Embedded Systems Laboratory Manual

ARM 9 TDMI

1. Laboratory Rules

a) Laboratory assessment:
- Presence during the laboratory is mandatory. One time unexcused absence is allowed within the semester.
- Students work on their exercises in one-person groups.
- Each student should have a suitable knowledge before starting the exercise. This is enough to know and understand the material presented during Embedded Systems lecture.
- Every exercise, with the exception of the first exercise, is individually assessed based on the quality of the source code and the submission date. The mark and remarks, if any, are recorded in a spread sheet.
- Every exercise is expected to be completed within one class (2 hours). If the time provided for the exercise is two classes (4 hours), it will be assessed during the second class of the exercise.
- The mark consists of the assessment of the source code (algorithm quality and code aesthetics) as well as the knowledge of the code, fundamentals of C programming and functions of the peripherals of the ARM microcontroller.
- In case of student’s lack of preparation for the class, they will not be allowed to continue the exercise. It is equivalent to an unexcused absence.
- The final mark is an average of all the marks received for each exercise.
- Being excessively late will be not tolerated.

b) Not allowed:
- Opening computer cases.
- Connecting or disconnecting cables without permission of the tutor.
- Carrying out any objects (including technical documentation) from the laboratory.
- Plagiarism of source code, functions or parts of functions and/or procedures from other groups or persons, Internet, etc.
- Presence of people not related to the class or the laboratory.
- Eating and drinking in the vicinity of computer hardware.

c) In case of breaking the rules in point b), a student might face the following consequences:
- The final mark of the evaluated exercise or presented program could be as low as 0% in case of plagiarism.
- Being removed from the class and failing the course.
- Being financially responsible for any repairs, if the damage was a result of intentional action.

2. Description of laboratory equipment

There is the following equipment in the laboratory:

1. Desktop Computers
2. AT91SAM9263-EK Evaluation Kits with 32-bit microcontrollers AT91SAM9263 with ARM9TDMI core.
3. Extension boards with peripheral devices: an encoder, thermometers and an LED display.
4. 12V/1A Power supply units.
5. JTAG adapters connecting the debug communication ports of the microcontrollers with the desktop computers.
6. A set of communication cables (EIA232, USB).

The evaluation kits consist of two modules:

1. A carrier board with an ARM microcontroller,
2. A module with an LCD display, an encoder and a thermometer.

Figure 1 presents a block diagram of the AT91SAM9263 microcontroller. It is equipped with memories such as:

- Volatile memory SRAM – 80 kB working at a full bus speed AMBA (0 waitstate),
- Volatile memory SRAM – 16 kB working at a full bus speed AMBA (0 waitstate),
- Non-volatile memory FLASH – 128 kB working at a full bus speed AMBA (0 waitstate).

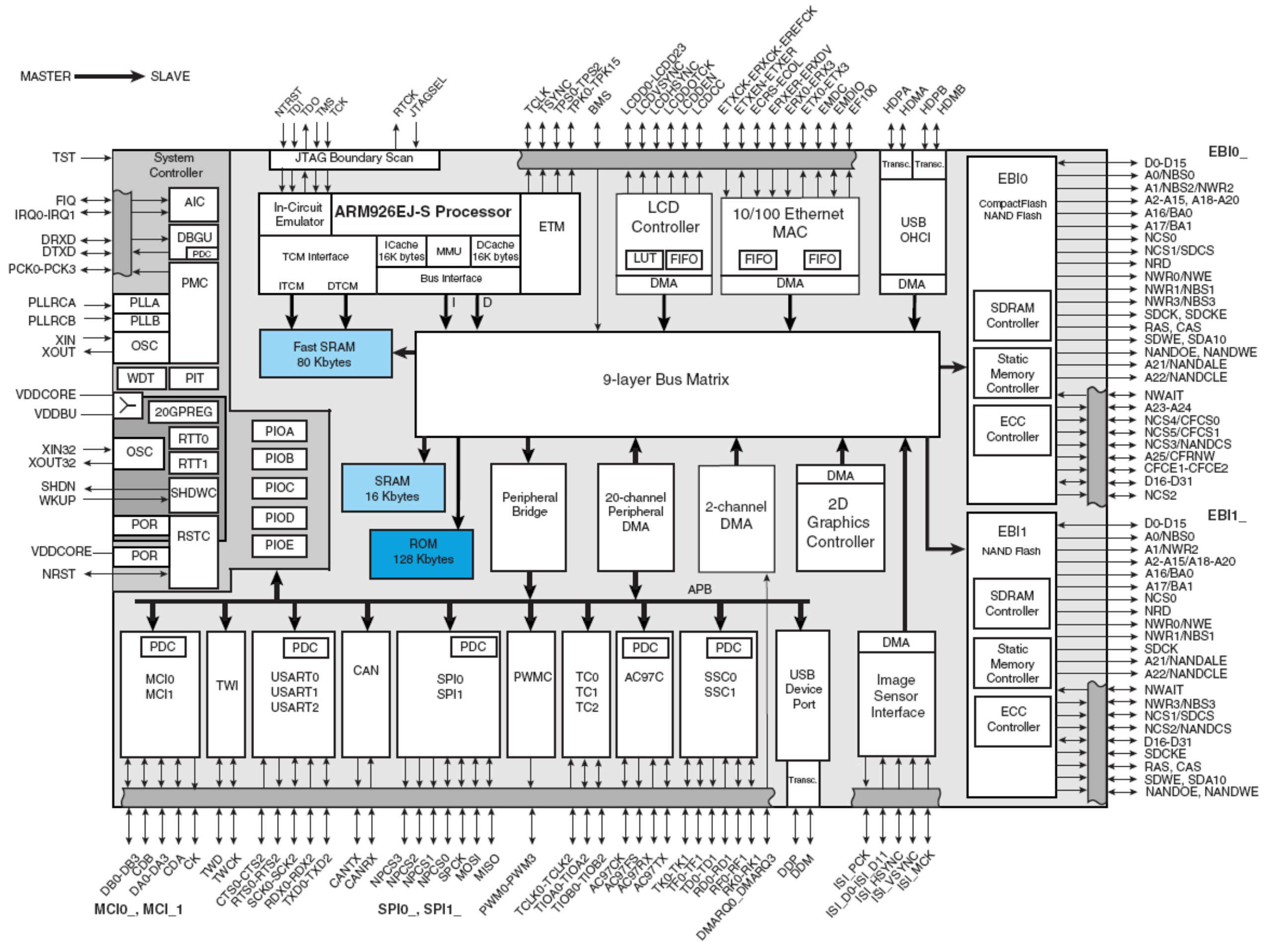

Fig. 1. AT91SAM9263 Block Diagram

The microcontroller works with the following external memories

- Dynamic memory SDRAM – 256 MB (8 M x 32 bit, available on the EBI 0 bus),
- Static memory PSRAM – 4 MB (2 M x 16 bit, available on the EBI 1 bus),
- Non-volatile memory NANDFLASH – 256 MB (256 M x 8 bit, available on the EBI 0 bus),
- Non-volatile memory NORFLASH – 8 MB (4 M x 16 bit, available on the EBI 0 bus),

Figure 2 presents the evaluation board with the elements used for communication with the external devices or the user.

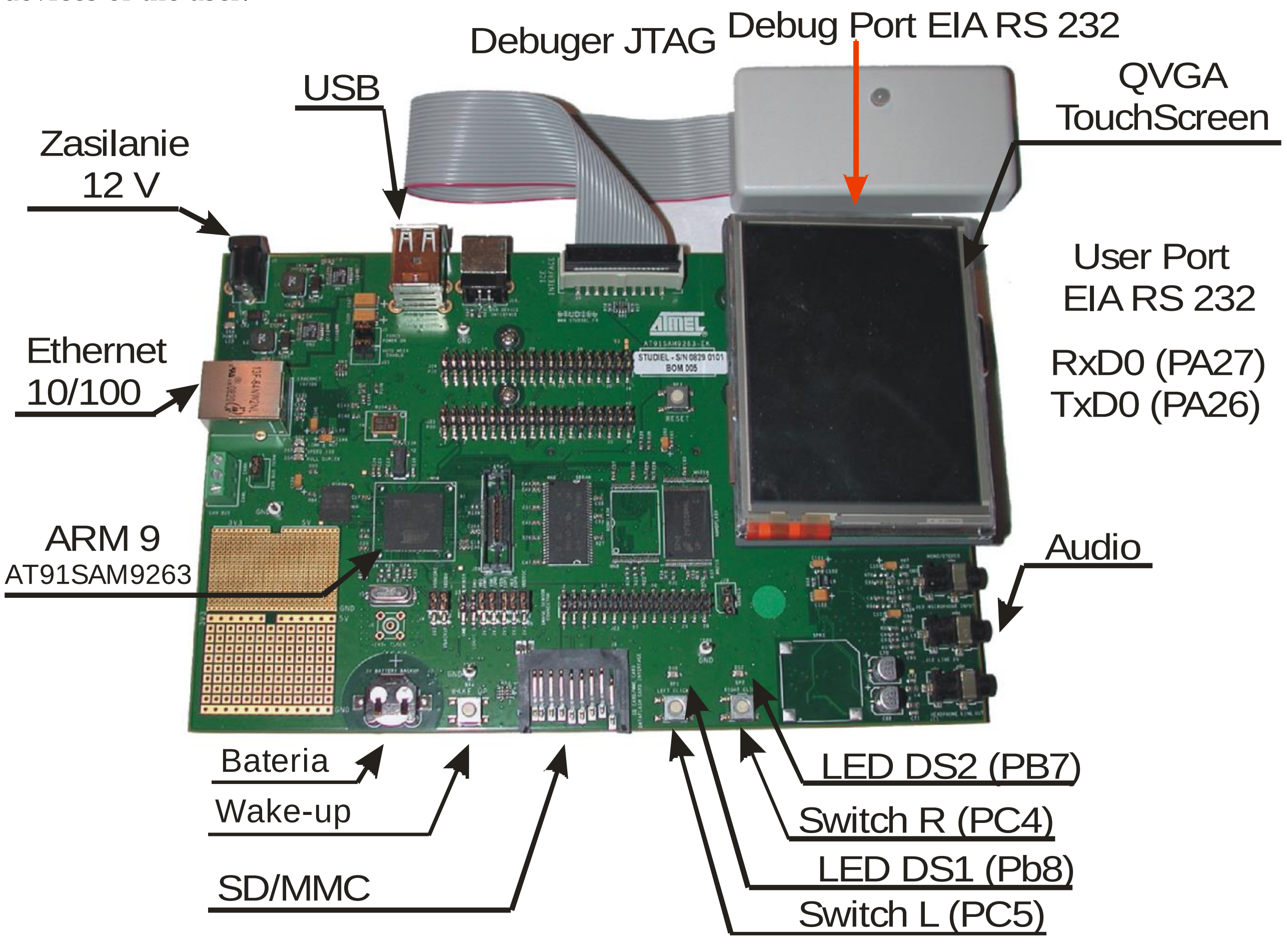

Fig. 2. AT91SAM9263 Evaluation Board

The board is equipped with a set of interfaces and peripheral devices:

- Interfaces:
  - Ethernet 100-base TX,
  - USB FS device Master,
  - 2 x USB FS Host,
  - CAN 2.0B,
  - EIA RS232 (general purpose),
  - EIA RS232 (for debugging and diagnostic purposes),
- Peripheral devices:
  - Display: 3.5" 1/4 VGA TFT LCD with a touch screen,
  - Audio codec: AC97 Audio,
  - Debug tool with JTAG interface,
  - Programming tool with JTAG interface,
  - SD/SDIO/MMC card slot.

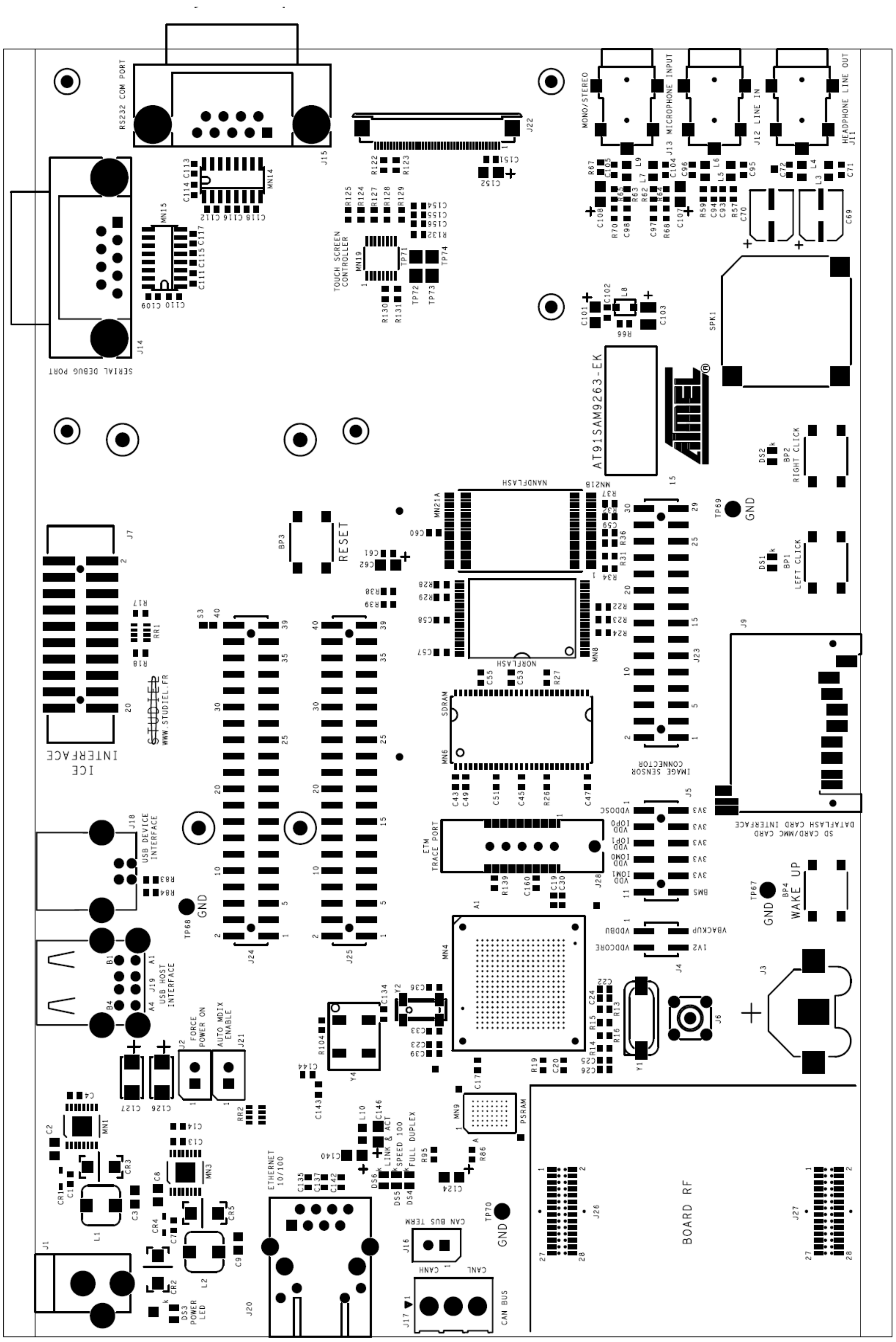

Fig. 3. Component placement on the evaluation board – Top Side

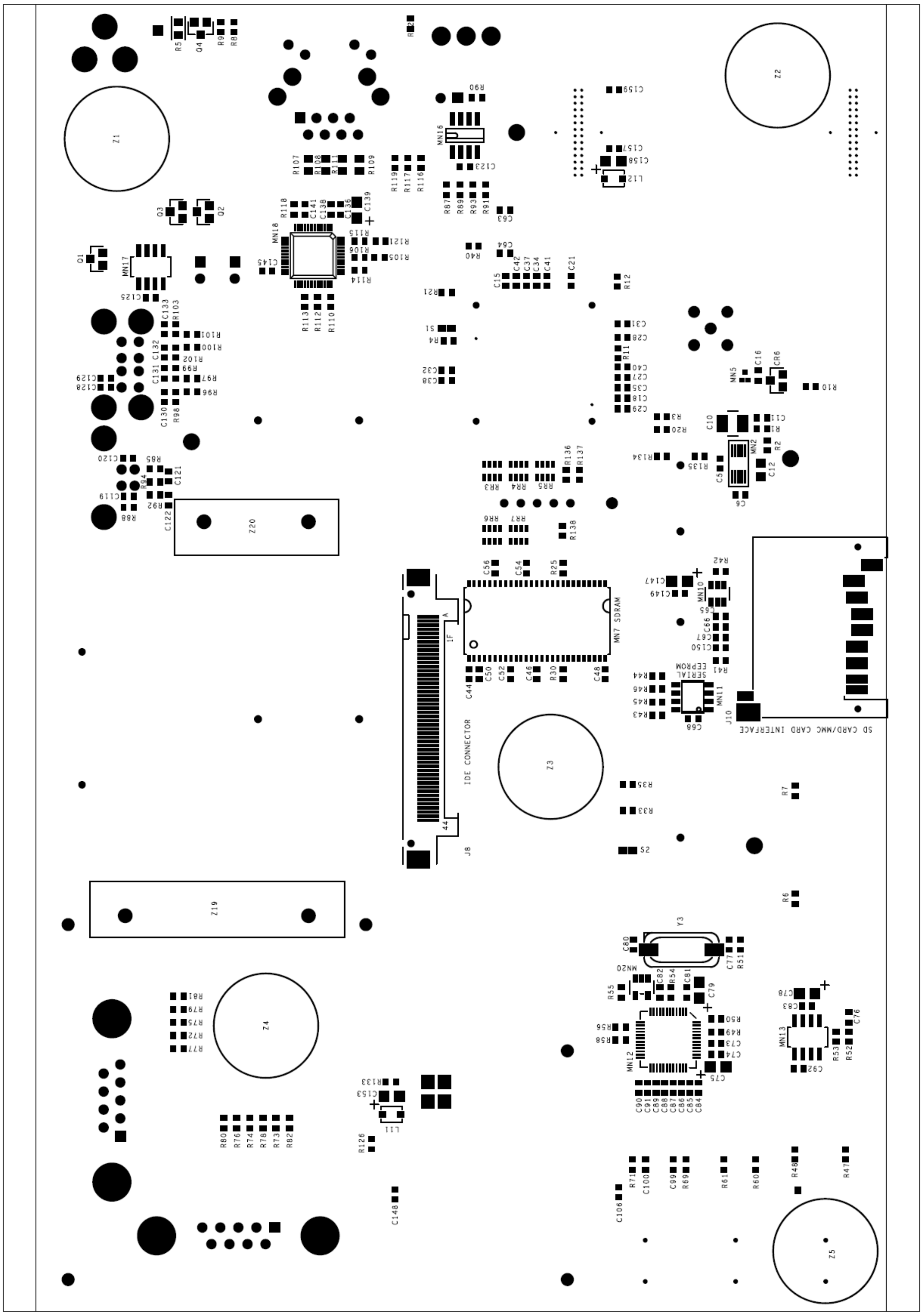

Fig. 4. Component placement on the evaluation board – Bottom Side

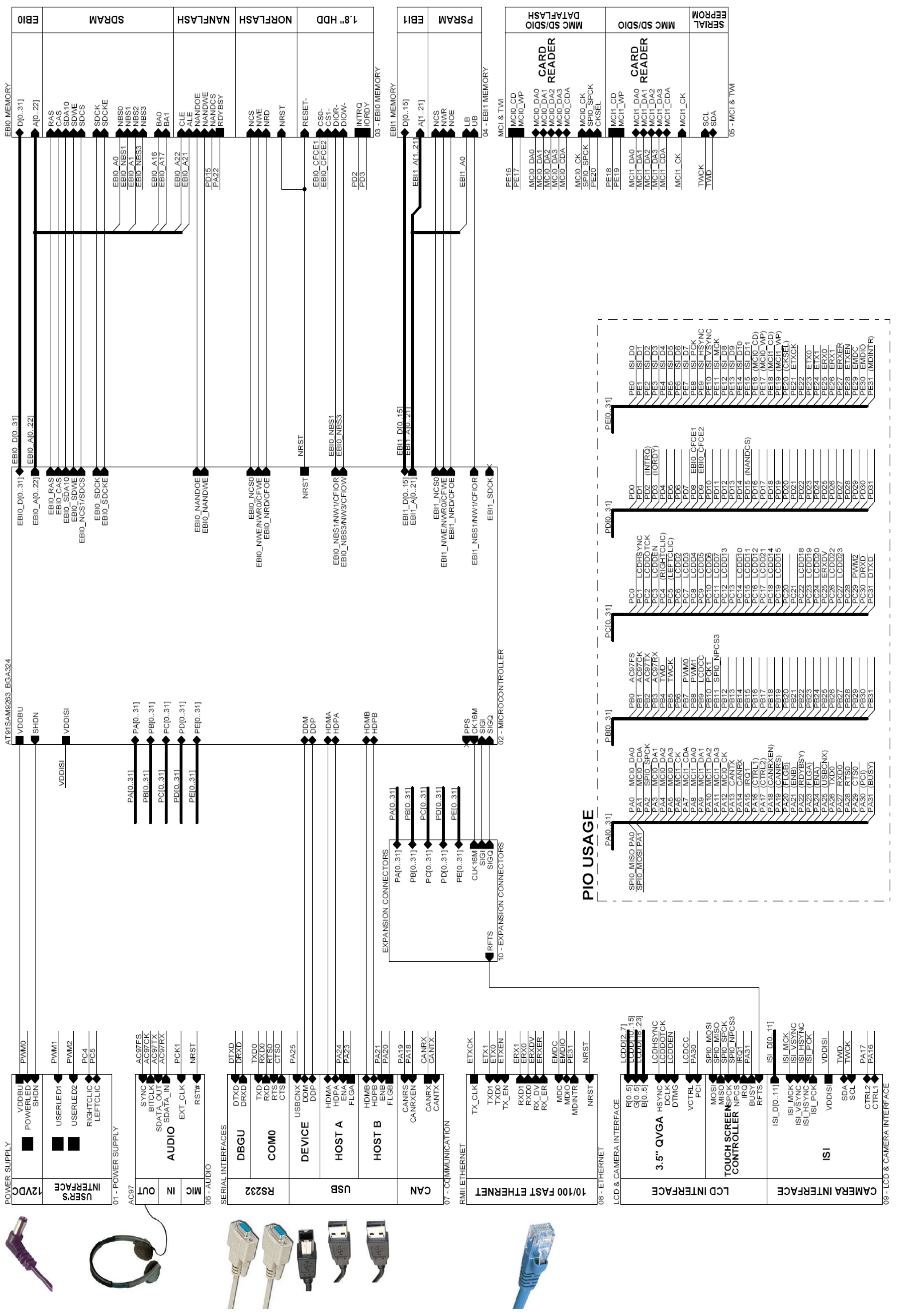

Fig. 5. AT91SAM9263-EK Block Diagram

In Figure 3 and 4 the component placement of top and bottom sides respectively is presented, while in Figure 5 the AT91SAM9263-EK Block Diagram with the included interfaces, memories and peripheral devices is shown.

3. Circuits used during the exercises with their control registers

During the laboratory, students will be expected to use General Purpose Input/Output ports, UART transmitters, timers, interrupt controllers and the external peripheral devices on the module board.

3.1. Components on the evaluation board

In figure 6 the circuits of the available LEDs and push buttons are shown. Names of the microcontroller ports used to control these components are given in Figure 2 in brackets. The detailed connectivity of push-buttons and two LEDs could be found in the Documentation for ARM At91SAM9263 starter-kit that is available in the Embedded Systems web page: http://fiona.dmcs.pl/es/.

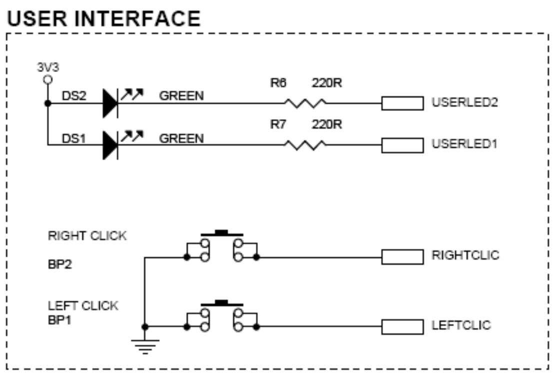

Fig. 6. Schematics representing the available LEDs and buttons

3.2. Components on the extension board

Figure 7 presents a photograph of the extension board (version 1) including double 7-segments LED display, thermometer, encoder and buzzer.

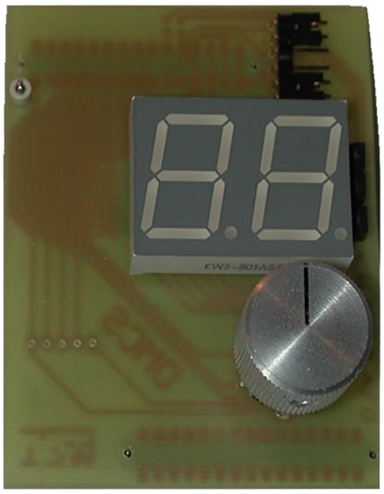
Fig. 7. Photo of the board with an encoder and a display

The extension board contains the following elements:

- Rotary encoder,
- Digital thermometer,
- LED display,
- Buzzer.

The schematic of the aforementioned elements is presented in Figure 8.

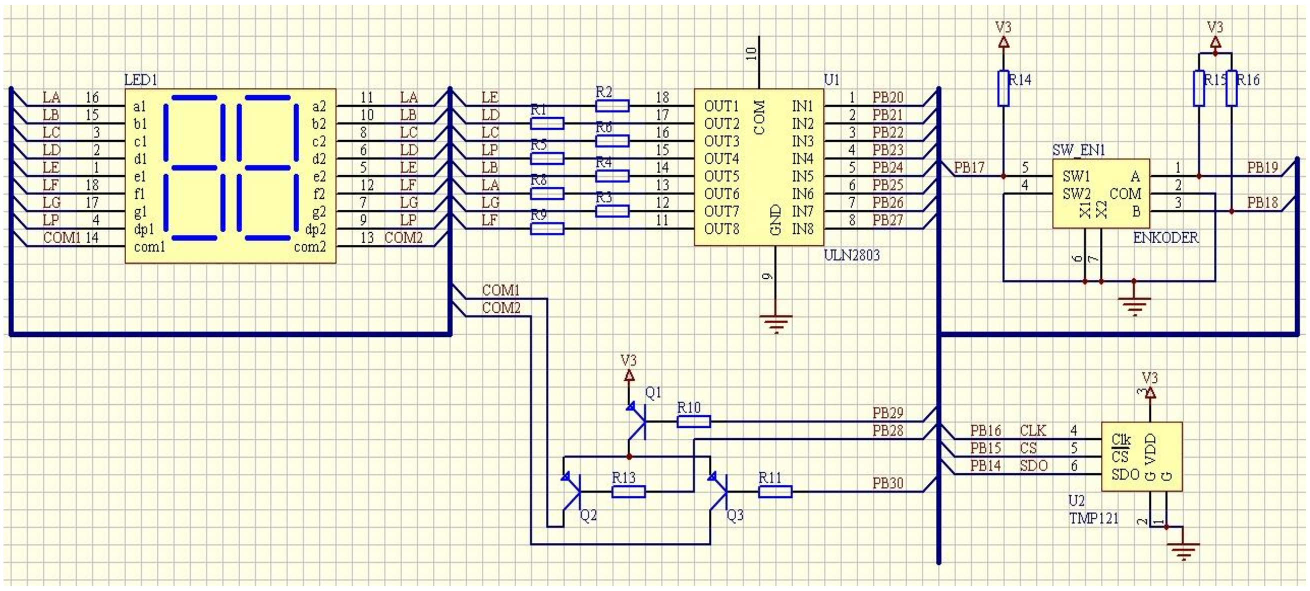

Fig. 8. Extension board schematics

3.2.1 LED segment display

The board contains two 7-segment LED displays, which can display digits or symbols and the dot near each digit. The appropriate segments of both displays are connected together. Only one digit can be displayed at the time. The digit is selected with p-n-p transistors Q2 and Q3. These transistors are activated by applying low level ('0') to pins PB28 and PB30, respectively. For testing purpose, one or both transistors can be constantly activated. Each segment can be controlled by correctly setting the output of a respective pin. Figure 9 presents the schematic view of display with segment names.

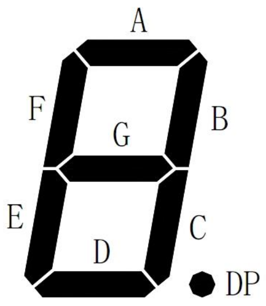

Fig. 9. LED display assignment

Display segment	Processor pin
Segment A	PB25
Segment B	PB24
Segment C	PB22
Segment D	PB21
Segment E	PB20
Segment F	PB27
Segment G	PB26
Dot (DP)	PB23

The display brightness can be controlled with PWM. This can be done in software or in hardware, by means of controlling Q1 transistor. For testing purposes this transistor can be enabled constantly (full brightness). Q1 transistor is enabled with low level (‘0’) on pin PB29.

To display digits or other symbols one must provide a function, which translates a sign (digit) to a combination of enabled segments, which can be done for example with an array. To use both displays one must display both digits alternately fast enough (with frequency of at least 60 Hz).

3.2.2 Rotary encoder

Rotary encoder available on the board consists of two independent elements: a monostable switch and a two-channel quadrature encoder.

Pressing the knob causes activation of the switch while turning the knob causes generation of a series of impulses. Pressing the button can be detected independently of turning the knob at the same time. When the knob is pressed, low level (‘0’) is set at the output of the encoder.

Depending on the direction of turns, the encoder impulses are as presented in the diagrams in Figure 10.

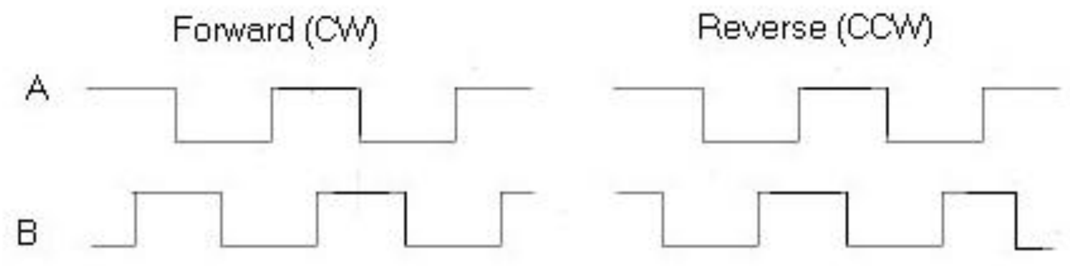

Fig. 10. Waveforms on the output of the encoder

To detect the direction of turns one must choose one of the channels as a reference channel and arbitrarily choose one of the edges (rising or falling), and then always check the state of the other channel at the selected edge. The encoder is connected to the processor as follows:

- Channel A: PB19
- Channel B: PB18
- Button: PB17

One can also measure the speed of turning the knob. To do it one must measure time between impulses (or frequency) of one of the channels.

3.2.3 Thermometer

TMP121 or TMP123 thermometer is installed on the board. The specification of the chip is available in printed form in the laboratory or on manufacturer’s website (www.ti.com). The thermometer can be used by generating signal waveforms in the software according to the specification. The thermometer uses the SPI (Serial Peripheral Interface) communication protocol.

The chip is connected as follows:

- Clock (CLK, SPI1_SPCK): PB14
- Data (SDO/MISO, SPI1_MISO): PB12
- Chip Select signal (CS, SPI1_NPCS0): PB15

The software implementation of SPI transceiver should be used to read the temperature form the device. The SPI protocol must be generated in the program using the GPIO functionality.

4. Using the software in the laboratories

The following tools, available in GNU license (http://www.gnuarm.org/), are used in the classes:

- GCC-4.3 toolchain (Linux):
  - Compile: gcc-4.3.2,
  - Utilities: binutils-2.19,
  - C/C++ libraries: newlib-1.16,
  - GDB-compatible debugger: insight-6.8.

4.1 ARM processors

- Assembly of programs in assembler
  - Go to the directory with source code
  - Type **arm-elf-asm** <*file name*>, to assemble the program. If assembly ends with success, elf file will be generated with the same name as source file
  - The *make* program can be used to simplify the compilation procedure

- Compiling C code
  - Go to the directory with source code
  - Type **arm-elf-gcc** < *file name* > with suitable parameters
  - The *make* program can be used to simplify the compilation procedure. The Makefile script is available on the Embedded Systems web page.

4.2 GDB Debugger

- GDB debugger is run with a command dependent on the architecture:
  - PC computer: gdb
  - ARM processor: **arm-elf-gdb**

  In case of ARM processors, OpenOCD must be run prior to debugging and be running for the whole time of debugging (e.g. in another terminal)
- To set breakpoint, use command *breakpoint* (shorthand b) <line number> or <function name>
- To run the program, use command *run* (shorthand r)
- After hitting a breakpoint, one can continue execution with command *continue* (c), or run in step by step entering into functions - command *step* (s) or without entering - command *next* (n)
- To return to executing top level function, one can use command *finish*
- To print the contents of a register or a variable one can use the command *print* (p)
- To interrupt the execution of the program one must press CTRL+C
- To close the debugger one must type *quit* (q)

Modifiers:
/x – display the result in hex

/t – display the result in bin
/d – display the result in dec

4.3 Minicom Terminal

- The Minicom program can be started with a command ***minicom***
- If the **minicom** program fails to start and exits with an error that it cannot save file „lock-file”, it should be started with parameter -t
- After the communication is initiated, the characters received from the microcontroller will be displayed in the terminal.
- To send a character an appropriate key on the PC keyboard must be pressed
- The window can be cleared with key combination ***CTRL+A C***
- The **minicom** program can be closed with key combination ***CTRL+A Q***
- The parameters of **minicom** program can be altered with key combination CTRL+A O
- At the first start of the program one should configure at least:
  - Serial port device /dev/ttyS0 or /dev/ttyS1. If the device is unknown, please try the command ***dmesg | grep tty***
  - Baud rate, parity and number of bits (in accordance to settings in the microcontroller)
  - Turn off data flow control, both hardware and software
- It is recommended to save the default configuration of the program, with the option ’save as dfl’
- If the serial port device changes during the execution of the program, one must save the configuration, terminate the program and restart it.