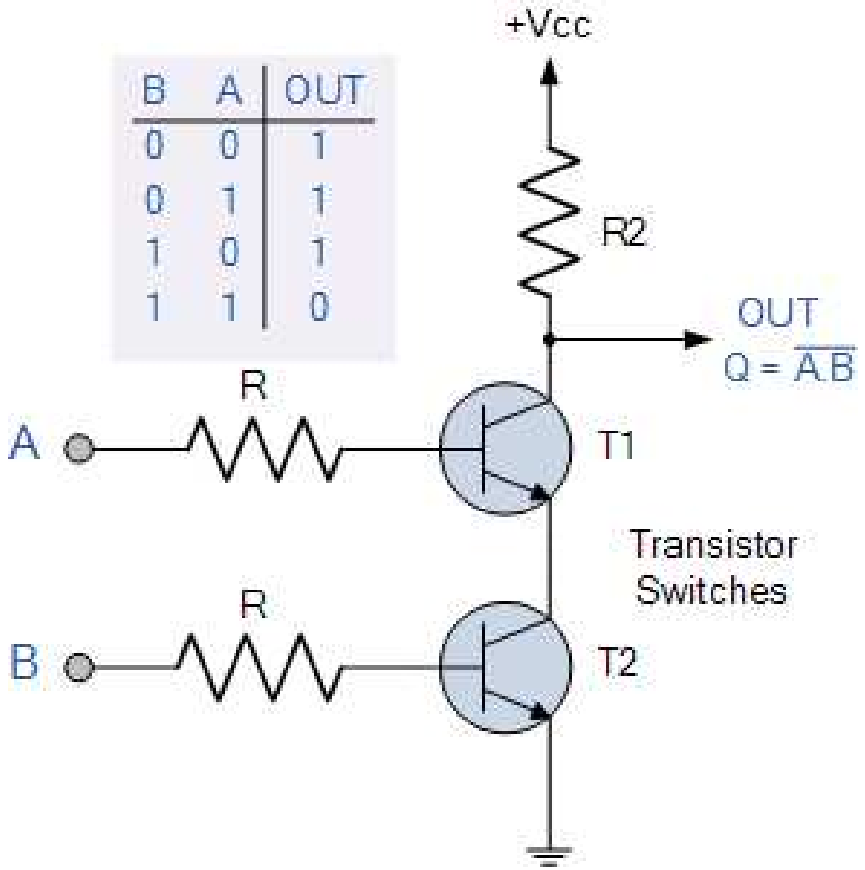
Boolean Algebra in Real World

AND operation



B	A	OUT
0	0	0
0	1	0
1	0	0
1	1	1

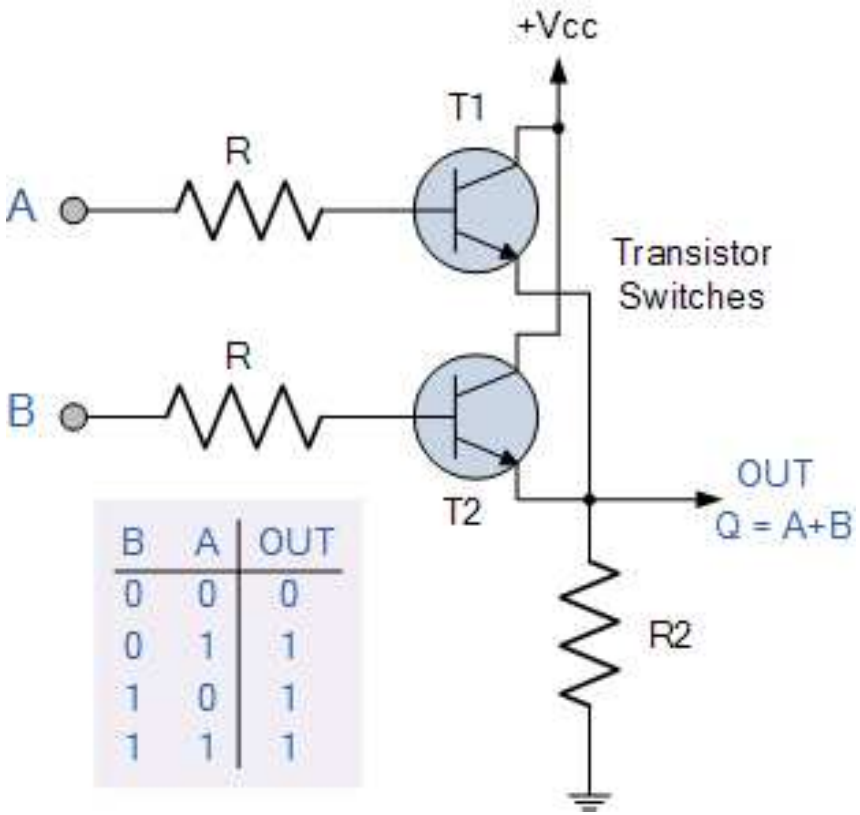
NAND operation



B	A	OUT
0	0	1
0	1	1
1	0	1
1	1	0

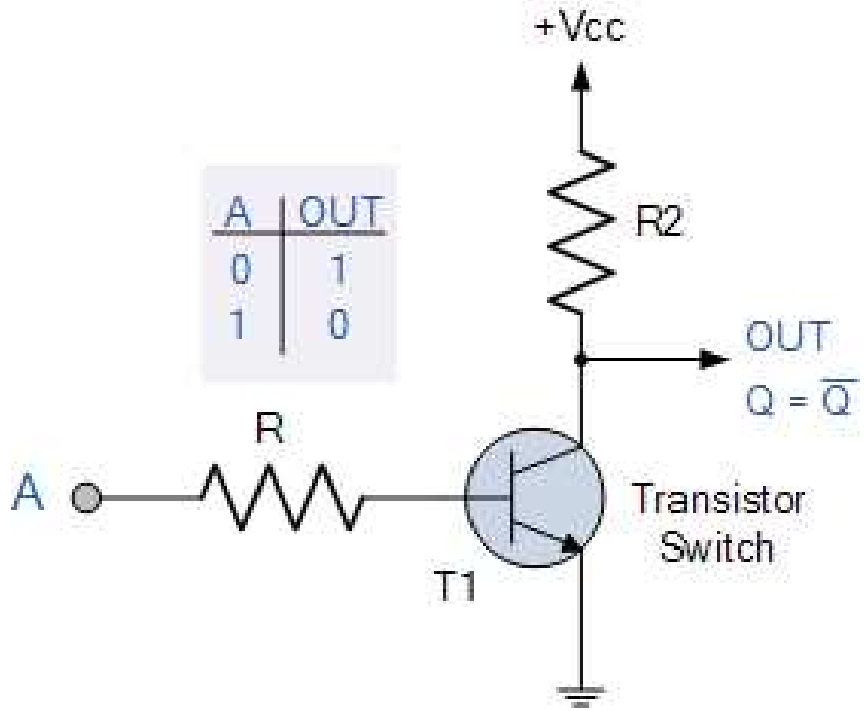
Boolean Algebra in Real World

OR operation



B	A	OUT
0	0	0
0	1	1
1	0	1
1	1	1

NOT operation

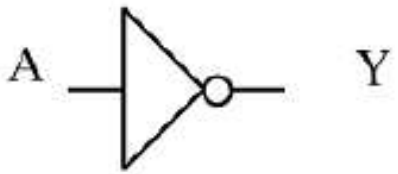


A	OUT
0	1
1	0

Logic Gates

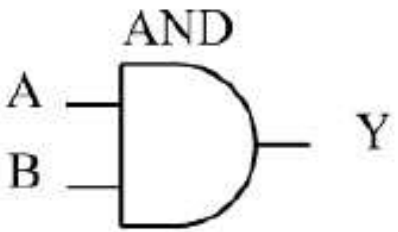
Basic Logic Gates

A	Y
0	1
1	0

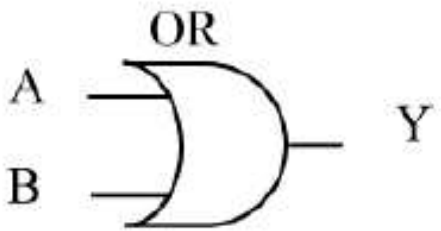


A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

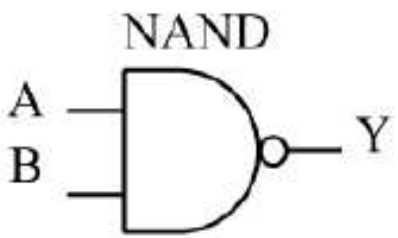
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



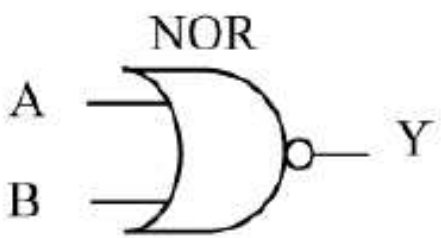
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1



A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0



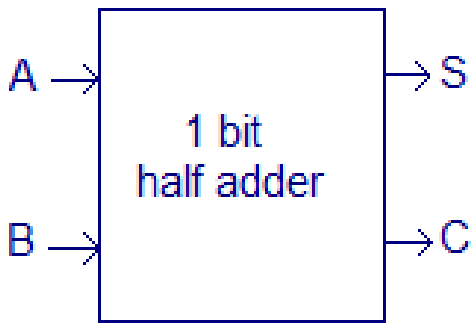
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0



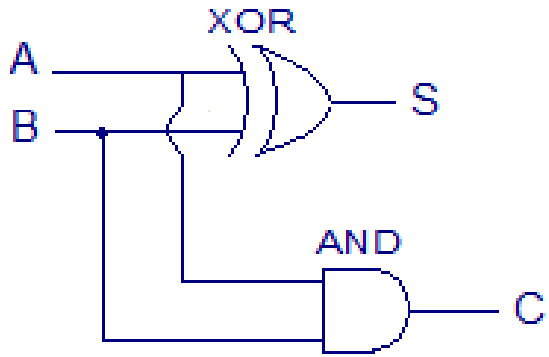
Arithmetic Operations

Inputs		Outputs	
A	B	S	C
0	0	0	0
1	0	1	0
0	1	1	0
1	1	0	1

Truth table

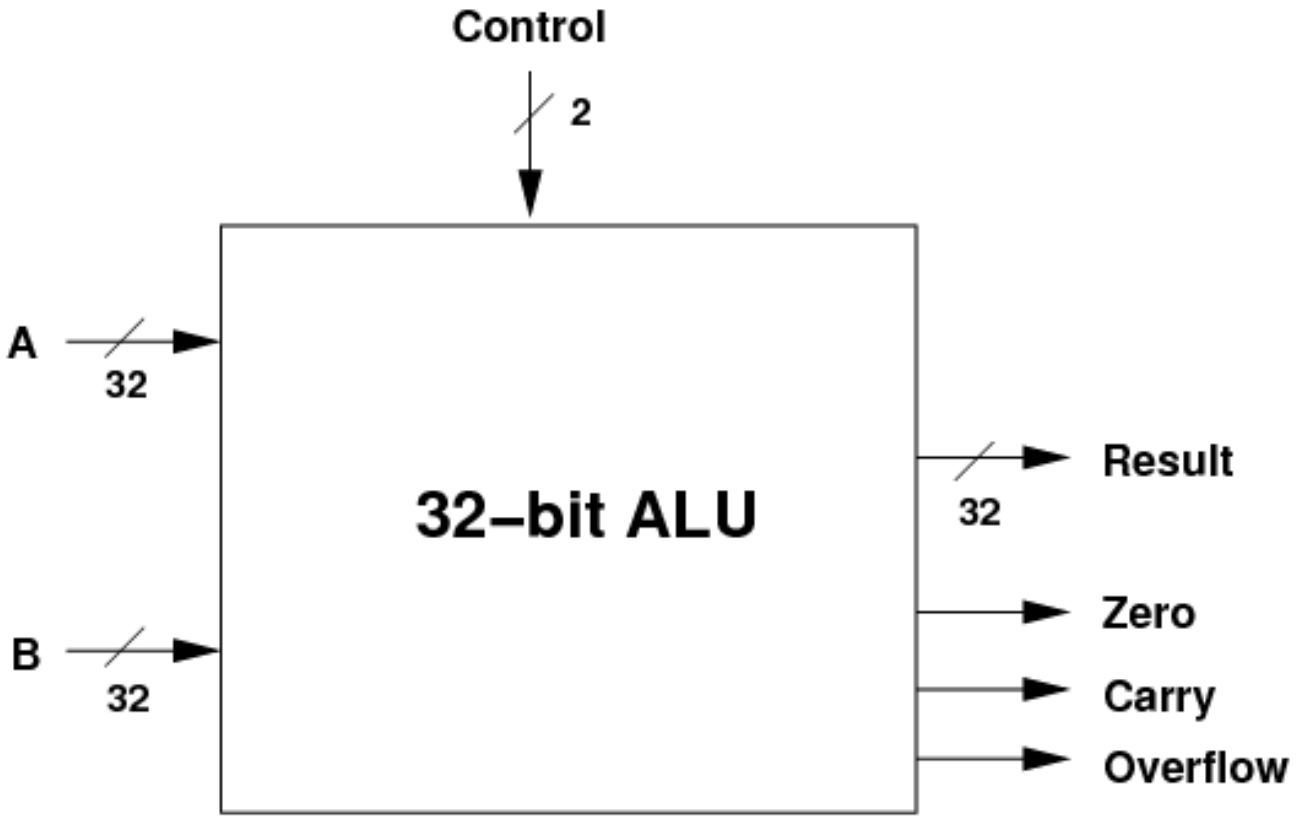


Schematic



Realization

Arithmetic Logic Unit (ALU)



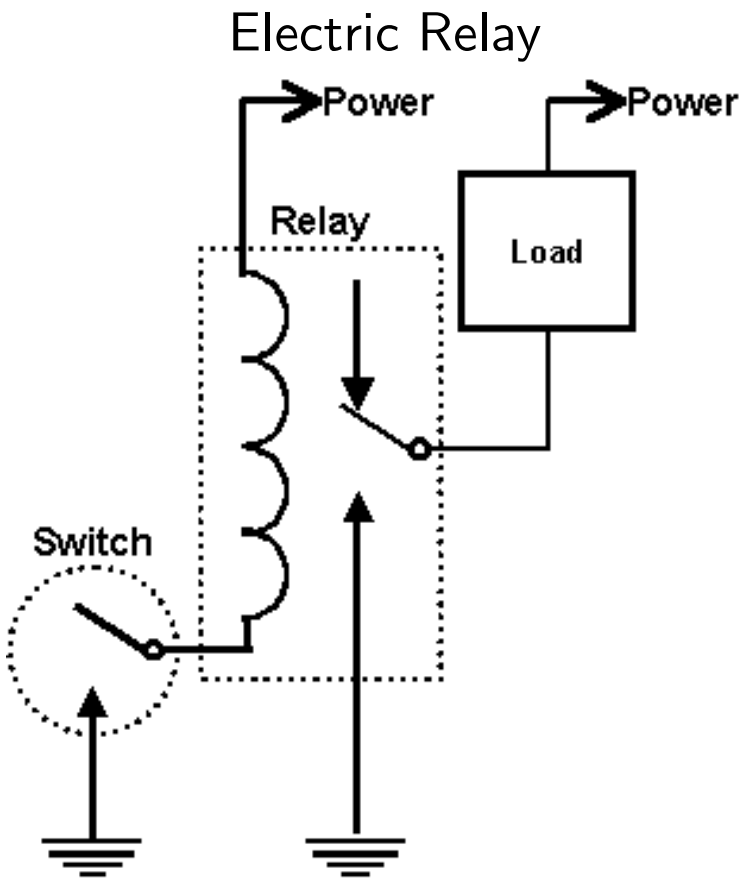
Programmable Logic Controllers

A **Programmable Logic Controller (PLC)** is a solid state control system that continuously monitors the status of devices connected as inputs. Based upon a user written program, stored in memory, it controls the status of devices connected as outputs.

PLC is a digital computer used for automation of electromechanical processes, such as control of machinery on factory assembly lines, control of amusement rides, or control of lighting fixtures.

Historical Logic Controllers – Electric Relays

- developed in 1890s
- used in telegraphic networks
- reconstruction of signal transferred over long distances
- used for control of machinery in first half of 20th century (**Logic Controllers**)



Historical Logic Controllers

First PLC:

- MODICON 084 (1968)
- Bedford Associates (Richard Morley)
- on order of General Motors
- automation of production lines
- 1972 – more than 10 PLC suppliers on world market
- strictly logical operations (digital)

Modern PLCs

Properties:

- hardware and software flexibility:
  - modularity
  - simplicity
  - effective programming
- operation stability and robustness
- in-operation maintenance and diagnostics
- re-usability for different applications
- I/O interface with standard signal parameters

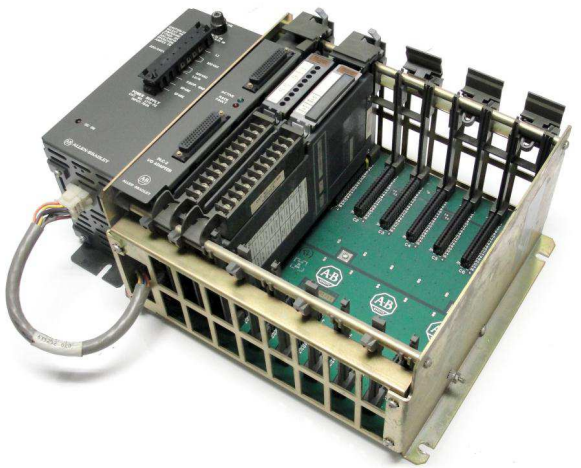
Sizes:

- small – units with up to 128 I/O's and memory up to 100 KB
- medium – up to 2048 I/O's and memory up to 1MB
- large – up to 8192 I/O's and memory up to 30 MB

PLC Construction Types

Types:

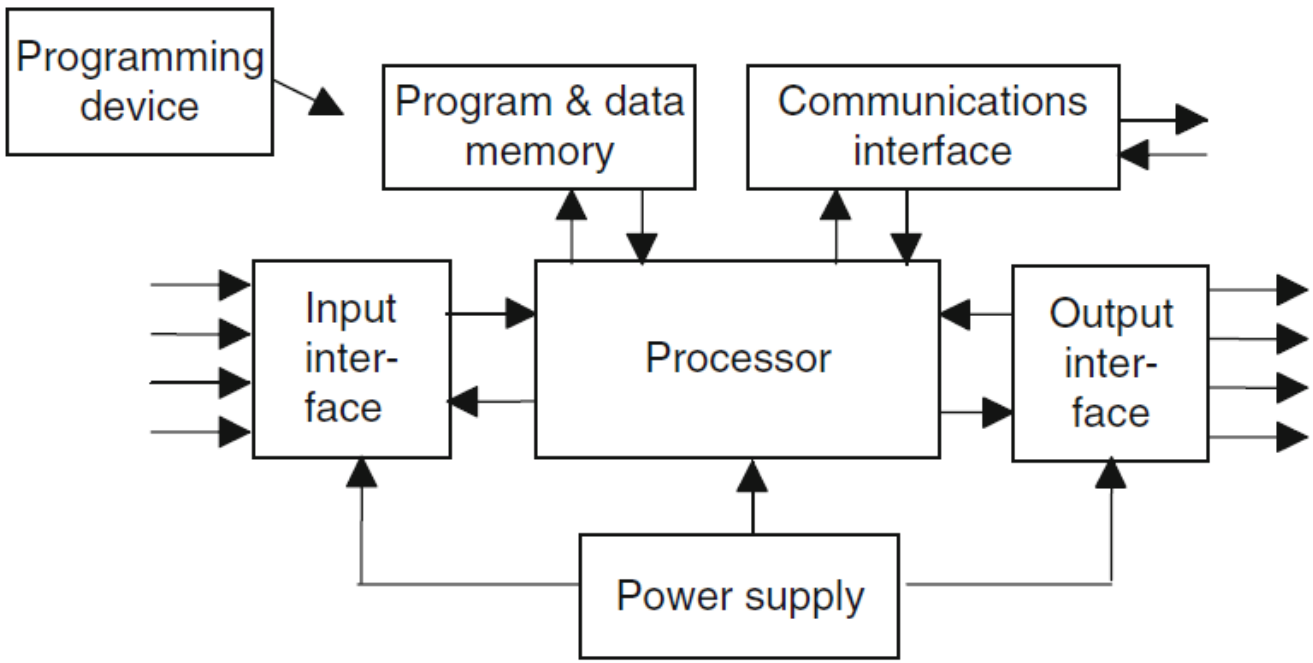
- unitary (monolithic)
  - very simple
  - cannot be extended by additional hardware
- modular
  - extendable by hardware modules
- rack mounted
  - extendable by hardware cards



PLC Hardware Architecture

PLC architecture consists of:

- Microprocessor (CPU)
- Memory
- Power supply
- Input interface
- Output interface
- Communication interface



Input/Output Modules

Many types of inputs and outputs can be connected to a PLC, and they can all be divided into two large groups, **analog** (discrete) and **digital**.

- **Digital inputs and outputs** – are those that operate due to a discrete or binary change - on/off, yes/no.
- **Analog inputs and outputs** – change continuously over a variable range - pressure, temperature, potentiometer.

I/O modules are available with various **numbers of field device points**, such as 4, 8, 16 and 32 point.

Optocouplers in the modules are used to isolate the module from the CPU.

Input/Output Module Types

- **AC Input** – uses AC voltage for input field device status
- **AC Output** – controls the ON/OFF state of AC output field devices such as relays, coils, and solenoids
- **DC Input (Discrete)** – uses DC voltage for input field device status
- **DC Input (Analog)** – the input is a variable DC signal level
- **DC Output (Discrete)** – discrete DC output modules control the ON/OFF states of DC output field devices
- **DC Output (Analog)** – provides a variable DC level

Input Devices

- Switches and push-buttons
- Discrete sensors
  - Limit switches
  - PV switches
  - Photoelectric sensors
  - Proximity sensors
- Analog sensors
  - Pressure sensors
  - Level sensors
  - Temperature sensors
- Encoders

Input Modules – Digital

- Number of inputs
  - 8, 16, 32, 64
  - groups 4, 8, 16
- Input voltage
  - AC, DC, AC/DC
  - 24V, 48V, 48-125V, 120V, 120V/230V
- Isolation
  - Galvanic (high speed)
  - Optical

Input Modules – Analog

- Number of inputs
  - 2, 4, 6, 8
  - individual / one common point
- Input voltage/current
  - $\pm$ 5 / $\pm$ 10V, 1-5 V, $\pm$ 20 mA, 0/4 - 20 mA
  - special – thermocouples, RTDs
- Resolution
  - 9, 12, 13, 14, 16 bits

Input Modules - Counters

- Number of inputs
  - 1 - 2 encoders
  - 8 standalone signals
- Input voltage
  - 5V, 24V
- Speed
  - 20 kHz, 500 kHz, 800 kHz
- Resolution
  - 0 to 32 bit:
  - 0 to 4 294 967 295 (0 to $2^32 - 1$)

Output Devices

- Valves
- Motor Starters
- Solenoids
- Actuators
- Control Relays
- Horns & Alarms
- Stack Lights
- Fans
- Counter
- Pumps
- Printers

Output Modules – Digital

- Number of outputs
  - 8, 16, 32, 64
  - groups 4, 8, 16
- Output voltage
  - AC, DC, AC/DC
  - 24V, 48V, 48-125V, 120V, 120V/230V
- Output current
  - 0.3A, 0.5A, 1.5A, 2A, 5A (relay)
- Isolation
  - Galvanic (high speed)
  - Optical

Output Modules – Analog

- Number of outputs
  - 2, 4, 8
  - individual / one common point
- Output voltage/current
  - $\pm$ 5 / $\pm$ 10V, 1-5 V, $\pm$ 20 mA, 0/4 - 20 mA
- Resolution
  - 11/12, 13/14, 16 bits

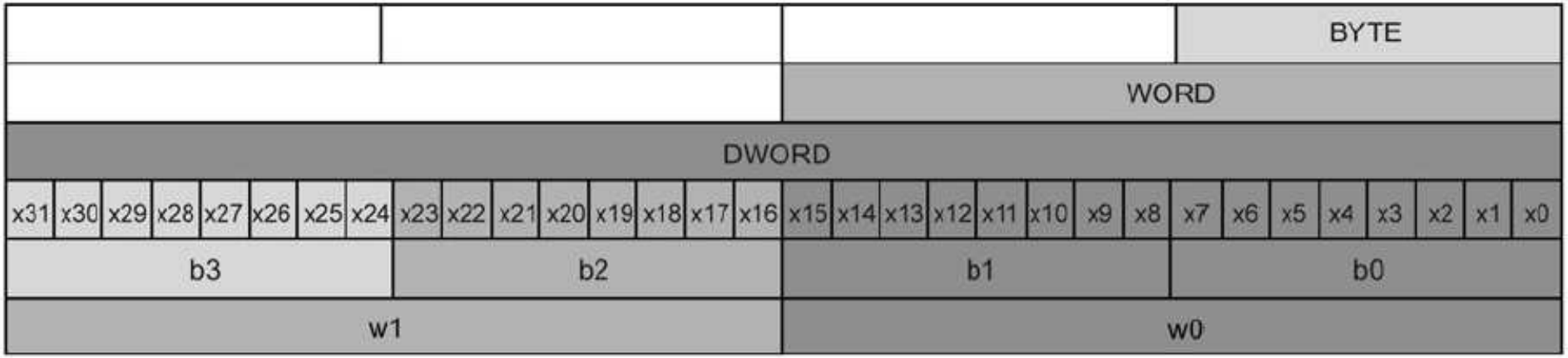
Physical Memory Types in PLCs

- RAM (Random Access Memory)
  - used by **CPU as an operating memory**
  - information lost in power off state
- ROM (Read Only Memory)
  - used to store PLC's basic **operating system / firmware**
  - holds information in power off state
- EPROM (Erasable Programmable Read Only Memory)
  - programmed out of the PLC, and then placed in the PLC
- EEPROM (Electrically-Erasable Programmable Read Only Memory)
  - can store programs like ROM. It can be programmed and erased using a voltage, so it is becoming more popular than EPROMs.
- Flash Memory
  - Newest type of EEPROM, but has similar functionality. Is mostly used for storing user defined programs before they are executed by CPU.

Memory Types in PLCs

- **Load memory** – is non-volatile storage for the user program, data and configuration. When a project is downloaded to the CPU, it is first stored in the Load memory area. This area is located either in a memory card (if present) or in the CPU.
- **Work memory** – is volatile storage for some elements of the user project while executing the user program. The CPU copies some elements of the project from load memory into work memory. This volatile area is lost when power is removed, and is restored by the CPU when power is restored.
- **Retentive memory** – is non-volatile storage for a limited quantity of work memory values. The retentive memory area is used to store the values of selected user memory locations during power loss. When a power down or power loss occurs, the CPU restores these retentive values upon power up.

Simple Data Types



Bit and bit-sequence:

- **Bool** is a Boolean or bit value.
- **Byte** is an 8-bit byte value
- **Word** is a 16-bit value
- **DWord** is a 32-bit double-word value

Integer data types:

- **USInt** (unsigned 8-bit integer) and **SInt** (signed 8-bit integer)
- **UInt** (unsigned 16-bit integer) and **Int** (signed 16-bit integer)
- **UDInt** (unsigned 32-bit integer) and **DInt** (signed 32-bit integer)

Real number data types:

- **Real** is a 32-bit Real number or floating-point value
- **LReal** (64-bit)

Date and time data types:

- **Date** is a 16-bit date value that contains the number of days since January 1, 1990
- **DTL** (date and time long - 12 bytes), **Time** (32-bit)

Character data types:

- **Char** is an 8-bit single character
- **String** (up to 254 characters)

Basic Addressing Memory Areas

Memory areas:

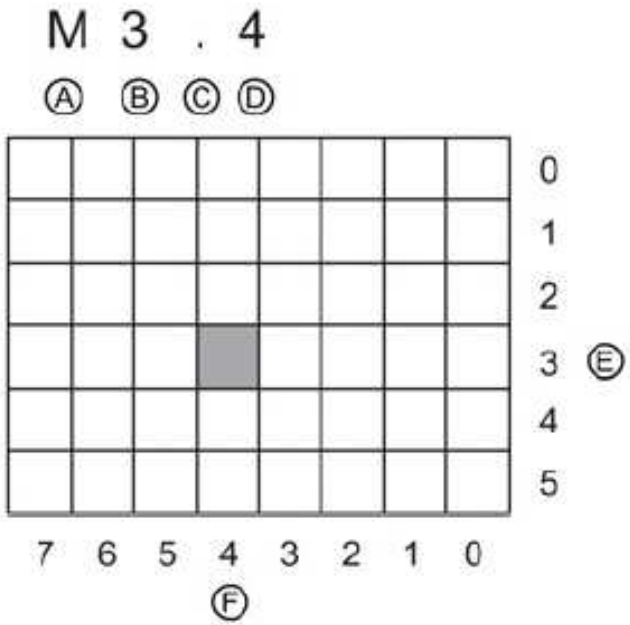
- Process image memory (image of input and output state of process)
  - **I** – process image **inputs**
  - **Q** – process image **outputs**
- **M** – control and data memory (bit memory)
- **L** – temporary data for a block local to that block
- **DB** – data memory and also parameter memory for FBs

Standard address structure:

[Memory area (I,Q,M,..)**][**Size (B,W)**][**Address**].[**Bit address**]**

Addressing Memory

Accessing a bit in the address for a Boolean value using a **dot** operator. Address contains only the memory area, the byte location, and the bit location for the data (such as I0.0, Q0.1, or M3.4).



Absolute address of a memory area:

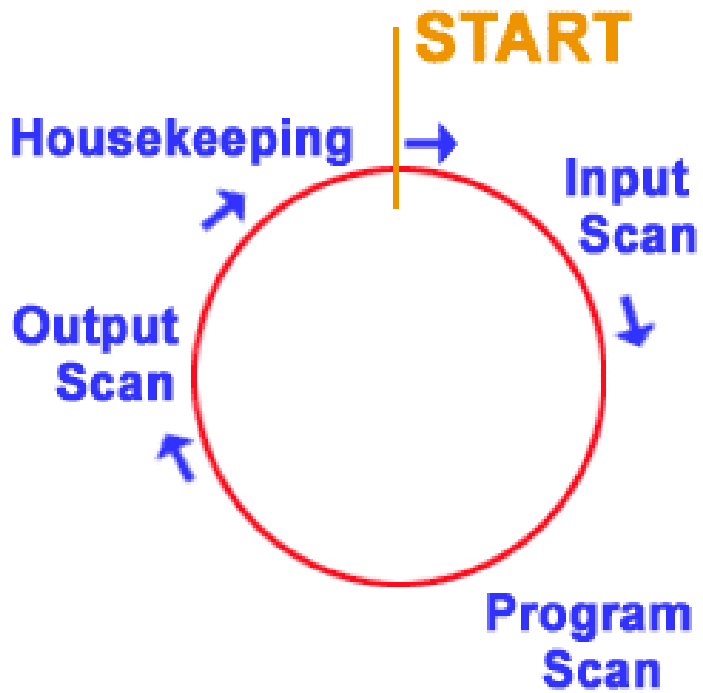
A	Memory area identifier
B	Byte address: byte 3
C	Separator ("byte.bit")
D	Bit location of the byte (bit 4 of 8)
E	Bytes of the memory area
F	Bits of the selected byte

PLC Operating Cycle

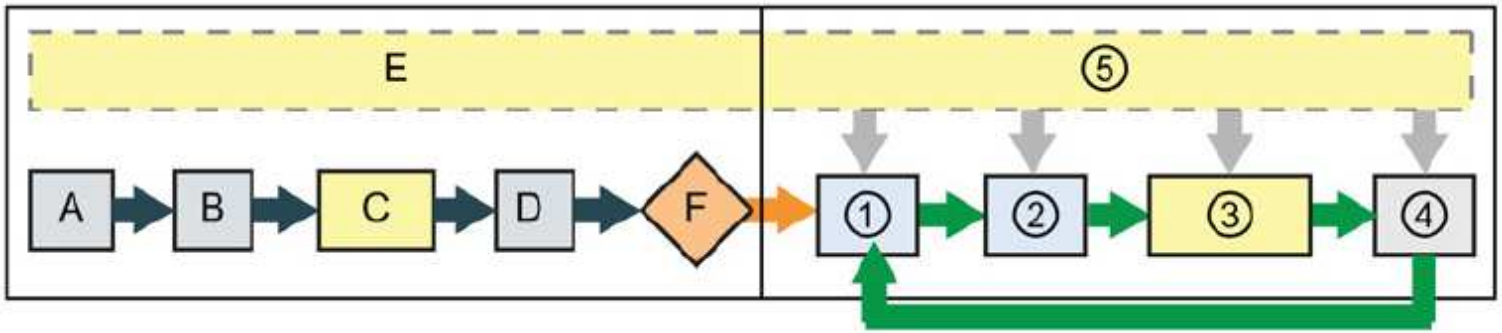
Operating cycle is repeating routine that comes after the hardware and software initialization.

Four Steps are incorporated in PLC Operation:

- **Input scan** – scans the state of the inputs
- **Program scan** – processes the program logic
- **Output scan** – energize (or de-energize) the outputs
- **Housekeeping** – this step includes communications, internal diagnostics, waiting, etc.



PLC Program Execution



- A Clears the I (image) memory area
- B Initializes the Q (image) memory
- C Initializes M memory, DBs and enables interrupts.
- D Physical inputs to I memory
- E Stores interrupts to queue for RUN
- F Enables the writing of Q memory to the physical outputs

- ① Writes Q memory to the physical outputs
- ② Copies the state of the physical inputs to I memory
- ③ Executes the program cycle OBs
- ④ Performs self-test diagnostics
- ⑤ Processes interrupts and communications during any part of the scan cycle

Operating Modes of PLC's CPU

The CPU has three modes of operation: **STOP** mode, **STARTUP** mode, and **RUN** mode. Status LEDs on the front of the CPU indicate the current mode of operation.

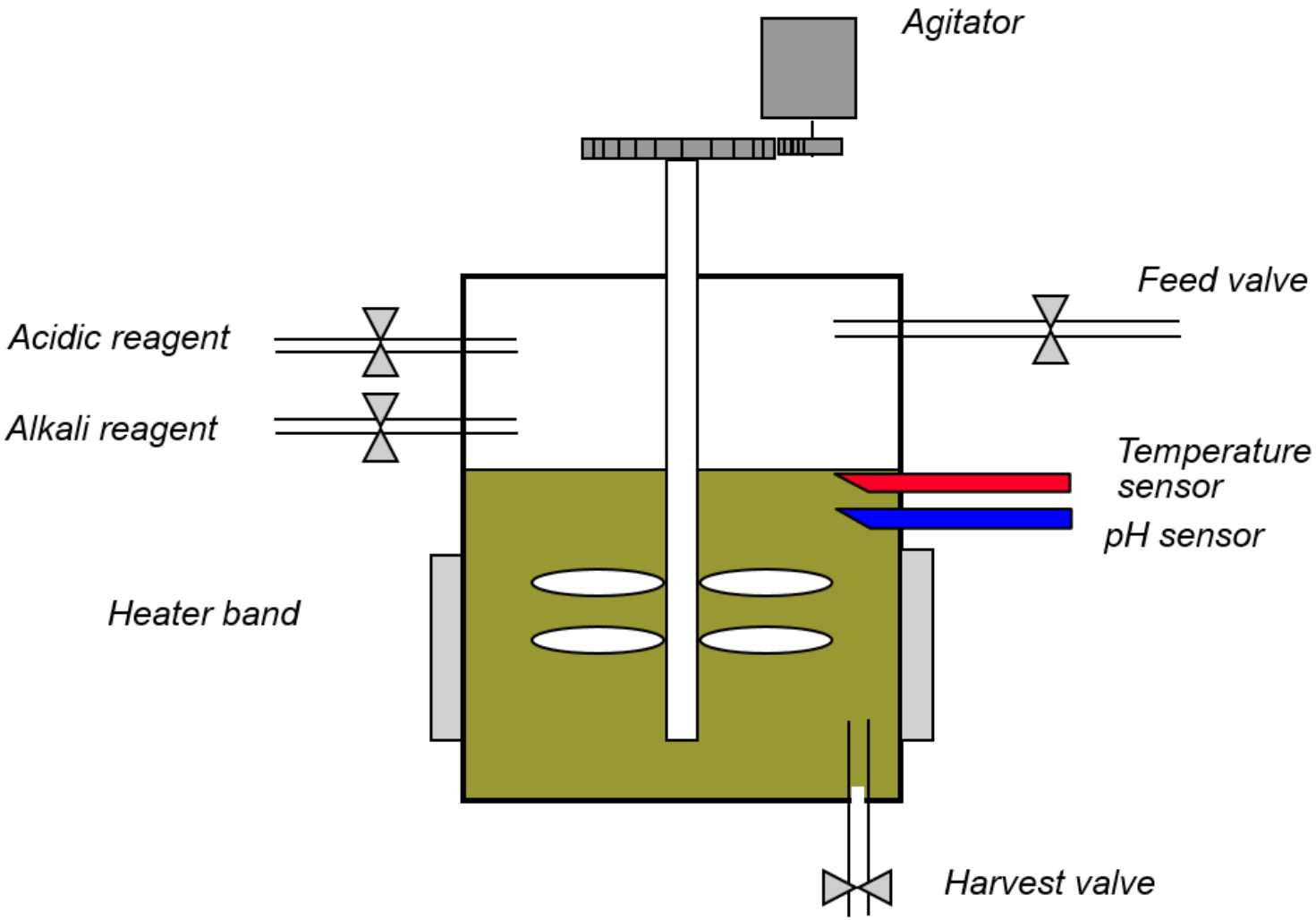
- **STOP mode** – the CPU is not executing the program, and project/program can be downloaded to PLC.
- **STARTUP mode** – the CPU executes any startup logic (if present). The CPU does not process interrupt events during the startup mode.
- **RUN mode** – the scan cycle executes repeatedly. Interrupt events can occur and the CPU can process them at any point within the program cycle phase.

Design of Program for Process Control

Now: Example of Fermentation Process Control

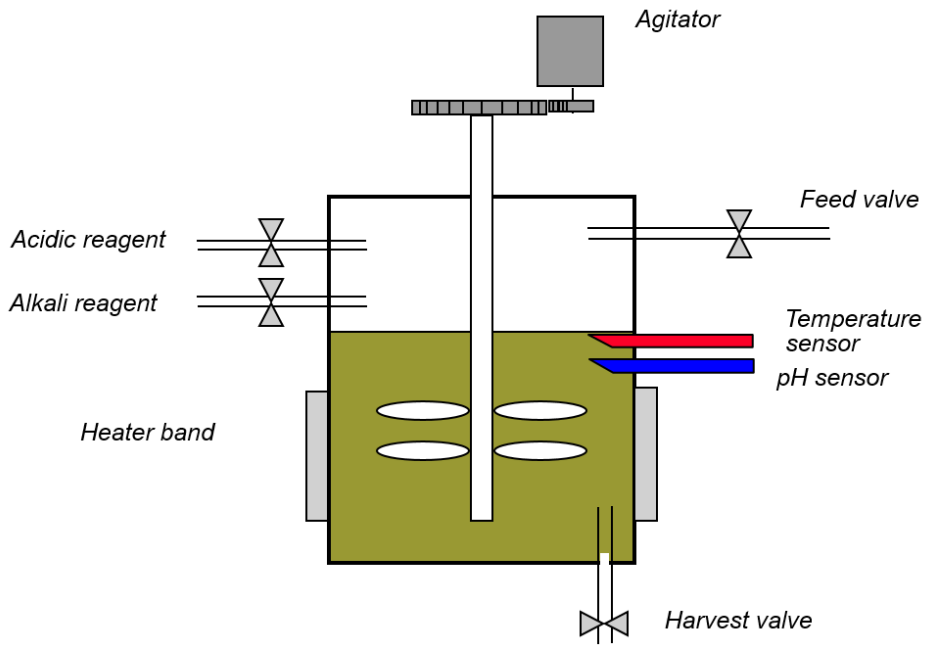
Next week: PLC programming techniques, Control algorithms, Communication

PLC Control – Fermentation Process



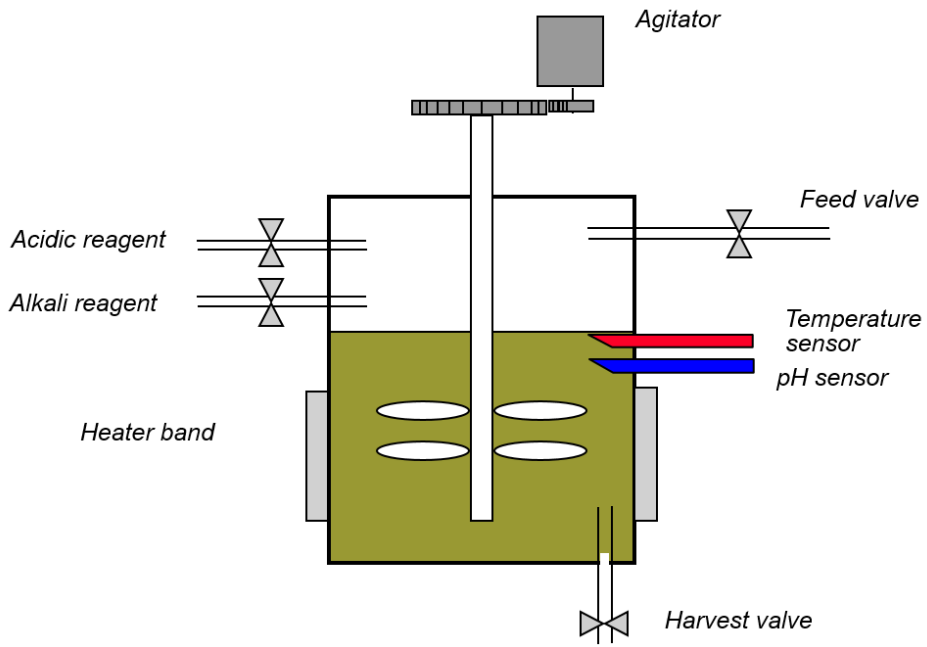
Step 1: Identification of system's interfaces

- Outputs (feedback):
  - temperature sensor
  - pH sensor
  - position of valves
  - motor speed (agitation)
- Inputs:
  - feed/harvest valve control
  - acidic/alkali valve control
  - motor power control
  - heater state control



Step 2: Coupling of process variables (PV) and control variables (CV)

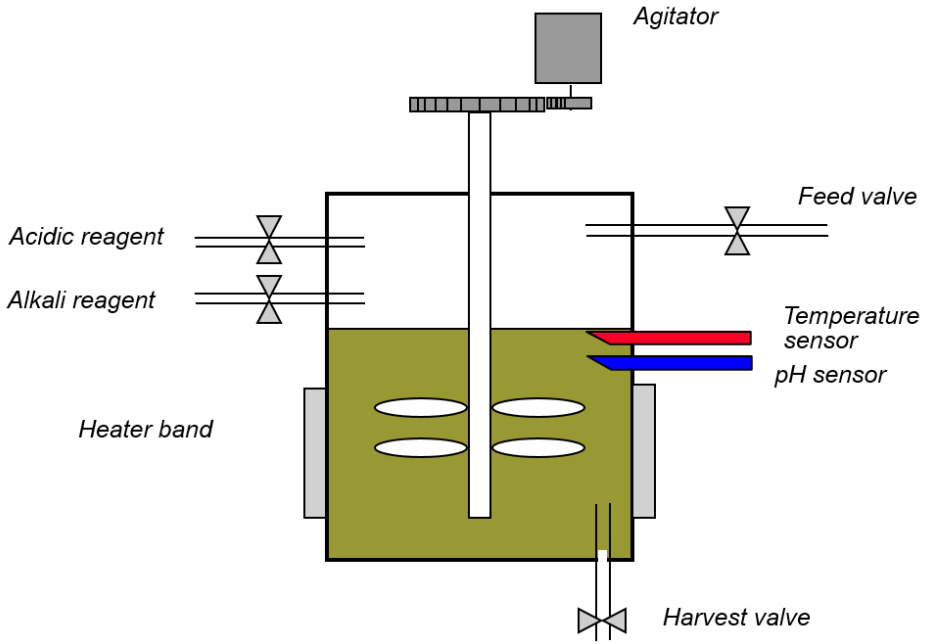
- **state of the heater** affects:
  - temperature
- **motor speed** affects:
  - agitation
- **acidic/alkali valves** affect:
  - pH value
- **feed/harvest valve** affect:
  - fill and harvest (operation stages)



Step 3: Definition of all operator interactions and overrides

For operator we define:

- a **“Start”** button
- a **“Stop”** button
- a **“Duration”** input

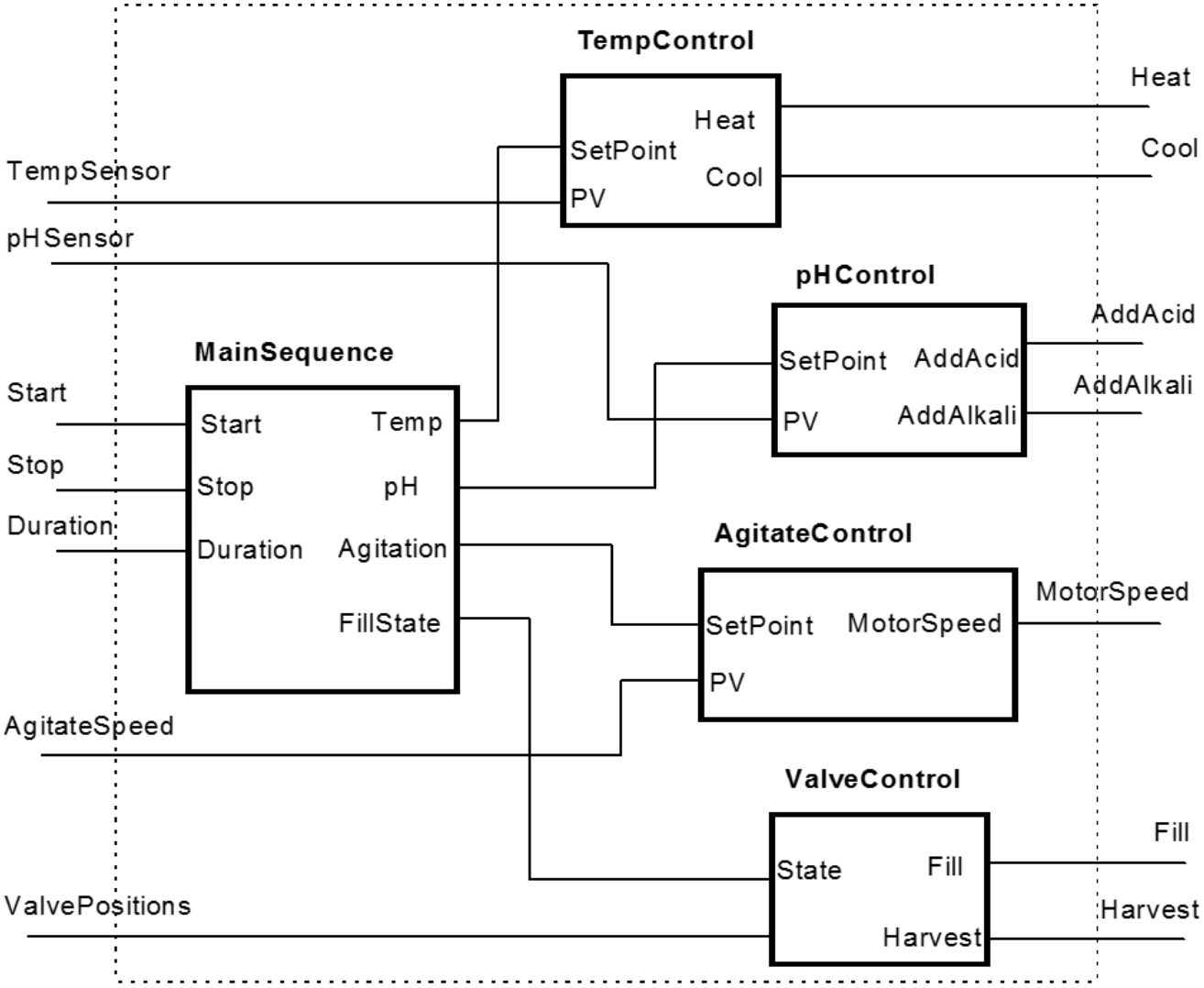


Step 4: Definition of program and function blocks

Each block performs a particular control task:

- **MainSequence** – program block that outputs sequential commands for function blocks. Setpoints for filling, heating, agitation, fermenting, harvesting, and cleaning will be generated here. This block also accepts operator commands.
- **ValveControl** – operating valves used to fill and empty the vessel
- **TemperatureControl** – for controlling the temperature
- **AgitatorControl** – agitator motor control
- **pHControl** – operating acidic/alkali valves

Step 5: Composition of program and function block into control algorithm



PLC Control – Fermentation Process

MainSequence in Sequential Function Chart (SFC)