EXAMPLE 1

Instruction Memory's file: 5th_fig460_MemEx1/inst.rom

This Mips pipeline implementation can handle data hazards and cannot handle branchs.

The user should set initial register values (linear). No data values are required.

Description: A simple sequence of five "add" instructions with no hazards.

ADD r1,r1,r2
ADD r3,r0,r2
ADD r4,r0,r2
ADD r5,r0,r2
ADD r6,r0,r2

ADD r1,r1,r2 – type R instruction

opcode = 0	rs = 1	rt = 2	rd = 1	sh = 0	func = 32
000000	00001	00010	00001	00000	100000

0x00220820

ADD r3,r0,r2 – type R instruction

opcode = 0	rs = 0	rt = 2	rd = 3	sh = 0	func = 32
000000	00000	00010	00011	00000	100000

0x00021820

ADD r4,r0,r2 – type R instruction

opcode = 0	rs = 0	rt = 2	rd = 4	sh = 0	func = 32
000000	00000	00010	00100	00000	100000

0x00022020

ADD r5,r0,r2 – type R instruction

opcode = 0	rs = 0	rt = 2	rd = 5	sh = 0	func = 32
000000	00000	00010	00101	00000	100000

0x00022820

ADD r6,r0,r2 – type R instruction

opcode = 0	rs = 0	rt = 2	rd = 6	sh = 0	func = 32
000000	00000	00010	00110	00000	100000

0x00023020

The hexadecimal code example is:

ADD r1,r1,r2 – 0x00220820
ADD r3,r0,r2 – 0x00021820
ADD r4,r0,r2 – 0x00022020
ADD r5,r0,r2 – 0x00022820
ADD r6,r0,r2 – 0x00023020

Calculations check (with linear initial register values):

```
ADD r1,r1,r2 – r1 = 3
ADD r3,r0,r2 – r3 = 2
ADD r4,r0,r2 – r4 = 2
ADD r5,r0,r2 – r5 = 2
ADD r6,r0,r2 – r6 = 2
```