A Nios II Simulator

Learning Goal: Getting familiar with Nios2Sim simulator.

Requirements: Nios2Sim Simulator.

1 Introduction

This document serves to introduce you to **Nios2Sim**, a small and handy Nios II simulator. This tool displays the memory and register contents at each step of the execution of an assembly program.

Table 1 lists the conventional functions and names[1] of the 32 registers of the Nios **Register File**.

Table 1: Registers of the **Register File**.

Register	Name	Function	Register	Name	Function
r0	zero	0x00000000	r16	s0	Saved Register
r1	at	Assembler Temporary	r17	s1	Saved Register
r2	v0	Return Value	r18	s2	Saved Register
r3	v1	Return Value	r19	s3	Saved Register
r4	a0	Register Arguments	r20	s4	Saved Register
r5	a1	Register Arguments	r21	s5	Saved Register
r6	a2	Register Arguments	r22	s6	Saved Register
r7	a3	Register Arguments	r23	s7	Saved Register
r8	t0	Temporary Register	r24	et	Exception Temporary
r9	t1	Temporary Register	r25	bt	Breakpoint Temporary
r10	t2	Temporary Register	r26	gp	Global Pointer
r11	t3	Temporary Register	r27	sp	Stack Pointer
r12	t4	Temporary Register	r28	fp	Frame Pointer
r13	t5	Temporary Register	r29	ea	Exception Return Address
r14	t6	Temporary Register	r30	ba	Breakpoint Return Address
r15	t7	Temporary Register	r31	ra	Return Address

[1] To improve the readability of the code, we extend the official Nios II registers naming: we added names to the registers r2 to r23, which were unnamed. These names are only supported by the **Nios2Sim** simulator.

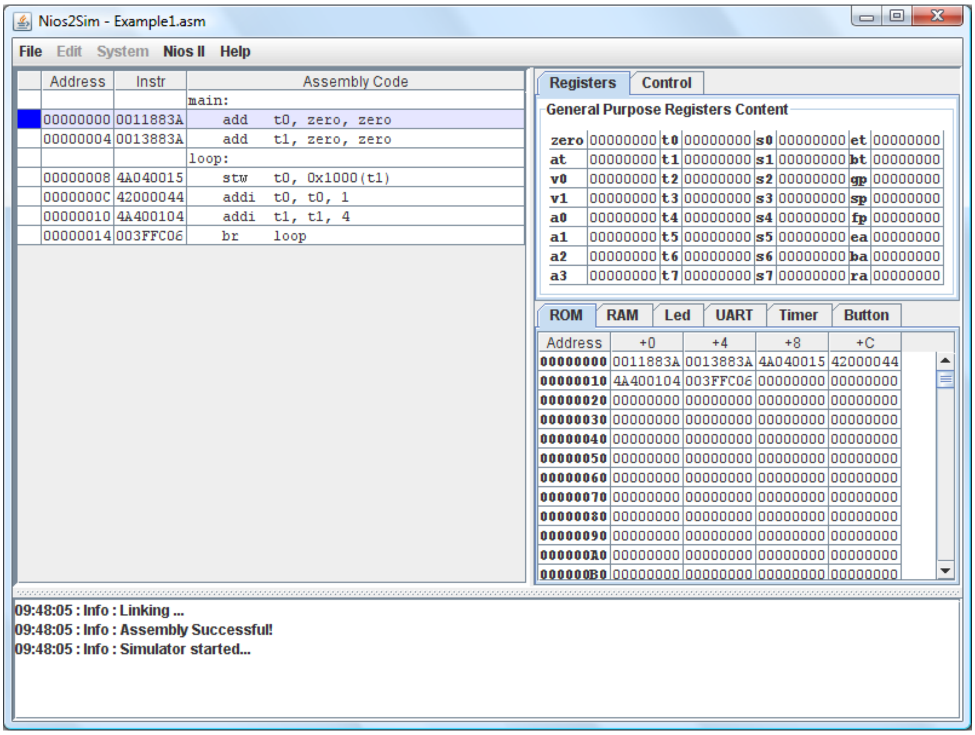
2 The Nios 2 Simulator

Let us now see how to simulate an assembly program step by step with the **Nios2Sim** simulator.

- Download the **Nios2Sim** simulator from the web page of the course.
- Execute the **Nios2Sim** simulator by double clicking on the `.jar` file. Note that you need a Java Runtime Environment (JRE) for that.
- Copy the following code example to the **Simulator**.

```
main:
    add     t0, zero, zero
    add     t1, zero, zero
loop:
    stw     t0, 0x1000(t1)
    addi    t0, t0, 1
    addi    t1, t1, 4
    br      loop
```

- Start the assembly by selecting Nios II > Assemble (Ctrl+1). If the assembly succeeded, you should see the message "**Assembly Successful!**".
- Select Nios II > Start Simulation (Ctrl+2) to start the simulation, which loads the simulation layout shown on Figure 1.



Address	Instr	Assembly Code
		main:
00000000	0011883A	add t0, zero, zero
00000004	0013883A	add t1, zero, zero
		loop:
00000008	4A040015	stw t0, 0x1000(t1)
0000000C	42000044	addi t0, t0, 1
00000010	4A400104	addi t1, t1, 4
00000014	003FFC06	br loop

zero	00000000	t0	00000000	s0	00000000	et	00000000
at	00000000	t1	00000000	s1	00000000	bt	00000000
v0	00000000	t2	00000000	s2	00000000	gp	00000000
v1	00000000	t3	00000000	s3	00000000	sp	00000000
a0	00000000	t4	00000000	s4	00000000	fp	00000000
a1	00000000	t5	00000000	s5	00000000	ea	00000000
a2	00000000	t6	00000000	s6	00000000	ba	00000000
a3	00000000	t7	00000000	s7	00000000	ra	00000000

Address	+0	+4	+8	+C
00000000	0011883A	0013883A	4A040015	42000044
00000010	4A400104	003FFC06	00000000	00000000
00000020	00000000	00000000	00000000	00000000
00000030	00000000	00000000	00000000	00000000
00000040	00000000	00000000	00000000	00000000
00000050	00000000	00000000	00000000	00000000
00000060	00000000	00000000	00000000	00000000
00000070	00000000	00000000	00000000	00000000
00000080	00000000	00000000	00000000	00000000
00000090	00000000	00000000	00000000	00000000
000000A0	00000000	00000000	00000000	00000000
000000B0	00000000	00000000	00000000	00000000

```
09:48:05 : Info : Linking ...
09:48:05 : Info : Assembly Successful!
09:48:05 : Info : Simulator started...
```

Figure 1: The simulation layout of the **Nios2Sim** simulator.

  - The table on the left lists for each line of the assembly code its corresponding address and **Nios II** instruction word. The blue line is the next instruction that the simulator will execute (i.e., the current PC address).

- The upper-right table lists the registers of the **Register File**. To edit a register value, double-click on it and enter a number. The values are represented in hexadecimal. Letting your mouse over a value displays a pop up box with the binary and decimal representations.
- The bottom-right table displays the memory content. The memory is organized in the same way as for the **Memories** and **Multicycle Nios II Processor** labs. You will find the LEDs module and the other peripherals, as if you were programming the board itself! Each memory module and peripheral has its own tab. Similarly to the registers, you can modify a value by double-clicking on it and get the different representations of this value by letting your mouse over it.

- You can execute the next step of the program by selecting Nios II > Execute a Step (Ctrl+E). Every change in the registers or in the memory during this step will be highlighted in red.
  - Execute some steps and observe the evolution of the memory and register contents.
- By selecting Nios II > Run (Ctrl+R), the simulator executes the program until it reaches a breakpoint or the maximal number of execution steps (500). Every change that occurs in the registers or in the memory during this interval will be highlighted in red.
  - Place a breakpoint inside the loop (at address `0x0008` for example). To do this, double-click on the desired line while in the simulation view.
  - Run the execution several times and observe the evolution of the memory and registers content.
- To terminate the simulation and return to the edition of the assembly code, select Nios II > End Simulation.