



معهد التكنولوجيا التطبيقية
INSTITUTE OF APPLIED TECHNOLOGY

Micro-Controllers

Module 1:

Introduction

To Microcontrollers

REVISED BY
IAT Curriculum Unit
August 2009

© Institute of Applied Technology, 2008

Micro-Controllers
Module 1
Introduction to Micro-Controllers

Module Objectives

- Introduce Micro-Controllers, their features and some applications
- Explain Micro-Controllers Architecture and pins' assignment
- Introduce Parallax Board of Education hardware (BOE)
- Introduce Basic Stamp Editor V2.0 software

Module Contents

- 1) What are Micro-Controllers?
- 2) Architecture of Micro-Controllers
- 3) Basic Stamp 2 Board of Education Hardware
- 4) Basic Stamp 2 Editor V2.0 software

Introduction

Around us there are many Electric and Electronics' devices (*with Buttons, LED's or LCD and some sort of control*), which perform different preplanned tasks. Examples of such devices are: Digital Cameras, iPods, modern car's panel, Washing machines, etc.

It is fair to assume that any equipment with buttons, LED or LCD must have a built in **device** which can control and follow up the whole process. One type of such **devices** is called **Microcontroller (μ con)**.

Examples of equipments with built in Microcontrollers (μ con):

1) In Electronic Gadgets (Devices):



Figure 1.1 Digital Camera



Figure 1.2 MP3 Player



Figure 1.3 I Pod



Figure 1.4 Mobile phone

2) In Cars: A modern car would have up to 7 μ con's for different control purposes such as:

- Engine control
- Cruise control

- Anti-Lock brakes System (ABS)
- Side Mirrors control
- 4-Wheel drive control
- Windows control
- Keyless car entry control



Figure 1.5 A modern car Panel

3) In Home Appliances:



Figure 1.6 Microwave



Figure 1.7 LCD TV



Figure 1.8 Washing machine



Figure 1.9 Cloth Dryer



Figure 1.10 Refrigerator



Figure 1.11 DVD Player and Remote

In the previous examples we can note that microcontrollers can perform many tasks in different applications; then a question may arise in our mind!

What are Microcontrollers?

Microcontrollers are "special purpose computers" that do one task well.

What are the things in which a Microcontroller is like a computer?

Like a computer a Micro-Controller has:

- CPU
- Memory
- Input and Output (I/O) devices

How are computers and microcontrollers different from each other?

Computers	Microcontrollers
The computer can handle many tasks and can run several programs.	The microcontroller can handle only one dedicated task

Microcontrollers are many!! Which one are we going to study?

Figure 2.1 shows the microcontroller we are going to use in this course. It is called *Basic Stamp2 (BS2)* and is developed by **Parallax**. The BS2 is programmed with **PBASIC**, where **PBASIC** is a version of the popular BASIC programming language, then;

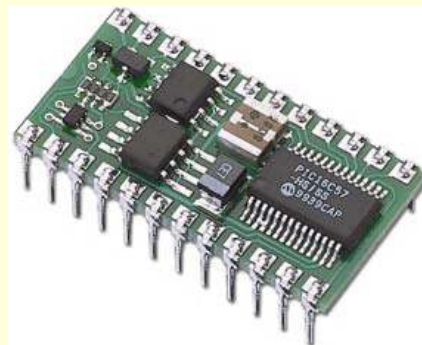


Figure 2.1 Basic Stamp2 (BS2)

PBASIC ≡ Parallax Beginner's All-Purpose Symbolic Instruction Code

We can also add that in the term *Basic Stamp*:

- **Basic** refers to the program used and
- **Stamp** because the Parallax microcontroller is as small as a postage stamp as shown in Figure 2.2



Figure 2.2 BS2 is as small as a postage stamp

Types of Microcontrollers:

The three most common microcontrollers for hobbyists are:

- 1) the **BS2** modules (BS2, BS2e, BS2sx, & BS2p) from **Parallax**,
- 2) the **PIC** series (PIC16, PIC18, & PIC24) from **Microchip**, and
- 3) the **AVR** series from **Atmel**

The special features of Microcontrollers are:

- 1) Can be embedded inside some other devices to control their actions
- 2) Can be dedicated to one task and run one specific program well.
- 3) low power devices
- 4) small and low cost

Architecture of Micro-Controllers

Microcontroller is a small computer on a single IC consisting of:

- 1) Central processing unit (CPU)
- 2) Memory
- 3) Input and Output ports (I/O pins)

1) Central processing unit (CPU)

The CPU is the "brain" that executes the programs. More over it can perform addition, subtraction, division, and multiplication operations.

2) Memory of Microcontrollers

The **memory** is the part of the μ con whose function is to store **data**.
The memory consists of cells called **Memory Locations (ML)**.

Memory Types

The **memory** is the part of the μ con whose function is to **store data** and it consists of cells called **Memory Locations (ML)**.

Every Micro-Controller has two types of memory:

- A) Short term memory (Volatile)
- B) Long term memory (Nonvolatile)

A) Short term memory (Volatile):

It loses data when the power is switched OFF and it is called RAM.

RAM is the short for **R**andom **A**ccess **M**emory.

The function of RAM: It is used to store temporary data that is not essential to run a device.

B) Long term memory (Nonvolatile):

Long term memory retains (keeps) the data even when the power is OFF and it is called EEPROM.

EEPROM = **E**lectrically **E**rasable **P**rogrammable **R**ead **O**nly **M**emory

The function of EEPROM: It is used to store the data that is essential to run a device.

3) Input and Output Ports (I/O Ports)

Input and output ports are the interface of the μ con to the outside world.

Via I/O ports you can connect various input/output devices to the μ con.

Example of Input Devices:

Switch, push button, or sensors

Example of Output Devices:

LCD, buzzer, or Indicators (such as LED)

Basic Stamp2 has **16 (I/O) ports**. Each one is **bi-directional**; that means it can be programmed to be an **Input** or an **Output** port.

Basic Stamp 2 Pins' Assignment

Normally Micro-Controllers are soldered into printed circuit boards (PCBs) and connected to various electrical and electronic components. It is very essential to know the pins assignment of the microcontroller used; so you can be able to connect the power, establish communication with PC, and connect the I/O devices to the right input and output pins. Pins' assignment of BS2 microcontroller is illustrated in Figure 3.

The Pins Assignment of BS2

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

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

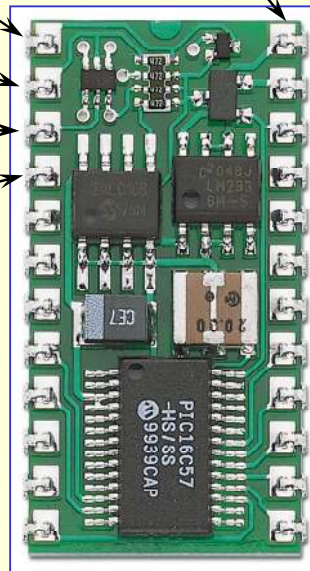


Figure 2.3 Pins Assignment of BS2

The Board Of Education (BOE)

To *power* and *program* the BS2; it needs to be integrated in a PCB board. The PCB board that comes with the BS2 is called **Board of Education (BOE)** and is shown in Figure 4. This board is fitted with components and IC 24-pins Socket to plug in the BS2 and easily establish the following:

- to connect a power supply to run the BS2
- to program the BS2 through **serial** or **USB** cable from PC
- to build electronic circuits and control them from the BS2
- to connect and control servo motors

The main parts of the Board of Education (BOE)

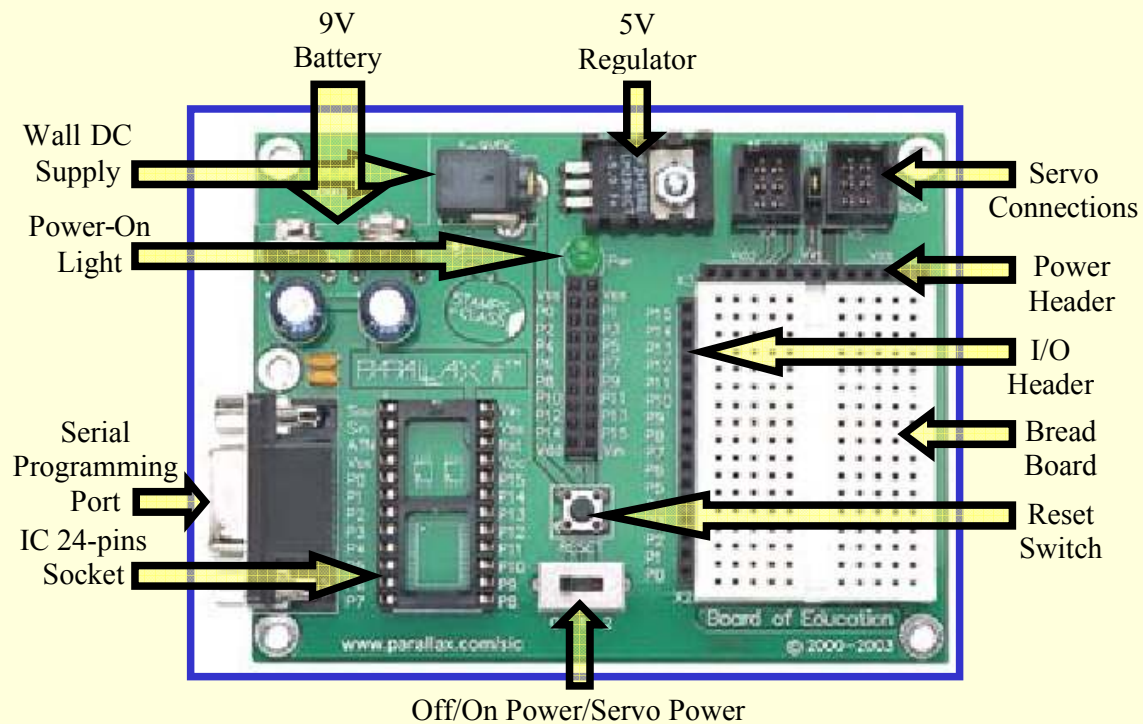


Figure 2.4 The main parts of the BOE

The BOE Part's Description	
Part	Description
1) IC 24-pins Socket	To plug in the BS μ con. Orientation of the μ con is very important.
2) Breadboard.	Breadboard is a white working area to plug in your experimental circuits. The Breadboard is separated into rows of five sockets in 2 columns. The 5-ockets in each row are internally connected.
3) I/O Header	The vertical black 16-sockets are labeled (P0→P15) and used to make connection to I/O pins of the BS2. It is referred to as I/O Header.
4) Power Header	The horizontal black sockets are used to supply power to the circuits you build and referred to as Power Header.
5) Selector switch	Selector switch to switch ON/OFF the BOE. It has 3-positions: 0 → OFF, 1 → ON, and 2 → Servo power ON
6) LED indicator	To indicate if the BOE is powered and switched ON.
7) Serial Programming Port	9 pin connector to connect your serial cable to PC
8) Battery socket	Battery socket to connect the 9V DC battery
9) Servo Connections	To connect up to 4-servo motors
10) Reset Switch	Press & release to start the program on your BS2 again
11) 5V regulator	To regulate (5.5→15V _{DC}) input voltage to 5V
<p>The Power Header is divided into 3-parts as follow:</p> <ul style="list-style-type: none">• $V_{DD} \Rightarrow$ provides +5 Volts DC• $V_{SS} \Rightarrow$ represents the GND (0 Volts)• $V_{IN} \Rightarrow$ Supply Voltage from battery (9V) or wall DC supply.	

Basic Stamp Editor Version 2.0

As mentioned earlier microcontrollers are special purpose computers that do only one task well. Without a suitable program, the μ con can not handle the required task and it is of no use. Therefore each μ con comes with special software used to program the unit. The special software (**PBASIC**) used to program the BS2 and other BS-modules, is called *Basic Stamp Editor* (Version 2.0 or higher). The version we are going to use is v2.3.9.

BS Editor Version 2.3.9 (v2.3.9)

Besides programming, this software can display messages sent by the Basic Stamp. This feature is very useful for troubleshooting purposes.

To program the BS2 means writing the set of commands that the μ con will understand and execute.

To Open the Basic Stamp Editor

Once you finish the setting of the BS Editor in your PC, the shortcut of the software appears on the desktop as shown in Figure 3.1. You can open the BS Editor by double clicking the shortcut. It has the look of most Windows' interface programs. A typical screen of BS Editor, consist of three main panels as shown in Figure 3.2.

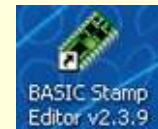


Figure 3.1
Shortcut to
BS Editor

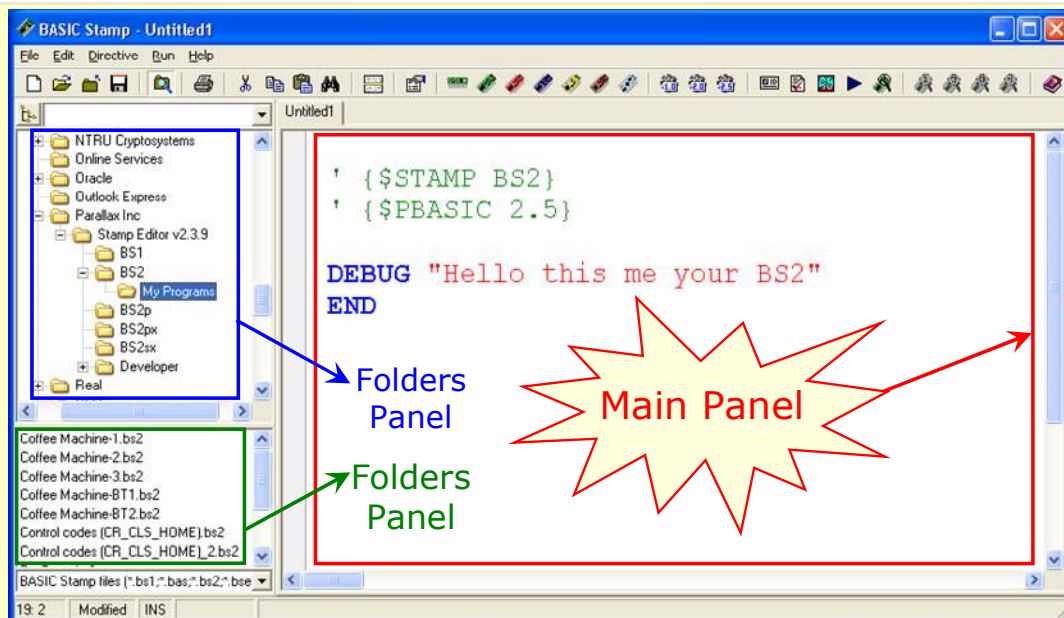
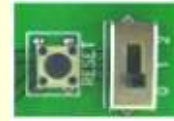


Figure 3.2 The typical screen of BS Software

To Program the BS2 always start with these 8-steps:

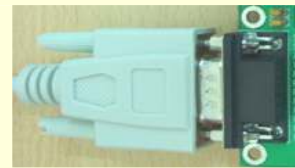
a) Set the 3-position switch to position-0, the BOE is OFF



b) Plug in the 9V battery



c) Connect your BS2 to the PC (or Laptop) through the serial (or USB) cable



d) Open the BS Editor (double click the shortcut on the Desktop)



e) Switch ON the power of the BOE (set the SW to position-1)

f) Identify your BS2 by clicking the ID Button



g) Select the Basic Stamp mode \Rightarrow BS2 \Rightarrow '**{\$STAMP BS2}**'

h) Select the PBASIC version \Rightarrow Ver-2.5 \Rightarrow '**{\$PBASIC 2.5}**'



Select by clicking on the icons (BS2 & Ver-2.5)

Now the BS Editor is ready for writing a program and the BOE is ready with a BS2 to download and run that program. This will be our task in Module-2

Appendix:

Memory Location: Each ML is identified with an **address** according to its location in the **memory**.

The address is used to **access** the data from the ML.

Data: means the contents stored in the ML.

Access the data: Means “write a data to” or “read a data from” the ML’s.

Register vs. memory:

All data must be moved from the **memory** into a **register** before it can be processed. For instance, if a certain set of numbers have to be added, first the numbers are moved into the registers, processed, and the result is also kept in a register.

The register has the addresses of a memory location where data is maintained.

Registers: is a temporary storage area *within the CPU*. Registers are ML’s that are used to store intermediate results of the performed operation.

Dynamics Architecture of Micro-Controllers

How things are working inside the Micro-Controller?

To illustrate this, consider the addition example (5+6+10+12).

In the memory, (RAM or EEPROM), all numbers are stored in memory locations as shown in the Figure 2:

ML00 = Memory location No.0 and it contains the data (5).

ML01 = Memory location No.1 and it contains the data (6).

ML02 = Memory location No.2 and it contains the data (10).

ML02 = Memory location No.3 and it contains the data (12).

▪
▪

ML14 = Memory location No.14

ML15 = Memory location No.15

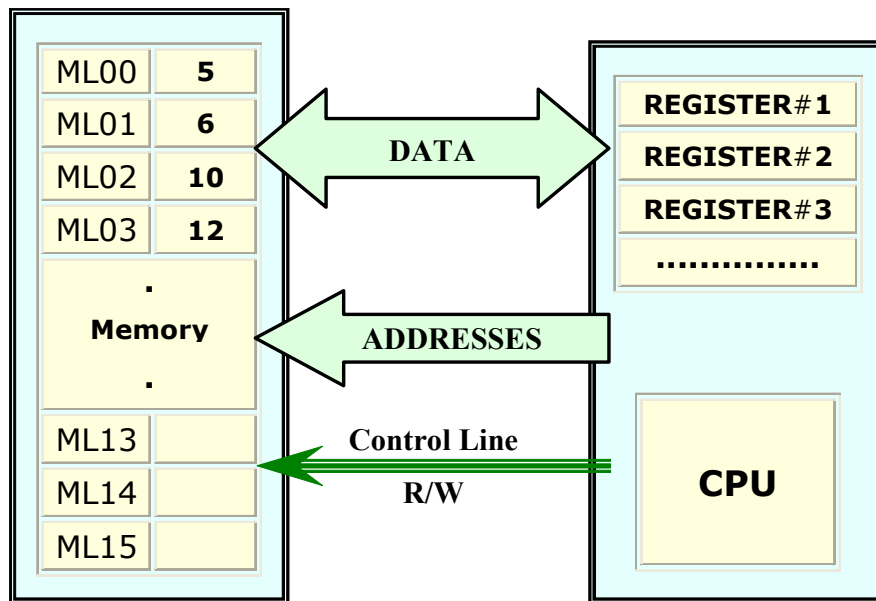


Figure 2 CPU and memory dynamics in the Micro-Controller

Step#1:

The CPU will read the contents of the ML01 and ML02 (5 and 6).

Then it perform the addition operation ($5 + 6$)

And store the result (11) in Register#1.

Result: Register#1 will contain the value 11.

Step#2:

The CPU will read the contents of the ML03 (10)

Then perform the addition operation ($10 + \text{Register\#1}$)

And store the result ($10 + 11 = 21$) in Register#2.

Result: Register#2 will contain the value 21.

Step#3:

The CPU will read the contents of the ML04 (12)

Then perform the addition operation ($12 + \text{Register\#2}$)

And store the result ($12 + 21 = 33$) in Register#3.

Result: Register#3 will contain the value 33.

Step#4: Finally, the final result will be written back to ML05.

Now the memory location No. 05 (ML05) contains the result (33).

The Data-Transfer Process of the Given Example:

- 1) The CPU will inform the EEPROM to **read the data** from the ML by setting the **control line** to **R** (Read).
- 2) The address bus will contain the ML address (ML01).
- 3) So, the content of ML01, (5), will be transferred via the data bus. The same steps will be repeated for the next ML's till the process is completed.
- 4) Once the final result has been calculated (= 33), the CPU will write this value back to ML05 of the EEPROM in the following manner:
- 5) The CPU will inform the EEPROM to **write the data** to the memory by setting the **control line** to **W** (Write).
- 6) The address bus will contain the ML address (ML05).
- 7) So, the final result (the data 33) will be transferred via the **data bus** to the specified memory location (i.e. ML05).