Course on BASCOM 8051 - (2)

Theoretic/Practical course on BASCOM 8051 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 5115 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 **FLIP** program, that means ***"FLexible In system Programming"***, released by **Atmel**.

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

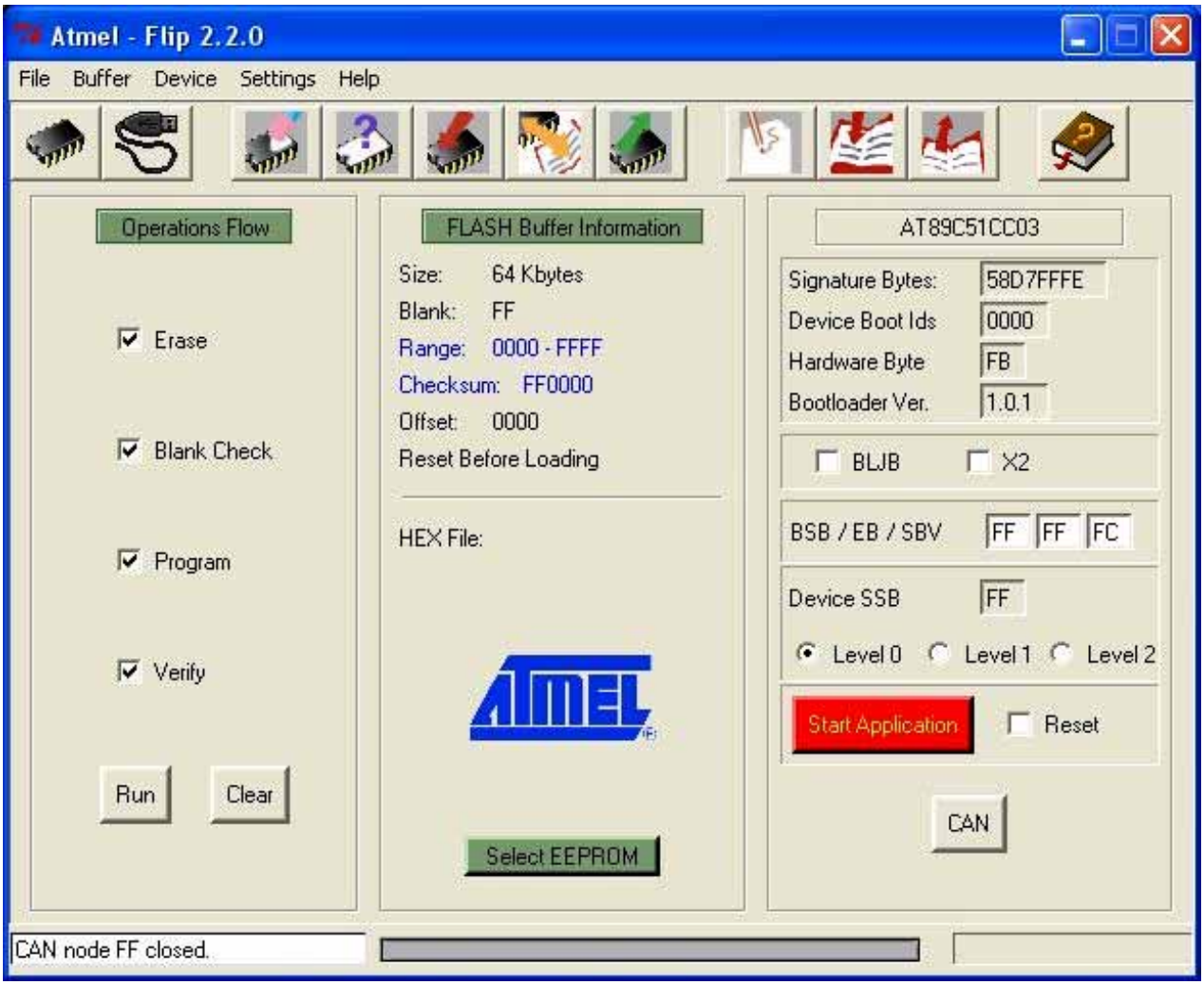
FLIP PROGRAM USE

The following steps describe how the compiled program can be saved on the **FLASH EPROM** of **GMM 5115 Mini Module**, through the proper **FLIP** 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 8051** development tool.

FLIP is a **PC** program, developed and released free of charge by **ATMEL**, that interact with the **Boot Loader** available on **Mini Module**, through one of the possible communication lines. By using **GMM 5115** this line is the **RS 232** asynchronous serial line, that must be obviously available also on **PC**. Thanks to this interaction the **FLIP** 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**.



1) Ensure that the **GMM 5115** asynchronous serial line is **RS 232** buffered, by configuring its dip switch as below described:

DSW1.2 -> ON
DSW1.3 -> ON
DSW1.4 -> OFF
DSW1.5 -> OFF

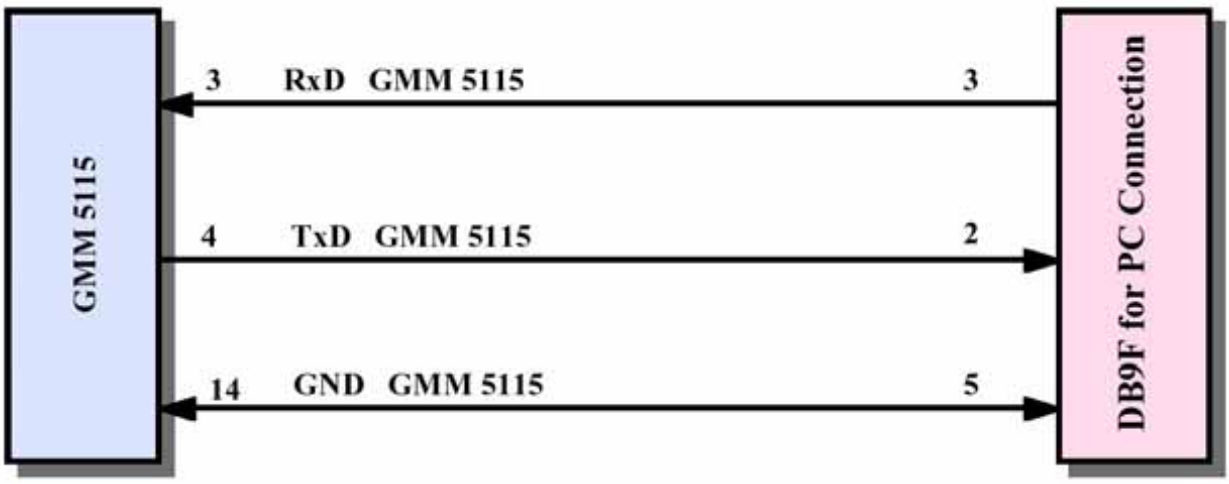
2) Find out the **PC** asynchronous serial line ready to use and the relative male **DB9** connector. Whenever the **PC** hasn't an **RS 232** serial line, please add it, for example by using suitable **USB <-> RS 232** converters.

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 5115** and **PC**, or on the other hand connect the two communication signals and reference ground signal.
When it is used a **GMM 5115** mounted on **GMM TST3** card, this connection is simply obtained with a simple **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 5115** is mounted on a board developed by user, the three communication signals of **PC** must be linked directly to **Mini Module** pin, as described in following figure.



Point to Point RS 232 connection example with PC

4) Search the **FLIP** program on the received **grifo®** **CD** disk, or download it directly from **ATMEL** web site:

http://www.atmel.com/dyn/resources/prod_documents/flip-2_4_6.zip.

On **ATMEL** web site are available different **FLIP** versions and the previous link point to last version without **JAVA** support, in order to simplify the use in this course.

5) Install the **FLIP** 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.

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

7) Select the **DEBUG** mode on **GMM 5115**, that is move **DSW1.1** in **ON** position.

8) Supply power to **GMM 5115** and check that its activity **LED DL1** is enabled and stay enabled during all the following steps.

9) Run the **FLIP** program installed at point **5** by using the proper links added in the **Windows** Start menu and/or desktop.

10) Press the first icon button in the left high corner of the displayed window, or alternatively select the ***Device | Select*** option menu; then select the microcontroller **T89C5115** in the displayed list and finally press **OK**.

11) Press the second icon button in the left high corner of the displayed window, or alternatively select the ***Settings | Communication*** option menu, then select in sequence: ***RS 232***, the serial ***Port: COMx*** of **PC** connected at point **3**, ***Baud: 115200*** and the press the ***Connect*** button. At this point the **FLIP** establish communication with **Mini Module Boot Loader** and it fills in some values inside its main window. Whenever the communication is not possible and after some seconds the ***Timeout Error*** window is displayed, You can test the following solutions: reduce communication speed from **115200** to **19200 Baud**; repeat the point from **6** to **11**; check the hardware serial connection by performing again steps from **1** to **3**.

12) Press the nineth icon button from the left high corner of the displayed window, or alternatively select the ***File | Load HEX File...*** option menu; in the shown dialog window select the **HEX** file to save on the **GMM 5115 FLASH EPROM**.

13) Select all the **4** check box inside the ***Operations Flow*** frame, in order to let **FLIP** execute, in sequence the four operations: Erase, Blank Check, Program, Verify.

14) At this point ensure that inside right frame of the **FLIP** main window, there are the following settings:

X2	*not selected*
BSB / EB / SBV	*00 FF FC*
Device SSB	*FF*

15) Press the ***Run*** button of the main window in order to start the **ISP** operations just selected.

16) Wait until the ISP operations are completely executed. Inside the status bar placed on window bottom, it is displayed the current operation together with a progressive bar that shows the execution progress; the check boxes become red during the execution and then green when the relative operation is completed. So the user must wait that ***Verify*** check box become green.

17) Close the **FLIP** program through the ***X*** in the right high corner of the window, or alternatively with ***File*** **|** ***Exit*** option menu.

18) Select the **RUN** mode on **GMM 5115**, that is move **DSW1.1** in **OFF** position and check that **LED DL1** is disabled.

19) At this point the new compiled program has been saved on **FLASH** memory and it is ready for execution. This happens with a power on or a Reset of **GMM 5115**.

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_BAS51_001.HEX***) must be saved on **GMM 5115 FLASH EPROM**, as described in previous paragraph, and when the Mini Module is restarted in **RUN** mode, 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.

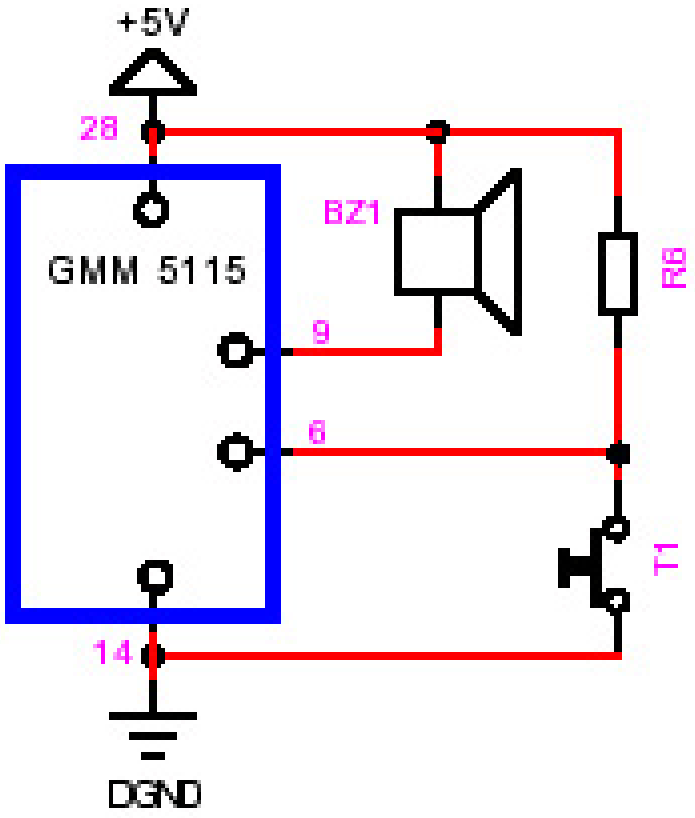
Example.002. Buzzer and Push Button Management.

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 5115**) connected **to BZ1 Buzze**r;
  - pin **12** of **GMM TST3 Z1** socket (= pin **6** of **GMM 5115**) 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**.

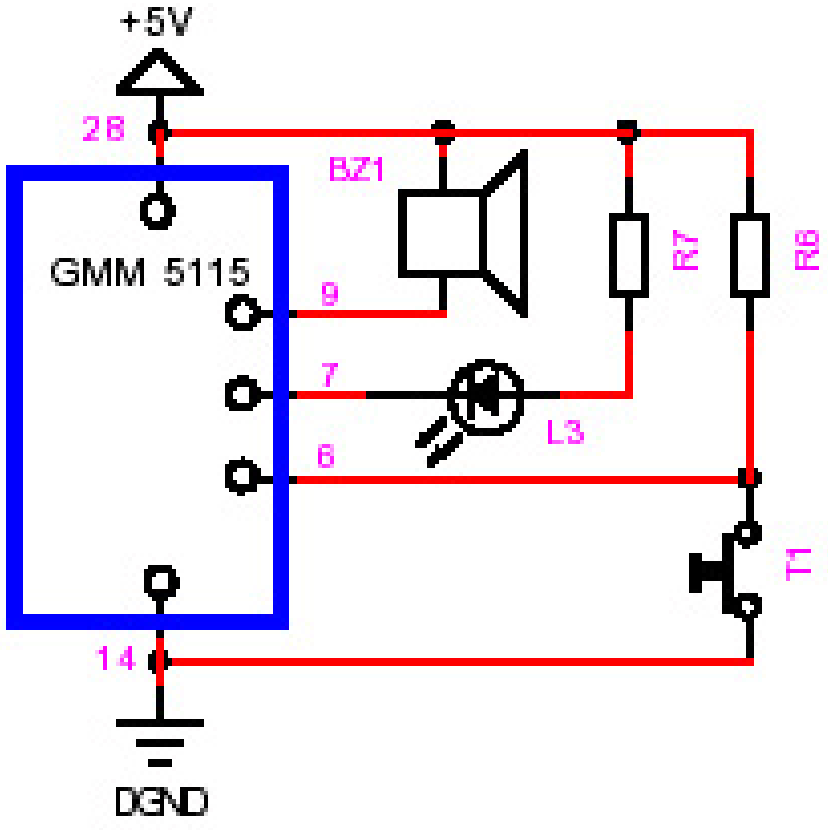
Example.003. LED, Buzzer and Push Button Management.

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 5115**) connected to **BZ1 Buzzer**;
  - pin **13** of **GMM TST3 Z1** socket (= pin **7** of **GMM 5115**) connected to green **LED L3** through **R7** and to green push button **T2**;
  - pin **12** of **GMM TST3 Z1** socket (= pin **6** of **GMM 5115**) 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**.