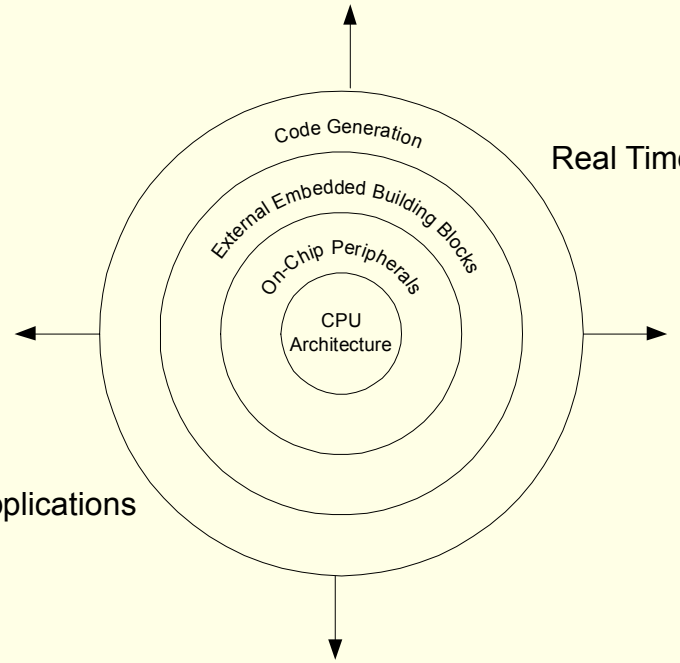
Introduction to SANGO

(R8C/Tiny System Simulator)

Contents

Introduction

SANGO - R8C/Tiny System Simulator gives an excellent simulation environment for the industry's most popular 16-bit microcontroller family, R8C/Tiny. It gives all required facilities to enable the system designers to start projects right from the scratch and finish them with ease and confidence.



SANGO is the total solution giving many state of art features meeting the needs of the designers possessing different levels of expertise. If you are a beginner, then you can easily learn about R8C/Tiny based embedded solutions without any hardware. If you are an experienced designer, you may find most of the required facilities built in the simulator enabling you to complete your next project without waiting for the target hardware.

Device Selection

Devices in the groups R8C/11, R8C/13, R8C/1A, R8C/1B, R8C/24 and R8C/25 of R8C/Tiny family are supported.

Program Editing

Powerful editing features for generating your programs and the facility to call an external Assembler to process input programs.

ClearView

ClearView facility gives all the internal architectural details in multiple windows. Information about the Program, Memory, Registers, SFRs are clearly presented in many windows to make you understand the program flow very easily.

Program Execution

A variety of program execution options including Single Stroke full speed execution, SingleStep, StepOver and BreakPoint execution modes give you total control over the target program.

ClearView updates all the windows with the correct and latest data and it is a convenient help during your debugging operations.

You may find this Simulator simplifies the most difficult operation of the program development, debugging, into a very simple and interesting task.

Simulation Facilities

Powerful simulation facilities are incorporated to complete your next embedded solution:

- CPU configuration for the application.
- Facility to develop the programs and also verify them.
- On-chip clock structure, interrupt facilities.
- All the on-chip peripherals including ports, timer/counters, communication facilities.
- Facilities are available to use all the features of the selected micons without any physical hardware.
- Also plenty of external embedded modules are simulated for the application.
  - Range of Plain Point LEDs and Seven Segment LED options.
  - LCD modules in many configurations.
  - Momentary ON keys.
  - A variety of keypads upto 4 X 8 key matrix.
  - Toggle switches.
  - All modes of onchip serial port communication facility.
  - IIC components including RTC, EEPROMs.
  - SPI Bus based EEPROM devices.

Code Generation Facilities

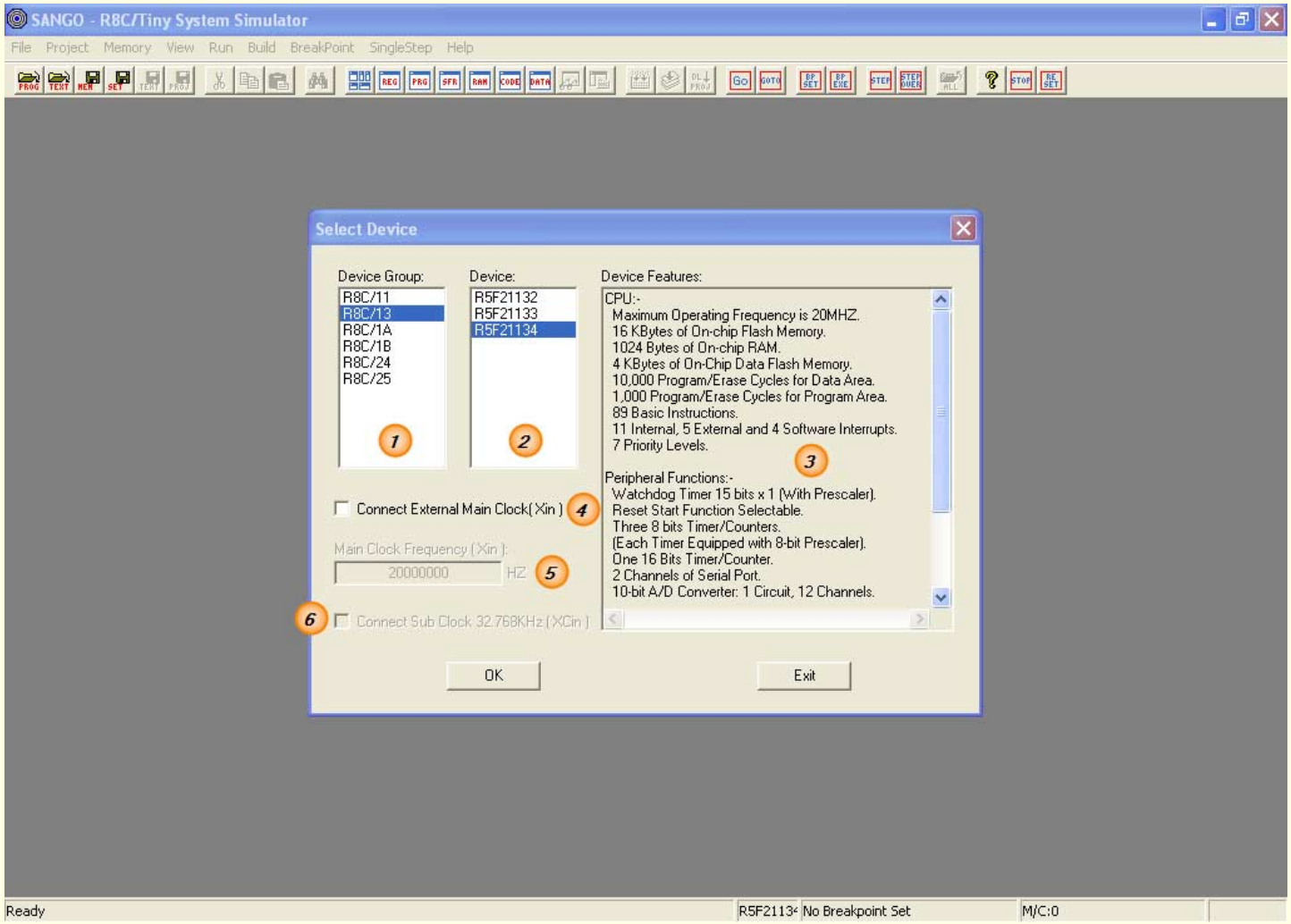
Powerful and versatile Code Generating facility enables you to generate the exact and compact code for many possible application oriented interfacing options.

You can simply define your exact needs and get the target code at a press of button at anywhere in your program flow. The code gets embedded into your application program automatically.

You are assured of trouble free working of final code in the real time.

- All on-chip peripherals including CPU clock selection.
- Interfacing IIC/SPI Bus devices.
- Range of keypads.
- Many LED/LCD interfacing possibilities.

Device Selection and Clock Setting



1. Select the device group from the list.

2. Select a particular device form the list of devices.

3. The features of the selected device will be displayed here.

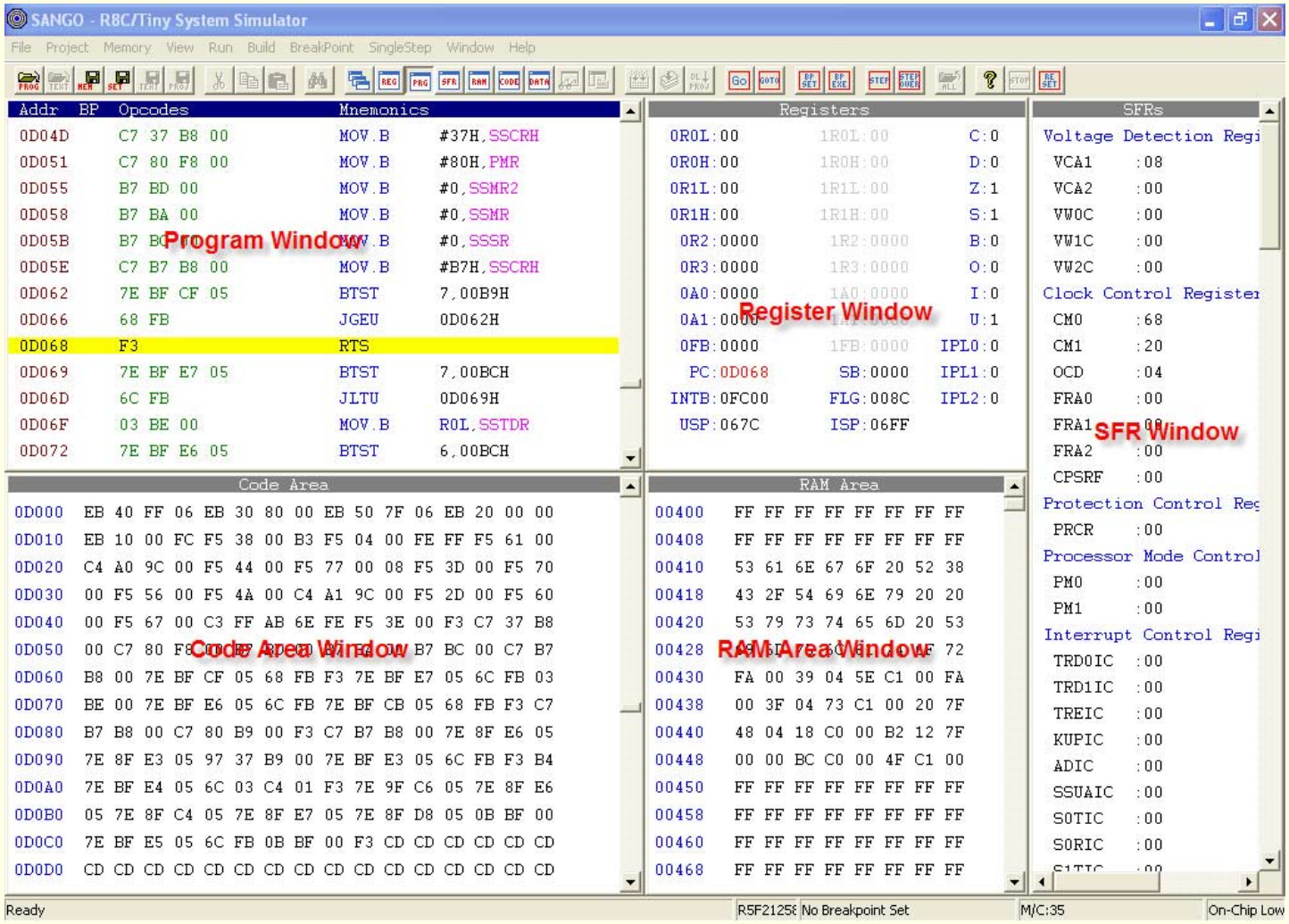
4. Check the check box to connect external clock to system.

5. Enter the frequency of the external clock in Hz.

6. Some of the devices have facility to connect a sub clock with frequency of 32.768 KHz. If the check box is enabled, the subclock will be connected to the controller.

Click OK button to start the simulator with the environment meant for the selected device.

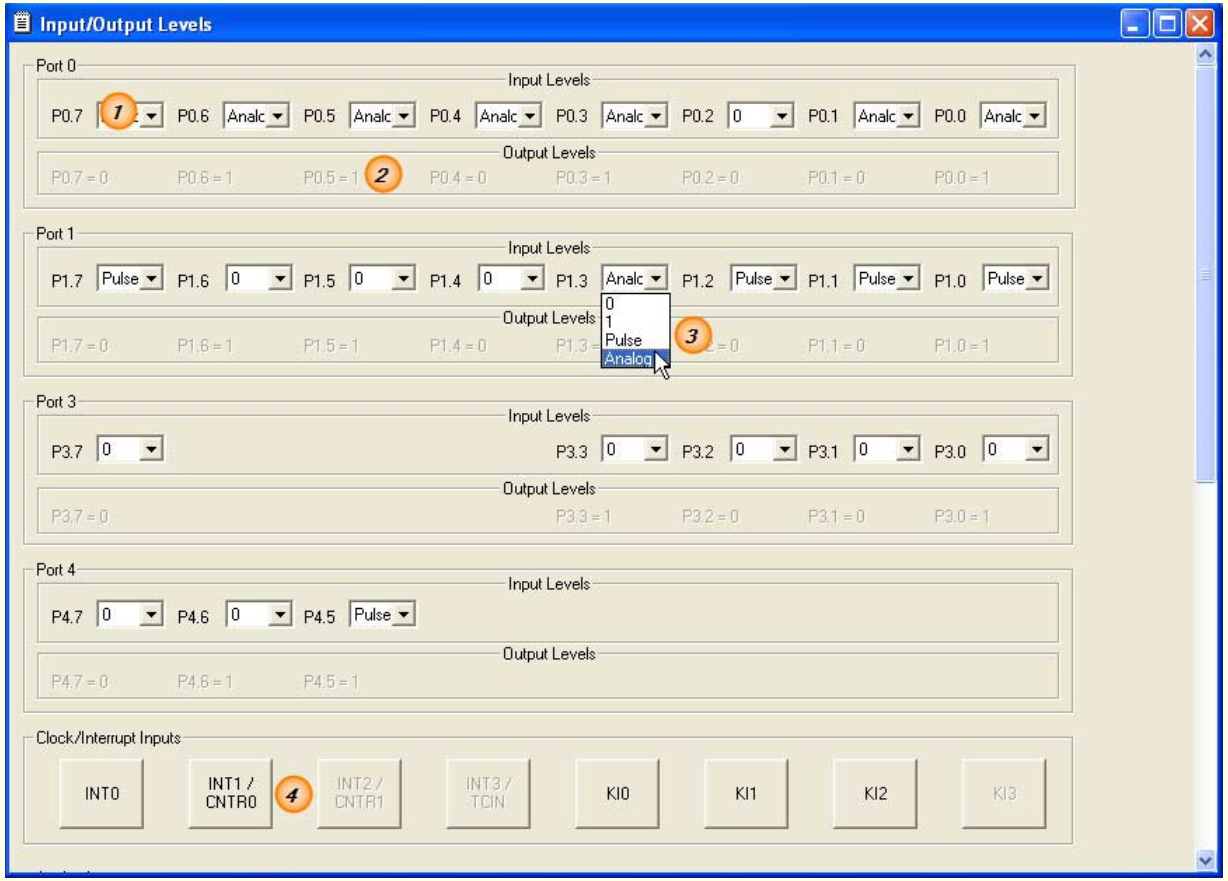
ClearView Window Structure



This is an optimized arrangement where windows are strategically placed in the display. Display area of the monitor is divided into five windows. Windows meant for **Program**, **Register**, **RAM Area**, **Code Area**, and **SFR** are placed in the ClearView. Size and position of the windows can't be changed. Scrolling facility is available wherever it is required.

This ClearView gives a complete picture on the internal architecture in a single screen while debugging the program code.

I/O Simulation Window



1. Displays the input levels of each port line and facility is available to set '0' or '1' level to the input line.

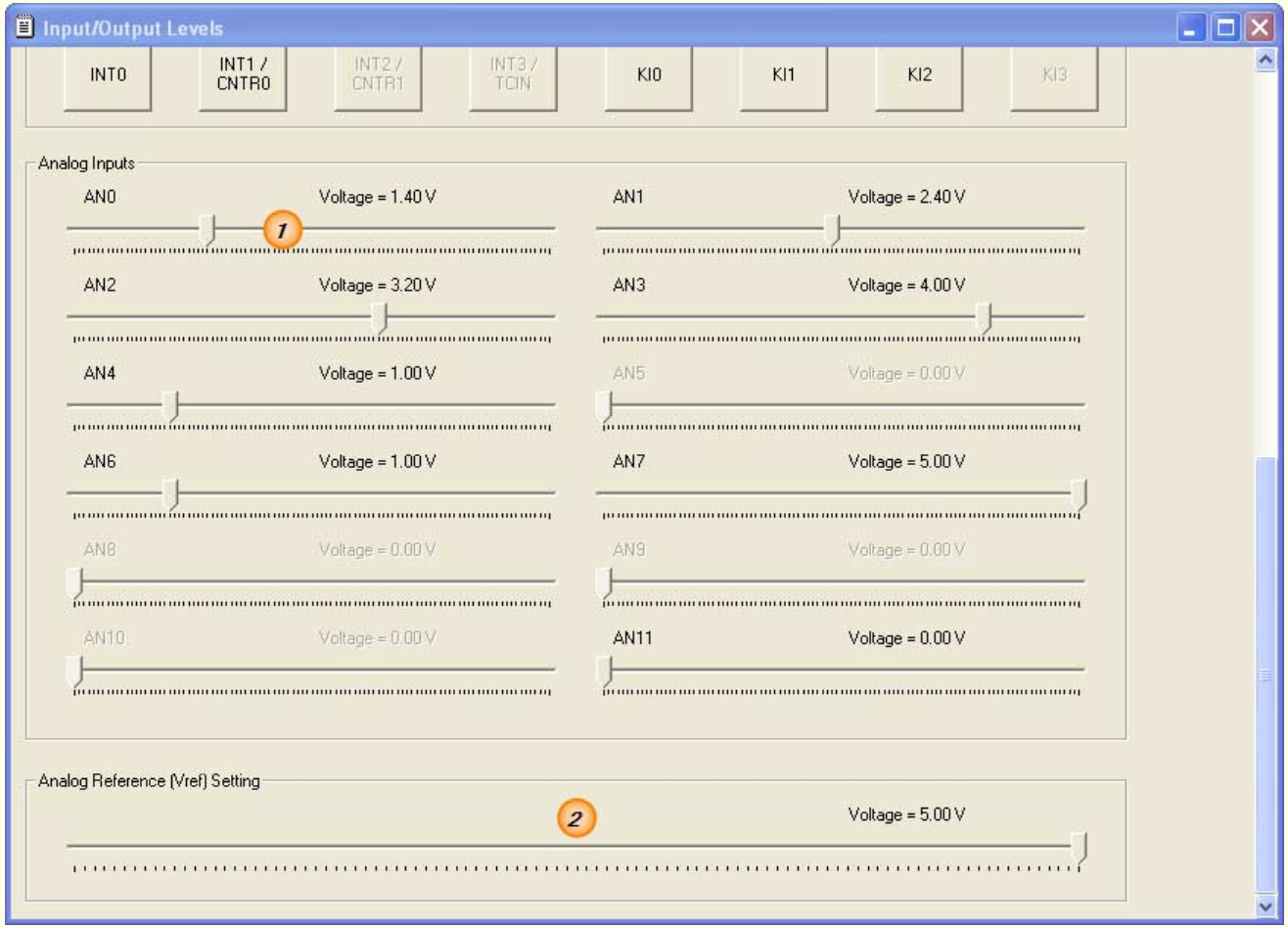
2. Indicates the output levels of all port lines.

3. To simulate ADC , a third option '**Analog**' to the respective port lines is provided. If this option is selected to a particular port line, a variable analog input source is enabled for that line along with the analog reference source.

4. To simulate interrupts and timers with external clock, a fourth option called '**Pulse**' is provided for the respective lines. A button is available to generate pulses to the timer or interrupt lines.

I/O window comes with all the ports of I/O lines with facility to simulate timers/counters at the respective I/O pins. Interrupt conditions can be simulated. Facility to test ADC is also available in this window.

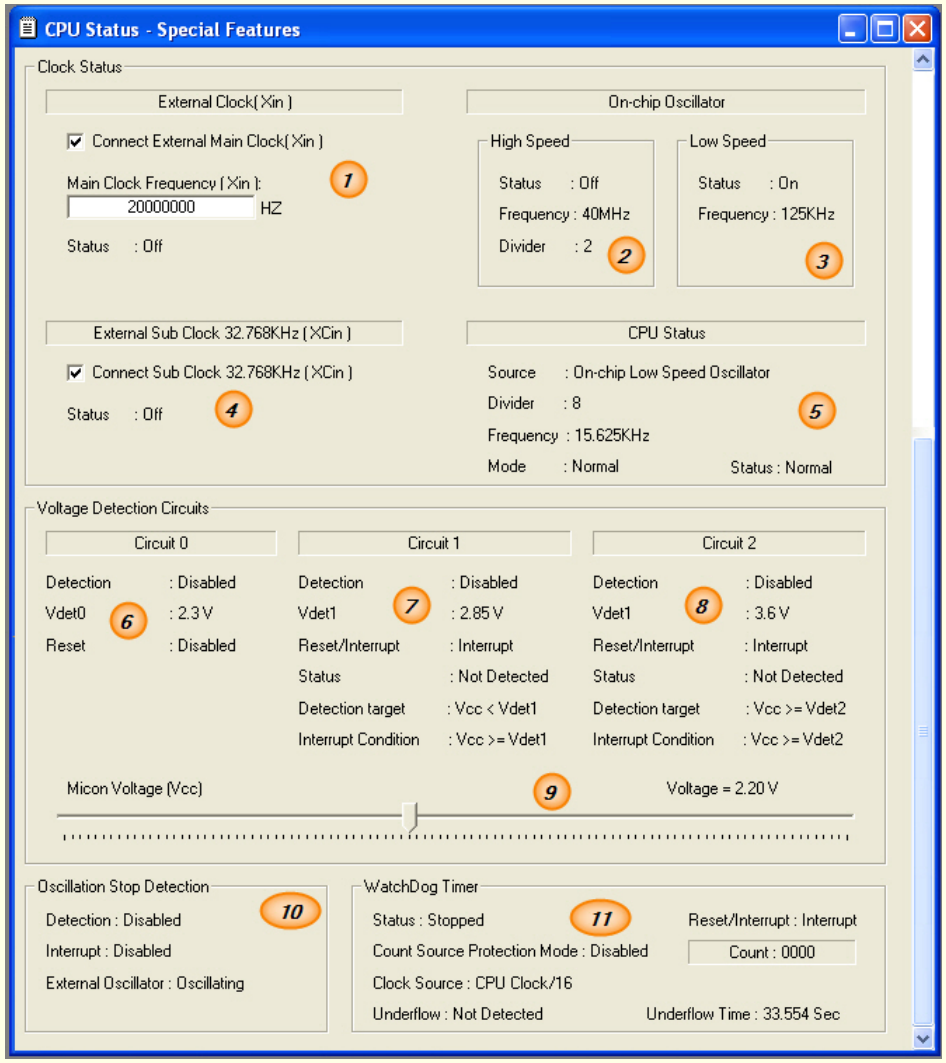
I/O Simulation Window



1 A variable source is connected to the port line, if '**Analog**' option of that line enabled. A slider control is provided for this purpose. Using this slider, the voltage level to the selected port line can be varied. The simulated voltage gets displayed. Maximum voltage input is 5V.

2 A variable source is connected to the **Vref** of the ADC using another slider. Maximum voltage input is 5V.

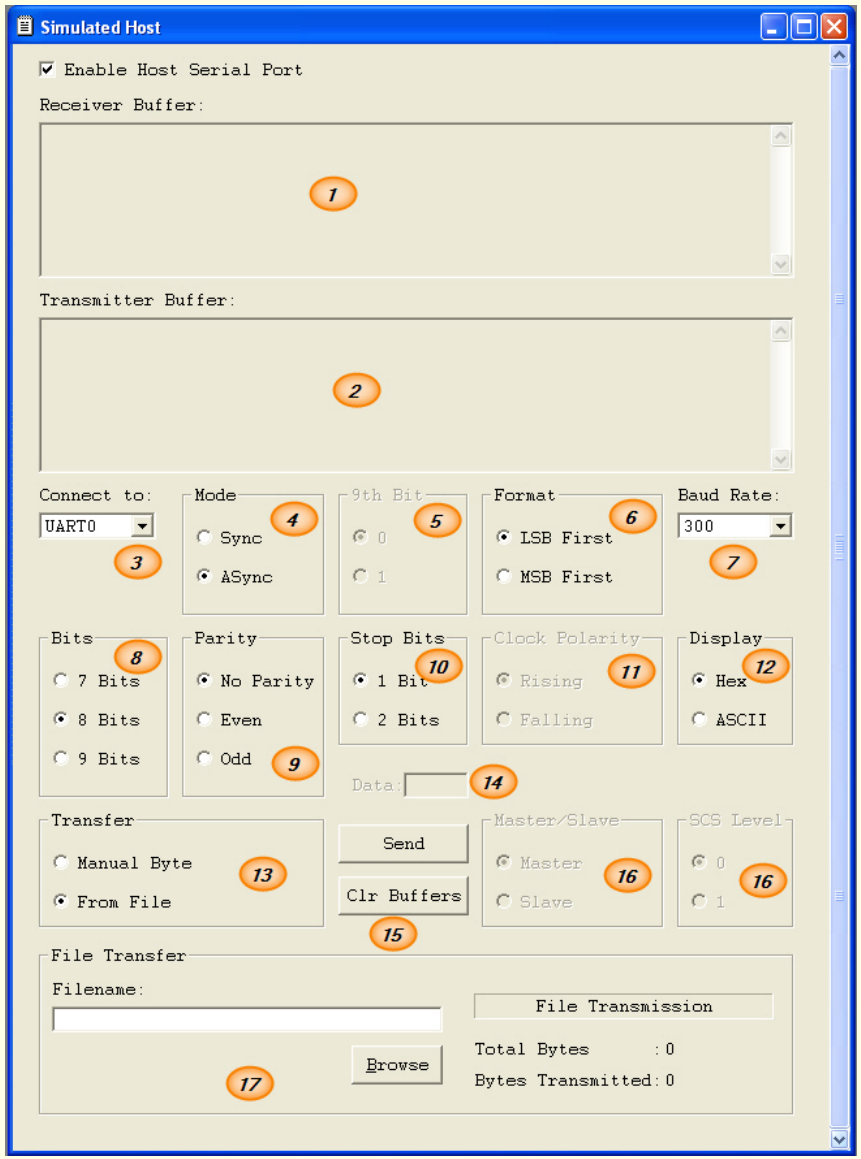
CPU Status Window



1. Gives the details of the external main clock, its status and its frequency. The frequency can also be edited here for the target application.
2. Displays the status of the high speed on-chip oscillator: The frequency of the high speed oscillator and its division ratio.
3. Displays the status of the low speed on-chip oscillator along with its frequency of oscillation.
4. Gives the details of the external sub clock, its status and its frequency.
5. Gives the details of selected CPU clock (either external main clock or on-chip oscillators or external sub clock)
6. Gives the details of the voltage detection circuit 0.
7. Gives the details of the voltage detection circuit 1.
8. Gives the details of the voltage detection circuit 2.
9. The CPU voltage can be increased or decreased using this slider. By altering the CPU voltage, the voltage detection operation can be similated and examined when using this window.
10. Gives details of the oscillation circuit, whether the circuit is oscillating or not and about the Oscillation Stop Detection Circuit.
11. The details of watchdog timer, its status, underflow time and the timer count value are indicated.

The CPU Status window gives the details of the CPU clock frequency, on-chip oscillators, voltage detection circuits, watchdog timer, etc.

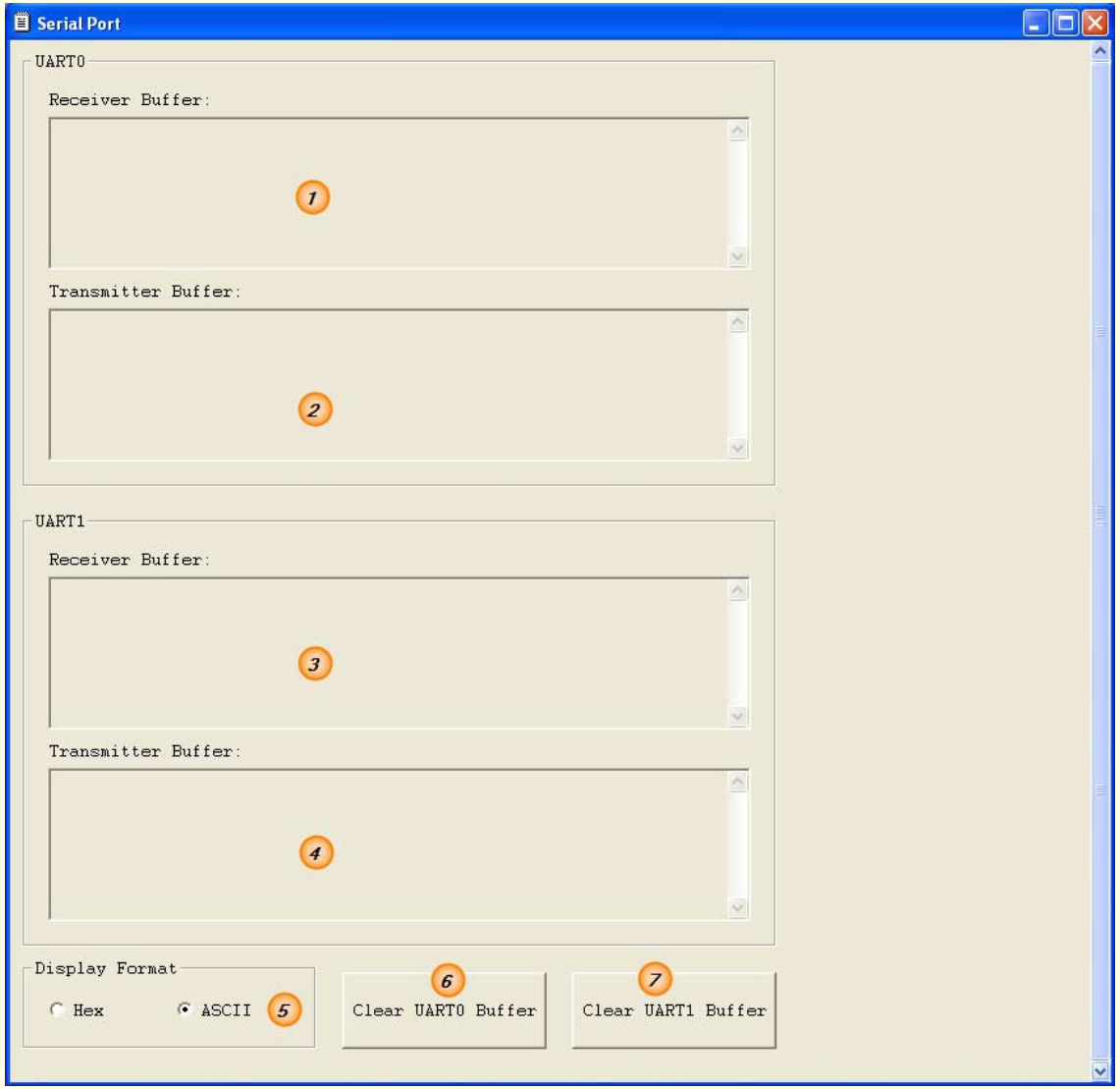
Simulated Host for Serial Port



Host window acts as the host to the serial port UART0, UART1 and SSU.

1. Displays the serially received data.
2. Transmits the data serially with the defined baud rate and format.
3. Select the channel UART0 or UART1 or SSU.
4. Select the mode Synchronous or Asynchronous.
5. Select the 9th bit data for multiprocessor communiaction
6. Choose the format – LSB first / MSB first
7. Choose the baud rate for transmission and reception.
8. Select the no of bits for Asynchronous mode 7 bits / 8 bits / 9 bits.
9. Select the parity type for Asynchronous mode – no parity / even / odd
10. Choose the stop bits for Asynchronous mode.
11. Choose the Clock polarity for Synchronous mode.
12. Choose the display format for both the buffers.
13. Host Setting – Transfer data from file or the entered byte in the window.
14. Enter data for transfer.
15. Click this button to clear both the buffers.
16. This setting applies for Synchronous mode and SSU mode. Select the host to simulate a master or a slave.
17. Select the file, to transfer from the host. The file transfer details are displayed.

On-chip Serial Port Simulation

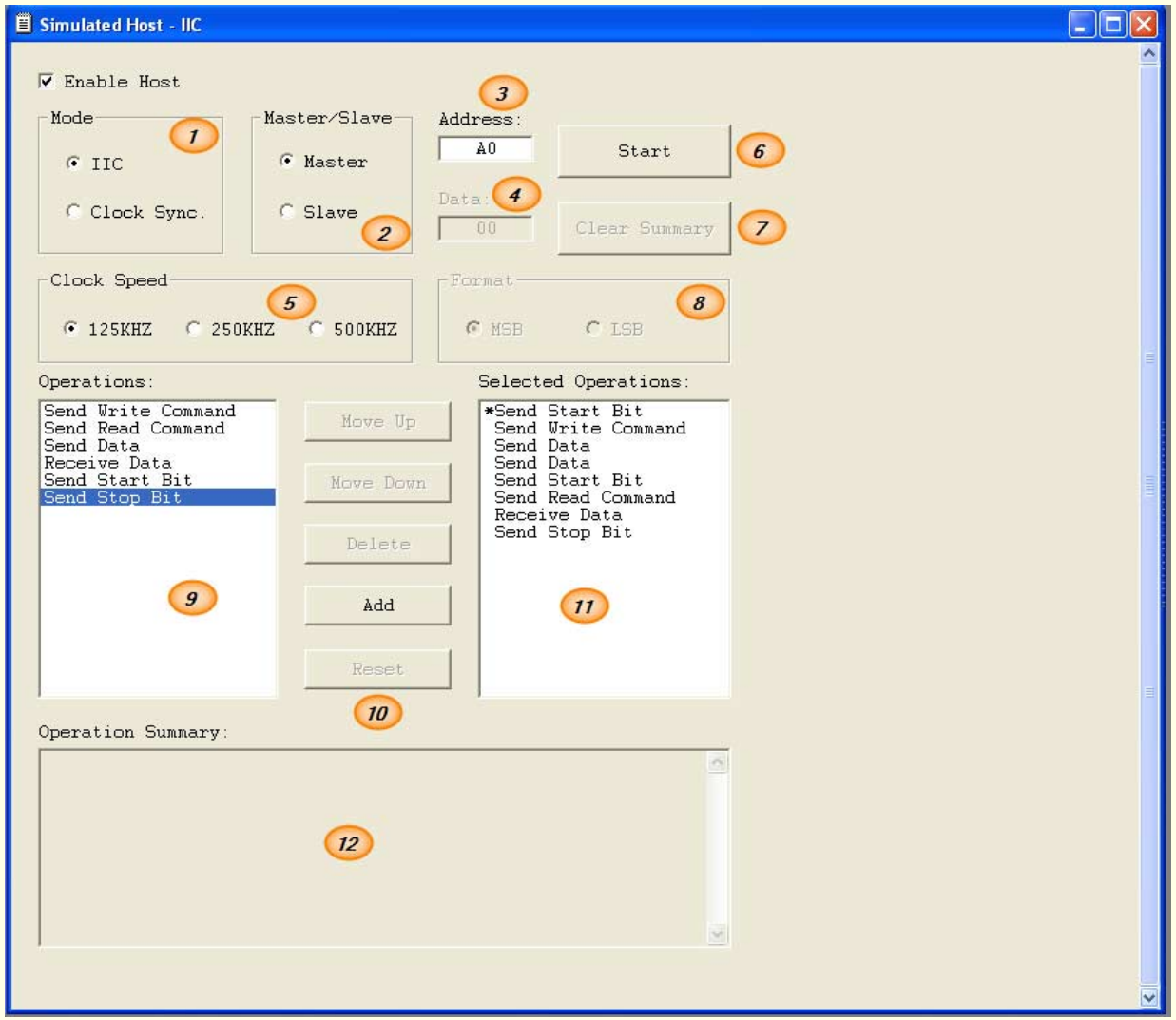


This window simulates both the channels of the serial port.

1. Receiver buffer for channel 0
2. Transmit buffer for channel 0
3. Receiver buffer for channel 1
4. Transmit buffer for channel 0
5. Select the display format for all buffers.
6. Clear both the buffers for channel 0

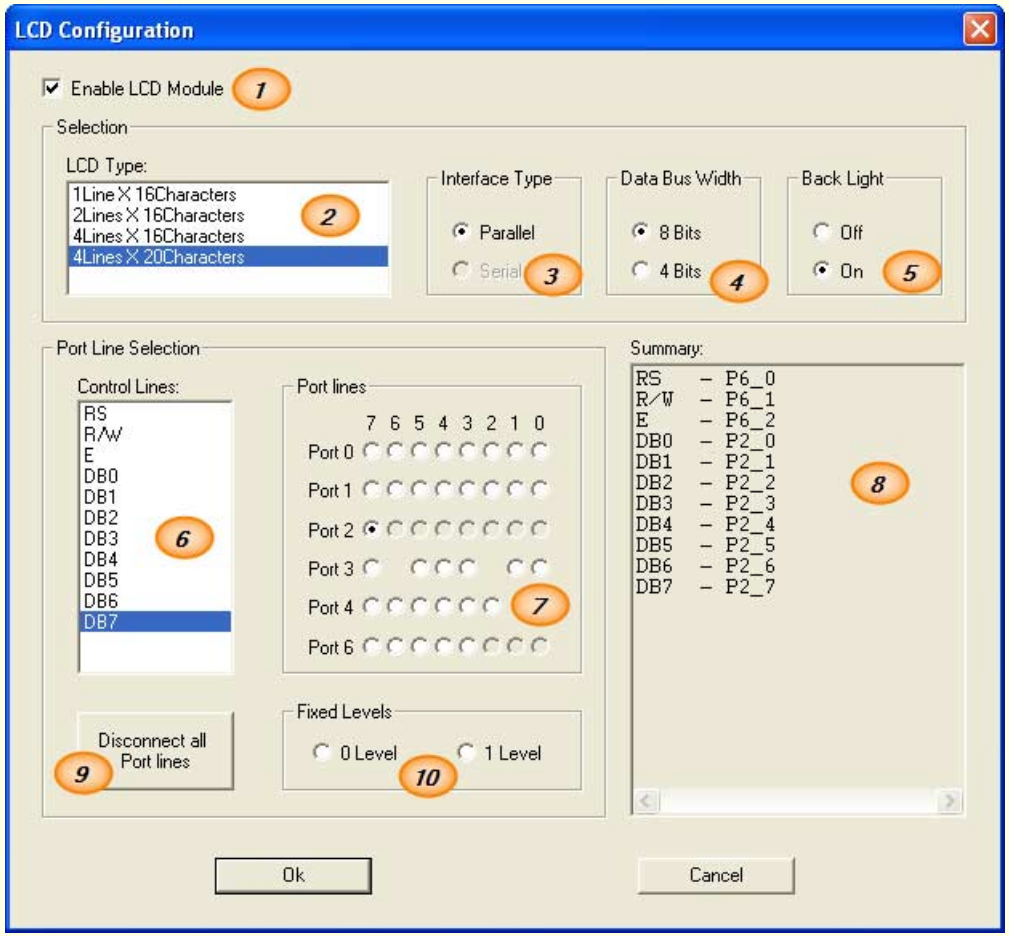
Simulated Host for the SSU

This window is the simulated host for the on-chip SSU.



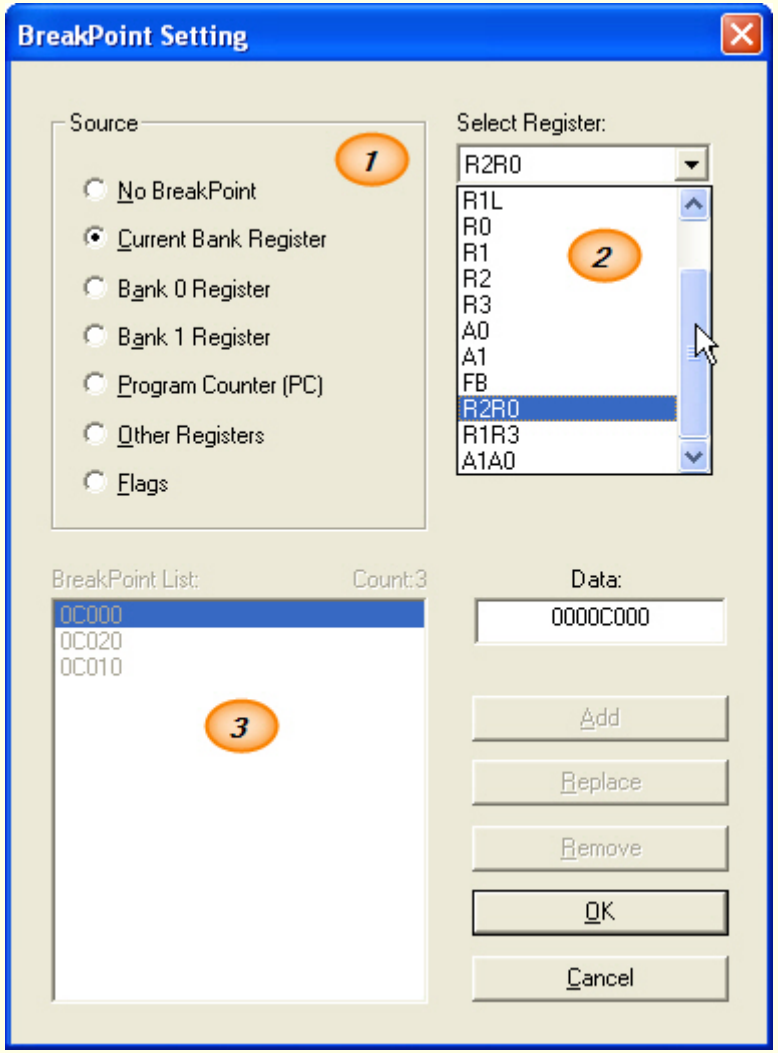
1. Enable the Simulated Host for the IIC module.
2. Select the mode: Maser or Slave.
3. Select the Slave address.
4. Enter the data to be sent.
5. Select the clock speed.
6. Click this button to start the operation.
7. Click this button to Clear the summary window.
8. Select the format – MSB/LSB
9. The available operations for the selected mode Master is listed here.
10. Use buttons Add, Delete, etc to add the operations to another list at the required sequence.
11. Displays the commands in the sequence selected by user.
12. The completed operations are listed here.

LCD Module Configuration for Simulation



1. Check the checkbox to enable the LCD module.
2. Select the LCD type.
3. Select the interface type as parallel.
4. Choose the data bus width from two options: 8bits or 4bits.
5. Switch on back light for LCD.
6. Displays the control and data lines of the LCD.
7. Allocate port lines for control and data lines of the LCD.
8. Displays the details of the port lines alloted for the LCD module.
9. Press this button to disconnect all the port lines from the LCD module.
10. Fixed level like ‘0’ or ‘1’ can be selected using this option to reduce the port line usage.

BreakPoint Setting



You can set BreakPoints in many places due to simulator's flexible design.

1 Displays the source for the breakpoint setting.

2 Displays the list of registers available for the selected source.

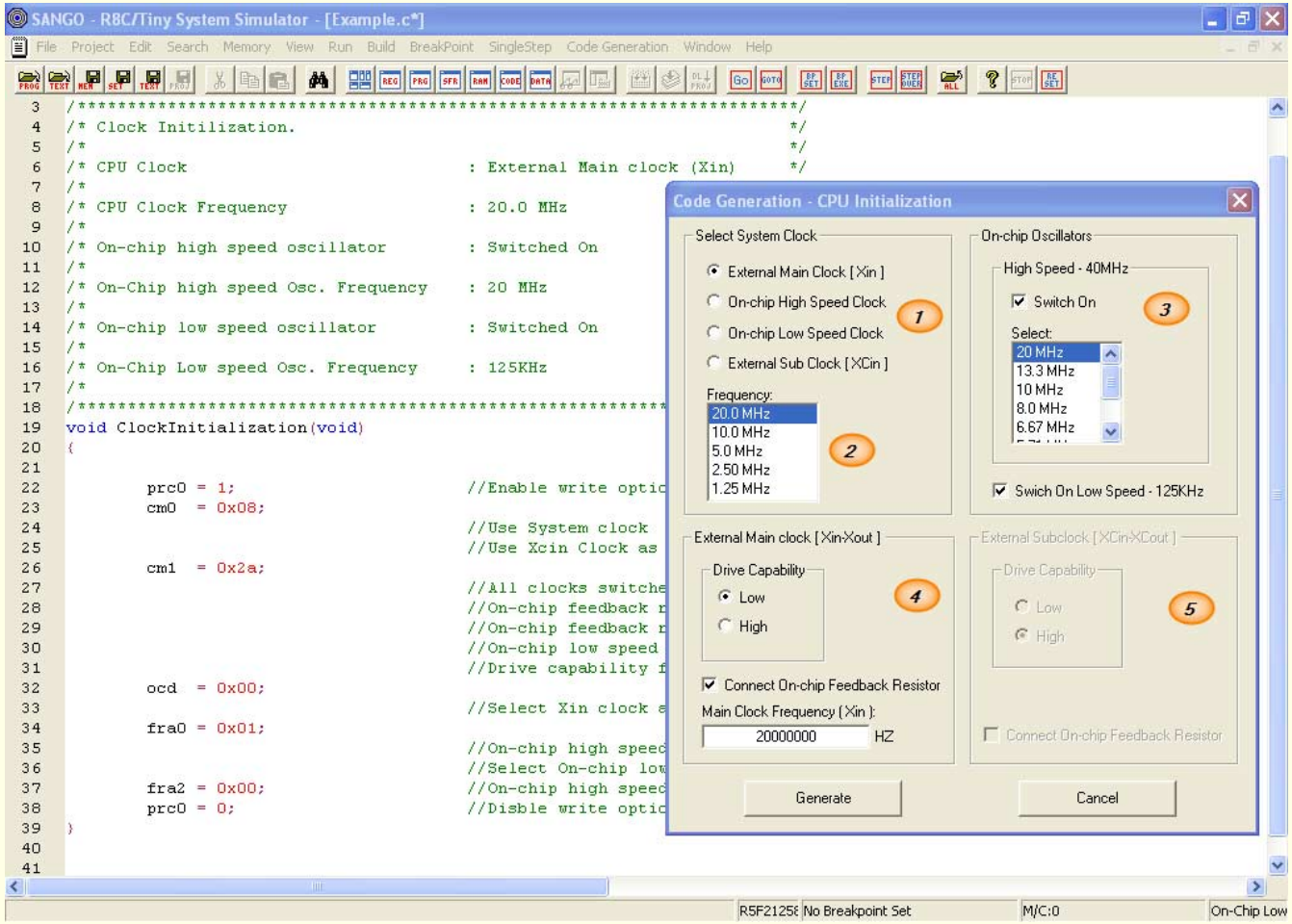
3 Displays the details of the selected BreakPoint.

For the PC, multiple breakpoints, upto 16, are possible. The PC breakpoint is indicated in the program window by a 'B' symbol

Code Generation

Code generation facility is available for all on-chip peripherals and external peripherals. Place the cursor at the point on the opened text file in the simulator.

From the Code generation menu any module is selected and after proper settings code is generated and gets pasted at the cursor point.

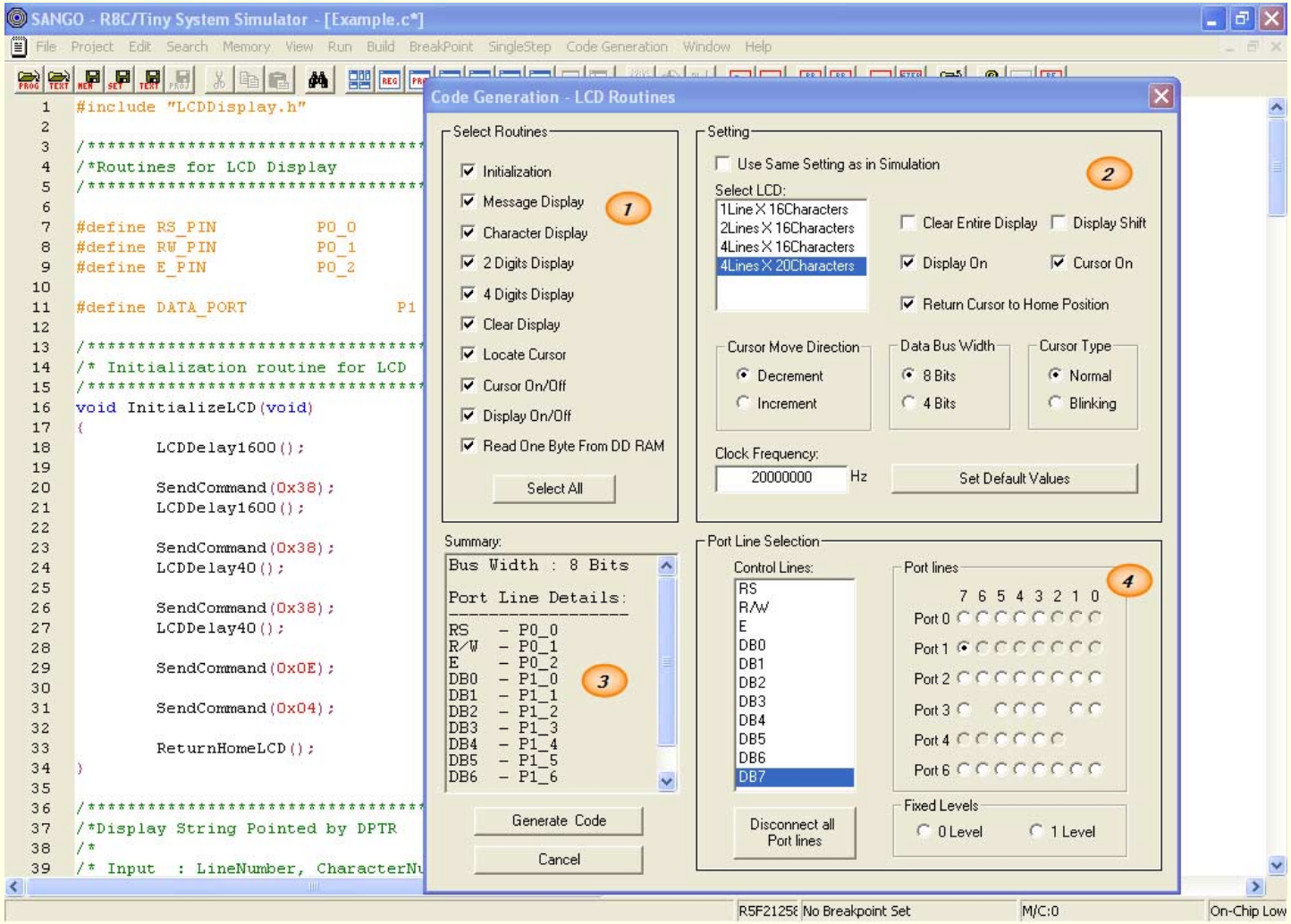


Code for CPU Initialization:

1. Select the system clock from the options available.
2. The selected clock and its divisions available are displayed in the list box. Choose the system frequency from the list.
3. Options are available for the on-chip oscillators. Select them if code is required.
4. Options for external main clock.
5. Options for external sub clock.

Click generate button to generate the code.

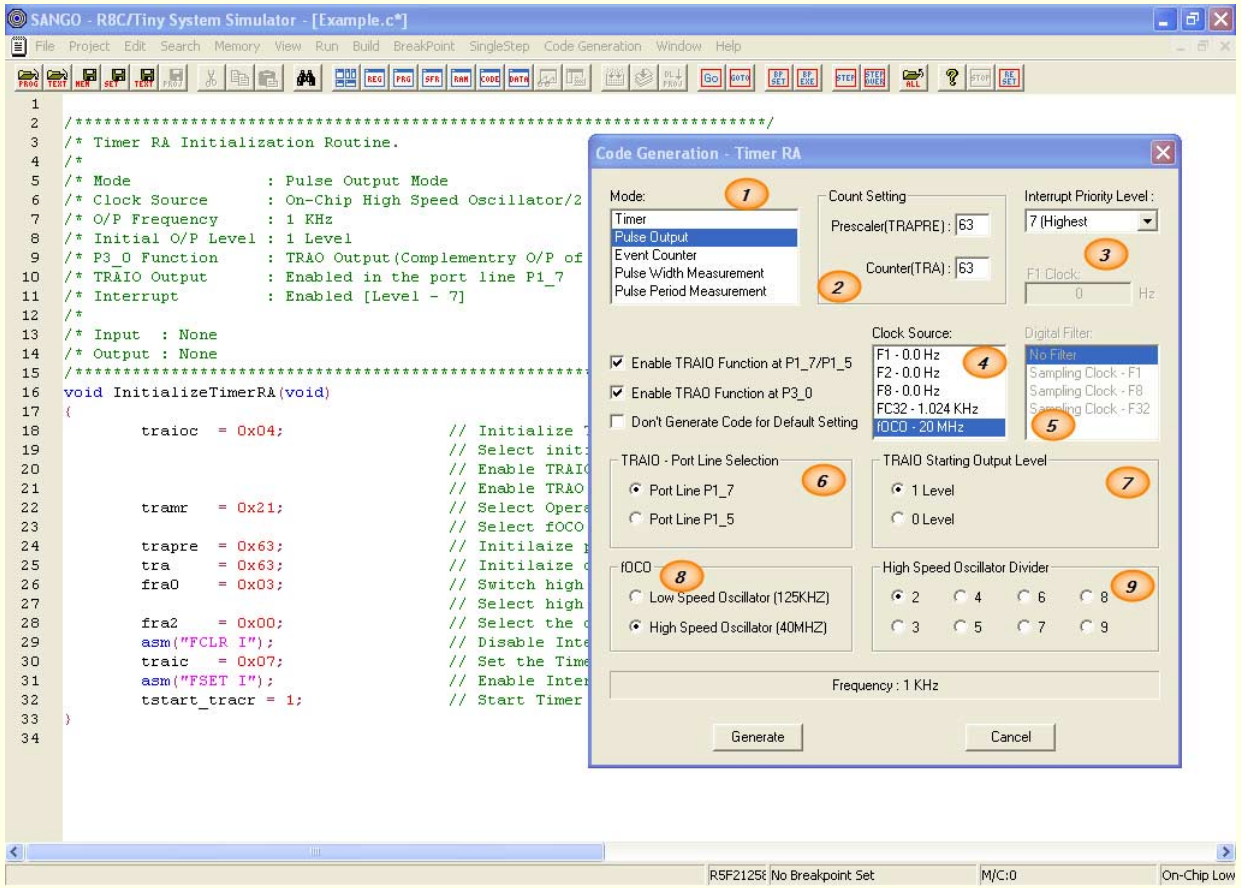
Code Generation for Two Lines by 16 Characters LCD Interface



1. Select the required routines.
2. Select the LCD type and initialization setting.
3. The control lines and selected port lines are displayed as summary.
4. Select the port lines for the control lines. Error checking also included. (ie. A single port line cannot be selected for two or more control lines)

Code Generation for Internal Peripheral - Timer RA

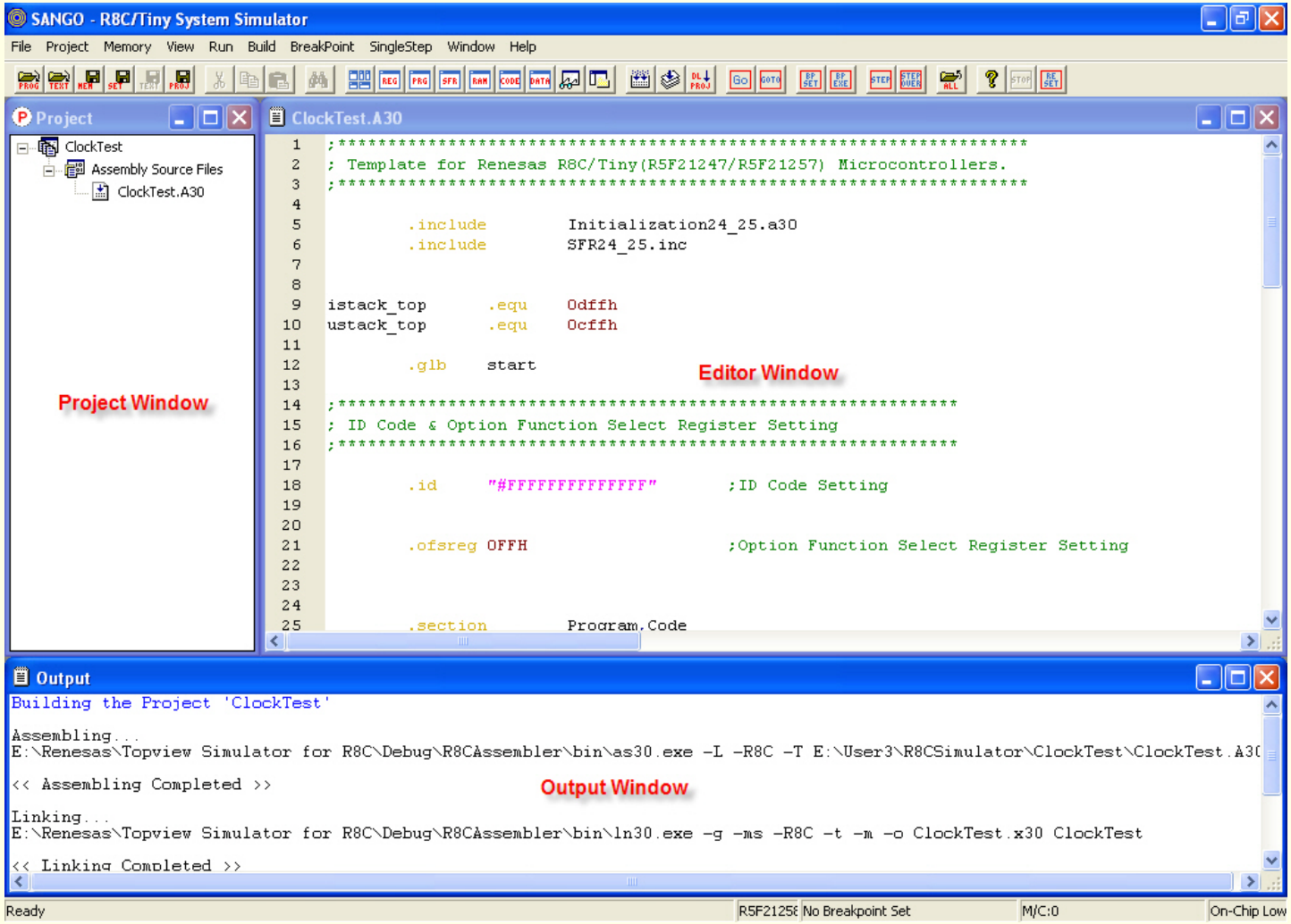
Code is generated for all the available modes of the Timer RA. For example the 'Pulse output mode" is discussed here.



1. Mode selection. Available modes are listed here.
2. Enter the prescalar and counter values. The frequency of the waveform gets displayed in a static box at the bottom. Adjust the counter values and set the required frequency.
3. Set the priority level for the timer RA overflow interrupt.
4. Select the clock source for timer.
5. Select the sampling clock for digital filter if required.
6. Select the port line for TRAIO output.
7. Select the output level for TRAIO pin.
8. Select the oscillator for fOCO clock.
9. Choose the clock divider

Click the generate button to generate the code.

Project Management



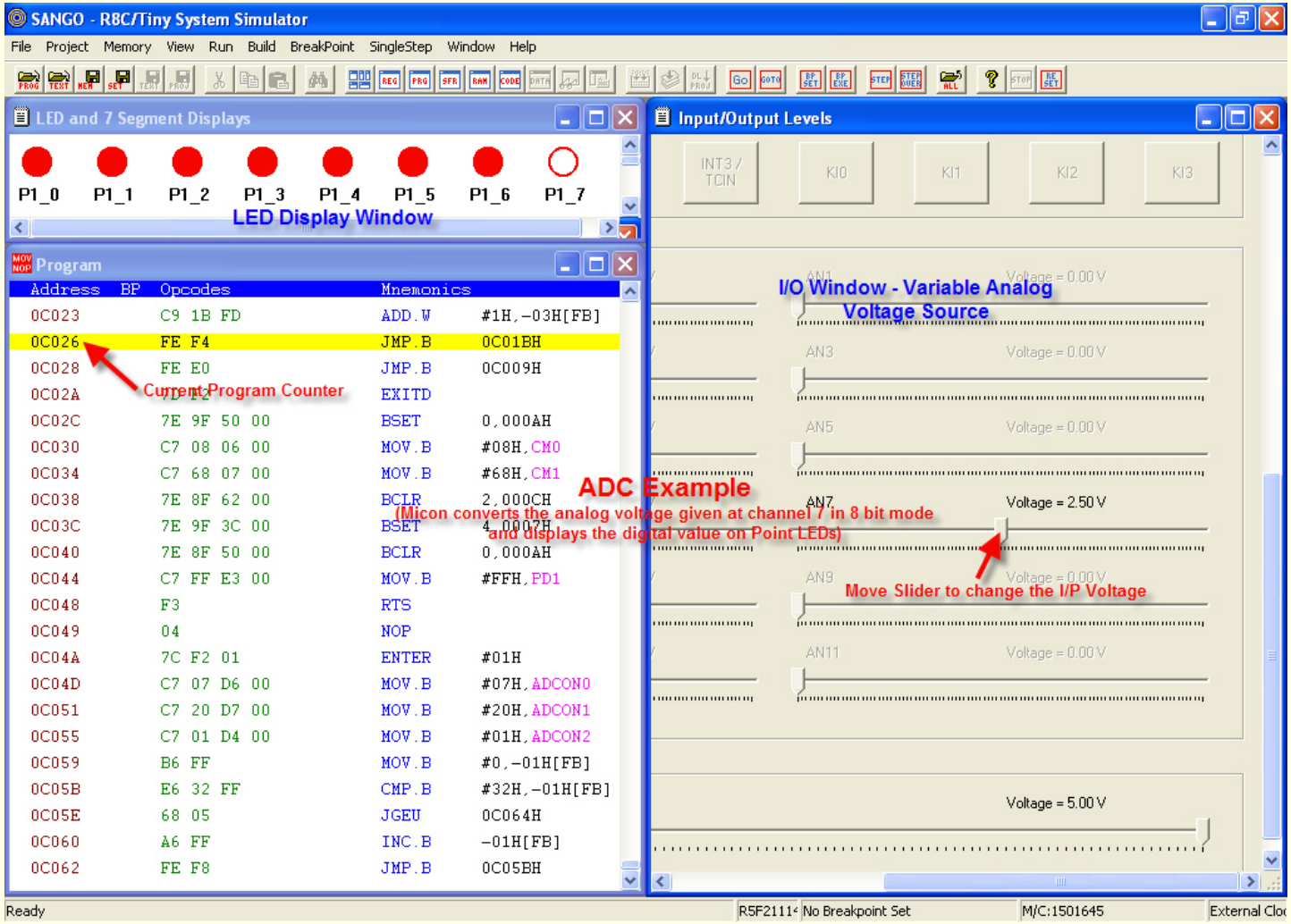
SANGO gives you facility to develop your programs right from the scratch. SANGO's built-in Text Editor takes care of program entry operations. You can also downl9oad any input program from the disk.

Program development starts by creating a new project. The project window gives the details of the files used in the project.

The editor window displays the contents of the 'C' files.

The project files can be compiled using the 'C' Compiler for R8C/Tiny. The compiler output is captured and displayed in the Output window. Use '**Dowload**' option to download the **mot** file in to the simulator for testing.

Example 1 - Study of On-chip ADC with 8 Numbers of Point LEDs

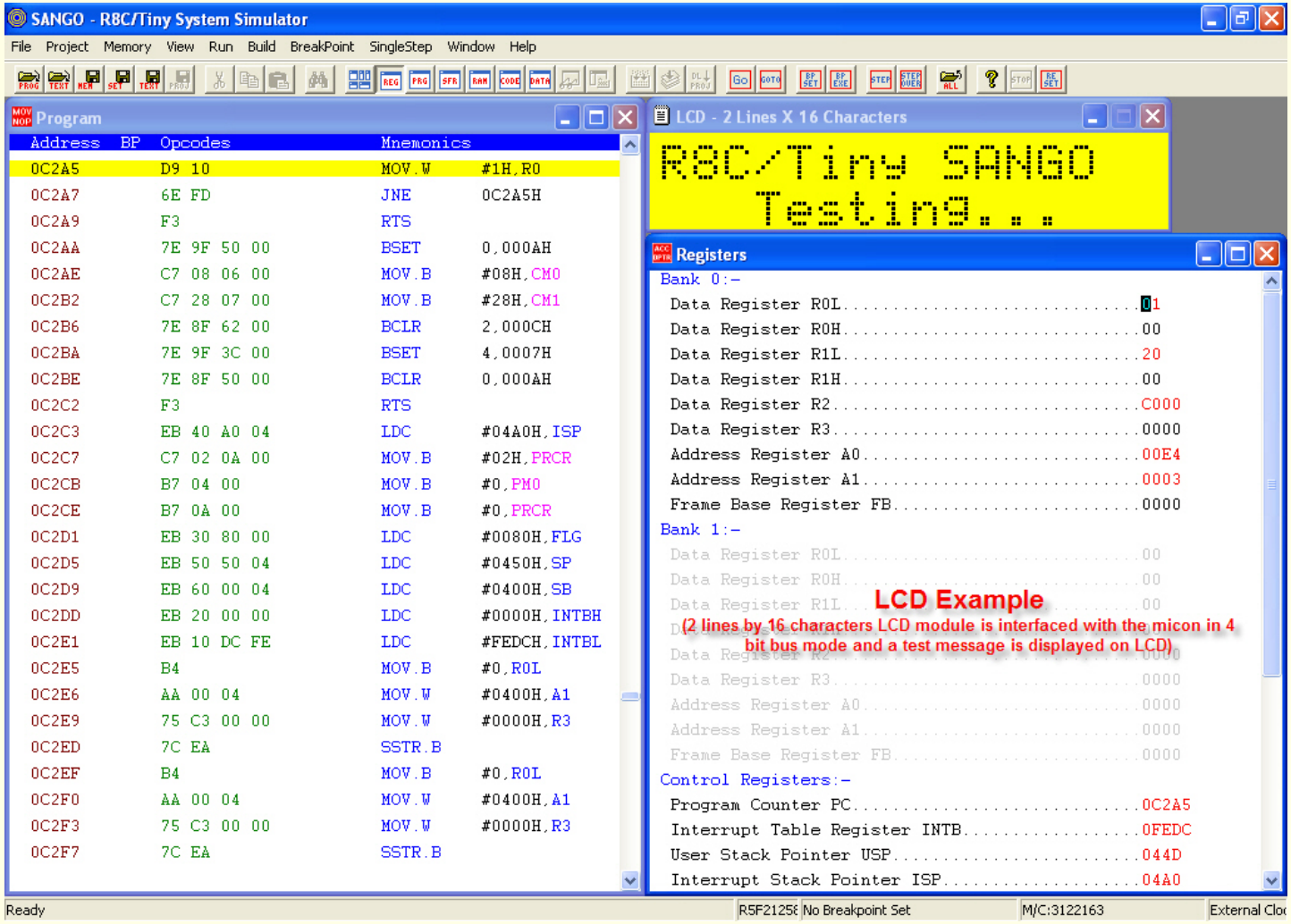


In this example the on-chip ADC is read and the digital data from ADC is displayed on eight numbers of point LEDs. The ADC is configured in 8 bit, one shot mode. The variable analog voltage can be given to the selected ADC input through the slider provided in the I/O window.

For study purpose, channel 7 of ADC is used and the point LEDs are connected to the port line P1_0 to P1_7.

In the program, the start conversion signal is given, and the converted digital data is read from ADC after getting the end of conversion signal. The read digital data is send to the port 1. This process is repeated continuously.

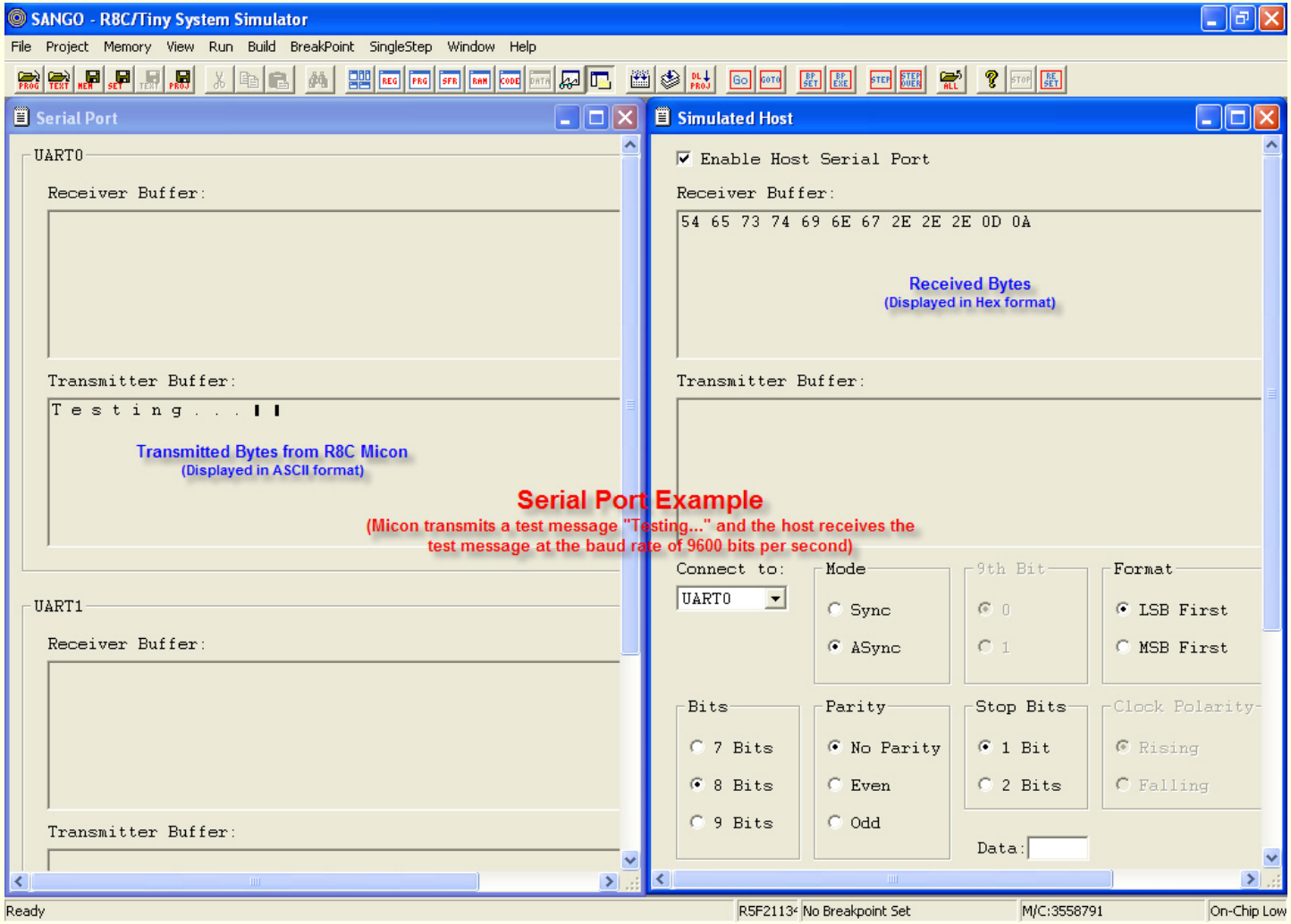
Example 2 - Two Lines by 16 Characters LCD Interface



This example deals with the interfacing of a two lines by 16 characters LCD module with micon in 4bit bus mode. Port lines P2_1 to P2_3 are connected to the control lines RS, R/W and E. The port lines P2_4 to P2_5 are used to send the data to LCD module.

In the program, the LCD module is first initialized in 2 lines, 4 bit bus mode. Then a 2 line message is displayed on LCD as shown in the figure.

Example 3 - Serial Port



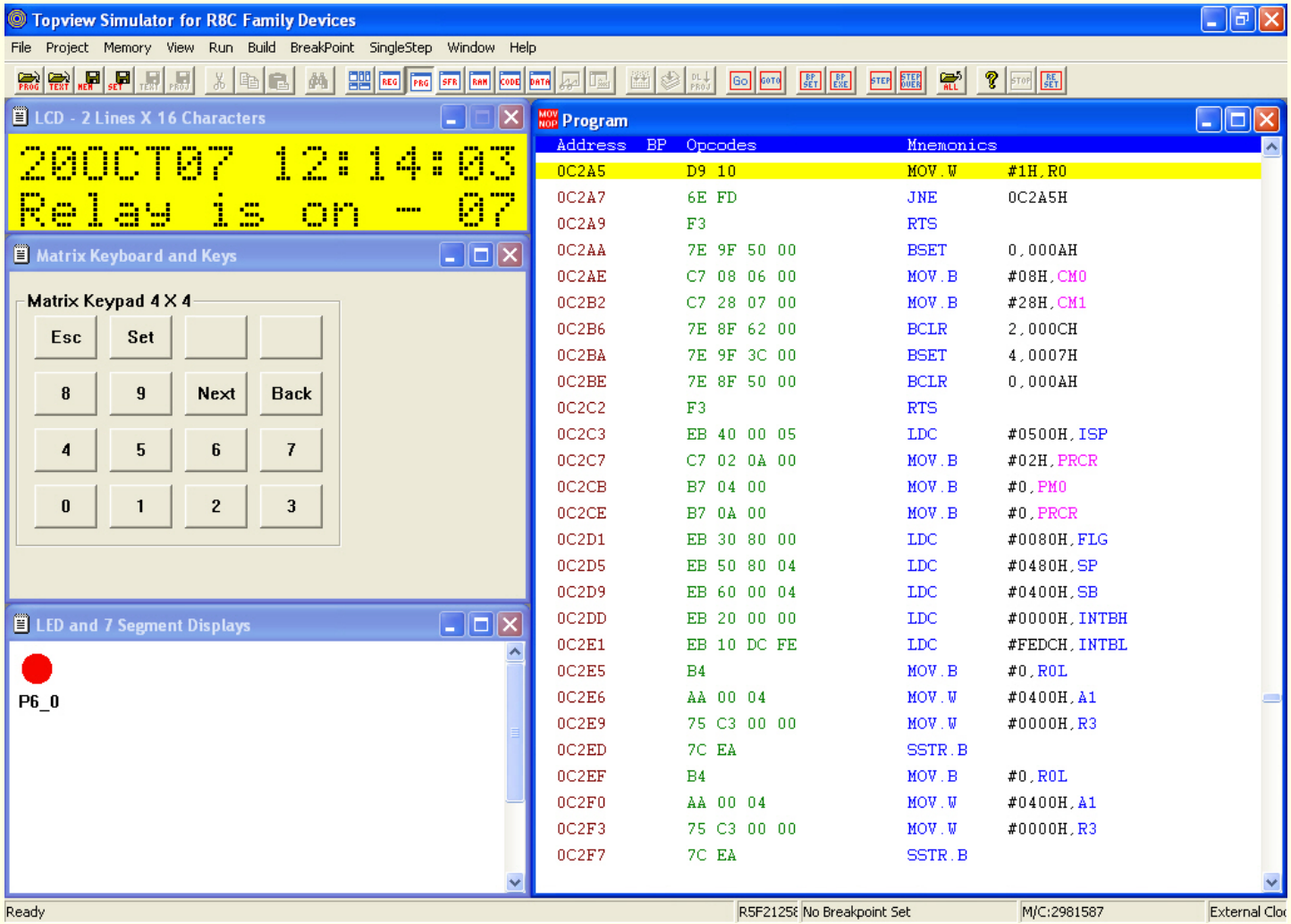
This study example gives a picture of the on-chip serial port and the simulated host serial port in asynchronous mode.

The UART0 of R8C/Tiny micon is used in this study at 9600 baud rate. In the program, the UART0 is initialized at 9600 baud and a message “Testing…” is sent to the host.

The host will receive the message and displayed the same on the screen. Data transmitted and received for both on-chip UART and the host are displayed on screen as shown.

There is an option to select the display format as either ASCII or Hex.

Project - Programmable Timer



This is an R8C/Tiny based useful timer meant for controlling a relay/solenoid as per timing schedule. It can be used in many applications where it is required to switch On/Off lights, motors and etc. at the fixed predefined timings.

The hardware is very simple. The R8C/Tiny micon controls every thing. There is an IIC RTC available to keep track of the current time. The LCM with 2 line X 16 size display combines with the keyboard to give the required interacting facility.