Course on BASCOM AVR - (2)

Theoretic/Practical course on BASCOM AVR Programming.
Author: DAMINO Salvatore.

BZ1 and T1 Management.

Once You had generated the first program of the course, and You understand its functionality, there is the need to save the compiled code on Microcontroller, in order to check its real execution.

This operation can be performed by connecting the **GMM TST3**, complete of **GMM AM08 Mini Module** mounted on **Z1** socket, to a **PC** through a serial line and the proper communication cable.

The communication is performed with an **RS 232** electric protocol. If Your **PC** has only the **USB** communication line, there aren't problems in fact it is sufficient get and use a standard **USB** to **RS 232** converter.

These cheap converter interface can be acquired from a normal shop that sell personal computer and accessories.

When the connection had been performed, You can start the saving operations by using the program named **AVR Bootloader grifo®**, released by **grifo®**

Below are detailed described all the operations required to correctly perform all the process.

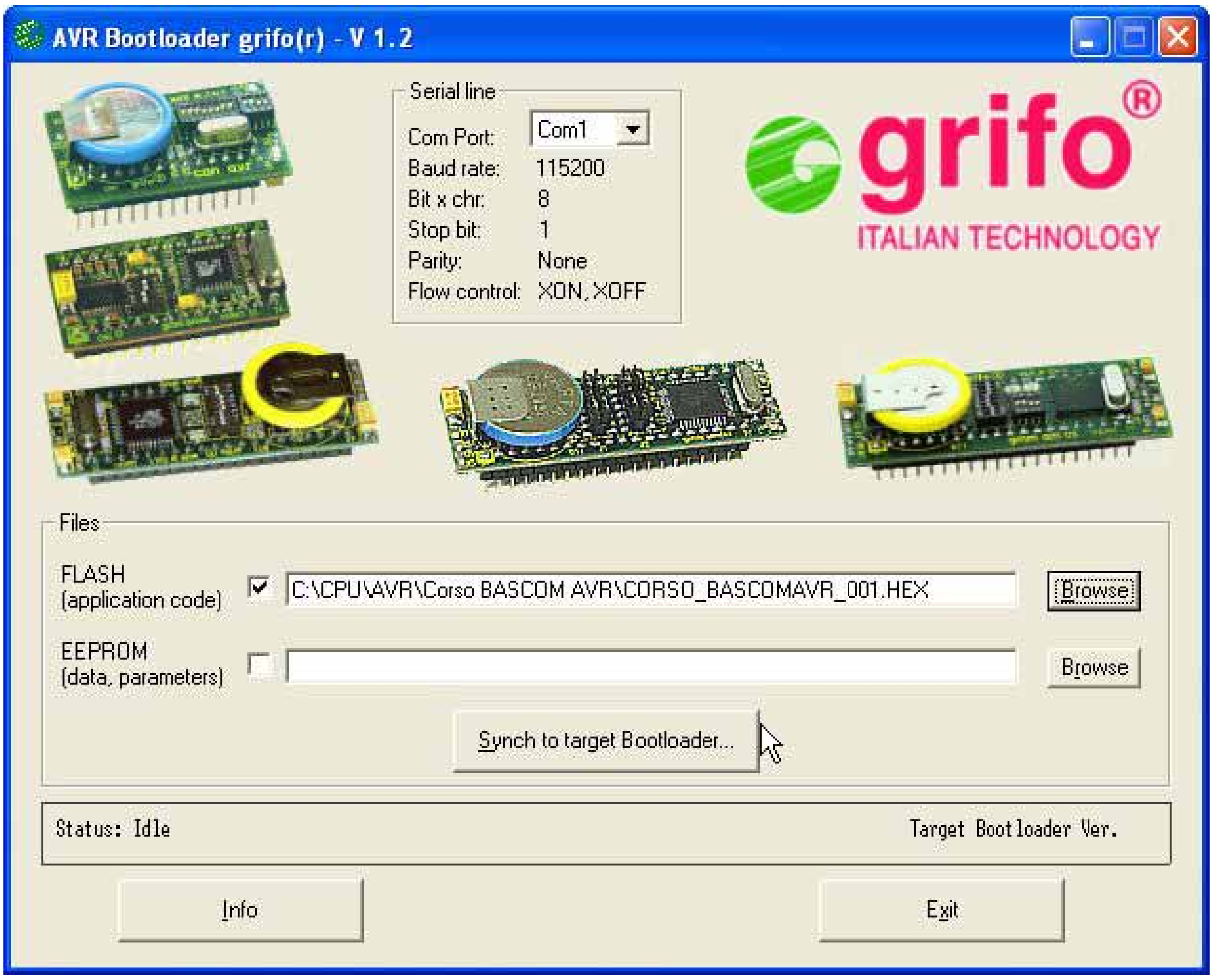
AVR Bootloader grifo® PROGRAM USE

The following steps describe how the compiled program can be saved on the **FLASH EPROM** of **GMM AM08 Mini Module**, through the proper **AVR Bootloader grifo®** program.

It is important to remind that the compiled program is the file with executable code for the Mini Module, in **HEX** Intel format, or in other words the file directly generated by **BASCOM AVR** development tool.

AVR Bootloader grifo® is a **PC** program, developed and released free of charge by **grifo®**, that interact with the **Boot Loader** available on **Mini Module**, through one serial communication line.

By using **GMM AM08** this line is the **RS 232** asynchronous serial line, that must be obviously available also on **PC**. Thanks to this interaction the **AVR Bootloader grifo®** is capable to manage the **Mini Module** memories, when it is already installed in the final system, without any other additional device. This management modality is named **ISP**, that means **In System Programming**.


AVR Bootloader grifo(r) - V 1.2
grifo®
ITALIAN TECHNOLOGY
Browse
Browse
Synch to target Bootloader...
Info
Exit

Serial line
Com Port: Com1
Baud rate: 115200
Bit x chr: 8
Stop bit: 1
Parity: None
Flow control: XON, XOFF

Files
FLASH (application code) ✓ C:\CPU\AVR\Corso BASCOM AVR\CORSO_BASCOMAVR_001.HEX
EEPROM (data, parameters)

Status: Idle
Target Bootloader Ver.

AVR Bootloader grifo® Main Window

1) Ensure that the **GMM AM08** asynchronous serial line is **RS 232** buffered, by configuring its dip switch as below described:

DSW1.1	**->**	**ON**
DSW1.2	**->**	**ON**
DSW1.3	**->**	**OFF**
DSW1.4	**->**	**OFF**

2) Find out the **PC** asynchronous serial line ready to use and the relative male **DB9** connector. Whenever the **PC** hasn't an **RS 232** serial line, please add it, for example by using suitable **USB <-> RS 232** converters.

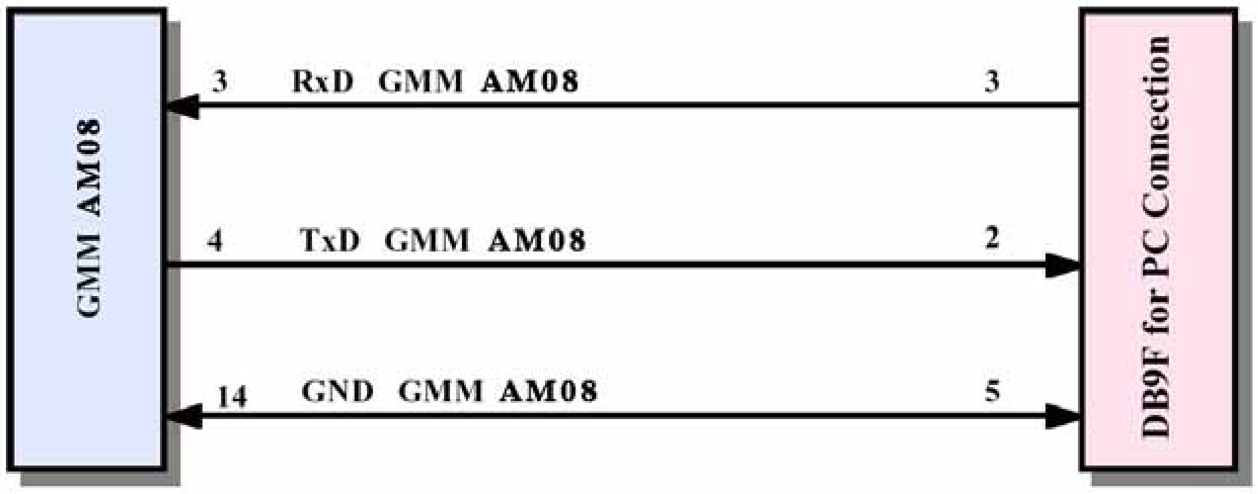
Moreover you must establish the name or number (**COMx**) of the found serial. For this purpose it can be comfortably used the window:

Start | *Settings* | *Control Panel* | *System* | *Hardware* | *Devices* | *COM & LPT Ports*.

3) Perform the serial connection between **GMM AM08** and **PC**, or on the other hand connect the two communication signals and reference ground signal.

When it is used a **GMM AM08** mounted on **GMM TST3** card, this connection is simply obtained with a standard **RS 232** extension cable, plugged in the **COMx** connector (find out at point **2**) on one side and in the **CN5** connector of **GMM TST3** on the other side.

Vice versa, when **GMM AM08** is mounted on a board developed by user, the three communication signals of **PC** must be linked directly to **Mini Module** pins, as described in following figure.



Point to Point RS 232 Connection Example With PC

4) Search the **AVR Bootloader grifo®** program on the received **grifo® CD** disk, or download it directly from web site, at the address:

   http://www.grifo.it/PUB/AVR/AVR_BL.ZIP

5) Unzip and install the **AVR Bootloader grifo®** program downloaded at previous point, on **PC** hard disk. Even if it isn't strongly required, during the installation we suggest to maintain and confirm the default options proposed by installation program.

   Whenever during the installation are displayed some windows that inform and/or require a confirmation about replacement of operating system files (**.DLL**), please ensure that the original versions are mantained.

6) Close all the executed programs that use the serial line **COMx** of **PC**, connected at point **3**.

7) Run the **AVR Bootloader grifo®** program installed at point **5**. If default settings have not been changed during installation, you can use the proper link added in the **Windows Start** menu: ***Start | Programs| grifo® | Avr Bootloader grifo®.***

8) Close the showing window appeared pressing ***Close*** button.

9) At this point there is the main window of **AVR Bootloadre grifo®** on screen that, in case of first execution, is without any settings. Select the communication line of **PC** connected at point **3**, in the the ***Com Port*** list.

10) Check the ***FLASH (application code)*** item and then choose the file to program in **FLASH** of **GMM AM08**, that is the file with **.HEX** described at the beginning of paragraph. After the button ***Browse*** pressure, select the file, through the proper dialog window appeared.

11)Press the button ***Synch to Bootloader...*** and immediately after turn off and on the **Mini Module** power supply or reset it. In order to synchronize the two systems it is required that power on or reset happens in **5** seconds from button pressure on **PC**; if this timeout elapses the program will shows a ***No Response from Target Bootloader***, and the user must close it with ***OK*** and then retry the synchronization.

If the synchronization fails even on next tries, it is suggested to check previous points and in details those dedicated to **RS 232** connection and serial line selection.

12) When the synchronization is done in the bottom status window appear the indication ***Sending FLASH file - lines remaining xxxx*** and the Bootloader version executed by **Mini Module**: ***Target Bootloader Ver. x.x***. While the second indication stay constant, the first one changes in decreasing the number of lines of **HEX** file to transmit, up to zero. At this point the file is completely downloaded to **Mini Module** and if errors are not occured, it is shown a window with the message ***FLASH file succesfully downloaded***.

13) The endurance of the transmission changes depending on file length and on communication line type, available on **PC**; for example if the used **COM** corresponds to an **USB<->RS232** converter, the average transfer time is really longer. In any case the user has to wait the condition described at previous point or, if it take too long time, stop the transmission with the ***Cancel*** button.

14) At this point the **FLASH** is programmed and the **AVR Bootloader grifo®** can be closed through the ***X*** in the right high corner of the window, or alternatively with the ***Exit*** button.

15) At this point the new compiled program has been saved and it is ready for execution. This happens automatically immediately after the programming but it can be executed again with a power on or a Reset of **GMM AM08**.

Start of Experimentation

Now You have all the elements required to start a profitable experimentation. You are capable to generate a program, save it on Microcontroller and check its right functionalities.

With the proposed example, the file obtained by compilation (***uk_BASAVR_001.HEX***) must be saved on **GMM AM08 FLASH EPROM**, as described in previous paragraph, and when the Mini Module is restarted, check that by pressing **T1** push button the **L3 LED** is enabled. If this conditions happen, congratulations, You have developed the first program successfully.

Next step to perform is to study and test all the other programs. These have been developed in order to increase readers knowledge and they offer rising experiences and complexity.

Example.002. Buzzer and Push Button Management.

Added Definitions:

None

Added Declarations:

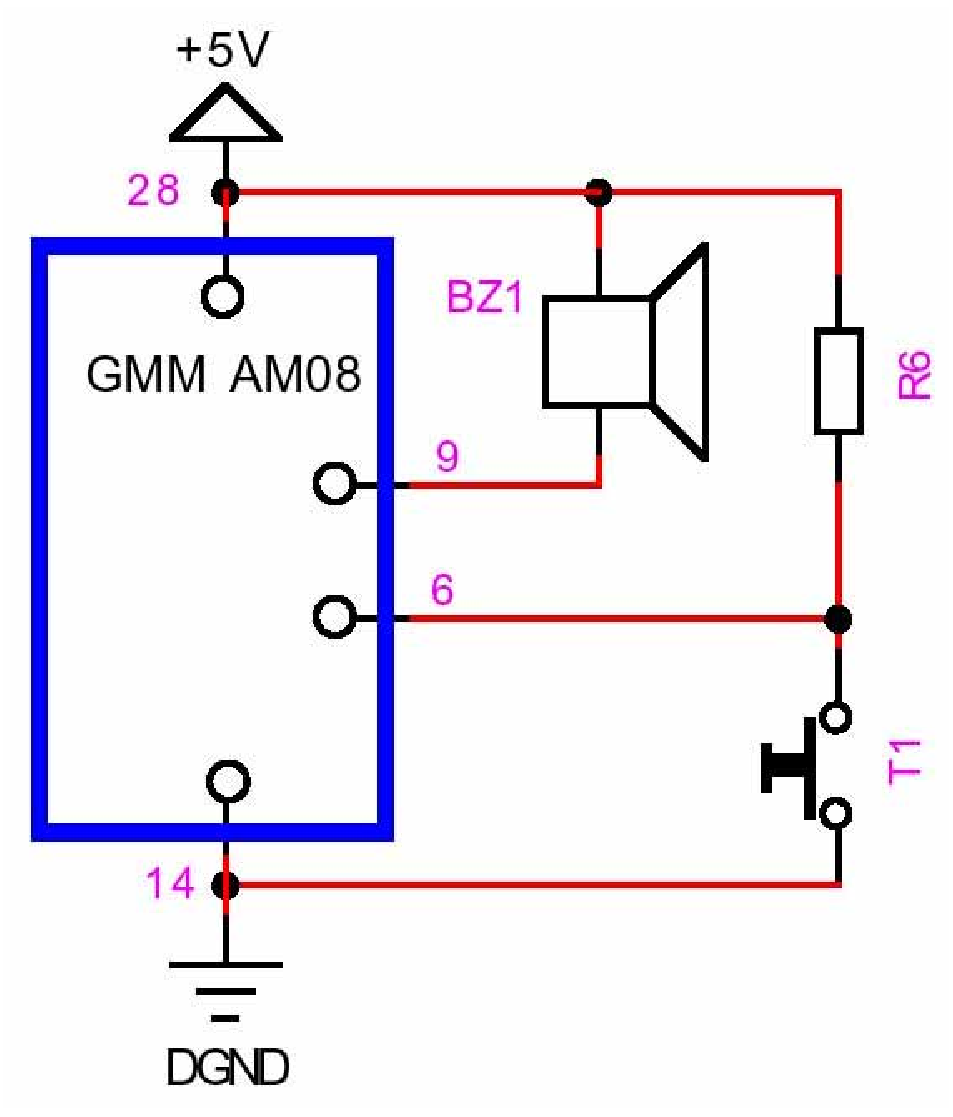
None

Added Instructions:

None

Added Operators:

None



Electric Diagram Used by Program.

This program executes simple operations on digital **I/O**, by using one **Push Button** and the **Buzzer BZ1** available on **GMM TST3**, as output device.

- The used **I/O** lines are:

    - pin **15** of **GMM TST3 Z1** socket (= pin **9** of **GMM AM08**) connected to **BZ1 Buzzer**;
    - pin **12** of **GMM TST3 Z1** socket (= pin **6** of **GMM AM08**) connected to **Red LED L2** through **R6** and to **Red** Push Button **T1**.

- The program manages the pin **6** as an input line and the pin **9** as an output line.

- At power on the **Buzzer** is silent.

- By pushing the **T1** push button, the **LED L2** is turned on by hardware, as it is electrically connected, and either the **Buzzer** will produce a sound.

- By releasing the **T1** push button the **Buzzer** will stop the sound.

On previous page it is available the electric diagram that must be realized in order to use the described **Example.002**.

The program functionality is similar to precedent **Example.001**. In fact it replaces the **L3** driving with **BZ1 Buzzer**.

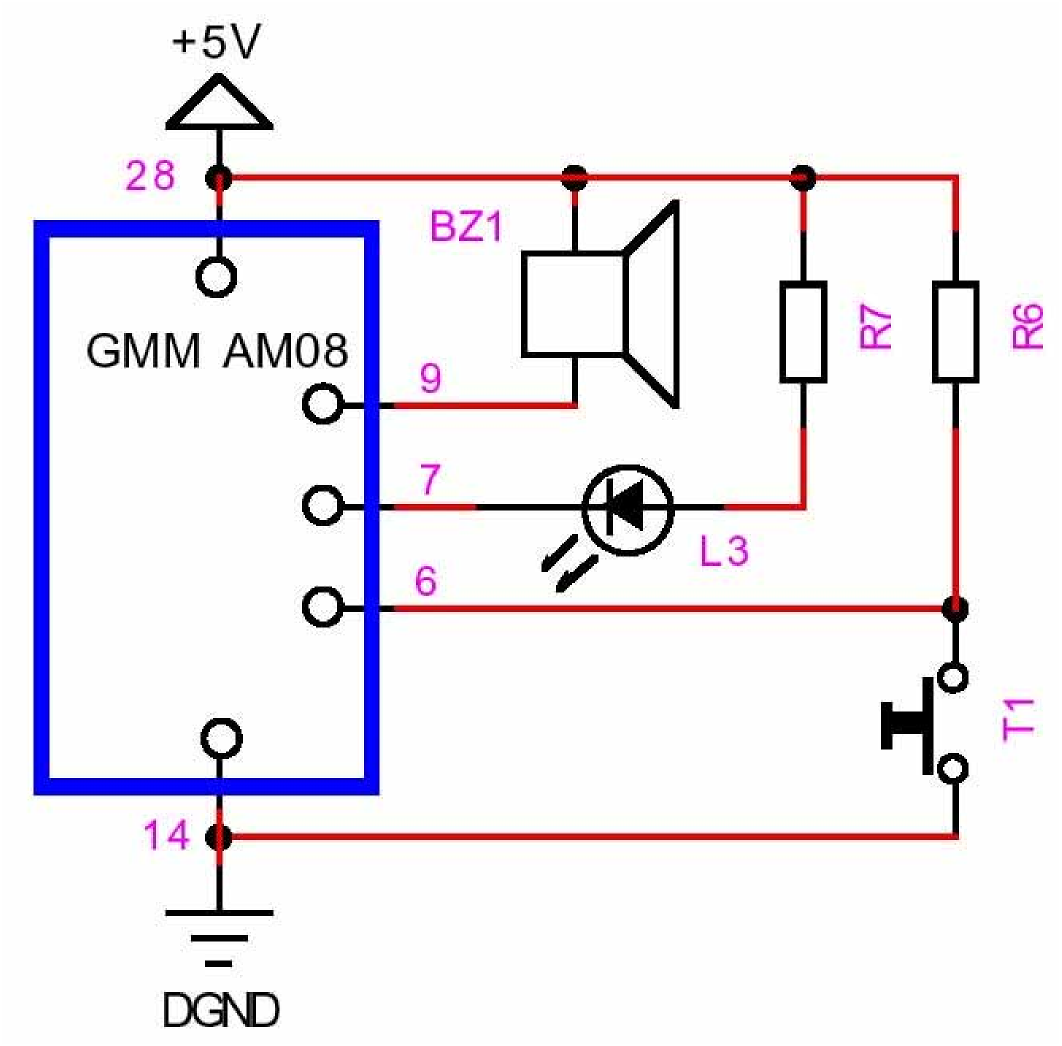
Example.003. LED, Buzzer and Push Button Management.

Added Definitions:
None

Added Declarations:
None

Added Instructions:
None

Added Operators:
None



Electric Diagram Used by Program.

This program executes simple operations on digital **I/O**, by using the **push button T1** as input device and the **Buzzer BZ1** plus **LED L3**, available on **GMM TST3**, as output devices.

- The used **I/O** lines are:

  - pin **15** of **GMM TST3 Z1** socket (= pin **9** of **GMM AM08**) connected to **BZ1 Buzzer**;
  - pin **13** of **GMM TST3 Z1** socket (= pin **7** of **GMM AM08**) connected to green **LED L**3 through **R7** and to **Green** Push Button **T2**;
  - pin **12** of **GMM TST3 Z1** socket (= pin **6** of **GMM AM08**) connected to **Red LED L2** through **R6** and to **Red** Push Button **T1**.

- The program manages the pin **6** as an input line and the pin **9**, **7** as output lines.

- At power on the **LED L3** is turned off and **Buzzer** is silent.

- By pushing the **T1** push button, the **LED L2** is turned on by hardware, as it is electrically connected, and either the **Buzzer** will produce a sound and L3 will be turned on.

- By releasing the **T1** the **Buzzer** will stop the sound and **L3** will turn off.

On previous page it is available the electric diagram that must be realized in order to use the described **Example.003**.

This program is the union of **Example.001** and **Example.002**. In fact it drives both the **L3 LED** and **BZ1 Buzzer**.