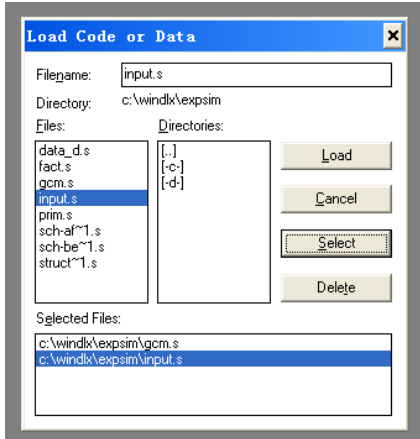
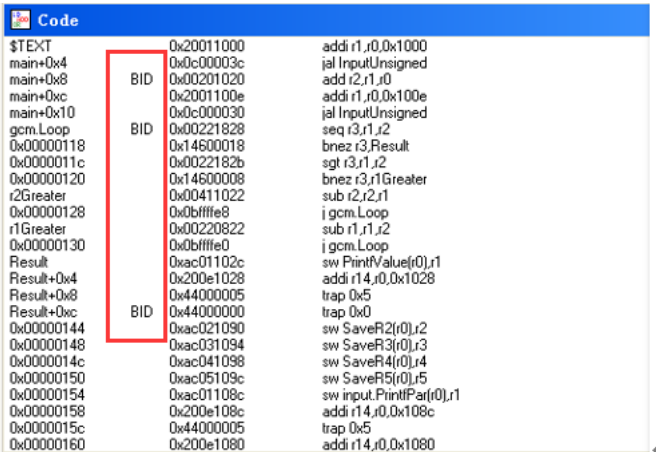
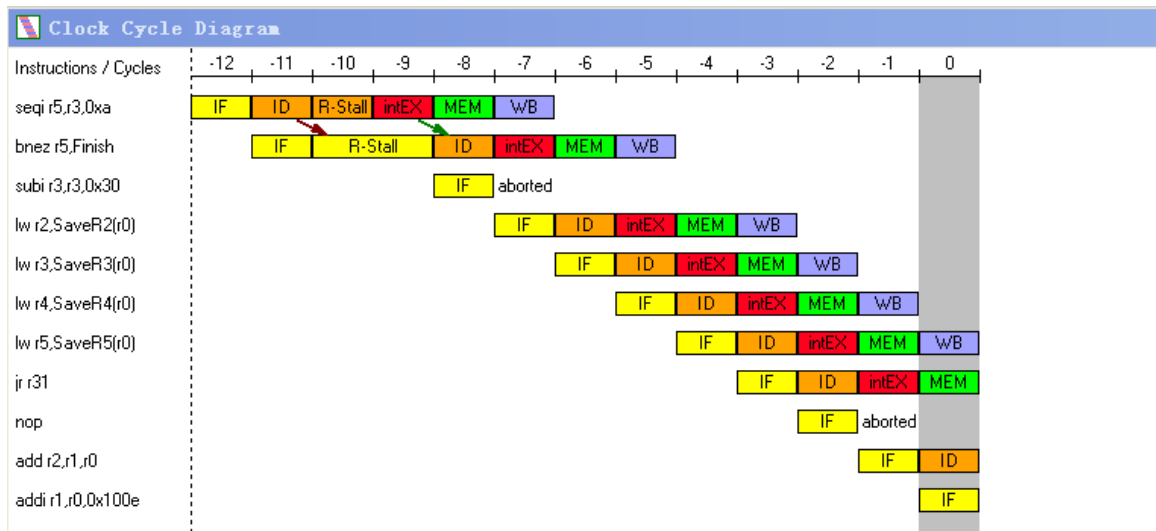
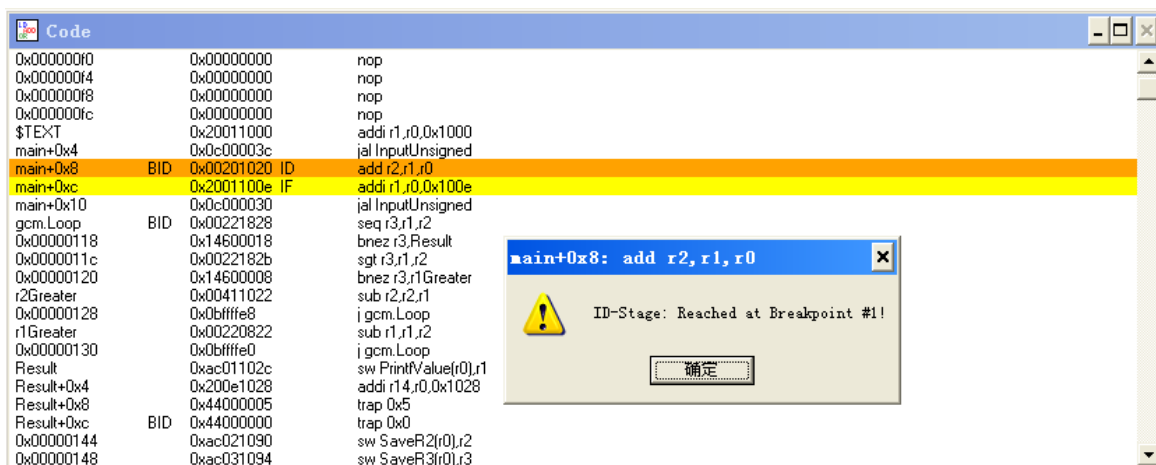
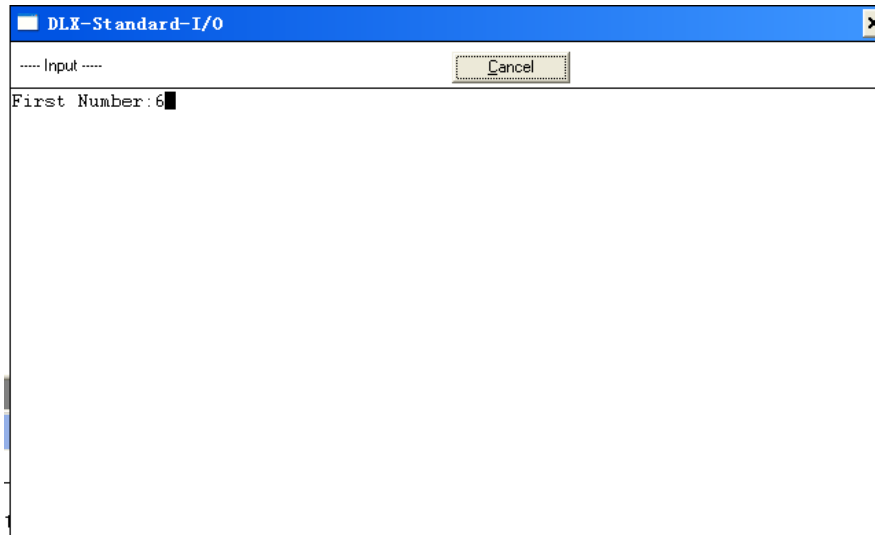


山东大学计算机科学与技术学院

计算机体系结构课程实验报告

学号：201800130031	姓名：来苑	班级：计科 1 班
实验题目：实验二 用 WinDLX 模拟器执行程序求最大公约数		
实验学时：2 学时	实验日期：2021. 5. 7	
实验目的： 通过本实验，熟练掌握 WinDLX 模拟器的操作和使用，清楚 WinDLX 五段流水线在执行具体程序时的流水情况，熟悉 DLX 指令集结构及其特点。		
硬件环境： 机房		
软件环境： Windows xp		
实验步骤与内容： 1. 配置 WinDLX，并将程序 gcm.s 和 input.s 装载如 WinDLX。 装载程序时要注意顺序。		
		
2. 设置断点 分别在 main+0x8(add r2,r1,r0)、gcm.loop(seq r3,r1,r2)和 result+0xc(trap 0x0)设置断点。		
		
3. 运行程序（第一组数 6 和 3）		

首先按下 F5，使程序运行到第一个断点处，即完成键入第一个数。



发现 Clock Cycle Diagram 窗口出现了红色和绿色箭头。

红色箭头是因为 seqi r5,r3,0xa 与指令 bnez r5,Finish 发生了冲突，所以 bnez 指令需要等待 seqi 指令完成执行，即完成 intEX 阶段才能通过定向得到 r5 寄存器的值，所以此时 bnez 指令发生了 R-Stall 暂停，直到 seqi 指令结束 intEX 阶段。而绿色箭头是因为这里使用了定向技术。

在第三条指令处出现了“aborted”。这是因为上一条指令 bnez 发生了跳转。所以该条指令暂停，留下了气泡。

同时，根据子程序 input.s，可以知道键入的数被保存在 R1 寄存器中，查看 Register 窗口验证。

Register									
PC=	0x00000110	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0
IMAR=	0x0000010c	R8=	0x00000000	R31=	0x00000108	F22=	0	D26=	0
IR=	0x2001100e	R9=	0x00000000	F0=	0	F23=	0	D28=	0
A=	0x00000006	R10=	0x00000000	F1=	0	F24=	0	D30=	0
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0		
B=	0x00000000	R12=	0x00000000	F3=	0	F26=	0		
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0		
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0		
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0		
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0		
FPSR=	0x00000000	R17=	0x00000000	F8=	0	F31=	0		
DMAR=	0x00000000	R18=	0x00000000	F9=	0	D0=	0		
SDR=	0x00000000	R19=	0x00000000	F10=	0	D2=	0		
SDRHI=	0x00000000	R20=	0x00000000	F11=	0	D4=	0		
LDR=	0x00000000	R21=	0x00000000	F12=	0	D6=	0		
LDRHI=	0x00000000	R22=	0x00000000	F13=	0	D8=	0		
R0=	0x00000000	R23=	0x00000000	F14=	0	D10=	0		
R1=	0x00000006	R24=	0x00000000	F15=	0	D12=	0		
R2=	0x00000000	R25=	0x00000000	F16=	0	D14=	0		
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0		
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0		
R5=	0x00000000	R28=	0x00000000	F19=	0	D20=	0		
R6=	0x00000000	R29=	0x00000000	F20=	0	D22=	0		

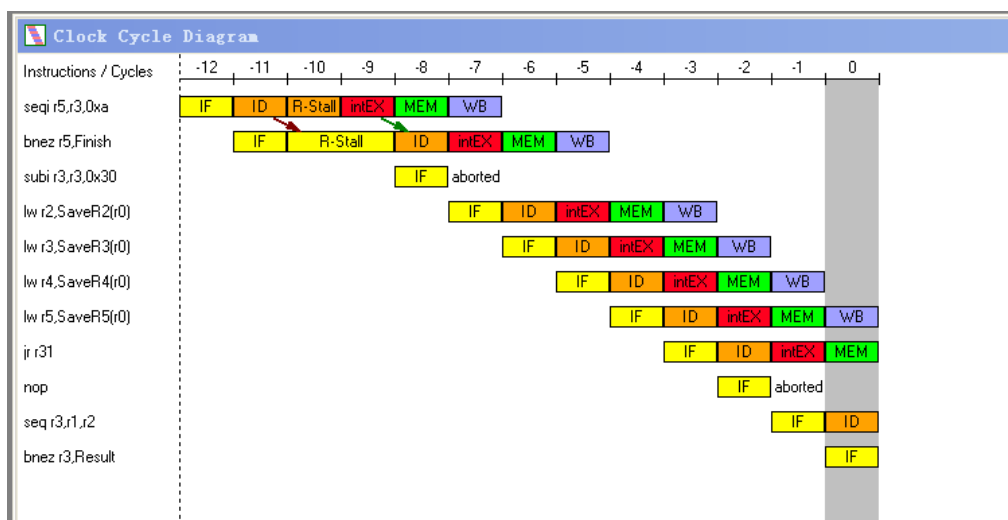
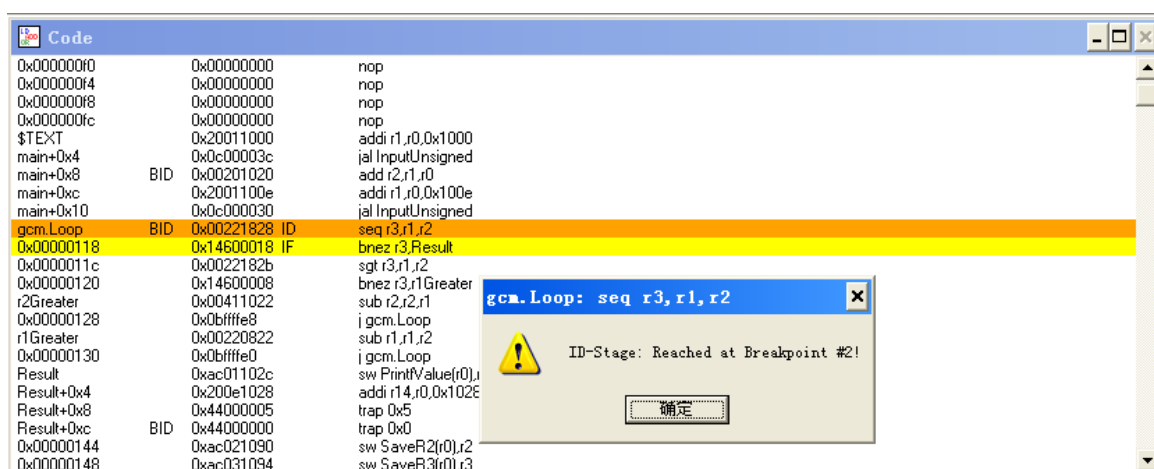
继续单步执行程序。

Code		
0x00000000	0x00000000	nop
0x00000004	0x00000000	nop
0x00000008	0x00000000	nop
0x0000000c	0x00000000	nop
\$TEXT	0x20011000	addi r1,r0,0x1000
main+0x4	0x0c00003c	jal InputUnsigned
main+0x8	BID 0x00201020 w8	add r2,r1,r0
main+0xc	0x2001100e MEM	addi r1,r0,0x100e
main+0x10	0x0c000030 riEX	jal InputUnsigned
gcm.Loop	BID 0x00221828	seq r3,r1,r2
0x00000118	0x14600018	bnez r3,Result
0x0000011c	0x0022182b	sgt r3,r1,r2
0x00000120	0x14600008	bnez r3,r1Greater
r2Greater	0x00411022	sub r2,r2,r1
0x00000128	0x0bffffe8	j gcm.Loop
r1Greater	0x00220822	sub r1,r1,r2
0x00000130	0x0bffffe0	j gcm.Loop
Result	0xac01102c	sw PrintValue(r0),r1
Result+0x4	0x200e1028	addi r14,r0,0x1028
Result+0x8	0x44000005	trap 0x5
Result+0xc	BID 0x44000000	trap 0x0
0x00000144	0xac021090 IF	sw SaveR2(r0),r2
0x00000148	0xac031094	sw SaveR3(r0),r3

Register									
PC=	0x00000148	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0
IMAR=	0x00000144	R8=	0x00000000	R31=	0x00000108	F22=	0	D26=	0
IR=	0xac021090	R9=	0x00000000	F0=	0	F23=	0	D28=	0
A=	0x00000000	R10=	0x00000000	F1=	0	F24=	0	D30=	0
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0		
B=	0x00000000	R12=	0x00000000	F3=	0	F26=	0		
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0		
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0		
ALU=	0x00000114	R15=	0x00000000	F6=	0	F29=	0		
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0		
FPSR=	0x00000000	R17=	0x00000000	F8=	0	F31=	0		
DMAR=	0x00000000	R18=	0x00000000	F9=	0	D0=	0		
SDR=	0x00000000	R19=	0x00000000	F10=	0	D2=	0		
SDRHI=	0x00000000	R20=	0x00000000	F11=	0	D4=	0		
LDR=	0x00000000	R21=	0x00000000	F12=	0	D6=	0		
LDRHI=	0x00000000	R22=	0x00000000	F13=	0	D8=	0		
R0=	0x00000000	R23=	0x00000000	F14=	0	D10=	0		
R1=	0x00000006	R24=	0x00000000	F15=	0	D12=	0		
R2=	0x00000006	R25=	0x00000000	F16=	0	D14=	0		
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0		
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0		
R5=	0x00000000	R28=	0x00000000	F19=	0	D20=	0		
R6=	0x00000000	R29=	0x00000000	F20=	0	D22=	0		

可以看到，第一个数（R1 寄存器中）被保存在了 R2 寄存器中，因为后面要将第二个数保存在 R1 寄存器中，所以这里要进行转存。

继续执行到下一个断点处。键入第二个数“3”。

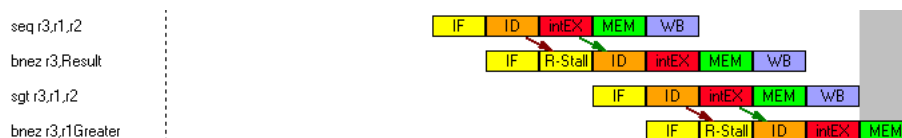


此处出现的“aborted”与上一个原因雷同。

查看寄存器，可以看到，第二个数保存在了 R1 寄存器中，第一个数保存在了 R2 寄存器中。

Register									
PC=	0x0000011c	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0
IMAR=	0x00000118	R8=	0x00000000	R31=	0x00000114	F22=	0	D26=	0
IR=	0x14600018	R9=	0x00000000	F0=	0	F23=	0	D28=	0
A=	0x00000003	R10=	0x00000000	F1=	0	F24=	0	D30=	0
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0		
B=	0x00000006	R12=	0x00000000	F3=	0	F26=	0		
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0		
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0		
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0		
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0		
FPSR=	0x00000000	R17=	0x00000000	F8=	0	F31=	0		
DMAR=	0x00000000	R18=	0x00000000	F9=	0	D0=	0		
SDR=	0x00000000	R19=	0x00000000	F10=	0	D2=	0		
SDRHI=	0x00000000	R20=	0x00000000	F11=	0	D4=	0		
IDR=	0x00000000	R21=	0x00000000	F12=	0	D6=	0		
IDRHI=	0x00000000	R22=	0x00000000	F13=	0	D8=	0		
R0=	0x00000000	R23=	0x00000000	F14=	0	D10=	0		
R1=	0x00000003	R24=	0x00000000	F15=	0	D12=	0		
R2=	0x00000006	R25=	0x00000000	F16=	0	D14=	0		
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0		
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0		
R5=	0x00000000	R28=	0x00000000	F19=	0	D20=	0		
R6=	0x00000000	R29=	0x00000000	F20=	0	D22=	0		

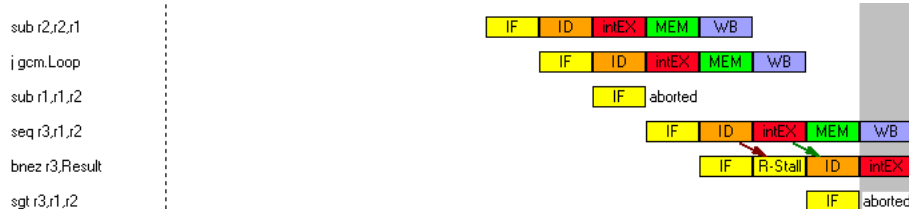
接下来，首先通过 seq 指令判断(R1)和(R2)是否相等，若相等会跳转到 Result 处，但此时不相等，所以继续顺序执行。然后再判断(R1)和(R2)谁更大，若(R1)更大则会跳转至 r1Greater 处，此处(R2)更大，所以顺序执行。



此处的两个红色箭头是因为上下两条指令都使用了 R3 寄存器，执行时有冲突；绿色箭头是因为使用了定向技术。

继续单步执行程序，执行(R2)=(R1)-(R2)，然后开始新一轮的循环，跳转到了 gcm.Loop，所以这里出现了一个”aborted”。

循环体中再对(R1)和(R2)比较大小，相等((R3)=1)，则计算结束，跳转到 Result 处。



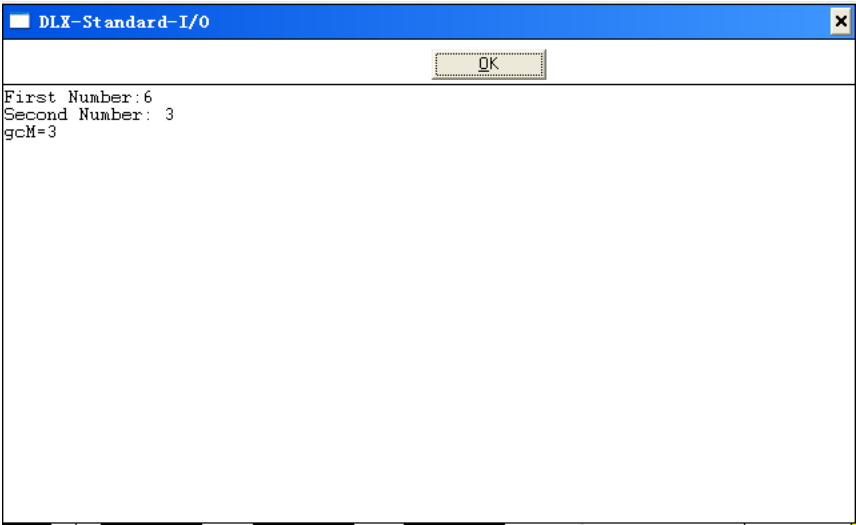
Register									
PC=	0x00000138	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0
IMAR=	0x00000134	R8=	0x00000000	R31=	0x00000114	F22=	0	D26=	0
IR=	0xac01102c	R9=	0x00000000	F0=	0	F23=	0	D28=	0
A=	0x00000000	R10=	0x00000000	F1=	0	F24=	0	D30=	0
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0		
B=	0x00000000	R12=	0x00000000	F3=	0	F26=	0		
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0		
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0		
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0		
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0		
FPSR=	0x00000000	R17=	0x00000000	F8=	0	F31=	0		
DMAR=	0x00000000	R18=	0x00000000	F9=	0	D0=	0		
SDR=	0x00000000	R19=	0x00000000	F10=	0	D2=	0		
SDRHI=	0x00000000	R20=	0x00000000	F11=	0	D4=	0		
IDR=	0x00000000	R21=	0x00000000	F12=	0	D6=	0		
IDRHI=	0x00000000	R22=	0x00000000	F13=	0	D8=	0		
R0=	0x00000000	R23=	0x00000000	F14=	0	D10=	0		
R1=	0x00000003	R24=	0x00000000	F15=	0	D12=	0		
R2=	0x00000003	R25=	0x00000000	F16=	0	D14=	0		
R3=	0x00000001	R26=	0x00000000	F17=	0	D16=	0		
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0		

运行程序直至第三个断点，即程序结束。

Address	PC	Instruction
0x00000000	0x00000000	nop
0x00000004	0x00000000	nop
0x00000008	0x00000000	nop
0x0000000c	0x00000000	nop
\$TEXT	0x20011000	addi r1,r0,0x1000
main+0x4	0x0c00003c	jal InputUnsigned
main+0x8	BID 0x00201020	add r2,r1,r0
main+0xc	0x2001100e	addi r1,r0,0x100e
main+0x10	0x0c000030	jal InputUnsigned
gcmLoop	BID 0x00221828	seq r3,r1,r2
0x00000118	0x14600018	bnez r3,r1,r2
0x0000011c	0x0022182b	sgt r3,r1,r2
0x00000120	0x14600008	bnez r3,r1,r2
r2Greater	0x00411022	sub r2,r2,r1
0x00000128	0x0bffffe8	j gcmLoop
r1Greater	0x00220822	sub r1,r1,r2
0x00000130	0x0bffffe0	j gcmLoop
Result	0xac01102c	sw PrintValue(r0),r1
Result+0x4	0x200e1028	addi r14,r0,0x1028
Result+0x8	0x44000005	trap 0x5
Result+0xc	BID 0x44000000	ID trap 0x0
0x00000144	0xac021090	IF sw SaveR2(r0),r2
0x00000148	0xac031094	sw SaveR3(r0),r3

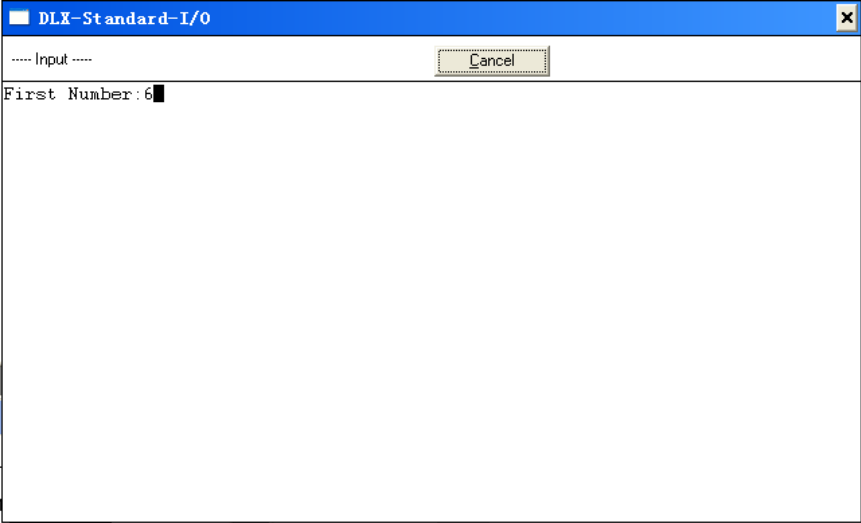


查看 execute/display DLX-I/O 中显示的结果:



4. 运行程序（第二组数 6 和 1）

首先按下 F5，使程序运行到第一个断点处，即完成键入第一个数。



Code

```

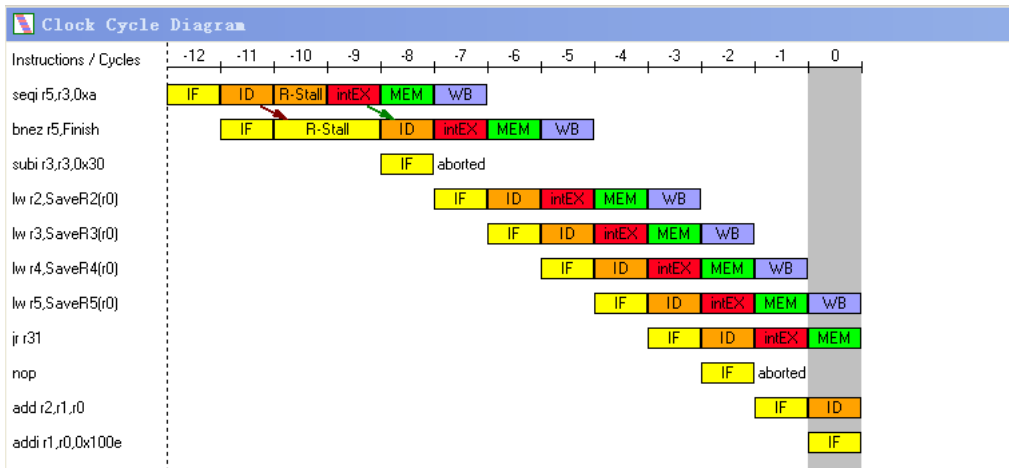
0x00000000 0x00000000 nop
0x00000004 0x00000000 nop
0x00000008 0x00000000 nop
0x0000000c 0x00000000 nop
$TEXT
main+0x4 0x20011000 addi r1,r0,0x1000
main+0x8 0x0c00003c jal InputUnsigned
main+0xc 0x2001100e addi r1,r0,0x100e
main+0x10 0x0c000030 jal InputUnsigned
gcm.Loop 0x00221828 seq r3,r1,r2
0x00000118 0x14600018 bnez r3,Result
0x0000011c 0x0022182b sgt r3,r1,r2
0x00000120 0x14600008 bnez r3,r1Greater
r2Greater 0x00411022 sub r2,r2,r1
0x00000128 0x0bffffe8 jgcm.Loop
r1Greater 0x00220822 sub r1,r1,r2
0x00000130 0x0bffffe0 jgcm.Loop
Result 0xac01102c sw PrintValue(r0),r1
Result+0x4 0x200e1028 addi r14,r0,0x1028
Result+0x8 0x44000005 trap 0x5
Result+0xc 0x44000000 trap 0x0
0x00000144 0xac021090 sw SaveR2(r0),r2
0x00000148 0xac031094 sw SaveR3(r0),r3

```

main+0x8: add r2, r1, r0

ID-Stage: Reached at Breakpoint #1!

确定



发现 Clock Cycle Diagram 窗口出现了红色和绿色箭头。

红色箭头是因为 seqi r5,r3,0xa 与指令 bnez r5,Finish 发生了冲突，所以 bnez 指令需要等待 seqi 指令完成执行，即完成 intEX 阶段才能通过定向得到 r5 寄存器的值，所以此时 bnez 指令发生了 R-Stall 暂停，直到 seqi 指令结束 intEX 阶段。而绿色箭头是因为这里使用了定向技术。

在第三条指令处出现了“aborted”。这是因为上一条指令 bnez 发生了跳转。所以该条指令暂停，留下了气泡。

同时，根据子程序 input.s，可以知道键入的数被保存在 R1 寄存器中，查看 Register 窗口验证。

Register

PC=	0x00000110	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0
IMAR=	0x0000010c	R8=	0x00000000	R31=	0x00000108	F22=	0	D26=	0
IR=	0x2001100e	R9=	0x00000000	F0=	0	F23=	0	D28=	0
A=	0x00000006	R10=	0x00000000	F1=	0	F24=	0	D30=	0
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0		
B=	0x00000000	R12=	0x00000000	F3=	0	F26=	0		
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0		
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0		
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0		
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0		
FPSR=	0x00000000	R17=	0x00000000	F8=	0	F31=	0		
DMAR=	0x00000000	R18=	0x00000000	F9=	0	D0=	0		
SDR=	0x00000000	R19=	0x00000000	F10=	0	D2=	0		
SDRHI=	0x00000000	R20=	0x00000000	F11=	0	D4=	0		
LDR=	0x00000000	R21=	0x00000000	F12=	0	D6=	0		
LDRHI=	0x00000000	R22=	0x00000000	F13=	0	D8=	0		
R0=	0x00000000	R23=	0x00000000	F14=	0	D10=	0		
R1=	0x00000006	R24=	0x00000000	F15=	0	D12=	0		
R2=	0x00000000	R25=	0x00000000	F16=	0	D14=	0		
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0		
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0		
R5=	0x00000000	R28=	0x00000000	F19=	0	D20=	0		
R6=	0x00000000	R29=	0x00000000	F20=	0	D22=	0		

继续单步执行程序。

Code		
0x00000000	0x00000000	nop
0x00000004	0x00000000	nop
0x00000008	0x00000000	nop
0x0000000c	0x00000000	nop
\$TEXT	0x20011000	addi r1,r0,0x1000
main+0x4	0x0c00003c	jal InputUnsigned
main+0x8	BID 0x00201020 WB	add r2,r1,r0
main+0xc	0x2001100e MEM	addi r1,r0,0x100e
main+0x10	0x0c000030 intEX	jal InputUnsigned
gcm.Loop	BID 0x00221828	seq r3,r1,r2
0x00000118	0x14600018	bnez r3,Result
0x0000011c	0x0022182b	sgt r3,r1,r2
0x00000120	0x14600008	bnez r3,r1Greater
r2Greater	0x00411022	sub r2,r2,r1
0x00000128	0x0bffffe8	j gcm.Loop
r1Greater	0x00220822	sub r1,r1,r2
0x00000130	0x0bffffe0	j gcm.Loop
Result	0xac01102c	sw PrintValue(r0),r1
Result+0x4	0x200e1028	addi r14,r0,0x1028
Result+0x8	0x44000005	trap 0x5
Result+0xc	BID 0x44000000	trap 0x0
0x00000144	0xac021090 IF	sw SaveR2(r0),r2
0x00000148	0xac031094	sw SaveR3(r0),r3

Register									
PC=	0x00000148	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0
IMAR=	0x00000144	R8=	0x00000000	R31=	0x00000108	F22=	0	D26=	0
IR=	0xac021090	R9=	0x00000000	F0=	0	F23=	0	D28=	0
A=	0x00000000	R10=	0x00000000	F1=	0	F24=	0	D30=	0
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0		
B=	0x00000000	R12=	0x00000000	F3=	0	F26=	0		
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0		
BT=	0x00000000	R14=	0x00001080	F5=	0	F28=	0		
ALU=	0x00000114	R15=	0x00000000	F6=	0	F29=	0		
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0		
FPSR=	0x00000000	R17=	0x00000000	F8=	0	F31=	0		
DMAR=	0x00000000	R18=	0x00000000	F9=	0	D0=	0		
SDR=	0x00000000	R19=	0x00000000	F10=	0	D2=	0		
SDRHI=	0x00000000	R20=	0x00000000	F11=	0	D4=	0		
IDR=	0x00000000	R21=	0x00000000	F12=	0	D6=	0		
IDRHI=	0x00000000	R22=	0x00000000	F13=	0	D8=	0		
R0=	0x00000000	R23=	0x00000000	F14=	0	D10=	0		
R1=	0x00000006	R24=	0x00000000	F15=	0	D12=	0		
R2=	0x00000006	R25=	0x00000000	F16=	0	D14=	0		
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0		
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0		
R5=	0x00000000	R28=	0x00000000	F19=	0	D20=	0		
R6=	0x00000000	R29=	0x00000000	F20=	0	D22=	0		

可以看到，第一个数（R1 寄存器中）被保存在了 R2 寄存器中，因为后面要将第二个数保存在 R1 寄存器中，所以这里要进行转存。

继续执行到下一个断点处。键入第二个数“1”。

DLX-Standard-I/O

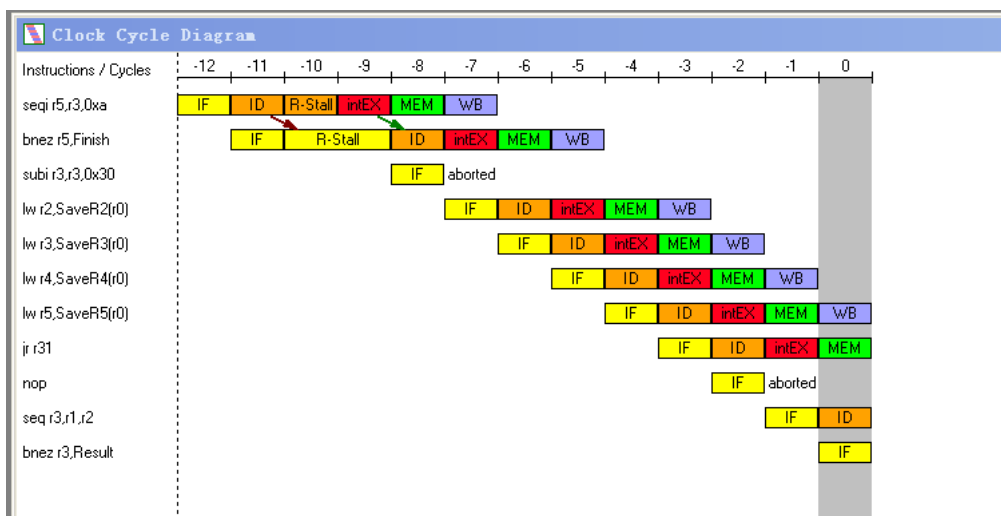
..... Input

Cancel

First Number: 6

Second Number: 1

Address	Hex	Assembly
0x00000000	0x00000000	nop
0x00000004	0x00000000	nop
0x00000008	0x00000000	nop
0x0000000c	0x00000000	nop
\$TEXT	0x20011000	addi r1,r0,0x1000
main+0x4	0x0c00003c	jal InputUnsigned
main+0x8	0x00201020	add r2,r1,r0
main+0xc	0x2001100e	addi r1,r0,0x100e
main+0x10	0x0c000030	jal InputUnsigned
gcm.Loop	0x00221828	ID seq r3,r1,r2
0x00000118	0x14600018	IF bnez r3,Result
0x0000011c	0x0022182b	sgt r3,r1,r2
0x00000120	0x14600008	bnez r3,r1Greater
r2Greater	0x00411022	sub r2,r2,r1
0x00000128	0x0bffffe8	j gcm.Loop
r1Greater	0x00220822	sub r1,r1,r2
0x00000130	0x0bffffe0	j gcm.Loop
Result	0xac01102c	sw PrintValue(r0),r1
Result+0x4	0x200e1028	addi r14,r0,0x1028
Result+0x8	0x44000005	trap 0x5
Result+0xc	0x44000000	trap 0x0
0x00000144	0xac021090	sw SaveR2(r0),r2
0x00000148	0xac031094	sw SaveR3(r0),r3

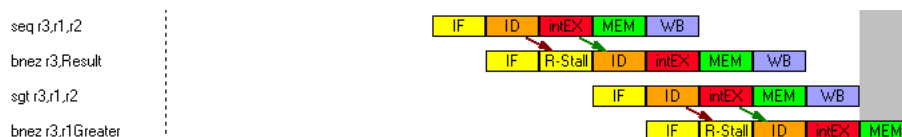


此处出现的“aborted”与上一个原因雷同。

查看寄存器，可以看到，第二个数保存在了 R1 寄存器中，第一个数保存在了 R2 寄存器中。

Register			
PC=	0x0000011c	R7=	0x00000000
IMAR=	0x00000118	R8=	0x00000000
IR=	0x14600018	R9=	0x00000000
A=	0x00000001	R10=	0x00000000
AHI=	0x00000000	R11=	0x00000000
B=	0x00000006	R12=	0x00000000
BHI=	0x00000000	R13=	0x00000000
BTA=	0x00000000	R14=	0x00001080
ALU=	0x00000000	R15=	0x00000000
ALUHI=	0x00000000	R16=	0x00000000
FPSR=	0x00000000	R17=	0x00000000
DMAR=	0x00000000	R18=	0x00000000
SDR=	0x00000000	R19=	0x00000000
SDRHI=	0x00000000	R20=	0x00000000
LDR=	0x00000000	R21=	0x00000000
LDRHI=	0x00000000	R22=	0x00000000
R0=	0x00000000	R23=	0x00000000
R1=	0x00000001	R24=	0x00000000
R2=	0x00000006	R25=	0x00000000
R3=	0x00000000	R26=	0x00000000
R4=	0x00000000	R27=	0x00000000
F21=	0	D24=	0
F22=	0	D26=	0
F23=	0	D28=	0
F24=	0	D30=	0
F25=	0		
F26=	0		
F27=	0		
F28=	0		
F29=	0		
F30=	0		
F31=	0		
D0=	0		
D2=	0		
D4=	0		
D6=	0		
D8=	0		
D10=	0		
D12=	0		
D14=	0		
D16=	0		
D18=	0		

接下来，首先通过 seq 指令判断(R1)和(R2)是否相等，若相等会跳转到 Result 处，但此时不相等，所以继续顺序执行。然后再判断(R1)和(R2)谁更大，若(R1)更大则会跳转至 r1Greater 处，此处(R2)更大，所以顺序执行。



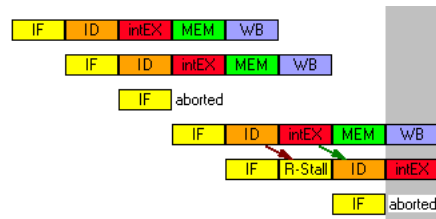
此处的两个红色箭头是因为上下两条指令都使用了 R3 寄存器，执行时有冲突；绿色箭头是因为使用了定向技术。

继续单步执行程序，执行(R2)=(R1)-(R2)，然后开始新一轮的循环，跳转到了 gcm.Loop，所以这里出现了一个”aborted”。

循环体中再对(R1)和(R2)比较大小。

一直循环，直到(R1)=(R2)。跳转到 Result。

```
sub r2,r2,r1
j gcm.Loop
sub r1,r1,r2
seq r3,r1,r2
bnez r3,Result
sgt r3,r1,r2
```



Register															
PC=	0x0000011c	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0						
IMAR=	0x00000118	R8=	0x00000000	R31=	0x00000114	F22=	0	D26=	0						
IR=	0x14600018	R9=	0x00000000	F0=	0	F23=	0	D28=	0						
A=	0x00000001	R10=	0x00000000	F1=	0	F24=	0	D30=	0						
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0								
B=	0x00000005	R12=	0x00000000	F3=	0	F26=	0								
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0								
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0								
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0								
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0								
FPSR=	0x00000000	R17=	0x00000000	F8=	0	F31=	0								
DMAR=	0x00000000	R18=	0x00000000												
SDR=	0x00000000	R19=	0x00000000												
SDRHI=	0x00000000	R20=	0x00000000												
LDR=	0x00000000	R21=	0x00000000												
LDRHI=	0x00000000	R22=	0x00000000												
R0=	0x00000000	R23=	0x00000000												
R1=	0x00000001	R24=	0x00000000												
R2=	0x00000005	R25=	0x00000000												
R3=	0x00000000	R26=	0x00000000												
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0								

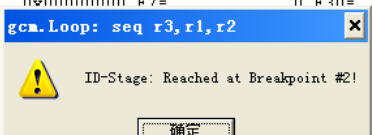
Register															
PC=	0x0000011c	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0						
IMAR=	0x00000118	R8=	0x00000000	R31=	0x00000114	F22=	0	D26=	0						
IR=	0x14600018	R9=	0x00000000	F0=	0	F23=	0	D28=	0						
A=	0x00000001	R10=	0x00000000	F1=	0	F24=	0	D30=	0						
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0								
B=	0x00000004	R12=	0x00000000	F3=	0	F26=	0								
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0								
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0								
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0								
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0								
FPSR=	0x00000000	R17=	0x00000000												
DMAR=	0x00000000	R18=	0x00000000												
SDR=	0x00000000	R19=	0x00000000												
SDRHI=	0x00000000	R20=	0x00000000												
LDR=	0x00000000	R21=	0x00000000												
LDRHI=	0x00000000	R22=	0x00000000												
R0=	0x00000000	R23=	0x00000000												
R1=	0x00000001	R24=	0x00000000												
R2=	0x00000004	R25=	0x00000000												
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0								
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0								

Register															
PC=	0x0000011c	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0						
IMAR=	0x00000118	R8=	0x00000000	R31=	0x00000114	F22=	0	D26=	0						
IR=	0x14600018	R9=	0x00000000	F0=	0	F23=	0	D28=	0						
A=	0x00000001	R10=	0x00000000	F1=	0	F24=	0	D30=	0						
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0								
B=	0x00000003	R12=	0x00000000	F3=	0	F26=	0								
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0								
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0								
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0								
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0								
FPSR=	0x00000000	R17=	0x00000000												
DMAR=	0x00000000	R18=	0x00000000												
SDR=	0x00000000	R19=	0x00000000												
SDRHI=	0x00000000	R20=	0x00000000												
LDR=	0x00000000	R21=	0x00000000												
LDRHI=	0x00000000	R22=	0x00000000												
R0=	0x00000000	R23=	0x00000000												
R1=	0x00000001	R24=	0x00000000												
R2=	0x00000003	R25=	0x00000000												
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0								
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0								

Register															
PC=	0x0000011c	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0						
IMAR=	0x00000118	R8=	0x00000000	R31=	0x00000114	F22=	0	D26=	0						
IR=	0x14600018	R9=	0x00000000	F0=	0	F23=	0	D28=	0						
A=	0x00000001	R10=	0x00000000	F1=	0	F24=	0	D30=	0						
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0								
B=	0x00000002	R12=	0x00000000	F3=	0	F26=	0								
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0								
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0								
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0								
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0								
FPSR=	0x00000000	R17=													
DMAR=	0x00000000	R18=													
SDR=	0x00000000	R19=													
SDRHI=	0x00000000	R20=													
LDR=	0x00000000	R21=													
LDRHI=	0x00000000	R22=													
R0=	0x00000000	R23=													
R1=	0x00000001	R24=													
R2=	0x00000002	R25=													
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0								
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0								



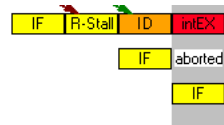
Register															
PC=	0x0000011c	R7=	0x00000000	R30=	0x00000000	F21=	0	D24=	0						
IMAR=	0x00000118	R8=	0x00000000	R31=	0x00000114	F22=	0	D26=	0						
IR=	0x14600018	R9=	0x00000000	F0=	0	F23=	0	D28=	0						
A=	0x00000001	R10=	0x00000000	F1=	0	F24=	0	D30=	0						
AHI=	0x00000000	R11=	0x00000000	F2=	0	F25=	0								
B=	0x00000001	R12=	0x00000000	F3=	0	F26=	0								
BHI=	0x00000000	R13=	0x00000000	F4=	0	F27=	0								
BTA=	0x00000000	R14=	0x00001080	F5=	0	F28=	0								
ALU=	0x00000000	R15=	0x00000000	F6=	0	F29=	0								
ALUHI=	0x00000000	R16=	0x00000000	F7=	0	F30=	0								
FPSR=	0x00000000	R17=													
DMAR=	0x00000000	R18=													
SDR=	0x00000000	R19=													
SDRHI=	0x00000000	R20=													
LDR=	0x00000000	R21=													
LDRHI=	0x00000000	R22=													
R0=	0x00000000	R23=													
R1=	0x00000001	R24=													
R2=	0x00000001	R25=													
R3=	0x00000000	R26=	0x00000000	F17=	0	D16=	0								
R4=	0x00000000	R27=	0x00000000	F18=	0	D18=	0								



```

bnez r3,Result
sgt r3,r1,r2
sw PrintValue(r0),r1

```

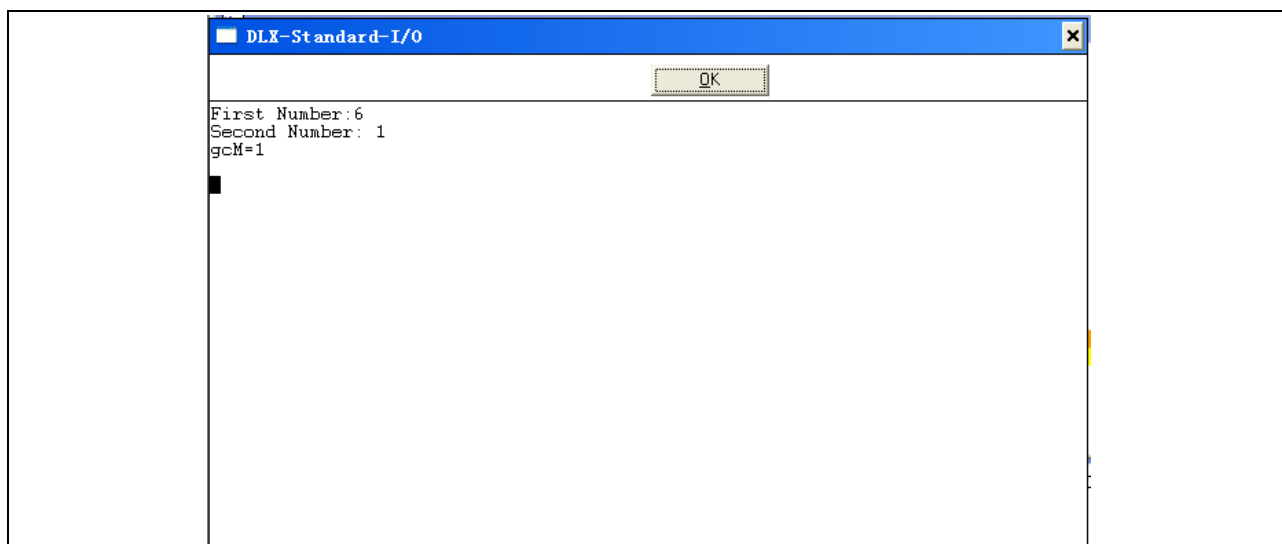


运行程序直至第三个断点，即程序结束。

Code			
0x00000000	0x00000000	nop	
0x00000004	0x00000000	nop	
0x00000008	0x00000000	nop	
0x0000000c	0x00000000	nop	
\$TEXT	0x20011000	addi r1,r0,0x1000	
main+0x4	0x0c00003c	jal InputUnsigned	
main+0x8	0x00201020	add r2,r1,r0	
main+0xc	0x2001100e	addi r1,r0,0x100e	
main+0x10	0x0c000030	jal InputUnsigned	
gcm.Loop	0x00221828	seq r3,r1,r2	
0x00000118	0x14600018	bnez r3,Result	
0x0000011c	0x0022182b	sgt r3,r1,r2	
0x00000120	0x14600008	bnez r3,r1Greater	
r2Greater	0x00411022	sub r2,r2,r1	
0x00000128	0x0bffffe8	j gcm.Loop	
r1Greater	0x00220822	sub r1,r1,r2	
0x00000130	0x0bffffe0	j gcm.Loop	
Result	0xac01102c	sw PrintValue(r0),r1	
Result+0x4	0x200e1028	addi r14,r0,0x1028	
Result+0x8	0x44000005	trap 0x5	
Result+0xc	BID 0x44000000	ID trap 0x0	
0x00000144	0xac021090	IF sw SaveR2(r0),r2	
0x00000148	0xac031094	sw SaveR3(r0),r3	



查看 execute/display DLX-I/O 中显示的结果：



结论分析与体会：

本次实验帮助我更深入理解了循环跳转的内部流程，在每次跳转时，流水线都会出现一个”aborted”和残留的一些气泡。

通过 WinDLX，可以从多个角度——代码角度、流水线时钟角度等——观察一个程序的执行，通过单步调试，可以对程序和计算机的理解更透彻。

同时，WinDLX 可以在多种配置下工作，可以改变流水线的结构和时间要求、存储器大小和其他几个控制模拟的参数，使用方便。