Z80 Simulator IDE
Getting Started

There are six examples bundled with Z80 Simulator IDE. They are located in application folder. This is short step by step guide for the beginners that will help them to test these examples and in that way explore the most important features of Z80 Simulator IDE.

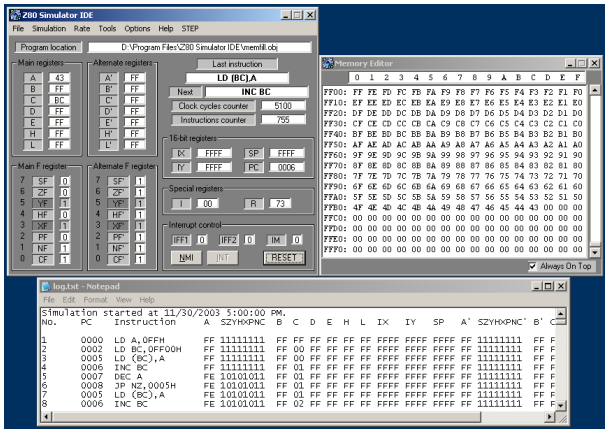
EXAMPLE 1

- Examine memfill.asm file from the application folder. This assembler routine fills the memory range FF00H - FFFFH with values FFH - 00H respectively.

```
        LD A,0FFH ;initial value in register A
        LD BC,0FF00H ;initial value in register pair BC
L1:     LD (BC),A ;load value in A to the memory location addressed by BC

        INC BC ;increment BC
        DEC A ;decrement A
        JP NZ,L1 ;loop until value in A is zero
        LD (BC),A ;load value 00H to memory location FFFFH
        HALT ;halt cpu
        .END
```

- Start Z80 Simulator IDE.
- Click on Tools\Assembler.
- Click on File\Open.
- Select memfill.asm file and click on Open. The assembler source program will be displayed in the editor.
- Click on Tools\Assemble. After the operation is completed the assembler will generate two files: memfill.lst (assembler listing with assembled opcodes) and memfill.obj (binary image of assembled routine ready to be loaded into memory). The output listing file memfill.lst will be displayed.
- Click on Tools\Load. That will load the program file memfill.obj into Z80 Simulator IDE memory.
- Close assembler window.
- Select the option Options\Enable logging.
- Select the option Options\Refresh Memory Editor.
- Click on Tools\Memory Editor. That will open the Memory Editor window. 13 bytes of the program are in focus.
- Reposition the windows on the screen to get better view, if necessary use Always On Top option on Memory Editor window.
- Using the scroll bar select the memory range FF00-FFFF.
- Check that Options\HALT Stops Simulation option is selected.
- Check that Rate\Fast simulation rate is selected. You can change the simulation rate even when the simulation is running.
- Click on Simulation\Start. The simulation will start immediately.
- Watch how the program affect the selected memory range.
- The simulation will automatically stop in about one minute.
- Log file log.txt created during program simulation and located in application folder will be displayed in Notepad.

- Screenshot: view



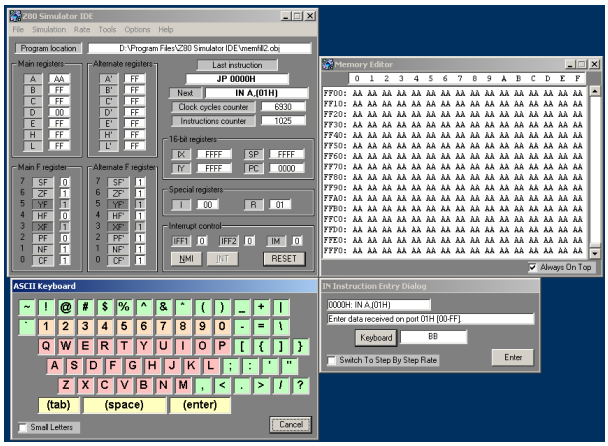
EXAMPLE 2

- Examine memfill2.asm file from the application folder. This is modified example 1. The value that will be used to fill the memory range FF00H - FFFFH is get from I/O port 01H. The routine is then repeated.

```
L2:     IN A,(01H) ;get value on port 01H to be used for memory fill
        LD D,0FFH ;initial value in counter register D
        LD BC,0FF00H ;initial value in pointer register pair BC
L1:     LD (BC),A ;load value in A to the memory location addressed by BC

        INC BC ;increment pointer BC
        DEC D ;decrement counter D
        JP NZ,L1 ;loop until value in D is zero
        LD (BC),A ;fill the last memory location FFFFH
        JP L2 ;repeat routine
        .END
```

- Start Z80 Simulator IDE.
- Click on Tools\Assembler.
- Click on File\Open.
- Select memfill2.asm file and click on Open. The assembler source program will be displayed in the editor.
- Click on Tools\Assemble. After the operation is completed the assembler will generate two files: memfill2.lst (assembler listing with assembled opcodes) and memfill2.obj (binary image of assembled routine ready to be loaded into memory). The output listing file memfill2.lst will be displayed.
- Close assembler window.
- Click on File\Load Program.
- Select memfill2.obj file and click on Open. That will load the program into Z80 Simulator IDE memory.
- Select the option Options\Refresh Memory Editor.
- Select the option Options\Prompt For Value Before IN Instruction.
- Click on Tools\Memory Editor. That will open the Memory Editor window. 17 bytes of the program are in focus.
- Reposition the windows on the screen to get better view, if necessary use Always On Top option on Memory Editor window.
- Using the scroll bar select the memory range FF00-FFFF.
- Check that Rate\Extremely Fast simulation rate is selected. You can change the simulation rate even when the simulation is running.
- Click on Simulation\Start. The simulation will start immediately.
- The simulator will display IN Instruction Entry Dialog where you should enter a value in the hex range 00-FF or click on Keyboard to easily get the ASCII code for any of the keyboard keys. Click on Enter to accept the selected value.
- Watch how the program affect the displayed memory range by loading the selected value to all those memory locations.
- The last two steps will be repeated until you stop the simulation by clicking on Simulation\Stop.

- Screenshot: view



EXAMPLE 3

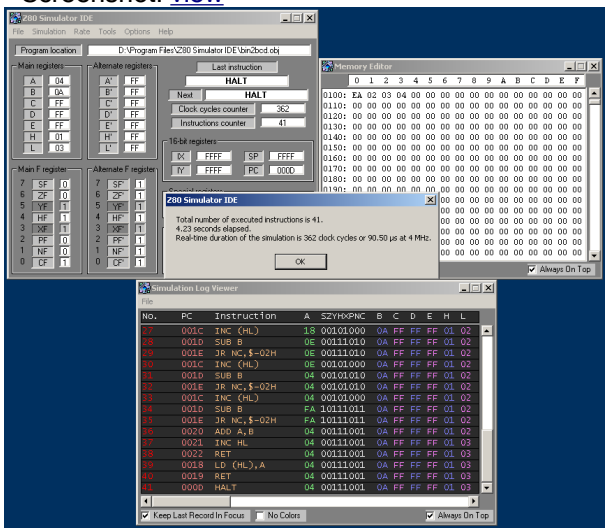
- Examine bin2bcd.asm file from the application folder. This example is a binary to BCD conversion routine. It is adapted from an illustrative program in section 11.6 of Z80 textbook 'The Z80 Microprocessor - Architecture, Interfacing, Programming, and Design' written by Prof. Ramesh S. Gaonkar - link.

```
START:  LD SP,STACK ;initialize stack pointer
        LD HL,BINBYT ;point HL index to where binary number is stored
        LD A,(HL) ;transfer byte
        LD HL,OUTBUF ;point HL index to output-buffer memory
        CALL BINBCD
        HALT

BINBCD:
        LD B,100 ;load 100 into register B (power of ten holding
register)
        CALL BCD ;call conversion for BCD3
        LD B,10 ;load 10 into register B
        CALL BCD ;call conversion for BCD2
        LD (HL),A ;store BCD1
        RET

BCD:
        LD (HL),0FFH ;load buffer with -1
STORE:  INC (HL) ;clear buffer first and increment for each subtraction
        SUB B ;subtract power of ten from binary number
        JR NC,STORE ;if number is larger than power of ten, go back and
add 1 to buffer
        ADD A,B ;if no, add power of ten to get back remainder
        INC HL ;go to next buffer location
        RET

        .ORG 0100H
BINBYT  .DB 234 ;example binary number to be converted into a BCD number
OUTBUF  ;output-buffer memory location

STACK   .EQU 0FFFFH ;definition of stack pointer initialization address
        .END
```

- Start Z80 Simulator IDE.
- Click on Tools\Assembler.
- Click on File\Open.
- Select bin2bcd.asm file and click on Open. The assembler source program will be displayed in the editor.
- Click on Tools\Assemble. After the operation is completed the assembler will generate two files: bin2bcd.lst (assembler listing with assembled opcodes) and bin2bcd.obj (binary image of assembled routine ready to be loaded into memory). The output listing file bin2bcd2.lst will be displayed.
- Close assembler window.
- Click on File\Load Program.
- Select memfill2.obj file and click on Open. That will load the program into Z80 Simulator IDE

memory.
- Select the option Options\Refresh Memory Editor.
- Click on Tools\Memory Editor. That will open the Memory Editor window.
- Reposition the windows on the screen to get better view, if necessary use Always On Top option on Memory Editor window.
- Using the scroll bar select the memory range 0100-01FF.
- Click on Tools\Simulation Log Viewer. That will open the Simulation Log Viewer window.
- Reposition the windows on the screen to get better view, if necessary use Always On Top option on Simulation Log Viewer window.
- Select the Keep Last Record In Focus option.
- Check that Rate\Fast simulation rate is selected. You can change the simulation rate even when the simulation is running.
- Check that Options\HALT Stops Simulation option is selected.
- Click on Simulation\Start. The simulation will start immediately.
- Watch how the program affect the selected memory range.
- The simulation will stop automatically when HALT instruction is reached.

- Screenshot: view



EXAMPLE 4

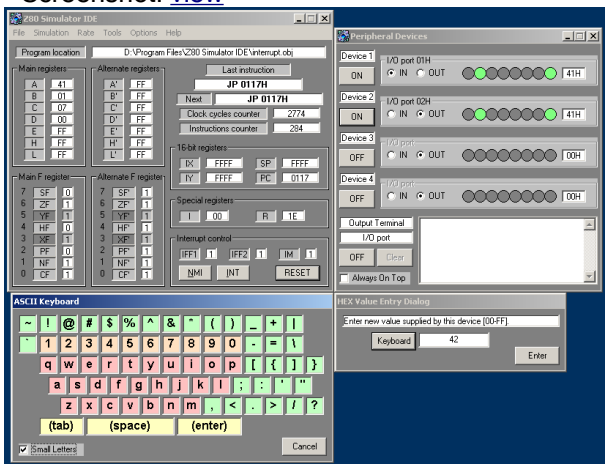
- Examine interrupt.asm file from the application folder. This example shows how peripheral devices and interrupts interfaces are used. The routine first sends five data bytes to port 02H (should be treated as the initialization of the peripheral device) and then responds to generated interrupts by echoing the value received on port 01H to port 02H.

```
        JP 0100H ;jump to main routine

        .ORG 0038H ;interrupt routine
        IN A,(01H) ;get the value from port 01H
        OUT (02H),A ;echo that value to port 02H
        EI ;enable interrupts
        RETI ;return from interrupt

        .ORG 0100H ;main rountine
        JR L1 ;jump over data area
L2:     .DB 0AH ;data byte 1
        .DB 0BH ;data byte 2
        .DB 0CH ;data byte 3
        .DB 0DH ;data byte 4
        .DB 0EH ;data byte 5
L1:     LD D,05H ;load counter register D
        LD BC,L2 ;load pointer register pair BC
L3:     LD A,(BC) ;get the data byte
        OUT (02H),A ;send it to port 02H
        INC BC ;increment pointer BC
        DEC D ;decrement counter D
        JP NZ,L3 ;loop until all data bytes are sent
```

```
        IM 1 ;set interrupt mode 1
        EI ;enable interrupts
L4:     JP L4 ;loop forever
        .END
```

- Start Z80 Simulator IDE.
- Click on Tools\Assembler.
- Click on File\Open.
- Select interrupt.asm file and click on Open. The assembler source program will be displayed in the editor.
- Click on Tools\Assemble. After the operation is completed the assembler will generate two files: interrupt.lst (assembler listing with assembled opcodes) and interrupt.obj (binary image of assembled routine ready to be loaded into memory). The output listing file interrupt.lst will be displayed.
- Close assembler window.
- Click on File\Load Program.
- Select interrupt.obj file and click on Open. That will load the program into Z80 Simulator IDE memory.
- Select the Rate\Normal simulation rate. You can change the simulation rate even when the simulation is running.
- UNSELECT the option Options\Prompt For Value Before IN Instruction.
- Select the option Options\Enable IN/OUT Instructions Logging.
- Click on Tools\Peripheral Devices. That will open the Peripheral Devices window.
- Reposition the windows on the screen to get better view, if necessary use Always On Top option on Peripheral Devices window.
- Turn on the Device 1 by clicking on its OFF button. Enter 01 for its port number.
- Select IN option button to configure it as an input device.
- Turn on the Device 2 by clicking on its OFF button. Enter 02 for its port number.
- Do not change its default configuration of an output device.
- Click on Simulation\Start. The simulation will start immediately.
- Watch the program sending 5 bytes 0A, 0B, 0C, 0D, 0E to the I/O device on port 02.
- By toggling individual bits of Device 1, prepare an arbitrary value (say AA). You can also prepare the value that will be supplied by this device by clicking on the value label next to the graphical representation. Then by clicking on Keyboard button you will be able to easily get the ASCII code for any of the keyboard keys.
- This program uses interrupts, so after the EI instruction INT button on interrupt interface is enabled.
- Click on INT button. That will send the interrupt signal to the Z80 Simulator, and the execution of the interrupt routine will start.
- The interrupt routine reads the byte supplied by the Device 1 and echoes it to the Device 2.
- The interrupt generation can be repeated.
- Click on Simulation\Stop to stop the simulation.
- Examine I/O log file io.txt from the application folder.

- Screenshot: view



EXAMPLE 5

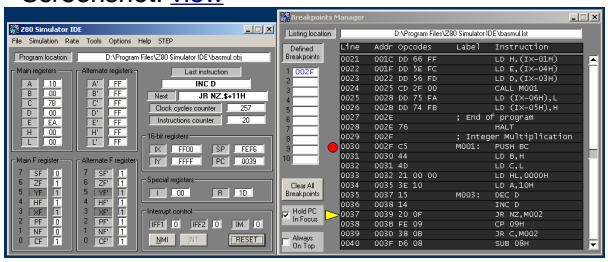
- Examine basmul.bas file from the application folder. This is Basic source program that will be compiled using integrated Basic compiler. It is an integer multiply routine that will multiply two integer numbers 123 and 234.

```
Dim a As Integer
Dim b As Integer
Dim c As Integer

a = 123 'First number
b = 234 'Second number
c = a * b
```

- Start Z80 Simulator IDE.
- Click on Tools\BASIC Compiler.
- Click on File\Open.
- Select basmul.bas file and click on Open. The basic source program will be displayed in the editor.

- Click on Tools\Compile. The compiler will generate basmul.asm file with assembler source.
- Close BASIC Compiler window.
- Click on Tools\Assembler.
- Click on File\Open.
- Select basmul.asm file and click on Open. The assembler source program will be displayed in the editor.
- Click on Tools\Assemble. After the operation is completed the assembler will generate two files: basmul.lst (assembler listing with assembled opcodes) and basmul.obj (binary image of assembled routine ready to be loaded into memory). The output listing file basmul.lst will be displayed.
- Click on Tools\Load. That will load the program file basmul.obj into Z80 Simulator IDE memory.
- Close assembler window.
- Check that Options\HALT Stops Simulation option is selected.
- Select the option Options\Refresh Breakpoints Manager.
- Click on Tools\Breakpoints Manager. That will open the Breakpoints Manager window.
- Reposition the windows on the screen to get better view, if necessary use Always On Top option on Breakpoints Manager window.
- Click on the line corresponding to 002F address to define the breakpoint on this instruction. That is the beginning of integer multiplication routine.
- Select the Hold PC In Focus option.
- Select the Rate\Fast simulation rate.
- Click on Simulation\Start. The simulation will start immediately.
- When the Z80 Simulator IDE reach the breakpoint it will automatically switch to Step By Step simulation rate.
- Click on Rate\Fast to continue with simulation.
- Watch the program execution on the Breakpoints Manager.
- The simulation will stop when the HALT instruction is reached. The result of multiplication 28782 (706EH) will reside in HL register pair and also in memory locations assigned to variable 'c'.

- Screenshot: view



EXAMPLE 6

- Examine basprint.bas file from the application folder. This is Basic source program that will be compiled using integrated Basic compiler. It is a single precision floating point multiply routine that will multiply two real numbers 9.876543 and 2.345678 and then send the formatted text showing the result of the operation to the I/O port number 1.

```
Dim a As Single
```

```
Dim b As Single
Dim c As Single

a = 9.876543 'First number
b = 2.345678 'Second number
c = a * b

Print 1, "Number ", a, CrLf
Print 1, "multiplied by ", b, CrLf
Print 1, "equals ", c, "!", CrLf
```

- Start Z80 Simulator IDE.
- Click on Tools\BASIC Compiler.
- Click on File\Open.
- Select basprint.bas file and click on Open. The basic source program will be displayed in the editor.
- Click on Tools\Compile. The compiler will generate basprint.asm file with assembler source.
- Close BASIC Compiler window.
- Click on Tools\Assembler.
- Click on File\Open.
- Select basprint.asm file and click on Open. The assembler source program will be displayed in the editor.
- Click on Tools\Assemble. After the operation is completed the assembler will generate two files: basprint.lst and basprint.obj. The output listing file basprint.lst will be displayed.
- Click on Tools\Load. That will load the program file basprint.obj into Z80 Simulator IDE memory.
- Close assembler window.
- Click on Tools\Peripheral Devices. That will open the Peripheral Devices window.
- Reposition the windows on the screen to get better view, if necessary use Always On Top option on Peripheral Devices window.
- Turn on the Output Terminal by clicking on its OFF button. Enter 1 for its I/O port number.
- Check that Options\HALT Stops Simulation option is selected.
- Select the Rate\Extremely Fast simulation rate.
- Click on Simulation\Start. The simulation will start immediately.
- PRINT routines are slow and it will take some time until the HALT instruction is reached. The program will display formatted text showing the result of the operation on the Output Terminal.

- Screenshot: view