Mechatronics

Topic #1

Intro to Mechatronics, microprocessor, PC, DAC, microcontroller, and BS2

Course Philosophy

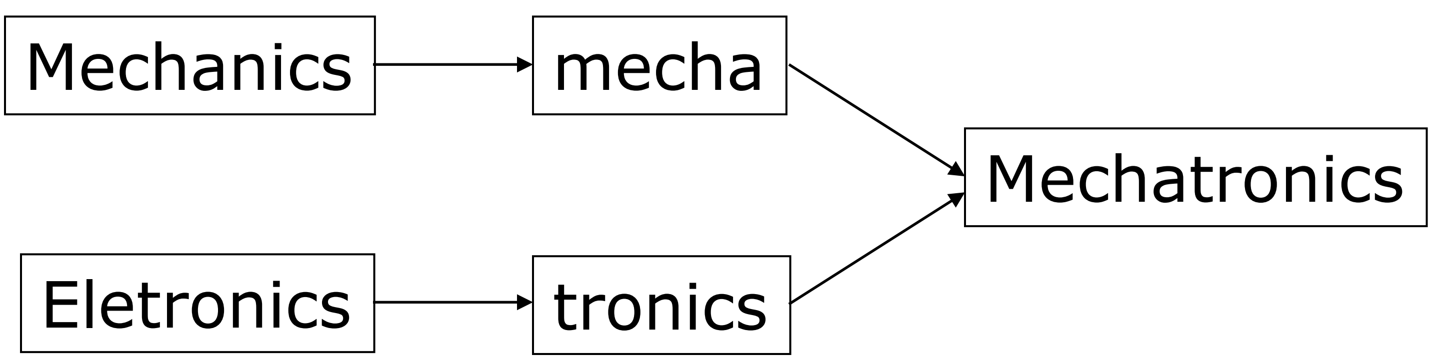
- Lecture:
  - Introduce/review foundational elements of mechatronics.
- Structured projects:
  - Illustrate and reinforce foundational elements of mechatronics.
- Final project:
  - Require design, modeling, simulation, analysis, refinement, prototyping, and validation of a mechatronics system.
  - Provide extensive hands-on integrative experience with various elements of mechatronics.

Course Objectives

- Understand foundational elements of mechatronics systems
- Understand operational principles of mechanical, electrical, electronic, and opto-electronic components
- Understand basic laws governing the operations of sensors and actuators including signal conditioning and power electronics
- Gain proficiency in Basic Stamps 2 (BS2) microcontroller—fundamentals, operation, programming, and interfacing
- Gain ability to integrate a variety of electrical and electronic components, sensors, and actuators to the BS2 microcontroller (e.g., filters, power electronics, analog to digital converter, digital to analog converter, digital I/O, etc.)
- Be able to work with control software (e.g., Matlab, Simulink, PBasic, etc.) for system modeling, control system design and analysis, data analysis, control prototyping, etc.
- Design, construct, and evaluate a prototype mechatronics system involving e.g., industrial automation, machinery monitoring/fault detection, embedded control, robotics, etc.

Mechatronics Defined — I

- "The name [mechatronics] was coined by Ko Kikuchi, now president of Yasakawa Electric Co., Chiyoda-Ku, Tokyo."
  - R. Comerford, "Mecha … what?" *IEEE Spectrum*, 31(8), 46-49, 1994.
- "The word, mechatronics is composed of *mecha* from mechanics and *tronics* from electronics. In other words, technologies and developed products will be incorporating electronics more and more into mechanisms, intimately and organically, and making it impossible to tell where one ends and the other begins."
  - T. Mori, "Mechatronics," *Yasakawa Internal Trademark Application Memo*, 21.131.01, July 12, 1969.



Mechatronics Defined — II

- “Integration of electronics, control engineering, and mechanical engineering.”
  - W. Bolton, *Mechatronics: Electronic Control Systems in Mechanical Engineering*, Longman, 1995.
- “Application of complex decision making to the operation of physical systems.”
  - D. M. Auslander and C. J. Kempf, *Mechatronics: Mechanical System Interfacing*, Prentice-Hall, 1996.
- “Synergistic integration of mechanical engineering with electronics and intelligent computer control in the design and manufacturing of industrial products and processes.”
  - F. Harshama, M. Tomizuka, and T. Fukuda, “Mechatronics-what is it, why, and how?- and editorial,” *IEEE/ASME Trans. on Mechatronics*, 1(1), 1-4, 1996.

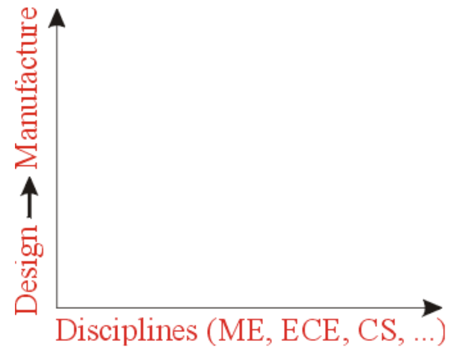
Mechatronics Defined — III

- "Synergistic use of precision engineering, control theory, computer science, and sensor and actuator technology to design <u>improved</u> products and processes."
  - S. Ashley, "Getting a hold on mechatronics," *Mechanical Engineering*, 119(5), 1997.
- "Methodology used for the <u>optimal design</u> of electromechanical products."
  - D. Shetty and R. A Kolk, *Mechatronics System Design*, PWS Pub. Co., 1997.
- "Field of study involving the analysis, design, synthesis, and selection of systems that combine electronics and mechanical components with modern controls and microprocessors."
  - D. G. Alciatore and M. B. Histand, *Introduction to Mechatronics and Measurement Systems*, McGraw Hill, 1998.
- Aside: Web site devoted to definitions of mechatronics:
  - http://www.engr.colostate.edu/~dga/mechatronics/definitions.html

Mechatronics: Our Working Definition

Mechatronics is the synergistic integration of sensors, actuators, signal conditioning, power electronics, decision and control algorithms, and computer hardware and software to manage complexity, uncertainty, and communication in engineered systems.

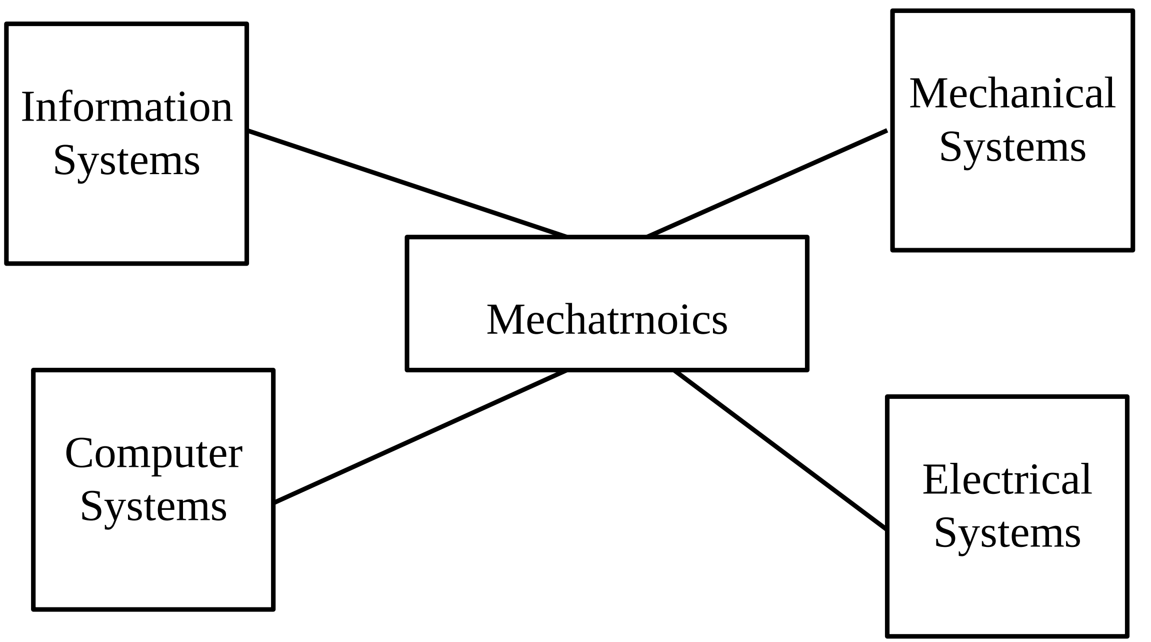
Product Realization Paradigm



- Engineered products frequently involve components from more than one discipline
- Traditional product realization
  - Discipline specific sequential process (design then manufacture)
  - Drawback: cost overruns due to redesign/re-tooling
- A better but still deficient approach
  - Discipline specific concurrent process (design for manufacturing)
  - Bottleneck: sub-optimal integration
- Mechatronics based product realization exploits
  - Integrated process founded upon interdisciplinary synergy

Disciplinary Foundations of Mechatronics

- Mechanical Engineering
- Electrical Engineering
- Computer Engineering
- Computer/Information Systems



Multi-/Cross-/Inter-Disciplinary

- Products and processes requiring inputs from more than one discipline can be realized through following types of interactions.
  - Multi-disciplinary: This is an additive process of brining multiple disciplines together to bear on a problem.
  - Cross-disciplinary: In this process, one discipline is examined from the perspective of another discipline.
  - Inter-disciplinary: This is an integrative process involving two or more disciplines simultaneously to bear on a problem.

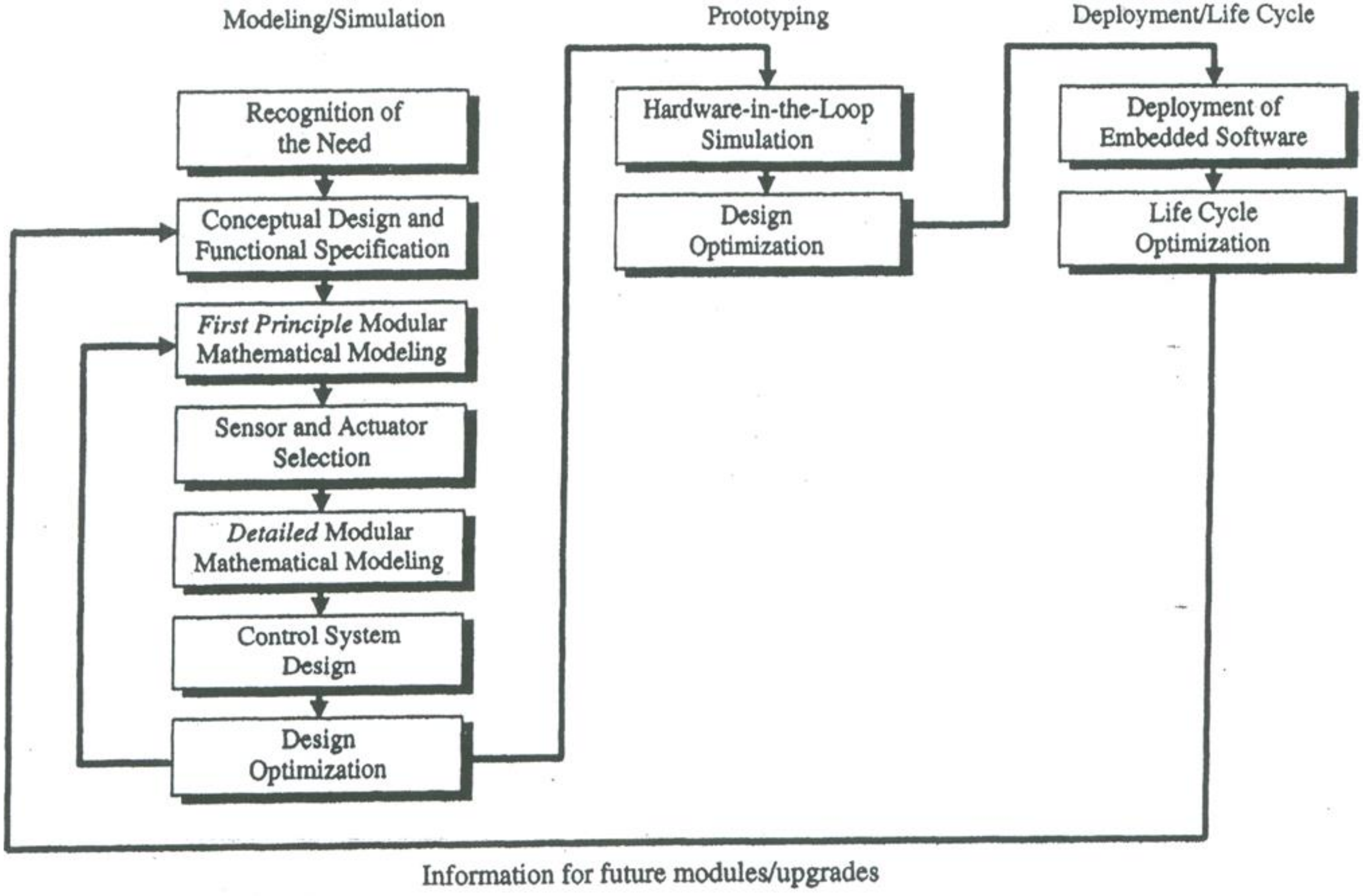
Sequential/Concurrent Product Realization

- Sequential and discipline specific concurrent design processes for product realization are at best multi-disciplinary calling upon discipline specialists to "design by discipline."
  - Design mechanical system "plant."
  - Select sensors and actuators and mount on plant.
  - Design signal conditioning and power electronics.
  - Design and implement control algorithm using electrical, electronics, microprocessor, microcontroller, or microcomputer based hardware.

Mechatronics-based Product Realization

- Systems engineering allows design, analysis, and synthesis of products and processes involving components from multiple disciplines.
- Mechatronics exploits systems engineering to guide the product realization process from design, model, simulate, analyze, refine, prototype, validate, and deployment cycle.
- In mechatronics-based product realization: mechanical, electrical, and computer engineering and information systems are integrated throughout the design process so that the final products can be better than the sum of its parts.
- Mechatronics system is not
  - simply a multi-disciplinary system
  - simply an electromechanical system
  - just a control system

Mechatronic Design Process



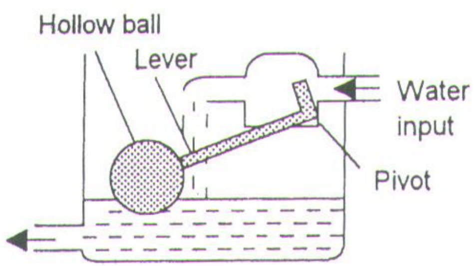
Evolution of Mechatronics as a Contemporary Design Paradigm

- Technological advances in design, manufacturing, and operation of engineered products/devices/processes can be traced through:
  - Industrial revolution
  - Semiconductor revolution
  - Information revolution

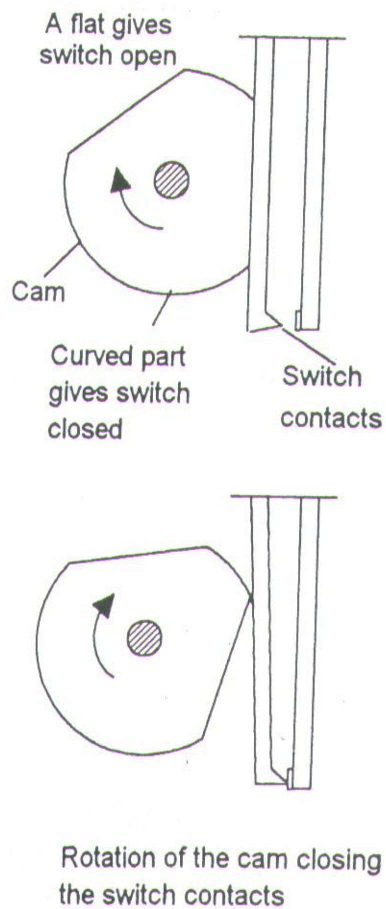
Industrial Revolution

- Allowed design of products and processes for energy conversion and transmission thus allowing the use of energy to do useful work.
- Engineering designs of this era were largely mechanical
  - e.g., operations of motion transmission, sensing, actuation, and computation were performed using mechanical components such as cams, gears, levers, and linkages).
- Purely mechanical systems suffer from
  - Power amplification inability.
  - Energy losses due to tolerances, inertia, and friction.

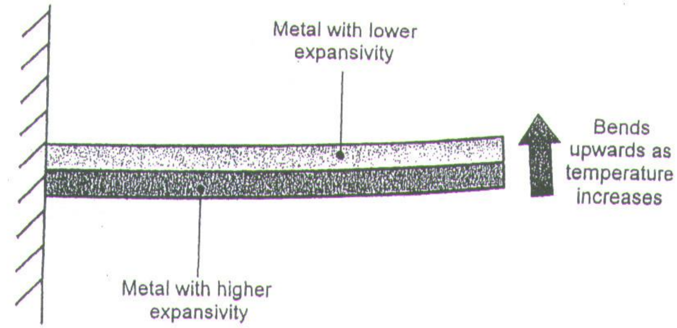
Examples of Predominantly Mechanical Designs



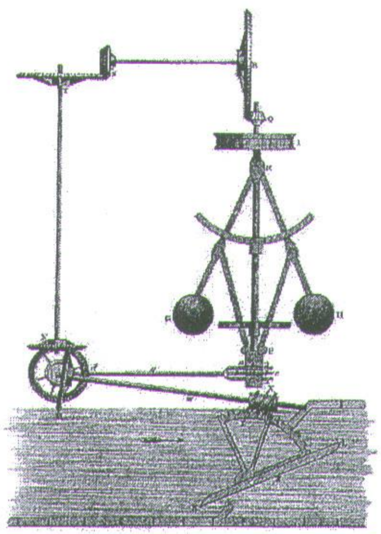
Float Valve



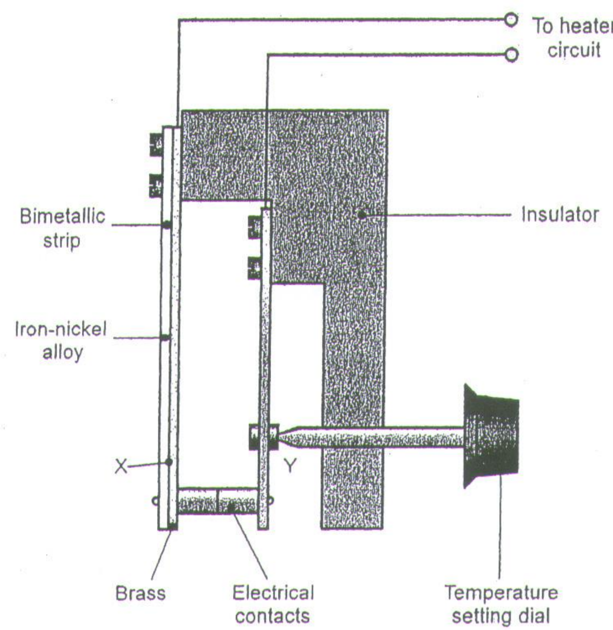
Cam Operated Switch



Bi-metallic Strip



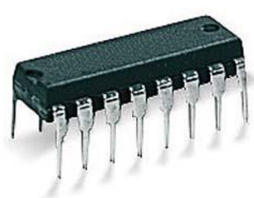
Watt's Governor



Thermostat

Semiconductor Revolution

- Led to the creation of integrated circuit (IC) technology.
- Effective, miniaturized, power electronics could amplify and deliver needed amount of power to actuators.
- Signal conditioning electronics could filter and encode sensory data in analog/digital format.
- Hard-wired, on-board, discrete analog/digital ICs provided rudimentary computational and decision-making circuits for control of mechanical devices.



An Integrated Circuit

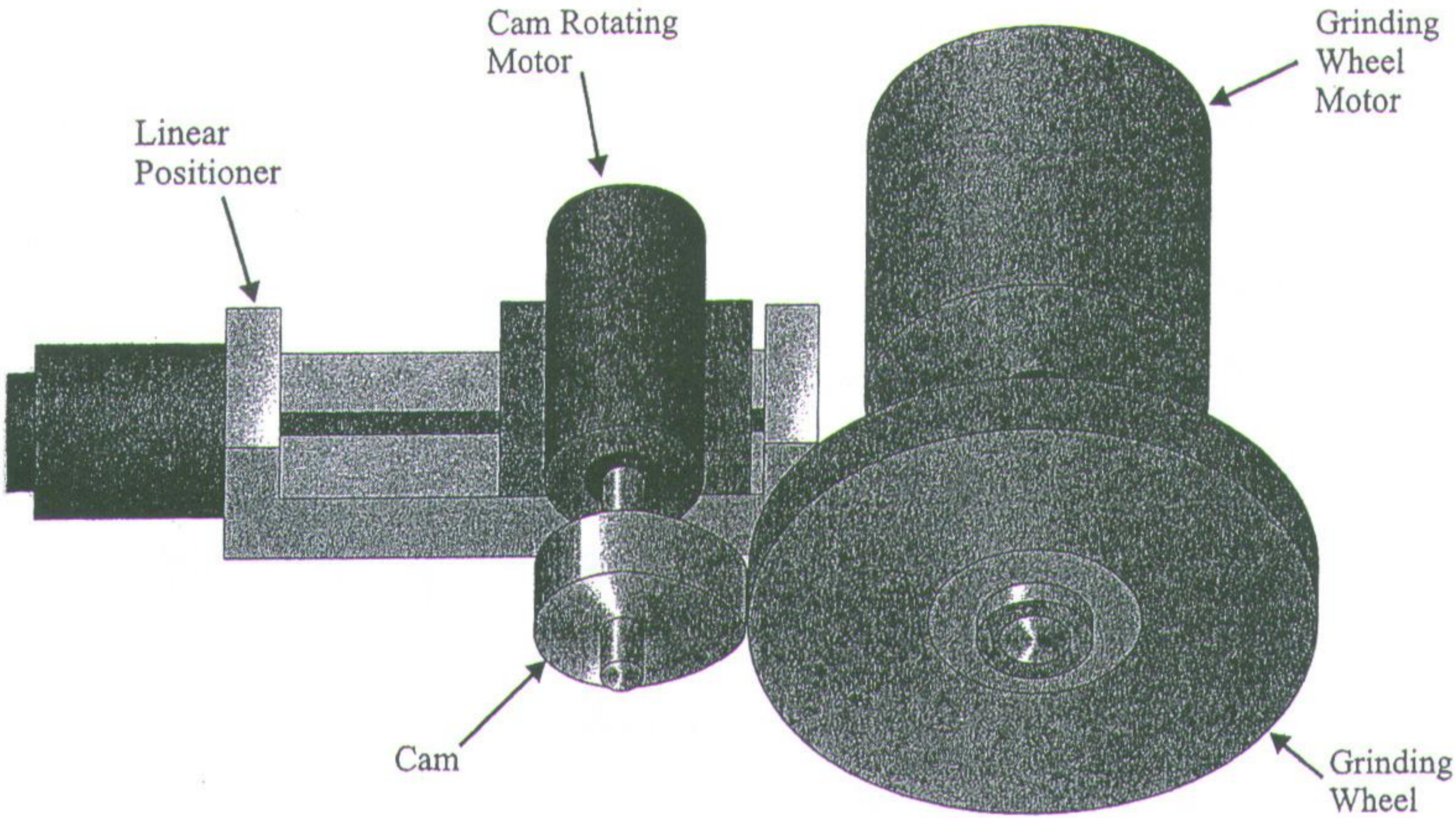
An A2D Converter

An Operational Amplifier

Information Revolution

- Development of VLSI technology led to the introduction of microprocessor, microcomputer, and microcontroller.
- Now computing hardware is ubiquitous, cheap, and small.
- As computing hardware can be effortlessly interfaced with real world electromechanical systems, it is now routinely embedded in engineered products/processes for decision-making.
  - Microcontrollers are replacing precision mechanical components, e.g., precision-machined camshaft that in many applications functions as a timing device.
  - Programmability of microcontrollers is providing a versatile and flexible alternative to the hard-wired analog/digital computational hardware.
  - Integrated computer-electrical-mechanical devices are now capable of converting, transmitting, and processing both the *physical energy* and the *virtual energy* (information).
- Result: Highly efficient products and processes are now being developed by judicious selection and integration of sensors, actuators, signal conditioning, power electronics, decision and control algorithms, and computer hardware and software.

Mechatronics Revolution: Example



Masterless Cam Grinder

Conventional v/s Mechatronics Design

conventional design	mechatronic design
added components	**integration of components (hardware)**
1 bulky 2 complex 3 cable problems 4 connected components	compact simple mechanisms bus or wireless communication autonomous units
simple control	**integration by information processing (software)**
5 stiff construction 6 feedforward control, linear (analog) control 7 precision through narrow tolerances 8 non-measurable quantities change arbitrarily 9 simple monitoring 10 fixed abilities	elastic construction with damping by electronic feedback programmable feedback (non-linear) digital control precision through measurement and feedback control control of non-measurable estimated quantities supervision with fault diagnosis adaptive and learning abilities

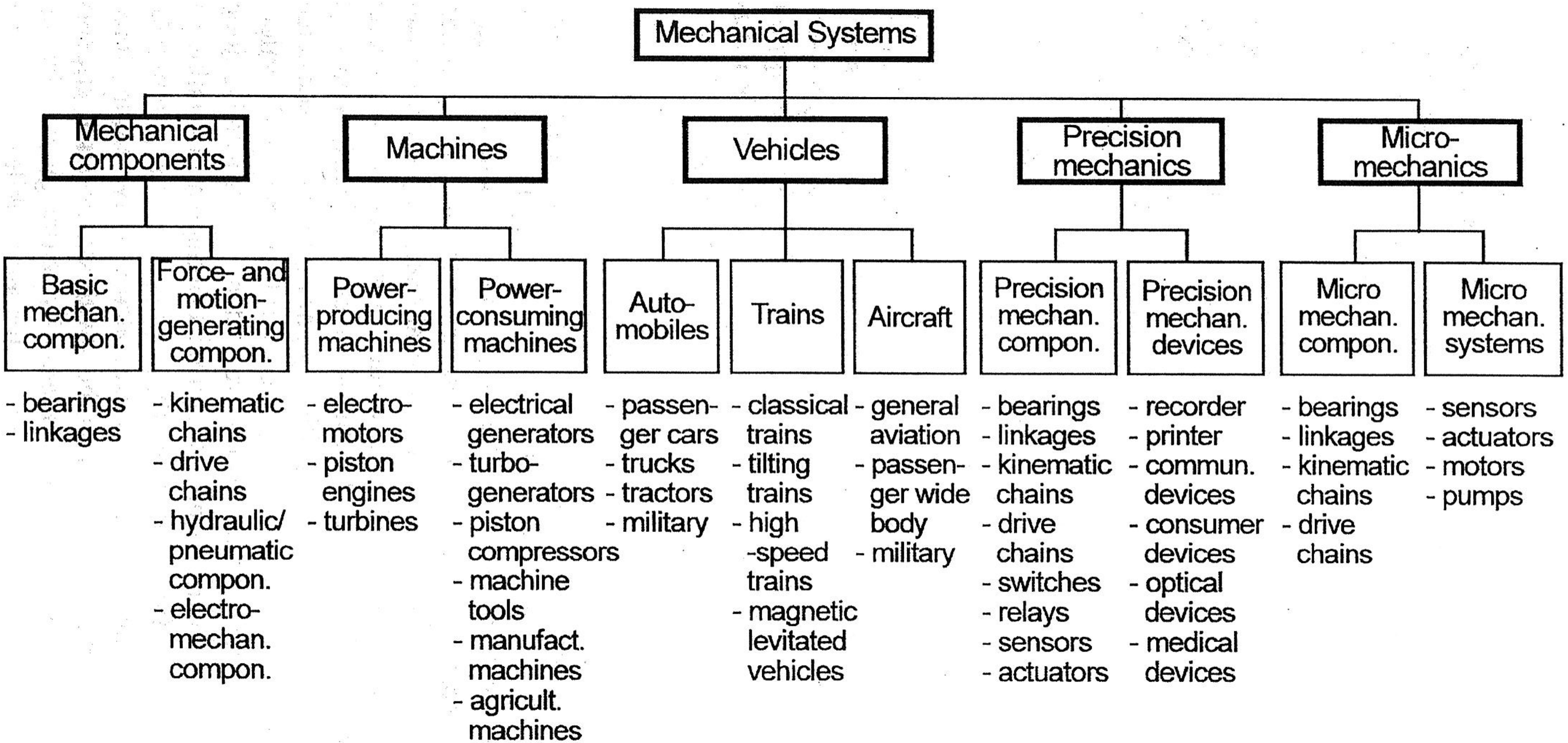
Conventional v/s Mechatronics Design: Realization Examples

conventional design	mechatronic design
added components	**integration of components (hardware)**
1 electromechanical typewriter 2 mechanically controlled injection pump with rotating piston 3 many wiring 4 belt-driven auxiliaries	electronic printer high pressure pump and magnetic injection valves (common rail) bus cable decentralized driven auxiliaries
simple control	**integration by information processing (software)**
5 stiff drivetrain 6 mechanical gas pedal 7 feedforward-controlled actuator 8 manual steering of cars during spinning 9 monitoring of exhaust gases through maintenance or inspection 10 rail vehicles	elastic drivetrain with algorithmic damping through engine control electronic non-linear throttle control feedback-controlled actuator with friction compensation feedback control of slip angle by state observer and individual wheel braking on-board misfire detection by speed measurement of engine crankshaft mobile vehicle with automatic navigation

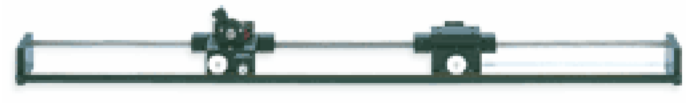
Elements of Mechatronics—Mechanical

- Mechanical elements refer to
  - mechanical structure, mechanism, thermo-fluid, and hydraulic aspects of a mechatronics system.
- Mechanical elements may include static/dynamic characteristics.
- A mechanical element interacts with its environment purposefully.
- Mechanical elements require physical power to produce motion, force, torque, flow, heat, etc.

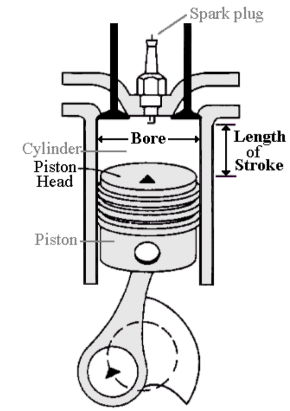
Elements of Mechatronics—Mechanical



Types of Motion

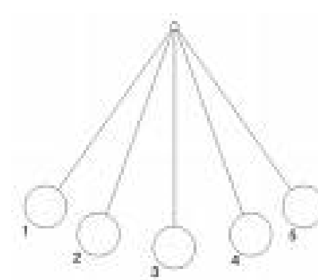


- Linear motion
  - motion in a straight line (example: train on a track)



- Reciprocating motion
  - linear motion that goes back and forth (example: pushing a slider-crank back and forth, such as the piston in an internal combustion engine)

- Rotary motion
  - circular motion (example: the hands of a clock moving, or a wheel on an axle)



- Oscillating motion
  - circular or arc-motion back and forth (example: the swing of a pendulum or the turning and release of a doorknob)

Inclined Plane

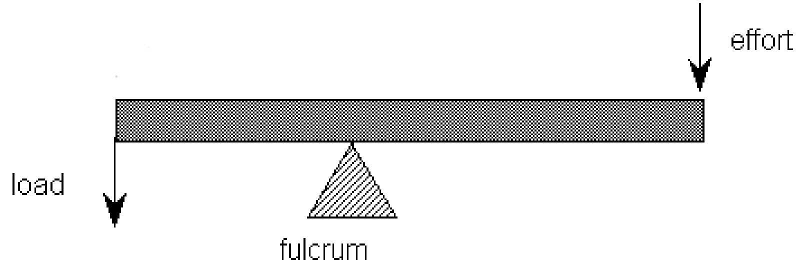
Wedge

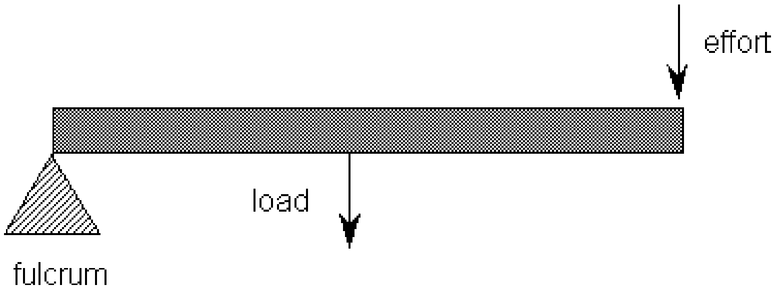
Worm Gear

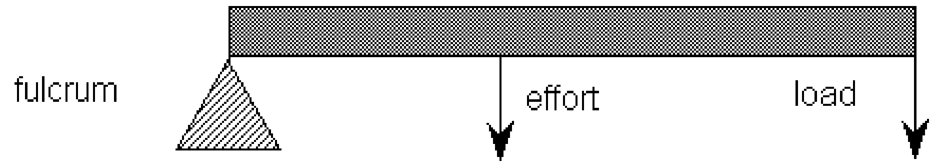
Wood Plane

Screw

Lever







Weighing Scale

Excavator

Scissors

Piano

Wheel and Axle

Waterwheel

Turbine

Windmill

Belt and Pulley

Chain Hoist

Crane

Elevator

Gears

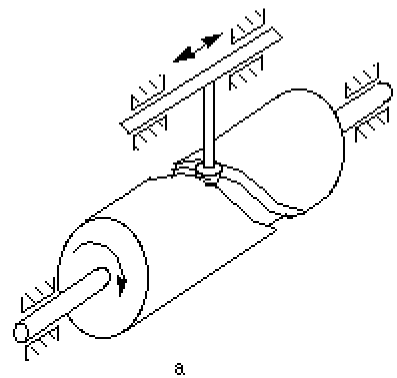
Bevel Gear

Spur Gear

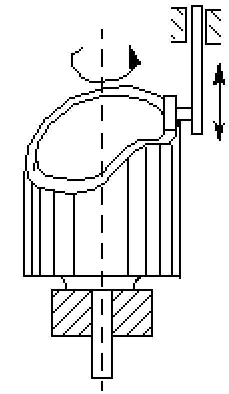
Helical Gear

Rack and Pinion

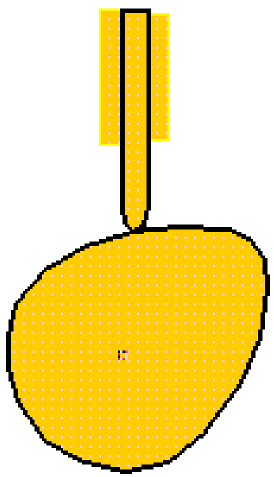
Cams



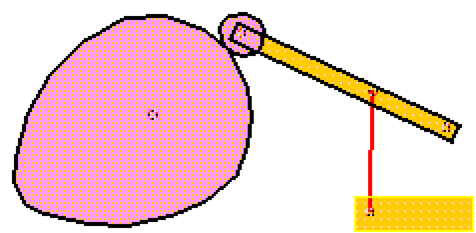
Cylindrical cam



End Cam



Translating Cam



Oscillating Cam

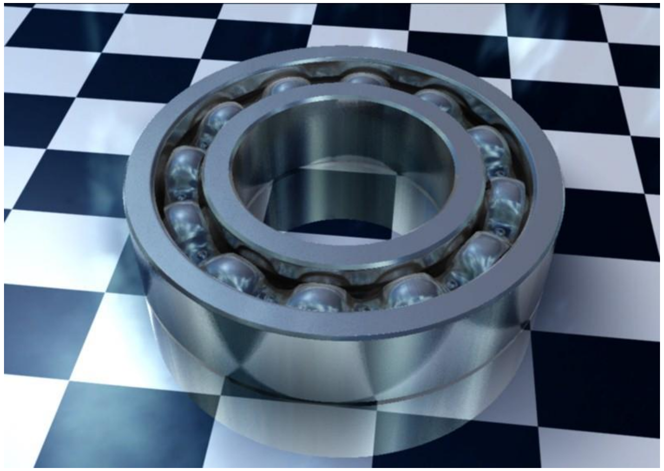
Springs

Leaf Spring

Washer Spring

Friction

Brake System



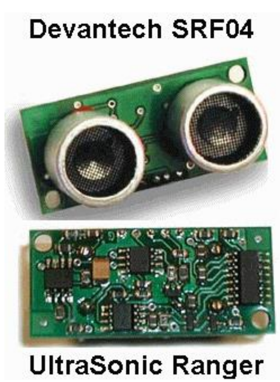
Bearing

Elements of Mechatronics—Electromechanical

- Electromechanical elements refer to:
  - Sensors
    - A variety of physical variables can be measured using sensors, e.g., light using photo-resistor, level and displacement using potentiometer, direction/tilt using magnetic sensor, sound using microphone, stress and pressure using strain gauge, touch using micro-switch, temperature using thermistor, and humidity using conductivity sensor
  - Actuators
    - DC servomotor, stepper motor, relay, solenoid, speaker, light emitting diode (LED), shape memory alloy, electromagnet, and pump apply commanded action on the physical process
- IC-based sensors and actuators (digital-compass, -potentiometer, etc.)

Flexiforce Sensor

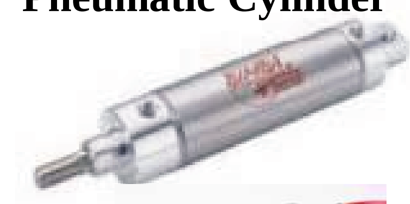
2-axis Accelerometer



GPS Receiver

Compass Module

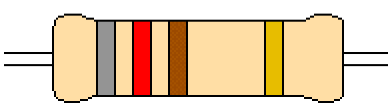
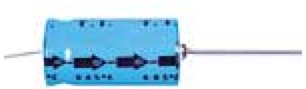
Pneumatic Cylinder

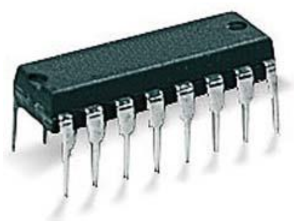
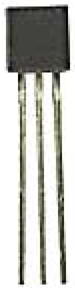


DC Motor

Elements of Mechatronics—Electrical/Electronic

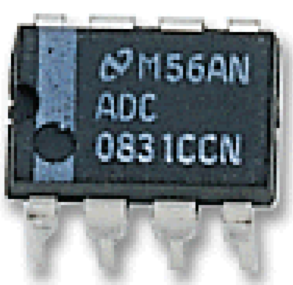
- Electrical elements refer to:
  - Electrical components (e.g., resistor (R), capacitor (C), inductor (L), transformer, etc.), circuits, and analog signals
- Electronic elements refer to:
  - analog/digital electronics, transistors, thyristors, opto-isolators, operational amplifiers, power electronics, and signal conditioning
- The electrical/electronic elements are used to interface electro-mechanical sensors and actuators to the control interface/computing hardware elements

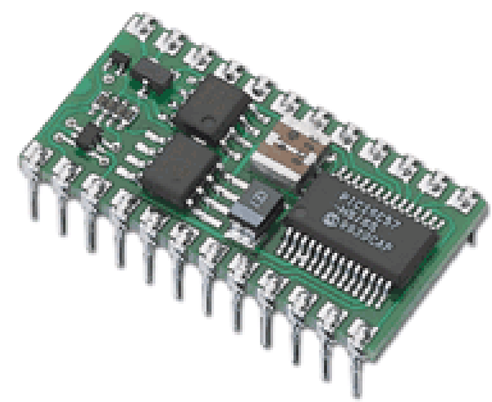



Elements of Mechatronics—Control Interface/Computing Hardware

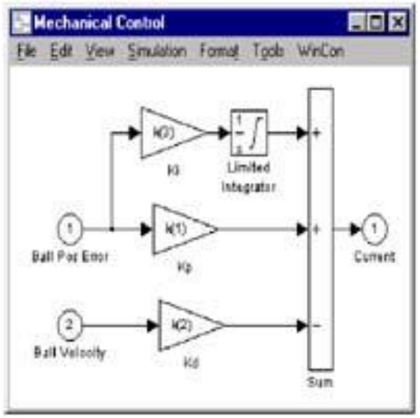
- Control interface/computing hardware elements refer to:
  - Analog-to-digital (A2D) converter, digital-to-analog (D2A) converter, digital input/output (I/O), counters, timers, microprocessor, microcontroller, data acquisition and control (DAC) board, and digital signal processing (DSP) board
- Control interface hardware allows analog/digital interfacing
  - communication of sensor signal to the control computer and communication of control signal from the control computer to the actuator
- Control computing hardware implements a control algorithm, which uses sensor measurements, to compute control actions to be applied by the actuator

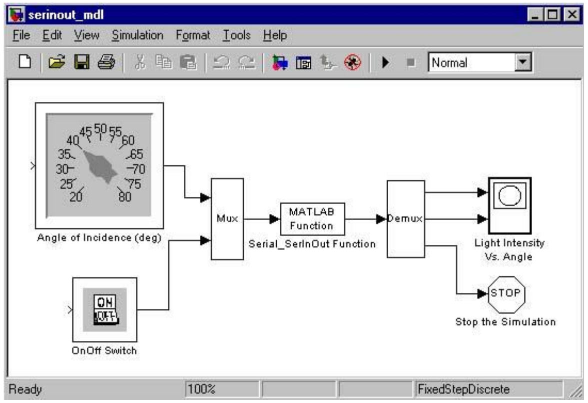




Elements of Mechatronics—Computer/Information System

- Computer elements refer to hardware/software utilized to perform:
  - computer-aided dynamic system analysis, optimization, design, and simulation
  - virtual instrumentation
  - rapid control prototyping
  - hardware-in-the-loop simulation
  - PC-based data acquisition and control



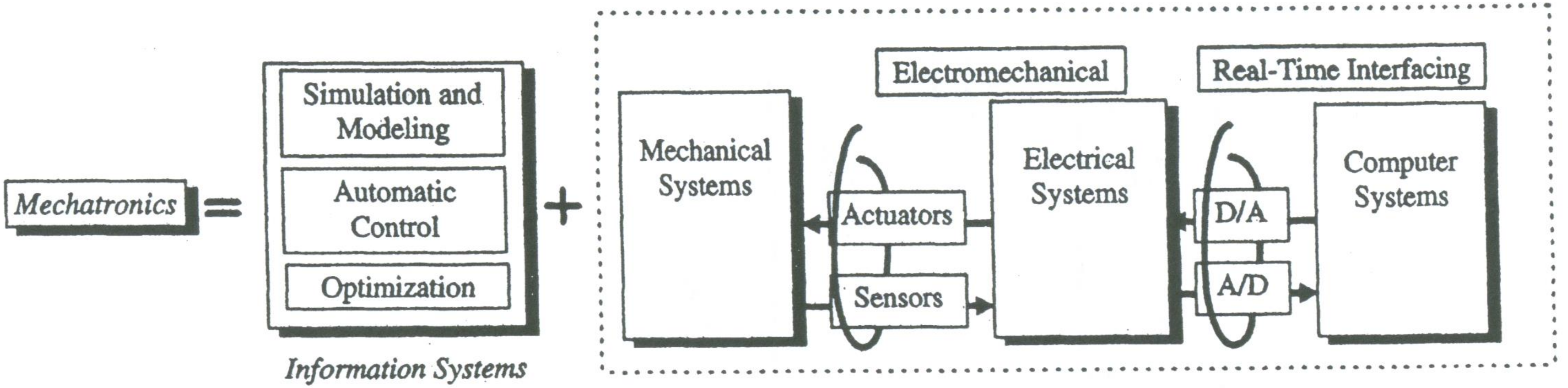


Elements of Mechatronics

- Typical knowledgebase for optimal design and operation of mechatronic systems comprises of:
  - Dynamic system modeling and analysis
    - Thermo-fluid, structural, hydraulic, electrical, chemical, biological, etc.
  - Decision and control theory
  - Sensors and signal conditioning
  - Actuators and power electronics
  - Data acquisition
    - A2D, D2A, digital I/O, counters, timers, etc.
  - Hardware interfacing
  - Rapid control prototyping
  - Embedded computing

Balance theory, simulation, hardware, and software

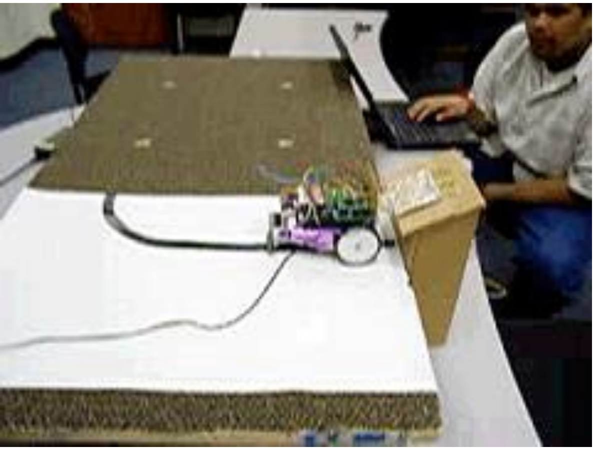
Key Elements of Mechatronics

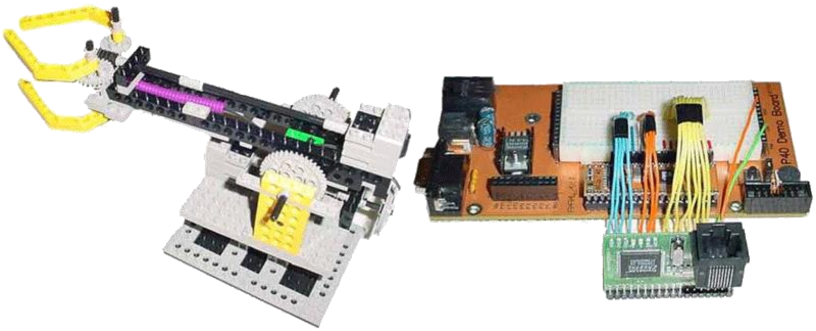


Mechatronics Applications

- Smart consumer products: home security, camera, microwave oven, toaster, dish washer, laundry washer-dryer, climate control units, etc.
- Medical: implant-devices, assisted surgery, haptic, etc.
- Defense: unmanned air, ground, and underwater vehicles, smart munitions, jet engines, etc.
- Manufacturing: robotics, machines, processes, etc.
- Automotive: climate control, antilock brake, active suspension, cruise control, air bags, engine management, safety, etc.
- Network-centric, distributed systems: distributed robotics, tele-robotics, intelligent highways, etc.

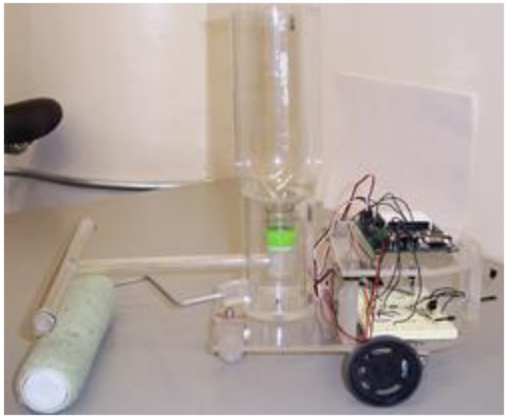
Sample Mechatronics Projects



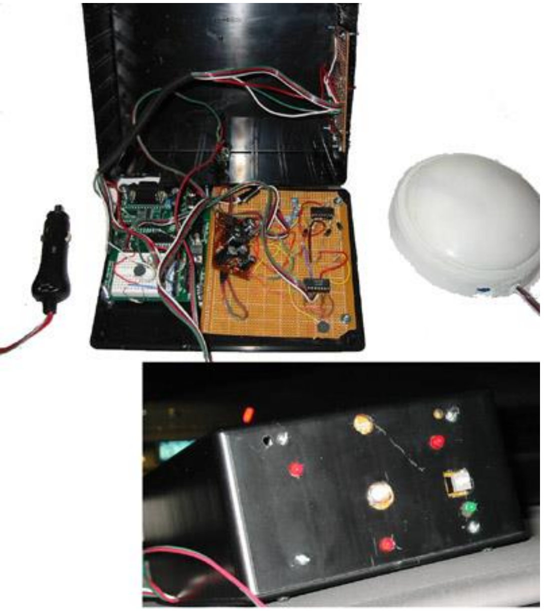


Remote Robot Arm Manipulation

Smart Irrigation System



Autonomous Polyurethane Applicator



Safe N Sound Driver

Smart Cane

Real-World Advanced Robots

Concrete Floor Surface Finishing Robot (Takenaka Corporation)

Autonomous Harvester (NREC @ CMU)

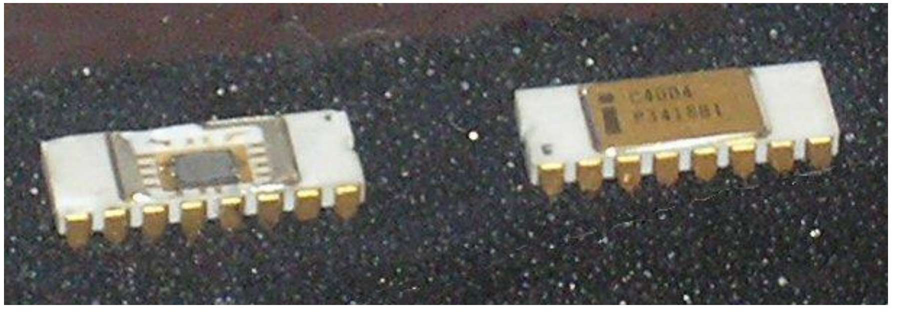
Robo-Mule (Boston Dynamics)

Robotic Excavator

Robo Mine-Sweeper

Microprocessor (μP) — I

- μP is a single VLSI (Very Large Scale Integration) chip.
  - It contains many digital circuits.
  - It is a complete computation engine fabricated on a single chip.
- Elements of a μP are:
  - Arithmetic/logic unit (ALU)
  - Instruction registers and decoders
  - Data registers
  - Control unit

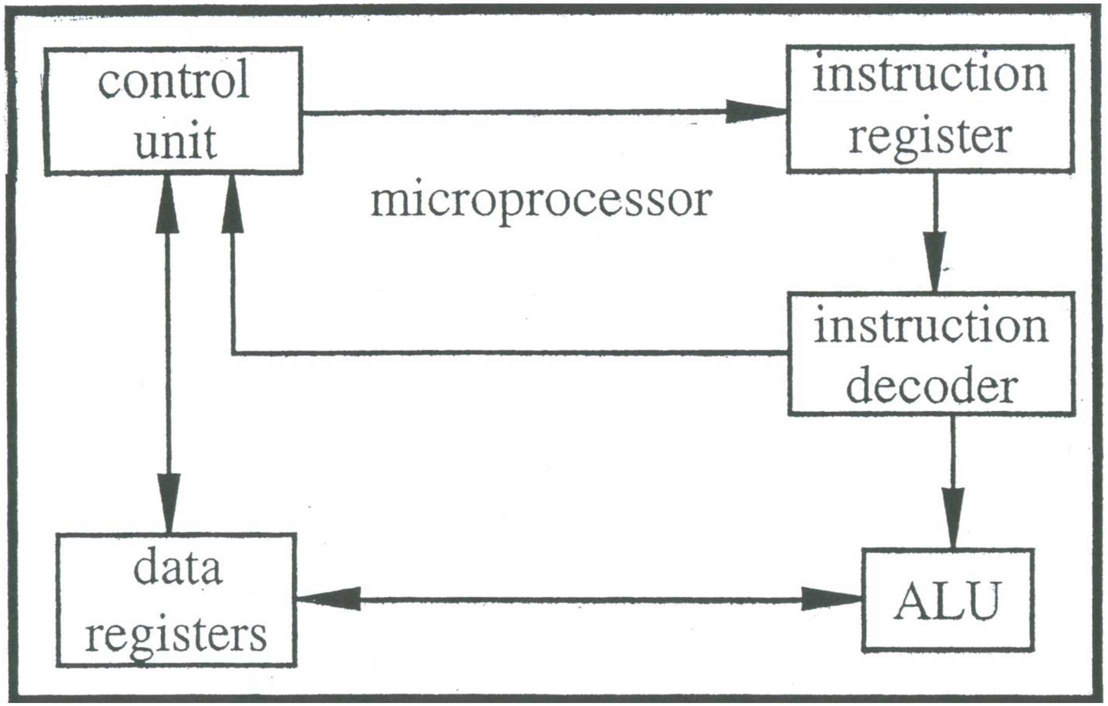


Intel 4004 (right)
Cover removed (left)

Microprocessor (μP) — II

- μP constitutes the central processing unit (CPU) of a microcomputer
- μP is programmed using:
  - Machine language
  - Assembly language
- Examples of μP:
  - Intel
    - 4004 (4-bit)
    - 8080, 8085 (8-bit)
    - 8088 (16-bit)
    - 80x86 (32-bit)
    - Pentium X (32-bit)
  - Motorola
    - 6800
  - Zilog
    - Z80

Microprocessor Architecture



How it Works

- Instructions tell ALU 2 things:
  - What to do
  - What data to do it to
- Data is IDed by its address
- Instructions are given in codes (that activate appropriate digital circuits in the control unit)

Microprocessor History — I

- First μP was introduced by Intel in 1971.
  - Intel 4004 (4-bit)
  - Killer app: portable calculators.
- Next came Intel's 8080 (8-bit) in 1974.
  - Killer app: desktop computers
- First major μP was 8088 (16-bit) introduced in 1979.
  - Killer app: IBM desktop computers in 1982
- Historical aside: Prior to the development of μPs, computers were built using discrete ICs!

Microprocessor History — II

Name	Date	Transistors	Microns	Clock speed	Data width	MIPS
8080	1974	6,000	6	2 MHz	8 bits	0.64
8088	1979	29,000	3	5 MHz	16 bits 8-bit bus	0.33
80286	1982	134,000	1.5	6 MHz	16 bits	1
80386	1985	275,000	1.5	16 MHz	32 bits	5
80486	1989	1,200,000	1	25 MHz	32 bits	20
Pentium	1993	3,100,000	0.8	60 MHz	32 bits 64-bit bus	100
Pentium II	1997	7,500,000	0.35	233 MHz	32 bits 64-bit bus	~300
Pentium III	1999	9,500,000	0.25	450 MHz	32 bits 64-bit bus	~510
Pentium 4	2000	42,000,000	0.18	1.5 GHz	32 bits 64-bit bus	~1,700

Microprocessor Functions

- μPs perform 3 primary functions/operations.
  - Mathematical operations: add, subtract, multiply, divide.
  - Move data from one location to another.
  - Make decisions and redirect the program flow based on decisions (i.e., based on decisions, redirect the program to a new set of instructions).

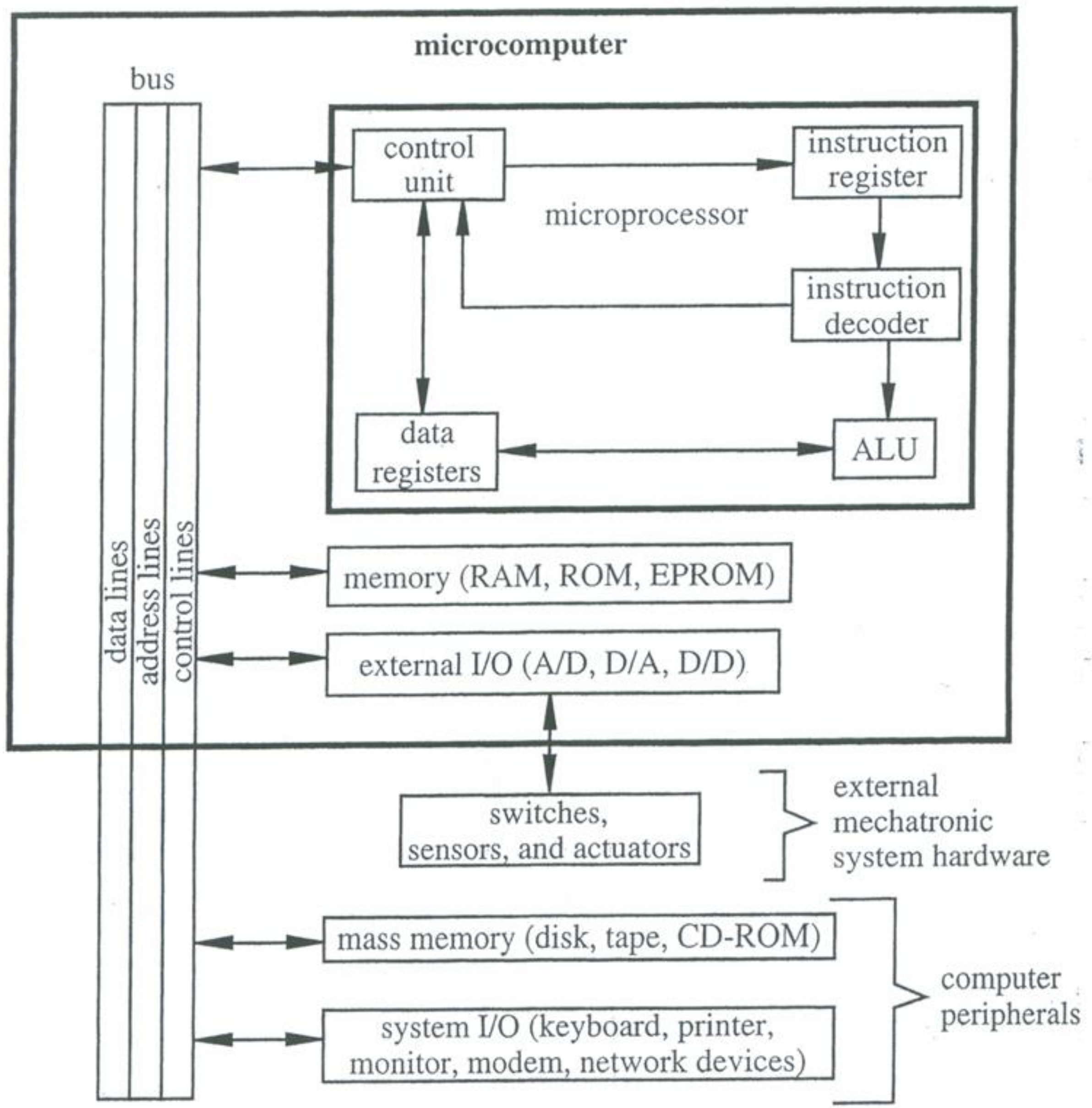
Microcomputer/Personal Computer/ Desktop Computer/Computer

- Computer is a μP packaged on a single circuit board with interfaces and memory chips.

- Types of computers: IBM and compatibles.
  - PC: “Personal computer” built around 8088/8086 μP with floppy disk drive.
  - XT: “Extended technology” built around 8088/8086 μP with hard disk drive.
  - AT: “Advanced technology” built around 80286 μP.

- Compared to μPs, computers are easy to program (use high level language of your choice, e.g., BASIC, FORTRAN, C, …).
- Modern day computers have significant computing power and excellent memory storage.
- PCs are purposefully designed to allow interaction and communication with humans.
- PCs are general purpose computing machines.

Microcomputer Architecture



Elements of PCs

- PCs consist of
  - μP
  - Memory
  - PC bus
  - External inputs/outputs (or peripherals/interfaces)
- Recall that the CPU of the μP performs the computing function.

Memory —RAM

- Random Access Memory (RAM)
  - Read/write volatile memory.
  - Volatile memory → its contents are lost when power is removed.
  - CPU can access contents of RAM at very high speed during program execution.
  - When a program is to be executed, it is temporarily stored in RAM for high speed execution.
  - All intermediate results of computations are also stored in RAM for fast retrieval.
  - RAM is usually quite costly since high speed access costs more money.

Memory —ROM

- Read Only Memory (ROM)
  - ROM is written onto only once at the factory.
  - Once ROM has been written onto it acts as a read only memory.
  - ROM is a nonvolatile memory → its contents are retained even when power is removed.
  - ROM stores program used by the microcomputer at its startup.
    - For example, on a PC Basic Input/Output System (BIOS) is stored in ROM.

Memory —Mass Storage

- The following fall under the category of mass storage memory:
  - Hard disk, floppy disk, tape drive, optical drive, etc.
  - This type of memory is both readable and write-able.
  - This type of memory is also nonvolatile.
  - This type of memory usually costs significantly less compared to RAM.
  - Downside: CPU takes significantly longer time to access the contents of mass storage memory *vis-à-vis* RAM.

PC Buses

- Industry Standard Architecture (ISA) bus dominated the PC market until mid to late 1990s.
- Micro-Channel Architecture (MCA) bus (IBM's proprietary design, used in PS/2)
- Extended ISA (EISA) bus
- Personal Computer Memory Card International Association (PCMCIA) bus
  - Originally intended for removable memory cards on portable computers.
  - Presently it is also used for interfacing miniature adaptor cards to PCs.
    - e.g., serial port, modem, network card, hard disk, data acquisition and control (DAC) card.
- Other buses: PCI (replaced ISA), IEEE 488, Centronics Parallel Port, Serial buses (RS232, RS422, RS485, USB), etc

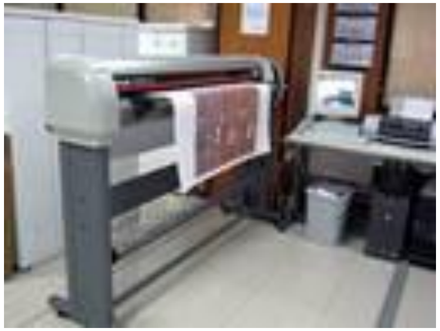
.

PC Expansion Buses Chart

Bus	*Address width (bits)*	*Data width (bits)*	*Standard clock rate (MHz)*	*Max. throughout at standard clock (MB/s)*	*Notes*
PC (XT)	20	8	8	8	Six IRQ lines. Three DMA channels.
ISA (AT)	24	16	8	16	Twelve IRQ lines. Seven DMA channels.
MCA	24	32	Variable (typically 10–20)	20–160	Maximum transfer rates achieved in data streaming mode. DMA implemented via bus mastering with up to 16 arbitrating devices.
EISA	32	32	8	33	Quoted throughput achieved in data streaming mode.
PCI	32	32 or 64	33 or 66	132[1]	Intelligent bus mastering with support for DMA. Quoted transfer rate is achievable in burst mode only.[1]

[1] For a 32-bit implementation running at 32 MHz. Maximum throughput increases proportionately for faster or wider versions of PCI.

External Interfaces/Peripherals



- Outputs such as:
  - Video monitor, speaker, printer, plotter.

- Inputs such as:
  - Keyboard, mouse, joystick, microphone, camera.

- These external interfaces of PCs allow humans to connect to, interact/communicate with the CPU.

Common Applications of PCs

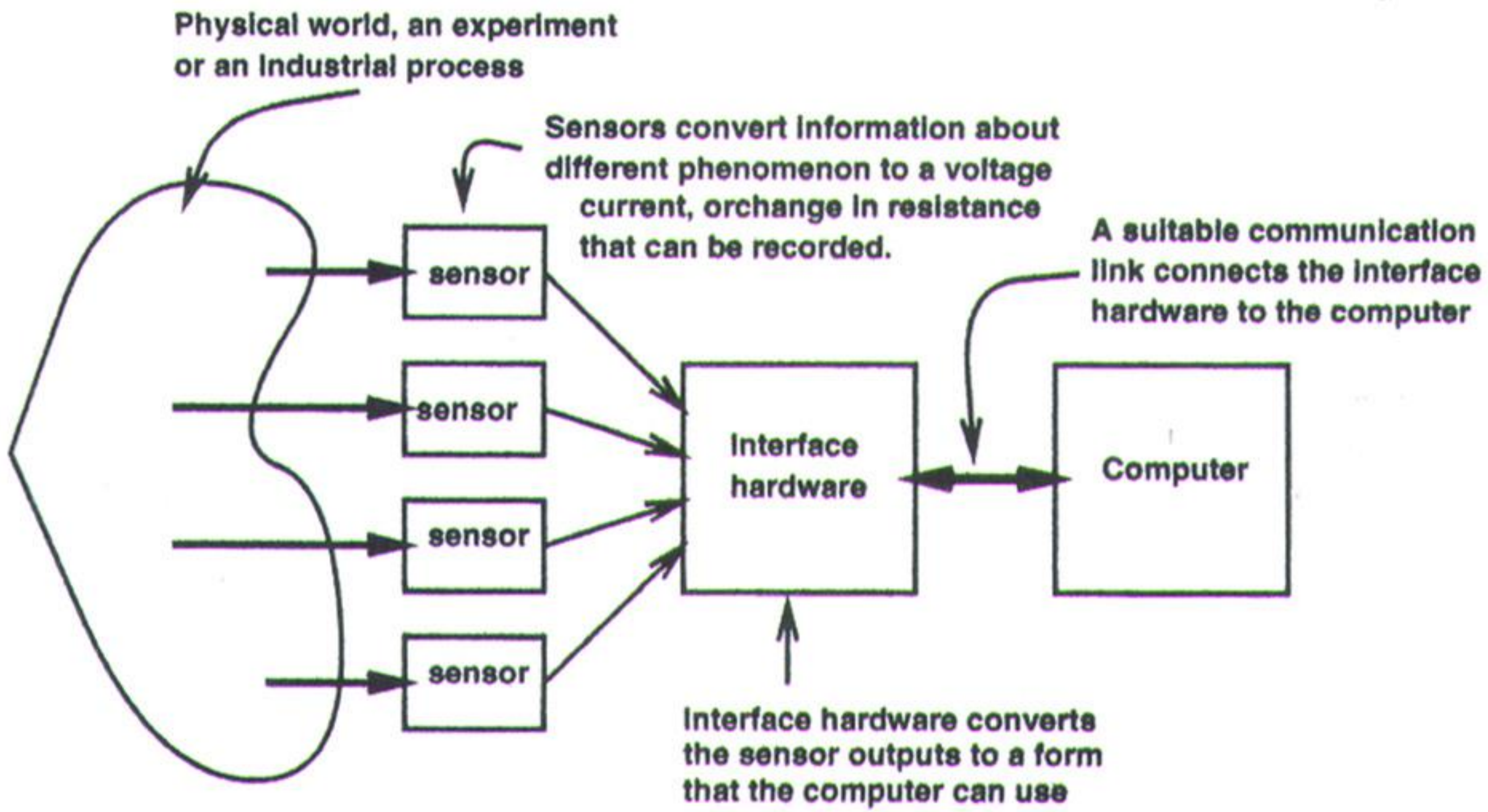
- Mathematical, scientific, engineering, and business computing
- Financial analysis
- Database management
- Word processing, desktop publishing
- Graphic design

Data Acquisition 101

- In common computer applications discussed earlier, *data* comes from a keyboard or a disk (where it is stored).
  - This type of data is prepared and entered in the computer by a human operator.
- Data acquisition (DAQ) has to do with getting data into computer from a real-world process *automatically* without human intervention.
  - DAQ is more than simple automation of the process of acquiring data.
- DAQ refers to automatic acquisition of real-world sensory information wherein
  - acquired data may be stored for later use
  - acquired data may be analyzed in real-time
  - acquired data may need on-line processing
  - acquired data may have to be presented in real-time
- DAQ is used for test instruments, condition monitoring of industrial machinery, process industry, medical instruments, environment monitoring, robotics, etc.
- DAQ can be used to develop virtual instruments for productivity enhancement.

PC-Based DAQ: A Pictorial Representation

Figure *Block diagram of a typical automated data acquisition system.*



PC-Based DAQ: Pros/Cons

- Advantages:
  - PC provides a flexible solution to DAQ problem since the it can adapt to changing needs of DAQ applications/users.
  - PCs have a wide market base.
  - Human resources with technical background in PC use readily available.
  - Numerous packages exist for scientific, mathematical, engineering, and statistical computing/analysis that could be readily used to analyze acquired data.
- Disadvantage:
  - PC platform is not ideal for DAQ applications requiring guaranteed/deterministic acquisition time.
  - Aside: The above drawback may be overcome to some extent using real-time kernel programs that allow real-time process to run with a high priority.

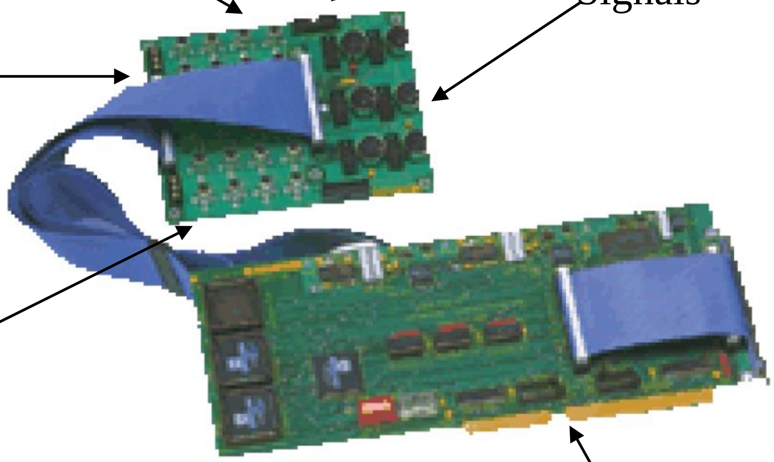
PC-Based DAQ: Selection Considerations

- Required data acquisition speed (slow versus very high): availability of corresponding sampling rate
- Data collection is event driven
- Connection to internet, intranet, telephone
- PC interface (Serial port, parallel port, PC bus, communication)
- Signal conditioning
- Number of analog input channels, voltage ranges, digital I/Os
- Resolution (# of bits of ADC)
- Accuracy
- Isolated inputs
- Plug-in DAQ cards (ISA, EISA, MCA, PCI, PCMCIA, …)
- Stand alone DAQ cards (communication-based)

Plug-in versus Stand-alone DAQ Cards

- Plug-in:
  - Easy installation and configuration
  - New plug-in cards even have plug-and-play capability
  - # of available slots in a PC limits extensibility of plug-in cards
  - Lack of robustness, ruggedization, and portability needed in a shop floor environment
- Stand-alone:
  - Processor independent can be connected to a PC or a *PLC*
  - Ruggedized to operate in hostile environment
  - Portable
  - Communicate to PC via Serial port, parallel port, TCP/IP, etc

Example: Plug-In & Stand Alone



MultiQ DAC Board

LabPro

Features of Plug-in DAQ Cards

Table *Features of plug-in data acquisition boards*

Least expensive method of computerized measurement and control.
High speed available (100kHz to 1GHz and higher).
Available in multi-function versions that combine A/D, D/A, digital I/O, counting, timing, and specialized functions.
Good for tasks involving low-to-moderate channel counts.
Performance adequate to excellent for most tasks, but electrical noise inside the PC can limit ability to perform sensitive measurements.
Input voltage range is limited to approximately ±10V.
Use of PC expansion slots and internal resources can limit expansion potential and consume PC resources.
Making or changing connections to board's I/O terminals can be inconvenient.

Features of Stand-alone DAQ Cards

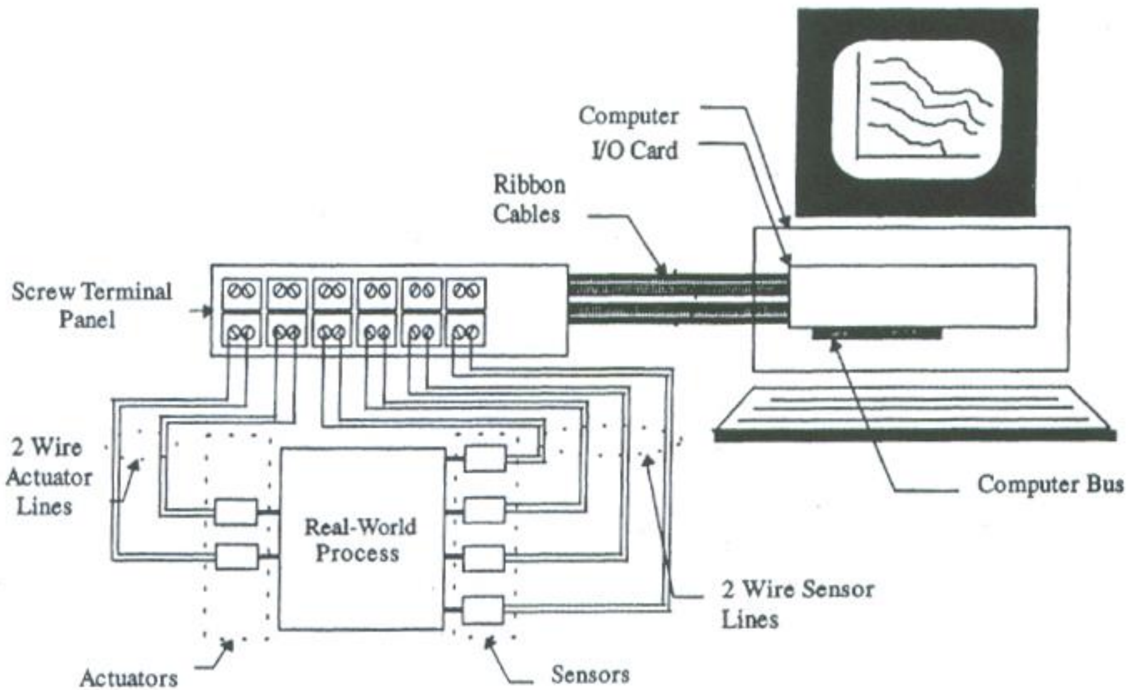
Table Features of external data acquisition chassis

Multiple board slots permit mixing-and-matching boards to support specialized acquisition and control tasks and higher channel counts.
Chassis offers an electrically quieter environment than a PC, allowing for more sensitive measurements.
Use of standard interfaces (IEEE-488, RS-232, USB, FireWire, Ethernet) can facilitate daisy chaining, networking, long distance acquisition, and use with non-PC computers.
Dedicated processor and memory can support critical “real-time” control applications or stand-alone acquisition independent of a PC.
Standardized modular architectures are mechanically robust, easy to configure, and provide for a variety of measurement and control functions.
Required chassis, modules, and accessories are cost-effective for high channel counts.
Some architectures have minimal vendor support, limiting the sources of equipment and accessories available.

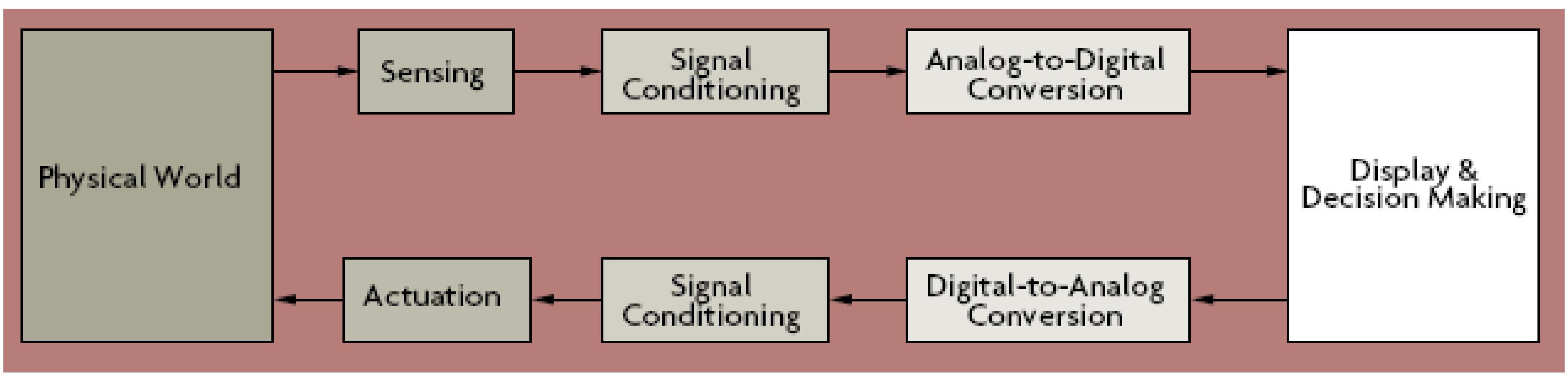
Data Acquisition and Control 101

- DAQ systems enable one to collect real-world data into the computer.
- DAQ systems are useful for monitoring and data analysis.
- On the other hand, if one needs to command a real-world device into action based on the measurement of some real-world phenomenon, then a DAQ system is not sufficient.
  - In this case one needs a data acquisition and control (DAC) system.
- A DAC system
  - collects data from sensors,
  - using computing resources of a PC or an on-board computer
    - processes sensory information
    - computes control command, and
  - commands control actuators.

DAC System: A Pictorial Representation



Major components of a DAC system and their interconnection for a system with four sensors and two actuators



Functional Diagram for a DAC system

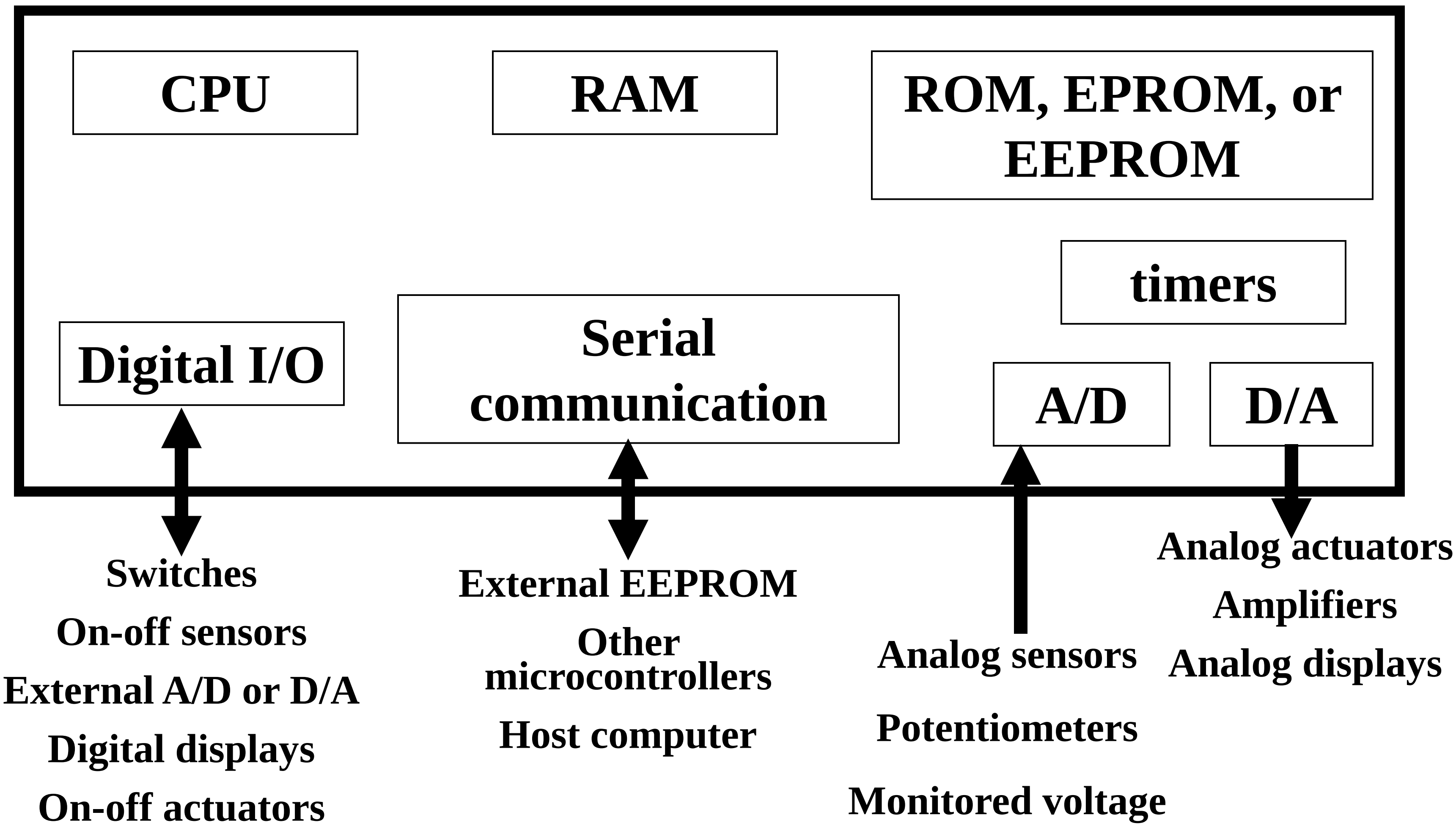
Microcontroller (μC)

- μC is a special purpose microcomputer.
- μC is a single IC containing many specialized and sophisticated circuits and functions.
- μCs are used in embedded computing applications to impart computing, decision making, and intelligence capabilities to machines, products, and processes.
- Whereas general purpose computers, i.e., PCs, are designed explicitly to interface with humans, μC are purposely designed to interface, interact, and communicate with
  - electrical/electronic devices;
  - sensor/actuators;.
  - machinery/equipment;
  - high-tech gadgets; etc.

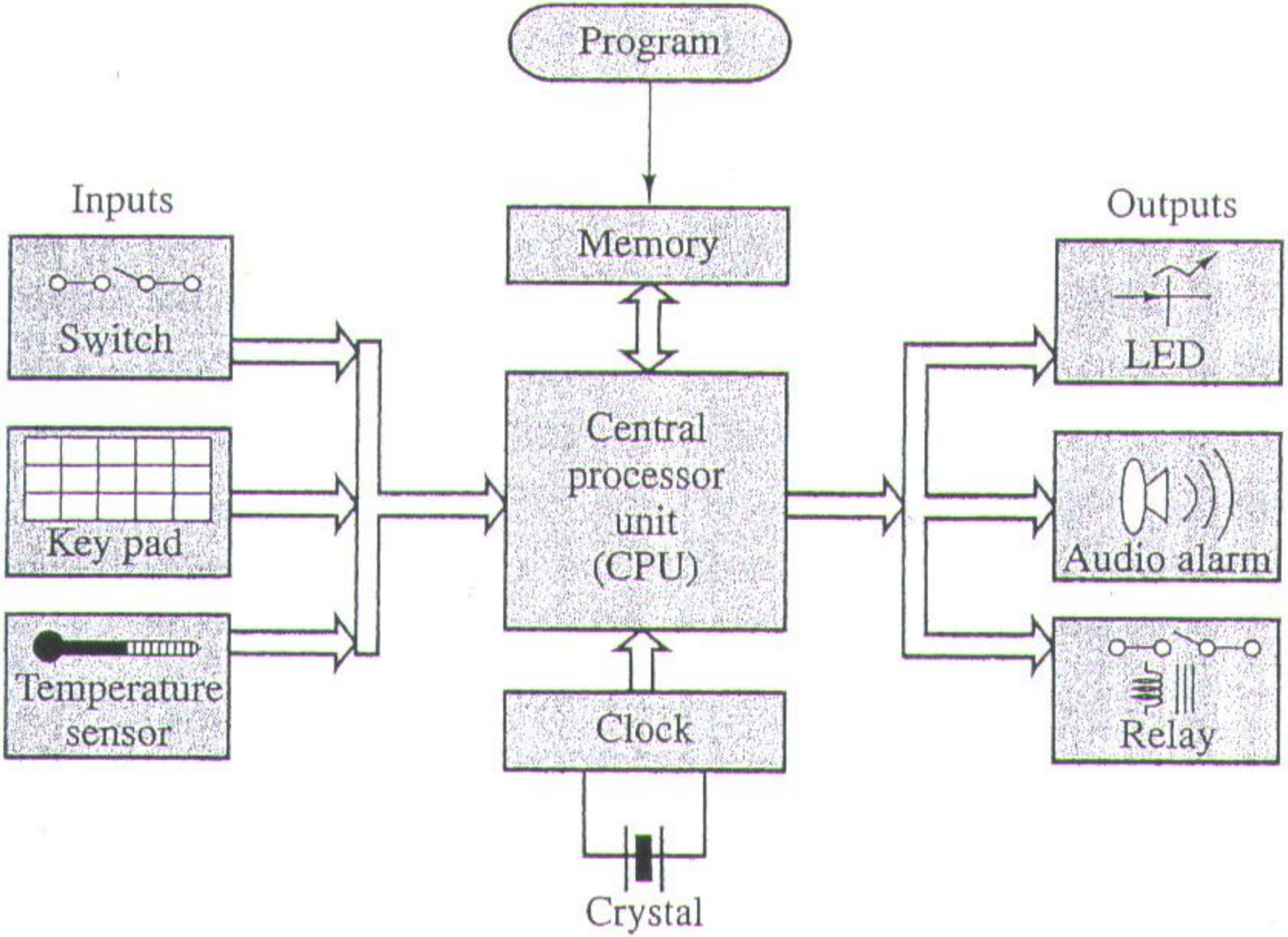
μC Elements

- According to one standard, a μC must contain at least two primary components:
  - RAM
  - CPU with instruction set (list of all commands and their corresponding functions)
- μCs may additionally have one or more of the following elements.
  - Various types of memory (ROM, EPROM, EEPROM, PROM)
  - I/O functionality
    - Digital I/O (to connect switches, LEDs, on-off sensors and actuators, etc.)
    - Analog I/O (high end μCs have built-in A2D, D2A, etc.)
    - Pulse-width-modulation outputs (PWM) to connect PWM-type actuators
    - Clock generators
    - Reset function
    - Watchdog timer (resets controller after a problem occurs)
    - Interrupt
    - Timers
    - Communication (RS-232, I^2C, 1-wire, etc.)

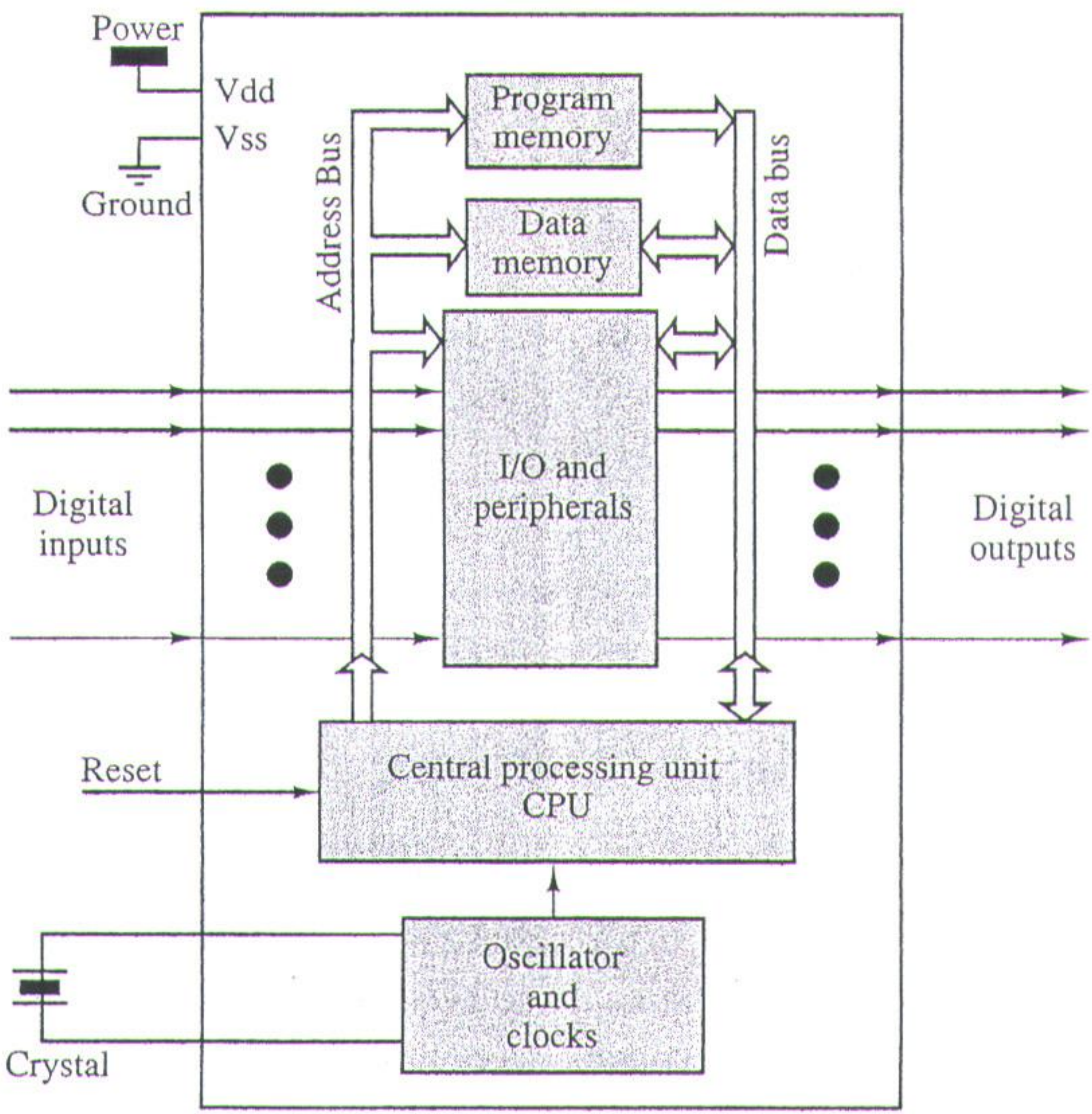
Components of a full featured μC



High Level Block Diagram of μC



μC Architecture

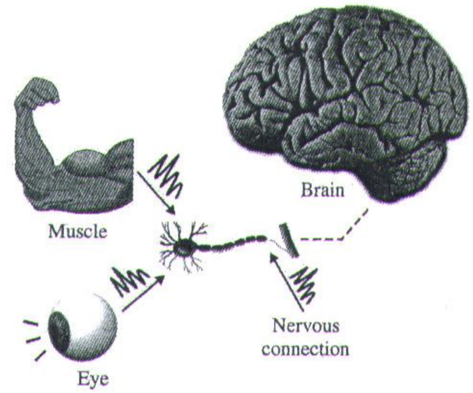


Some Applications of µCs

- Consumer products:
  - Microwave oven (key pad as input device, LCD as an output device, and of course control of microwave oven it self)
  - Smart washing machine (control wash cycle by monitoring quality of drain water and control dry cycle by monitoring humidity level of outlet air)
  - VCR, cell phone, home automation, security, etc.

- Medical electronics
- Aerospace
- Automobiles
- Manufacturing industry
- Laboratory automation

μC and Human Brain Analogy



- Humans have 5 ***senses***: touch, smell, taste, hearing, vision.
- In real-world, humans interact with their environment by sensing the environment:
  - e.g., a person driving a car *sees* (eye is vision sensor) a herd of sheep on a road.
- Human sensory organs transmit *sensed info* to brain via neural pathways that connect sensory organ to the brain.
- Human ***brain*** processes the sensory signal and undertakes a ***decision*** to steer the car to avoid collision with the sheep.
- Human brain sends this signal to
  - ***Muscles*** in the driver's legs, arms, hands, etc., through neural pathways.
  - The driver steers the car and avoids collision with the sheep.
- Similar to human brain, a μC performs decision making and it interacts with the environment through sensors and actuators that are connected to it. Thus, a μC like a human brain imparts intelligence to unanimated objects.

μC Function

- Microcontrollers are specifically designed to:
  - Read input devices, such as buttons and sensors.
  - Process data or information.
  - Control output devices, such as lights, displays, motors and speakers.

Why use μCs for Mechatronic Applications?

- Cost
- Processing power
- Flexibility

Cost Issues —I

- Cost of a general purpose desktop PC: approx. $500+.
- Plug-in DAC/stand-alone DAC cost: several hundred dollars to over a thousand dollars (each A2D channel costs $75+).
- Software: DAQ/DAC drivers, data processing/analysis software (LabVIEW, Simulink-RTW, etc.), real-time kernel for Windows O.S. (e.g., VentureCOM) — easily over $500.
- Is it appropriate to spend all this amount to impart intelligence to a $400 washing machine?
- Although one of a kind mobile robot may demand and justify this type of expense, the standard desktop PC form factor is not suitable for mobile robotics application.
  - If one alternatively selects PC104-based DAC hardware, costs are even higher (low demand, low volume → higher cost).

Cost Issues —II

- Possible alternative: Application specific integrated circuits (ASIC) instead of PC-based DAQ/DAC.
- ASICs require engineering design for each particular problem at hand from scratch.
  - Engineering design fee
  - Custom-made IC die
- Costs of ASICs can be several tens to hundreds of dollars depending on production volume.
- Is it appropriate to install a $30 brain on a $20 toaster to add intelligence to it?
  - In consumer products costs issue is very sensitive since one way to operate, survive, thrive, and profit in an increasingly competitive marketplace is to keep prices low.
  - Even when a company innovates by adding intelligence to an otherwise routine consumer product, it has to do so with smallest cost increase!
- ASICs may not provide solution to the aforementioned cost problem.

Cost Issues —III

- Consider use of μC
- Peripheral Interface Controllers (PIC) from Microchip Inc.
  - PIC16C57
    - Qty 1: $7.50
    - Qty 100: $4.50
    - Qty of several 1000: $3.50
- Consider 8049 μC which costs approx. $0.95
- Finally circuits built using discrete logic components are not very versatile and usually more expensive than μC.

Processing Power Issue

- Processing power issues:
  - Clock speed
  - # of clock cycles to execute one instruction
  - Instruction set
- A diverse variety of μCs are available with variations in clock speed, instruction set, etc. Depending on an application at hand, one needs to select a suitable μC.
- How?
  - μC A has 10 MHz clock speed, it takes 10 clock cycles to execute each instruction → μC A processes each instruction @ the speed of 1 MHz.
  - μC B has 5 MHz clock speed, it takes 2 clock cycles to execute each instruction → μC B processes each instruction @ the speed of 2.5 MHz.
  - Choice between A and B?

Instruction Set

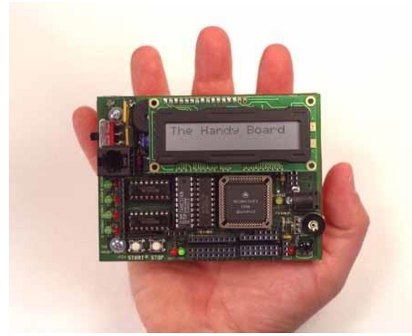
- If a particular μC has a sufficiently complete instruction set it may be that case that
  - One can perform operations such as multiply and divide using just one instruction.
  - Clock speed
- Alternatively, if another μC of same processing speed per instruction does not have a complete instruction set, then it may be the case that multiply and divide operations require multiple instructions → slower speed.
- Sufficiently complete instruction set will require a larger ROM to store the instruction set.
  - Usual design trade-off (no free lunch).

Flexibility of μC

- A μC-based design provides the ability to fix design after it has been completed.
- A μC provides flexible firmware thus μC functionality can be changed by downloading new firmware.
- A μC represents a flexible hardware to which additional devices can be added on the unused pin. Not only that I/O pins can be reassigned with relative ease.

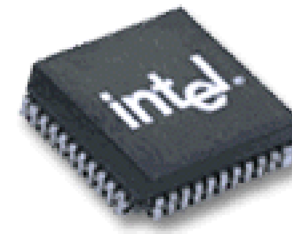
μC Manufactures/Vendors

Microchip: PIC series μC



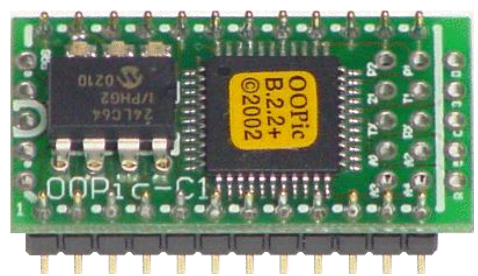
Motorola: 68HC11

(MIT Handy board)



Intel: 8096, MCS 96296

Dallas Semiconductor: TINI



OOPic

BasicX

- Aside:
  - Each μC manufacturer has a diverse variety of μCs with differing parts, features, and functionality.
  - The various μCs are marketed as: low end, mid range, and high end.

Arduino

μC Selection

- Requirement of task at hand
  - Memory, speed, I/O, other interfaces, floating point computation
- Availability of software development tools
  - Emulators, programmers, compilers
- Availability of parts and components
- Existing user base and support services
- Availability of application notes
- Power requirements
- Clocking and reset requirements
- Cost

PIC μC

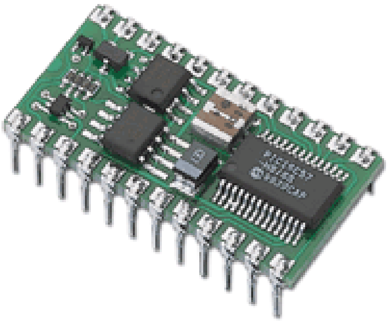
- Peripheral Interface Controller (PIC)
- Code developed for low-end PIC μC is upward compatible
- PIC device packaging
  - Dual Inline Package (DIP): plastic package containing one-time programmable (OTP) EPROM
  - Ceramic package with glass window (JW): contains EPROM which is erased by exposing glass window to UV light
  - Surface mount technology: component pins are soldered on raw card
  - Chip on board technology

Memory Types

- Erasable Programmable ROM (EPROM): OTP DIP package and UV erasable and re-programmable (JW). Use for fast access.
- Electrically EPROM (EEPROM): No need of UV light exposure for erasing program. In BS2 we download our program to EEPROM.
- ROM: Program built into the chip at the factory. μC instruction set is stored in ROM.
- PIC's CPU: Harvard architecture
  - Variable and I/O register/memory are separate from program memory.
- PIC series μCs:
  - 8bit: PIC10, PIC12, PIC16, PIC18
  - 16 bit: PIC24F, PIC24H, dsPIC30, dsPIC33
  - 32bit: PIC32

Basic STAMP μC

- Name originates from μC's footprint approx. the size of a typical postage stamp
- Simple and easy to use μC platform
- Uses a PIC-based PBasic interpreter
  - PBasic is a form of programming language adapted from Basic for embedded computing applications (**P**arallax **Basic** → **PBasic**)
- Two main lines:
  - Basic Stamp 1 (BS1) introduced in 1992
  - Basic Stamp 2 (BS2) introduced in 1995 (our focus)



Basic STAMP: Overview

- 9 volt battery power or 9-12 volt wall adapter
- Digital I/Os: 16 TTL compatible devices can be interfaced
- All 16 digital I/Os are reprogrammable
  - Under user program control, direction of any I/O pin can be changed
  - That is, a pin can be changed from input pin to output pin, and vice-versa, on the fly
- Package: 24 pin DIP module useful for hobby purpose, educational environment, etc. Other packages are available for OEMs.
- BS2 is based on PIC16C57 μC
  - PIC16C57 runs at 20 MHz (5 MIPS)
  - BS2 executes 3000 PBasic instructions per second

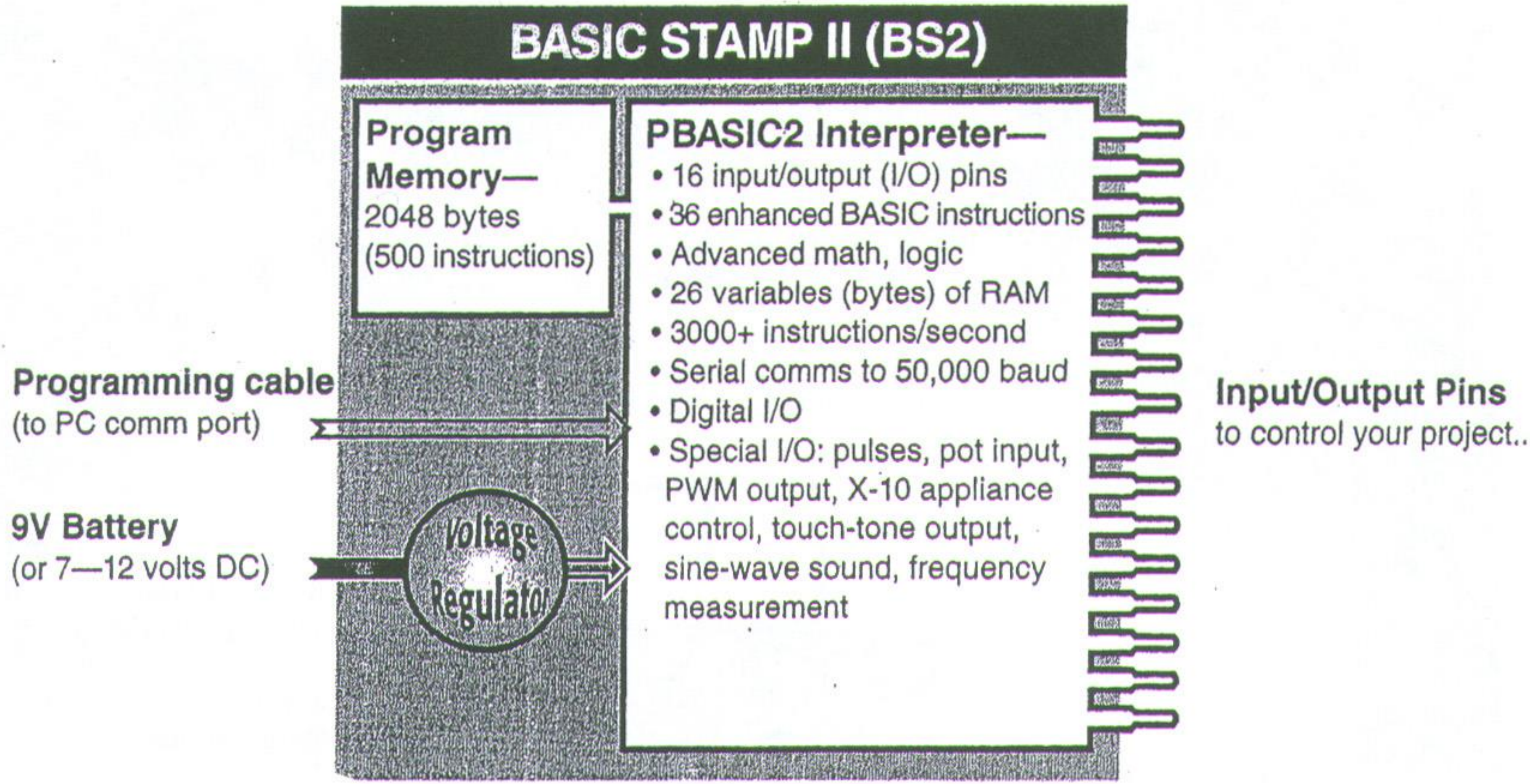
Basic STAMP: PBasic interpreter

- PIC16C57 serves as the hardware platform
- Parallax's firmware, i.e., program, is stored on PIC16C57 ROM and turns the PIC hardware into an interpreter chip
- PBasic language instruction set is permanently stored on PIC's ROM

Basic STAMP: Memory

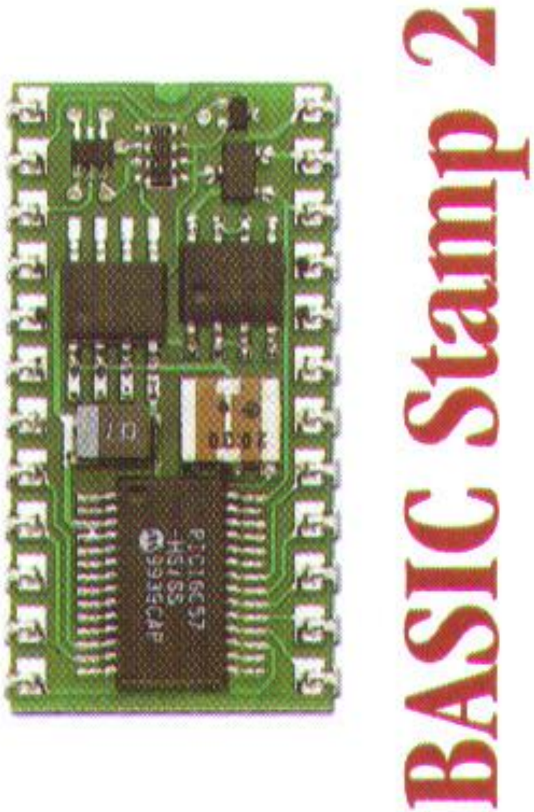
- Memory for storing user programs?
  - PIC’s ROM used up for storing interpreter firmware
  - Use a non-volatile EEPROM to store user programs (size: 2048 bytes ≅2KB)
  - 2KB EEPROM can store up to 500 lines of PBasic code
  - If all of 2KB is not used for code storage, then unused portion can store field data, even on long-term basis!
  - After the program is downloaded, the program stays in EEPROM even after power is removed
  - EEPROM can be reprogrammed 10 million times
  - Only one program can be stored in EEPROM at a given time

BS2 Specs —I



BASIC STAMP II (BS2)
Program Memory—
2048 bytes
(500 instructions)
PBASIC2 Interpreter—
• 16 input/output (I/O) pins
• 36 enhanced BASIC instructions
• Advanced math, logic
• 26 variables (bytes) of RAM
• 3000+ instructions/second
• Serial comms to 50,000 baud
• Digital I/O
• Special I/O: pulses, pot input, PWM output, X-10 appliance control, touch-tone output, sine-wave sound, frequency measurement
Programming cable
(to PC comm port)
9V Battery
(or 7—12 volts DC)
Voltage Regulator
Input/Output Pins
to control your project...

BS2 Specs —II



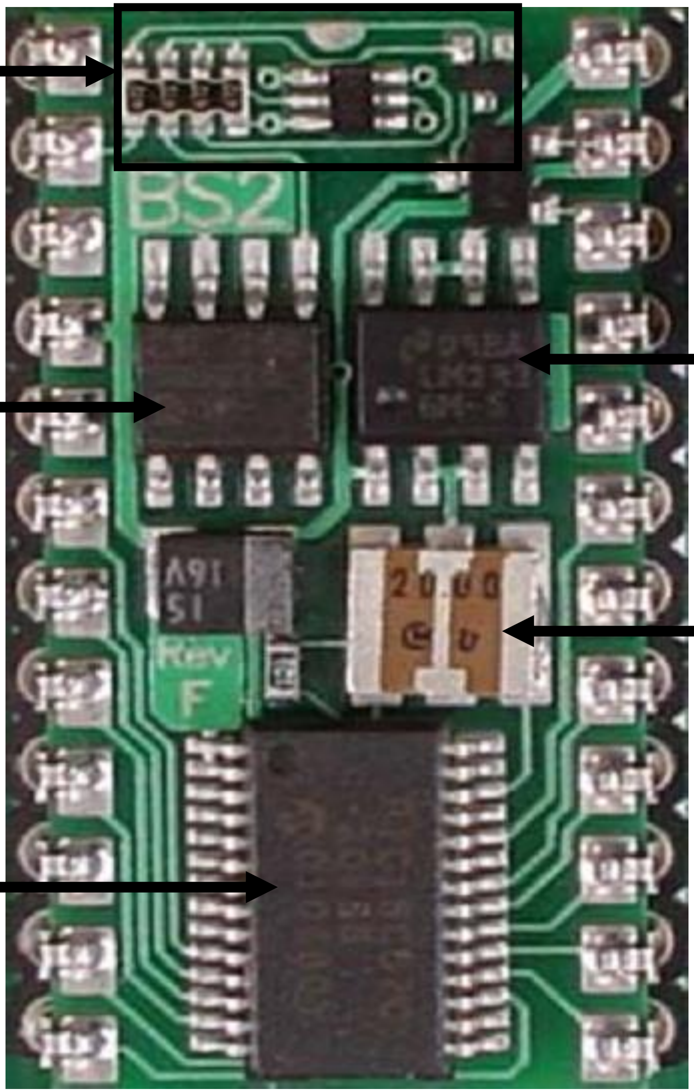
Package:	24-pin DIP
Environment:	0°-70° C (32 -158° F)
Microcontroller:	Microchip PIC16C57
Processor Speed:	20 MHz
Program Execution Speed:	~4,000 instructions/sec
RAM Size:	32 Bytes (6 I/O,26 Variable)
EEPROM (Program) Size:	2K Bytes,~500 instructions
Number of I/O Pins:	16 + 2 Dedicated Serial
Voltage Requirements:	5 - 15 vdc
Current Draw @ 5V:	8 mA Run / 100 µA Sleep
Source/Sink Current per I/O:	20 mA / 25 mA
Source/Sink Current per unit:	40 mA / 50 mA per 8 I/O pins
PBASIC Commands:	36
PC Programming Interface:	Serial Port (9600 baud)
DOS Text Editor:	STAMP2.EXE
Windows Text Editor:	Stampw.exe

BS2 Anatomy —I

Serial Signal Conditioning
Conditions voltage signals between PC serial connection (+/- 12V) and BASIC Stamp (5V)

EEPROM
Stores the tokenized PBASIC program.

Interpreter Chip
Reads the BASIC program from the EEPROM and executes the instructions.



5V Regulator
Regulates voltage to 5V with a supply of 5.5VDC to 15VDC

Resonator
Sets the speed at which instructions are processed.

BS2 Anatomy —II

Pin 1: S_{OUT}

Transmits serial data during programming and using the DEBUG instruction

Pin 2: S_{IN}

Receives serial data during programming

Pin 3: ATN

Uses the serial DTR line to gain the Stamps attention for programming.

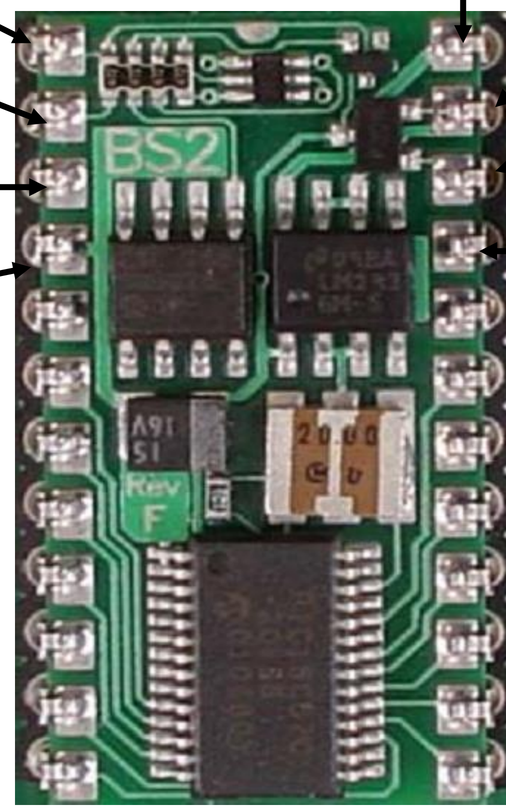
Pin 4. V_{SS}

Communications Ground (0V).

Pins 5-20:

Input/Output (I/O) pins P0 through P15

P0
P1
P2
P3
P4
P5
P6
P7



Pin 24. V_{IN}

Un-regulated input voltage (5.5-15V)

Pin 23. V_{SS}

Ground (0V)

Pin 22. RES

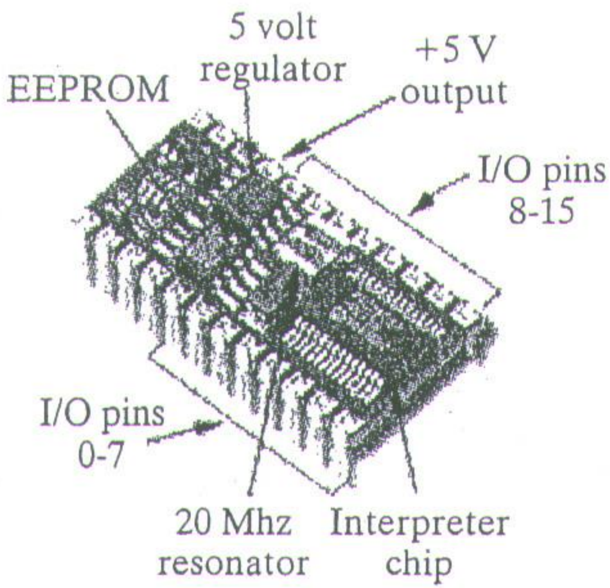
Reset- LOW to reset

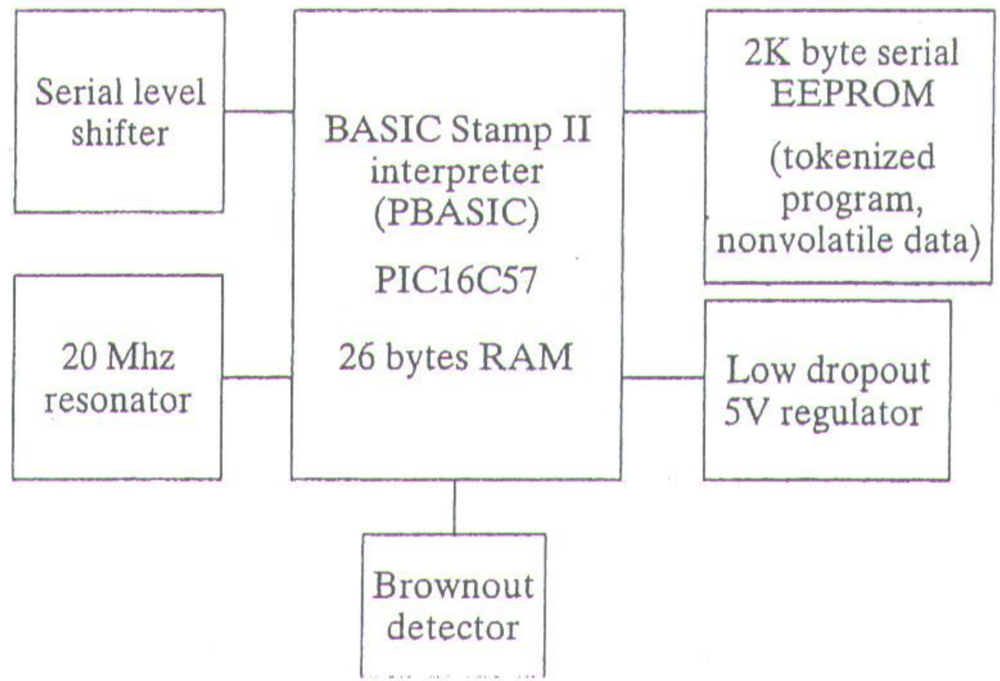
Pin 21. V_{DD}

Regulated 5V.

P15
P14
P13
P12
P11
P10
P9
P8

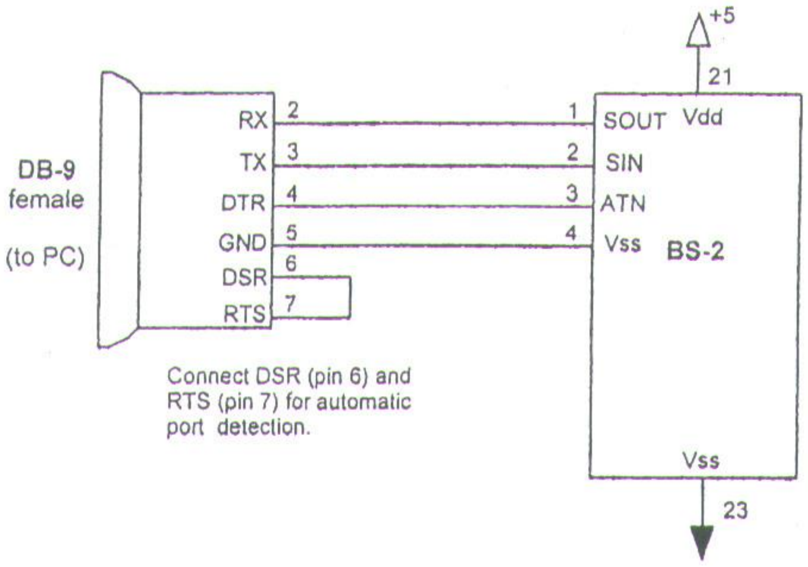
BS2 Anatomy —III

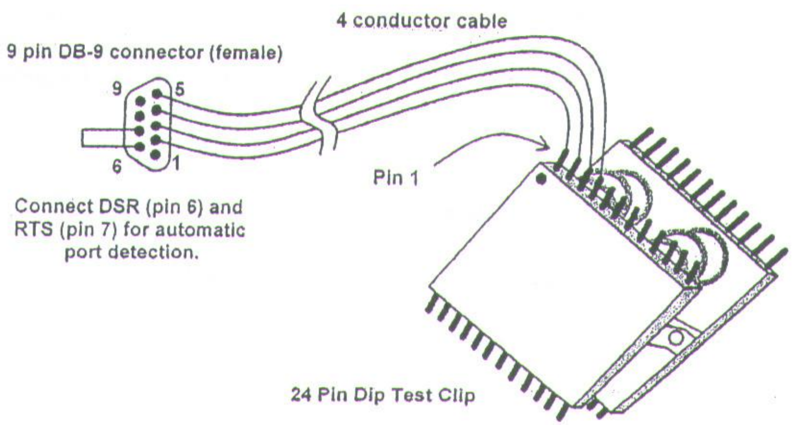




BS2 Development Hardware

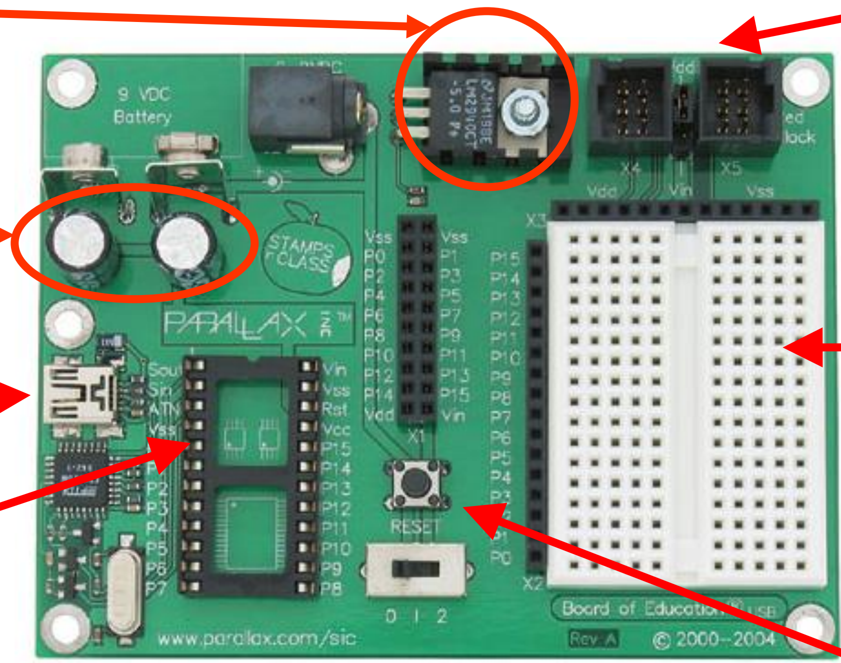
- Develop your project circuit on a breadboard, printed circuit board, or perf board.
- Program BS2 using DB9 programming cable
  - Connect one end of serial cable to PC's serial port
  - Connect other end of serial cable to Pins 1, 2, 3, and 4 of BS2
    - by hard wire connection
    - through a 24-pin Dip Clip which houses BS2
    - by connecting with a DB9 connector mounted on a breadboard which also houses BS2
      - wasted real space does not justify such a design except for prototyping purpose.





BS2 Development Hardware from Parallax: Carrier/Experiment Board

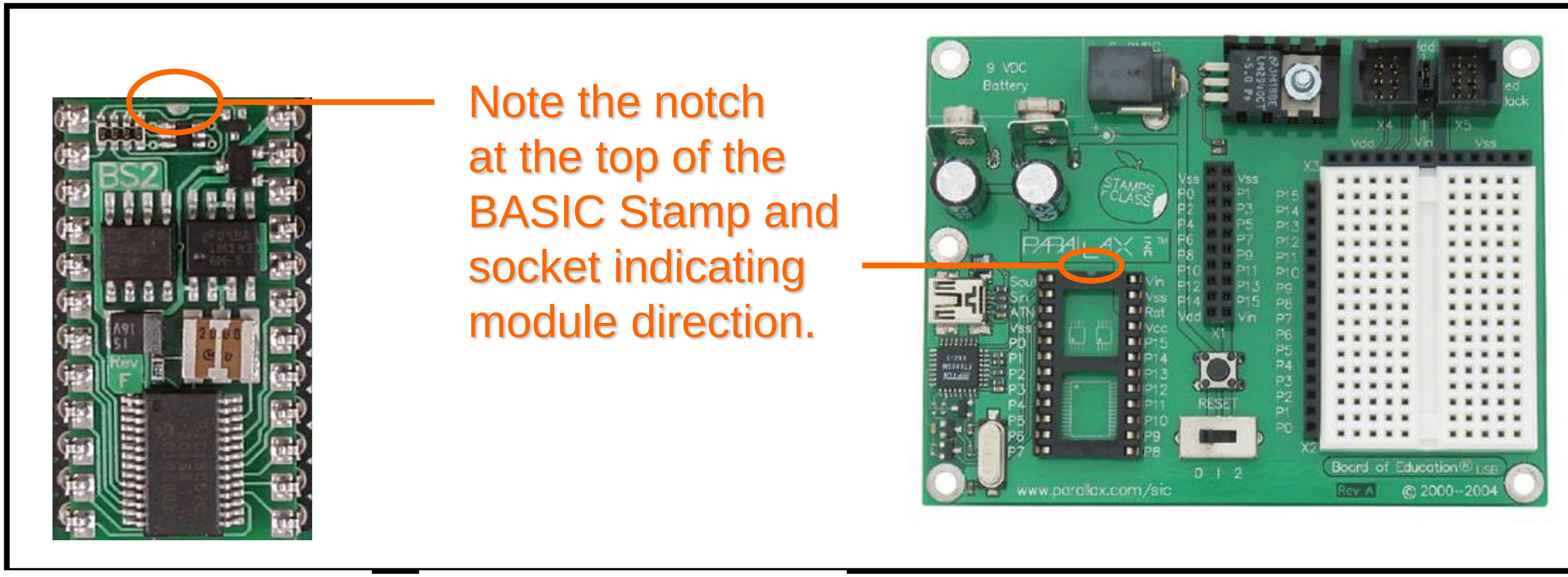
- Power and communication circuits for BS2 can be designed by the user for low-budget projects.
- For starters, a variety of carrier/experimenter boards are available from Parallax that speed development and testing.
- The Board of Education (BOE) will be used throughout this course.

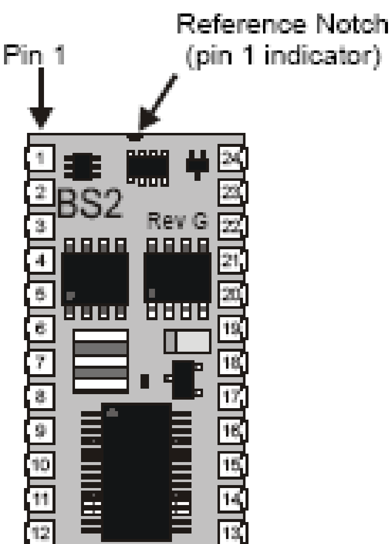


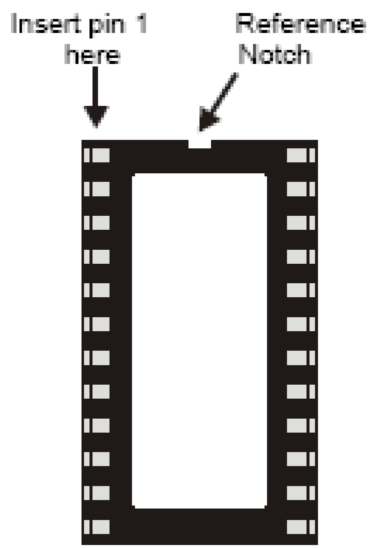
Board of Education

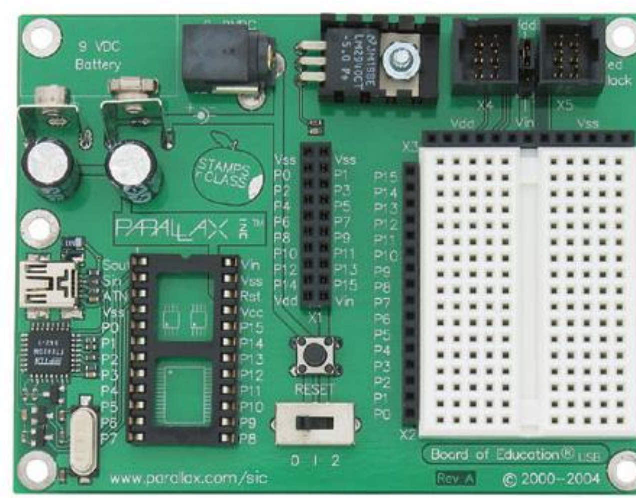
Quick Start: Installing BS2 on BOE

- Notch on BS2 must match with socket on BOE for correct installation direction



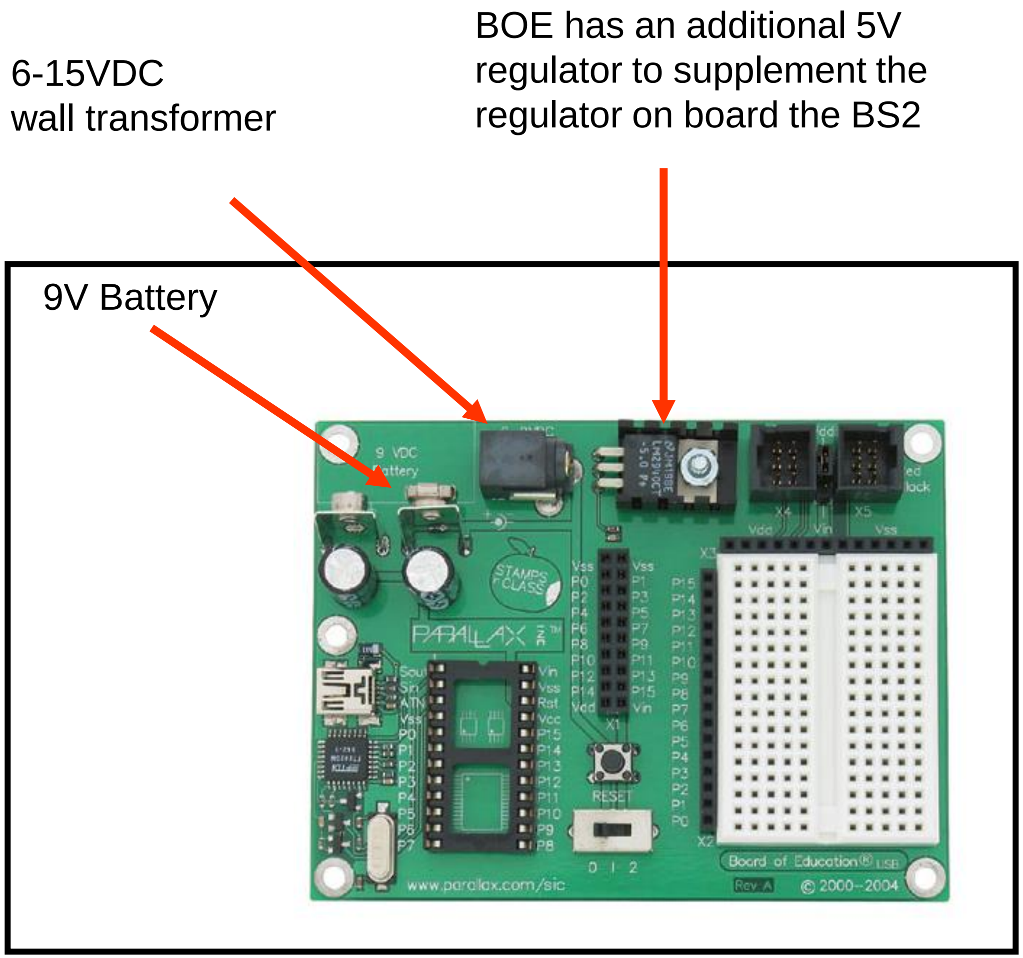






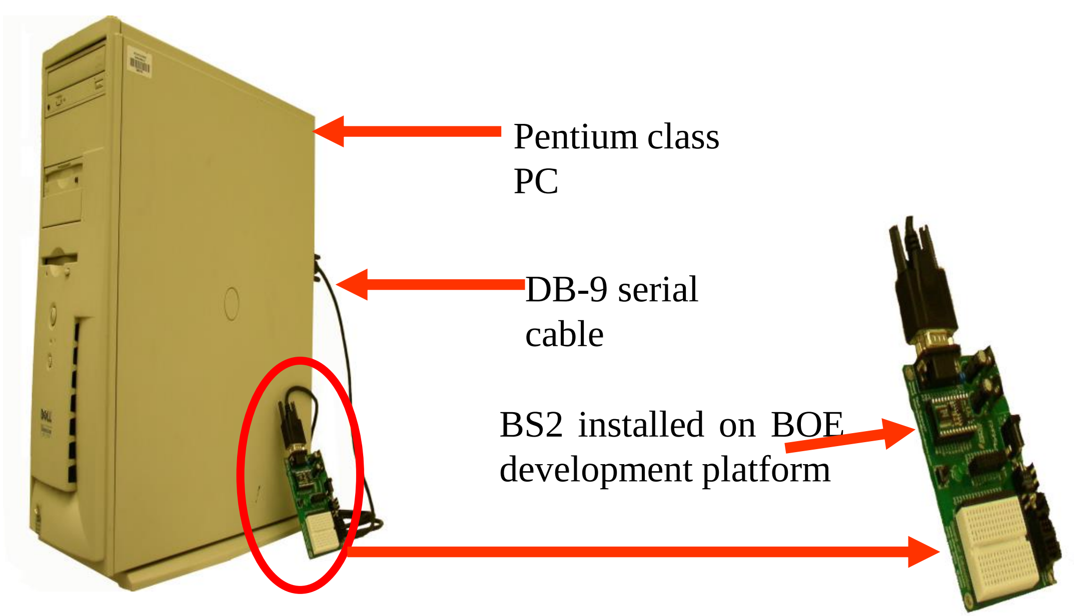
Quick Start: Power Connections

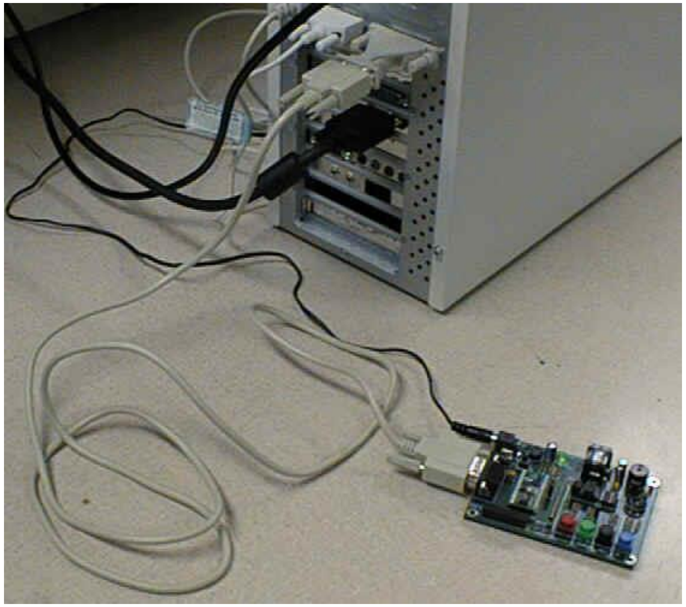
- The Board of Education may be powered using a 9V battery or a wall transformer.



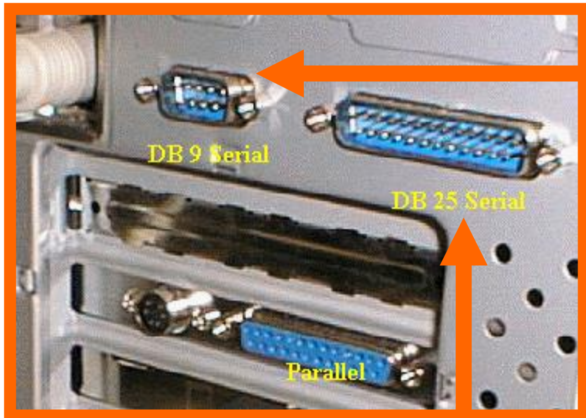
Quick Start: Data Connections

- A serial cable (modem cable) connects the BS2 and the PC's serial communication port (COM port).
  - Serial means that data is sent or received one bit at a time.
  - The serial cable is used to download the stamp with the program written in the text editor and is sometimes used to display information from the BASIC Stamp using the DEBUG instruction.
  - Ensure that you use are using a 'Full Modem' cable (pins 2 and 3 do not cross from end-to-end) as opposed to a 'Null-Modem' cable (pins 2 and 3 cross).
  - There are different connectors for different computer hardware.

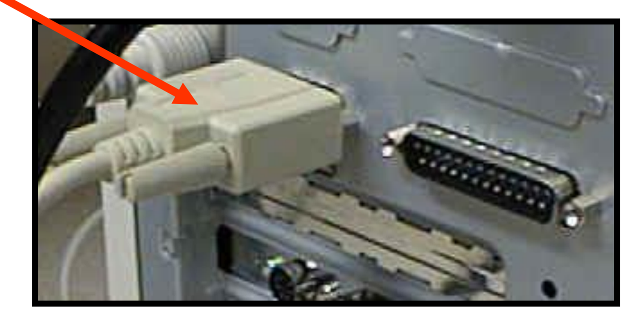


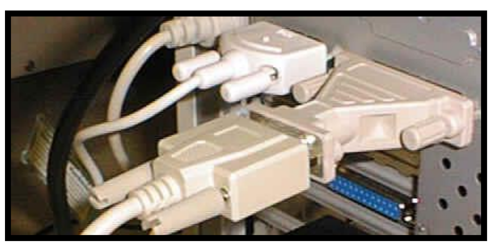


Quick Start: Serial Data Connectors

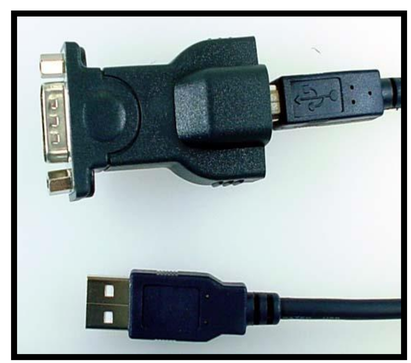


The cable is typically connected to an available DB 9 COM port.





A DB 25 to DB 9 adapter may be needed on older systems



Newer systems may only have USB ports and require a USB-to-Serial Adapter.

Quick Start: Summary

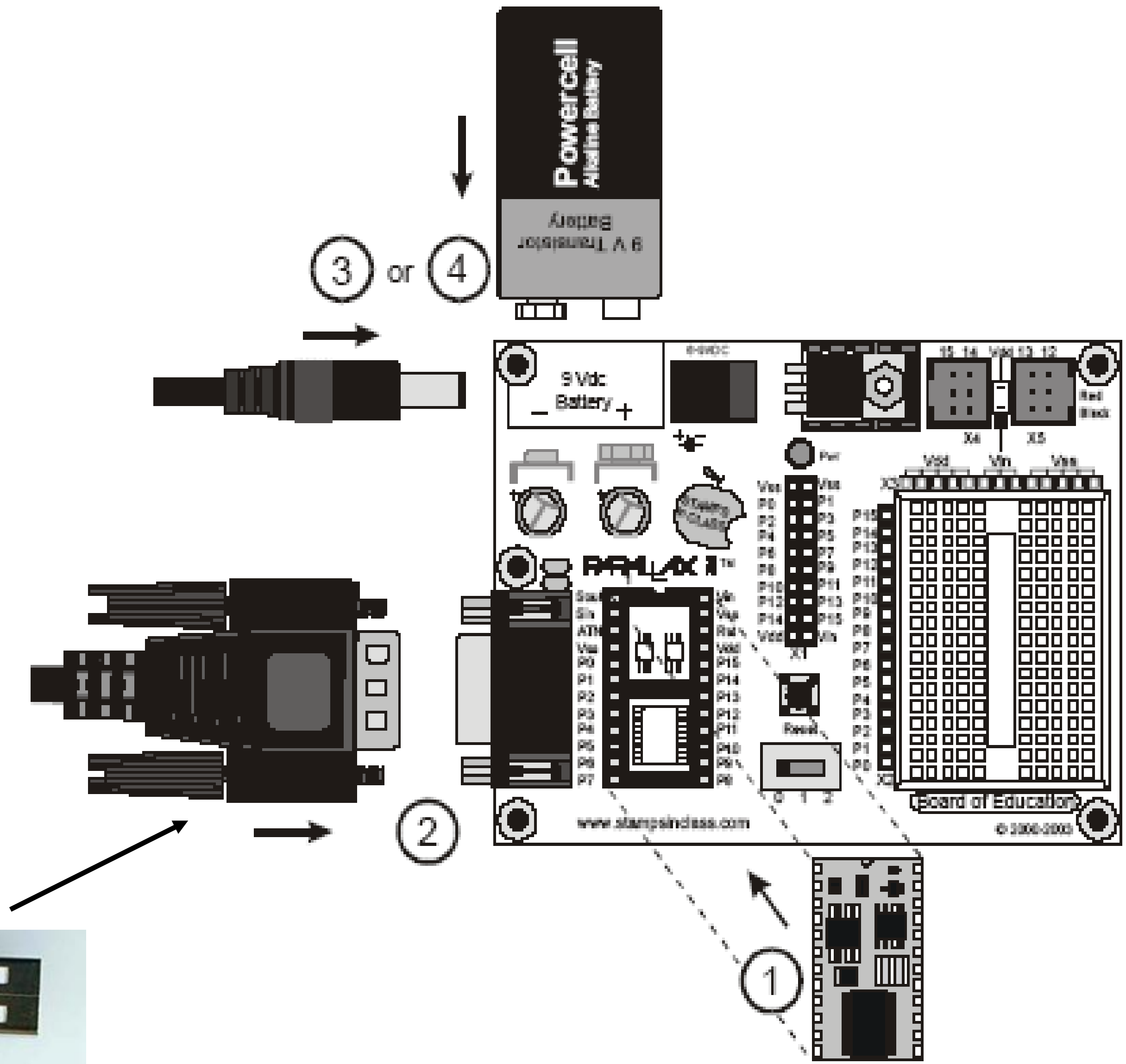
1) Insert the BASIC Stamp module into its socket, being careful to orient it properly.

2) Connect the 9-pin female end of the serial cable to an available serial port on your computer, and then connect the male end to the Board of Education. Note: you cannot use a null modem cable.

3) Plug in the 6-9 V 300mA center-positive power supply into the barrel jack.

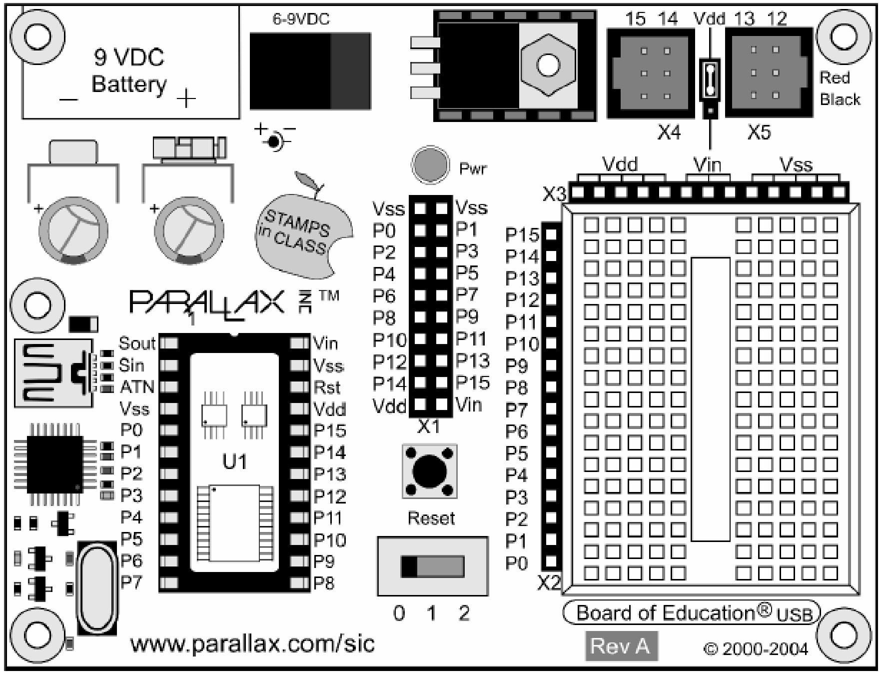
OR

4) Plug a 9 volt battery into the 9 VDC battery clip.



Newer systems may only have USB ports

Board of Education: USB



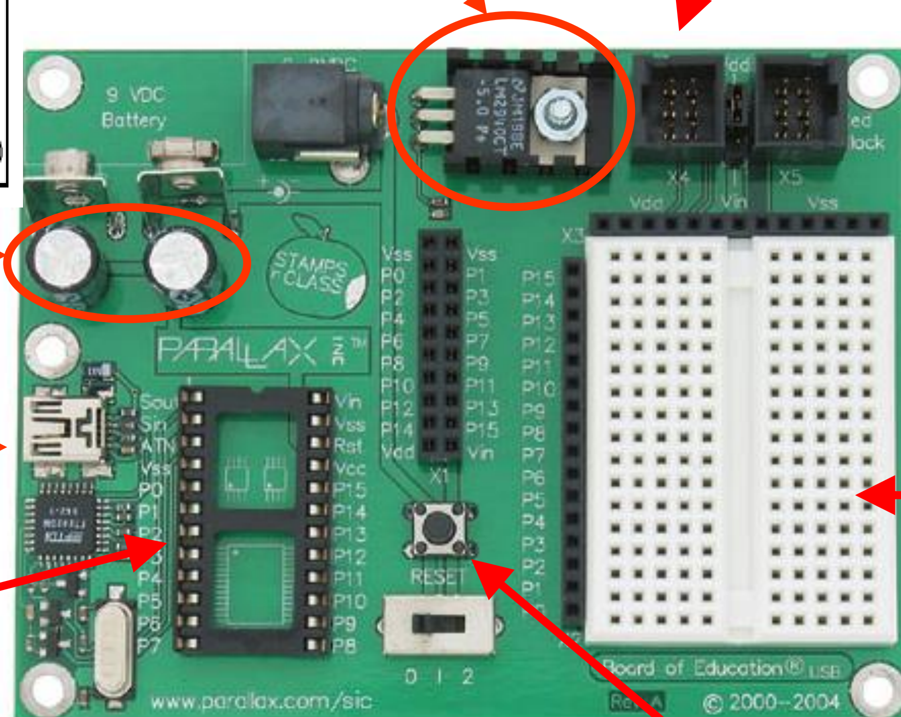
Servomotor connector

5V Voltage regulator

Caution!!

Capacitors

USB port on newer BOE



BS2 socket

Reset button

Breadboard

Quick Start: Software Installation

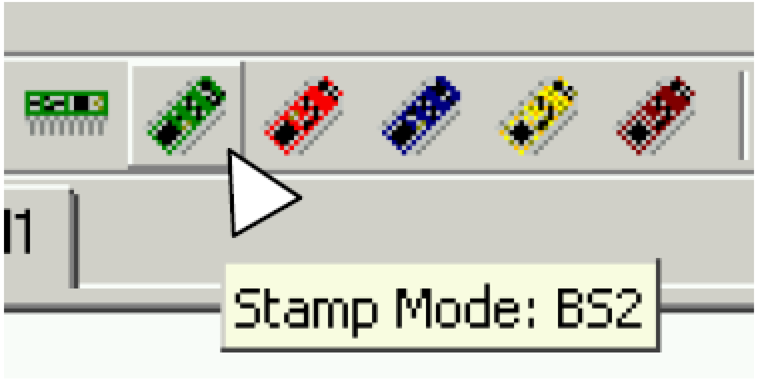
- On Parallax CD, go to Software → Basic Stamp → Windows section to locate latest version of the Basic Stamp Editor software. Click the install button and follow prompts to install and run the software.
- On Parallax web site, go to www.parallax.com → Support → Downloads → Basic Stamp Software. Now look in the Software for Windows section for the latest version of the Basic Stamp Editor software. Click the download button and follow prompts to install and run the software.
- PBasic code for BS2 is written using Windows Stamp Editor: Stampw.exe.

Quick Start: Hello World—I

- Enter the $STAMP and $PBASIC directives from the Basic Stamp Editor's toolbar
  - When using BS2 and PBasic 2.5, the following directives should be placed at the start of the code

' {$STAMP BS2}

' {$PBASIC 2.5}



Click on the icon that corresponds to your BASIC Stamp model to automatically place the $STAMP directive in your program.

Click on the icon for the PBASIC language version that is compatible with your BASIC Stamp model.

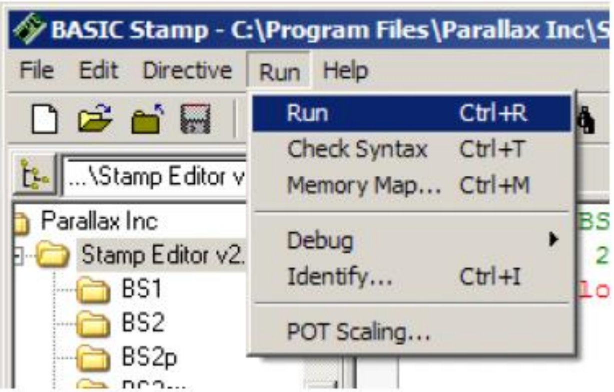
Quick Start: Hello World—II

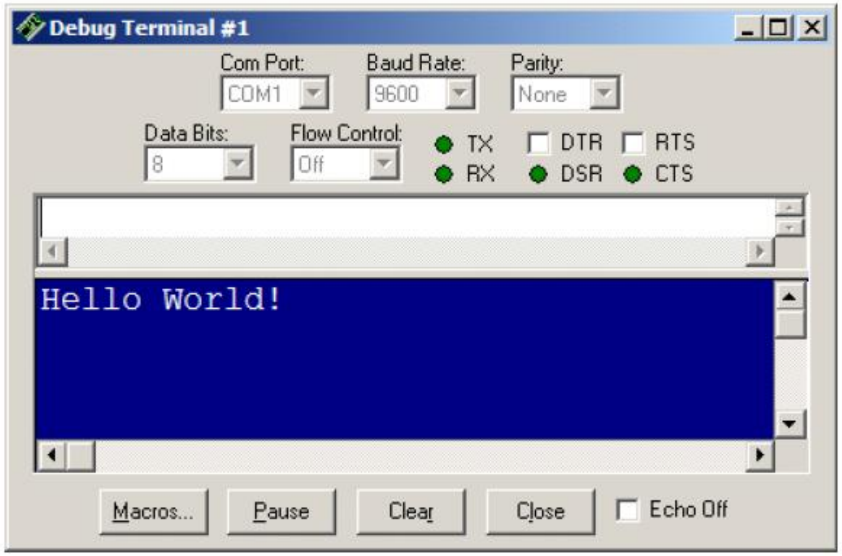
- Type the line DEBUG “Hello World!” below the compiler directives.

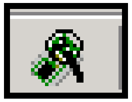
  ' {$STAMP BS2}

  ' {$PBASIC 2.5}

  DEBUG "Hello World!"
- Debug instruction sends serial data back to the PC on the serial cable. A *debug terminal window* opens on the PC to display the returning data.
- Tokenize and download this program into the BASIC Stamp.
  – Select Run → Run from the menu bar
  – Press CTRL-R from the keyboard or
  – Click on the Run ► icon on the toolbar
- A progress bar window should appear. Then a debug window should appear and display “Hello World!”
- The "Hello World!" text appearing on the debug terminal window is sent from the BASIC Stamp, through the programming cable, to the PC.
- The DEBUG button may be used to manually open a DEBUG window.







Hands-on Exercises

DEBUG BASIC Stamp Syntax and Reference Manual 2.2	p159 — p169
DEBUG format BASIC Stamp Syntax and Reference Manual 2.2	p159
DEBUG basic examples BASIC Stamp Syntax and Reference Manual 2.2	p162
DEBUG conversion formatter BASIC Stamp Syntax and Reference Manual 2.2	p163
DEBUG cursor position BASIC Stamp Syntax and Reference Manual 2.2	p168
DEBUGIN BASIC Stamp Syntax and Reference Manual 2.2	p171 — p174
Programming Essentials & The Elements of PBASIC Style: StampWorks Manual	p.11—24
Getting Started BASIC Stamp Frequently Asked Questions	All