

Freescal FAE75 Training Genesi Pegasos II: Using SimG4+

Top Changwatchai and Jack Chen

June 10, 2004

Freescal™ and the Freescal logo are trademarks of Freescal Semiconductor, Inc.
All other product or service names are the property of their respective owners. © Freescal Semiconductor, Inc. 2004



Presentation Overview

- **What is SimG4+?**
- **Preparation**
 - Directory layout
 - Examples and tool flow
- **Step 1: Compile source code**
- **Step 2: Run simulator**
- **Step 3: View pipeline**
- ***(Optional)* Analysis example**

What is SimG4+?

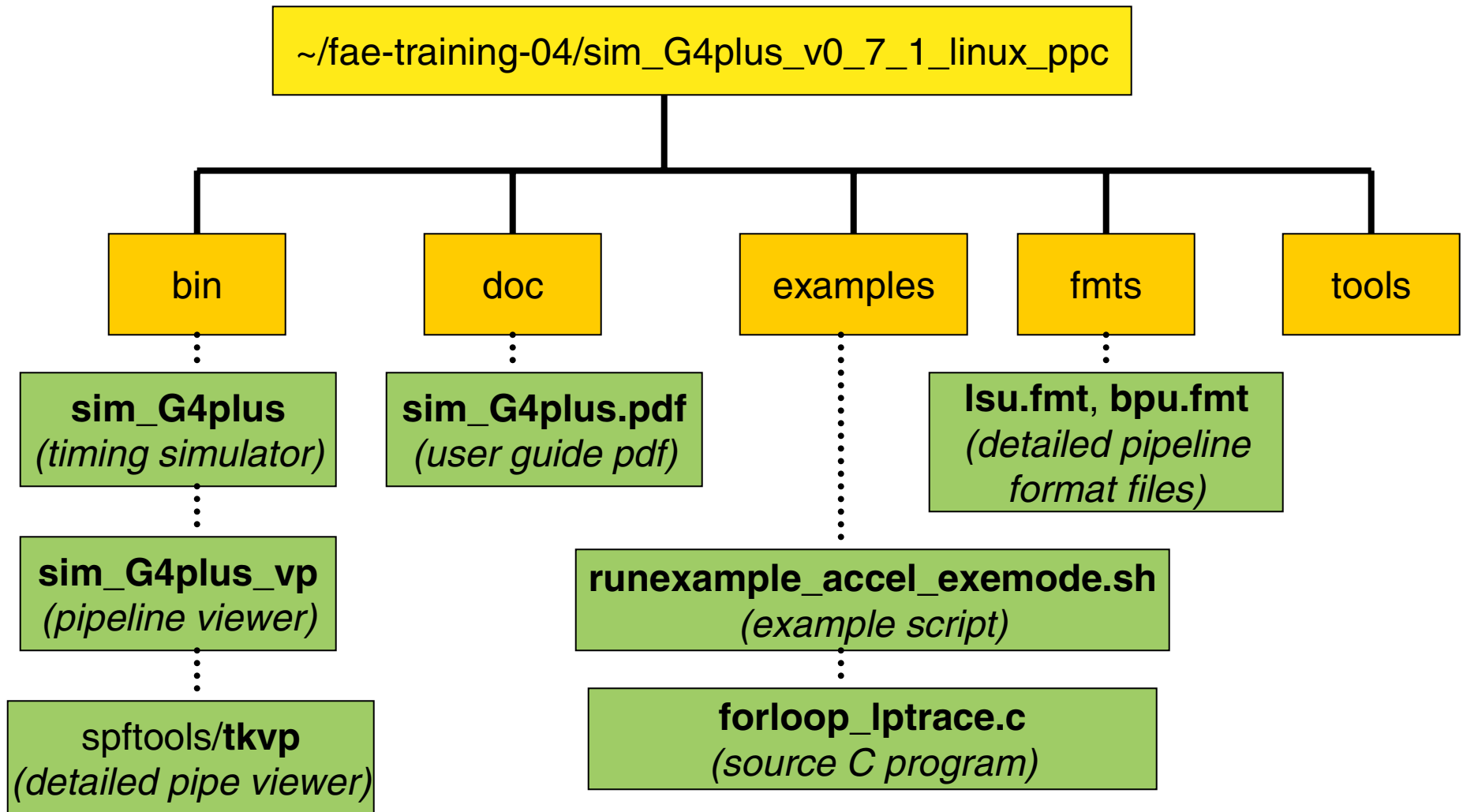
- **SimG4+ is a cycle-accurate, execution-driven timing simulator of the G4+ microprocessor**

- “cycle-accurate” = hardware is simulated at cycle-by-cycle granularity
- “execution-driven” = simulator consumes an executable program

- **Uses**

- Application tuning
- Library development/tuning
- Compiler optimization development

Directory Layout



Login!

1. login as username: **guest** password: **guest**
2. **\$ which sim_G4plus**

Go to example directory!

1. **\$ cd ~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples**
2. **\$ ls**

(TIP: to use command and filename completion, type first few characters of name, then press <tab> to complete)

```
guest@debian:~$ which sim_G4plus
/home/guest/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/bin/sim_G4plus
guest@debian:~$ cd ~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples/
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ ls
GNUmakefile          forloop_lptrace
README.examples     forloop_lptrace.c
forloop.reference.output  hello.tte
forloop_accel_exemode.pipeout  runexample_accel_exemode.sh
forloop_accel_exemode.stats    runexample_exemode.sh
forloop_accel_exemode.txt      runexample_lptrace.sh
forloop_exemode.c
```

View the Example Script

\$ less runexample_accel_exemode.sh

(TIP: scroll up and down with arrow keys, pgup/pgdown, 'q' to quit)

Steps:

1. Compile source code

2. Run simulator

3. View pipeline

```
#!/bin/sh

# This script demonstrates the tool sequence for generating a pipeline
# output from an lptrace instrumented .c program using the accelerated
# execution-driven mode of sim_G4plus. No trace files are generated
# in this sequence.
#
# This script expects to be run on a Linux PowerPC platform because
# the binary needs to be built using gcc for PowerPC.
#
if [ "`uname -sm`" != "Linux ppc" ]; then
    echo
    echo This script must be run on a Linux PowerPC platform
    echo Please examine this script to understand how to invoke the tools
    echo for performance analysis.
    echo
    exit 1
fi

# Compile instrumented program (.c -> executable)
echo "Compiling forloop_lptrace.c..."
# The generated executable must be linked statically
gcc -O -o forloop_lptrace forloop_lptrace.c -static

if [ ! -x ../bin/sim_G4plus ]; then
    echo "ERROR: Expected sim_G4plus to be here: ../bin/sim_G4plus"
    exit 1
fi

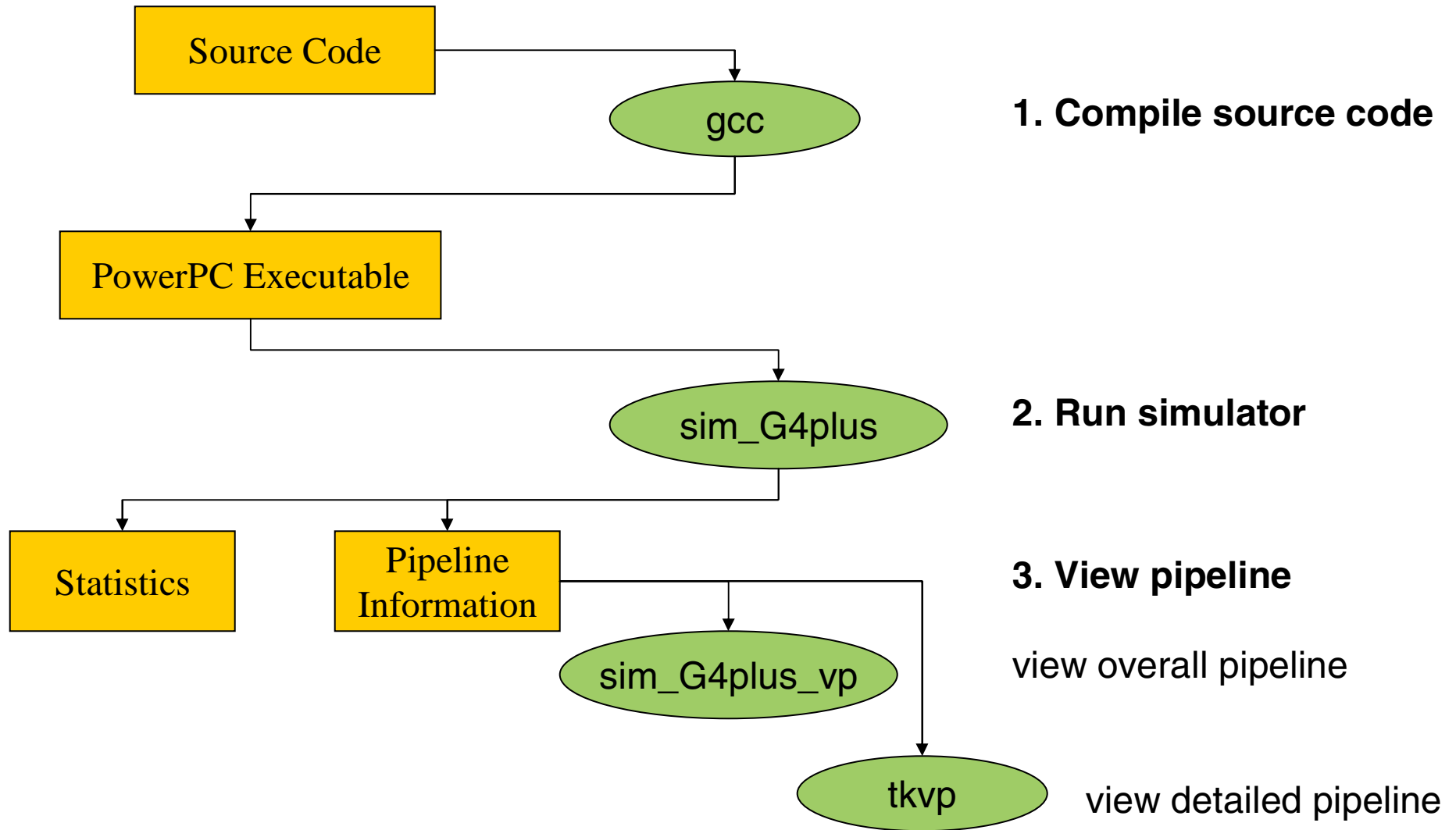
echo "Running on the simulator..."
# Run through simulator (the lptraceout) (the .c state)
../bin/sim_G4plus -a -p forloop_accel_exemode.pipeout \
    forloop_lptrace > forloop_accel_exemode.stats

if [ ! -x ../bin/sim_G4plus_vp ]; then
    echo "ERROR: Expected sim_G4plus_vp to be here: ../bin/sim_G4plus_vp"
    exit 1
fi

echo "Generating pipeout dump..."
# Dump pipeout (pipeout)
../bin/sim_G4plus_vp -r 60 -n forloop_accel_exemode.pipeout > forloop_accel_exem
ode.txt

echo
```

SimG4+ Tool Flow



Step 1: Compile Source Code

View Source Code Before Compiling

\$ less forloop_lptrace.c

or

\$ vi forloop_lptrace.c

- **start and stop markers are illegal opcodes understood by the simulator**

- **start code: 0x14000001**
- **stop code: 0x14000002**

- **the resulting executable will run only on the simulator**

```
/*
 * forloop.c
 *
 * Compile using:
 *   gcc -O forloop.c -o forloop -static
 *
 * The purpose of this program is to demonstrate the
 * instrumentation for tracing a for loop.
 */

int main (void) {
    unsigned int i;
    unsigned int j;
    unsigned int x = 0x0000100;

    /* start tracing */
    asm (".long 0x14000001");
    j = 0;
    for (i = 0; i < 32; i++) {
        if (x & 0x1) {
            j++;
        }
        x >>= 1;
    }

    /* stop tracing */
    asm (".long 0x14000002");
    return j;
}
forloop_lptrace.c (END)
```

start marker

stats/pipeline collected

stop marker

Compile Source Code

1. `$ gcc -o forloop_lptrace forloop_lptrace.c -static`

specify executable to output

specify input source file

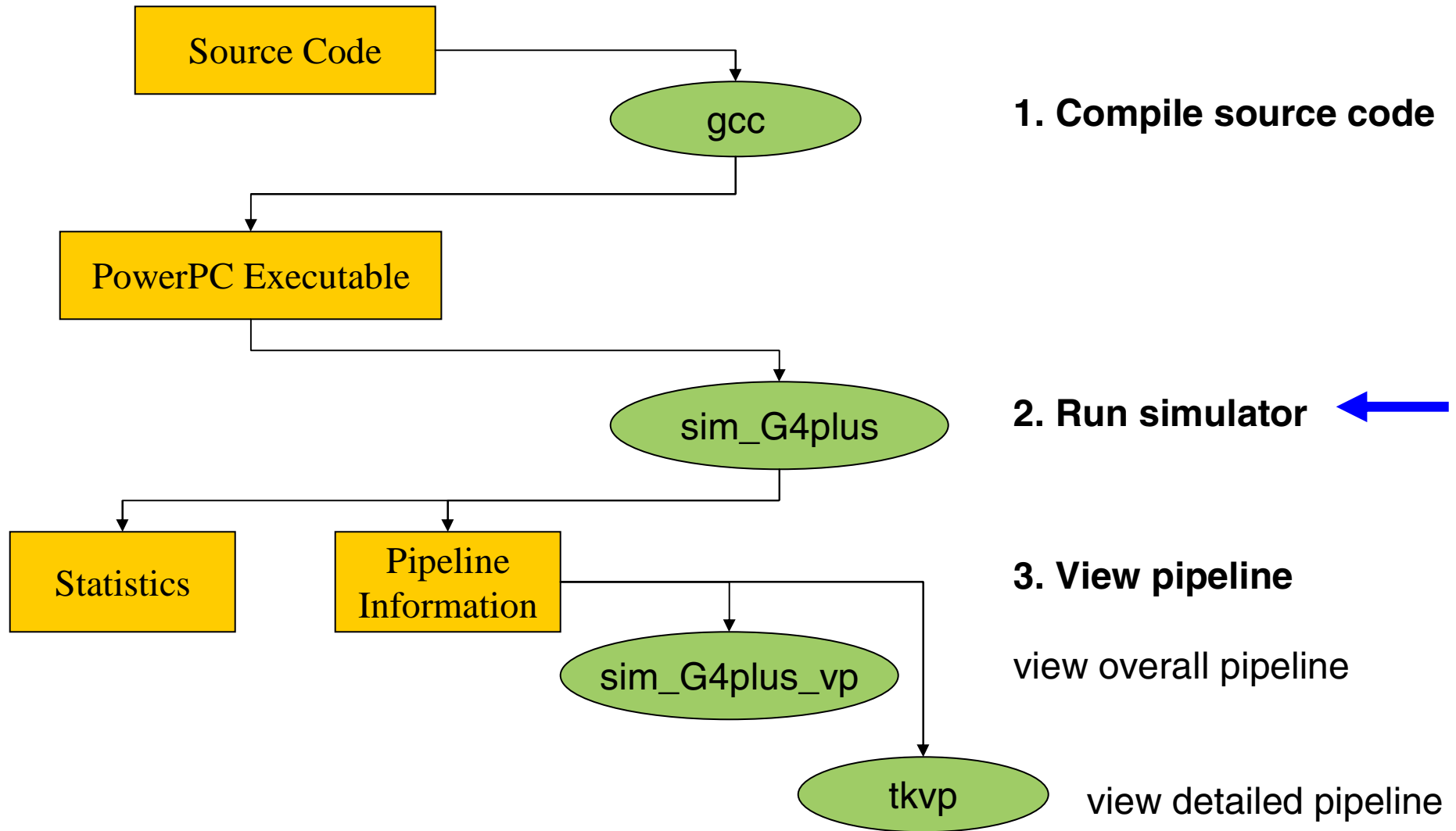
must link statically
(simulator complains if it is given a dynamically-linked executable)

- `$ file forloop_lptrace`

(Optional) verify the file type of the executable generated
(ELF, 32-bit MSB, PowerPC, statically linked)

```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ gcc -o forloop_lptrace forloop_lptrace.c -static
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ file forloop_lptrace
forloop_lptrace: ELF 32-bit MSB executable PowerPC for cisco 4500,
version 1 (SYSV), for GNU/Linux 2.2.0, statically linked not strip
ped
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ █
```

SimG4+ Tool Flow



Step 2: Run Simulator

Simulator Options: “-h” to access help

```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ sim_G4plus -h  
sim_G4plus expiration date: Sun Nov 14 23:59:59 2004
```

```
sim_G4plus v0_7_1
```

```
The performance model for the MPC7447A  
Copyright 2003 Motorola.  
Motorola Confidential Proprietary.
```

Usage

```
Usage: sim_G4plus [options] [input_tracefile]
```

```
Options for controlling simulation runs and output:
```

```
-c <clocks>      How many clock cycles to run (default is seq.max_cycles)  
-d path1:path2  Search path for parameter files  
-e <eventfile> Set the param seq.pevents_file to <eventfile>  
-f <filename>   Use a parameter file  
-p <filename>   Write pipeline info for post-processing to <filename>  
-s param=value  Set the runtime parameter 'param' to 'value'  
-a              Run a PowerPC ELF binary in accelerated mode
```

```
Options for displaying status (no simulation done):
```

```
-E              Print the registered events, then exit  
-r              Print runtime parameters then exit  
-R              Print runtime parameters with descriptions then exit  
-v              Print version message then exit  
-h              Print help message then exit
```

```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ █
```

Simulator Options: “-v” to access version

```
quest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ sim_G4plus -v  
sim_G4plus expiration date: Sun Nov 14 23:59:59 2004
```

Expiration Date

```
sim_G4plus v0_7_1
```

Version

```
The performance model for the MPC7447A  
Copyright 2003 Motorola.  
Motorola Confidential Proprietary.
```

```
Run on host : debian  
Current Time : Mon Jun 7 22:49:51 2004
```

```
quest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ █
```

Simulator Options: “-r”, “-R” (verbose) to view parameters

```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ sim_G4plus -r
sim_G4plus expiration date: Sun Nov 14 23:59:59 2004

sim_G4plus v0_7_1

The performance model for the MPC7447A
Copyright 2003 Motorola.
Motorola Confidential Proprietary.

# *** General Simulation Parameters ***
sim.trace_type = auto
sim.max_cycles = 0
sim.pipeout_file =
sim.pipeout_start = 1
sim.pipeout_stop = 0
sim.pevents_file =
sim.pevents_enable = *.*
sim.pevents_append = false
sim.testcase_num = 0
# sim.bus_ratio = 1
sim.strace_on = false

# *** Sequencer Parameters ***
# seq.sanity_mode = false
seq.enable_data_dependent = true

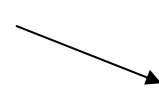
# *** Fetch Parameters ***
fet.start_delay = 0
icache.mode = infinite

# *** Branch Parameters ***
bpu.perfect_direction_prediction_trace_mode = false
bpu.enable_bht = true
bpu.enable_link_stack = true
bpu.enable_btlic = true
bpu.enable_branch_folding = true

# *** LSU Parameters ***
dcache.mode = infinite

# *** MSS Parameters ***
# l2cache.mode = infinite
# l2cache.first_beat_latency = 9
# l2cache.subsequent_beat_latency = 1
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ █
```

Parameter



Default value

Run Simulator

```
$ sim_G4plus -a -p forloop.pipeout forloop_lptrace > forloop.stats
```

generate pipeline output file

start simulation in accelerated mode

(don't collect stats/pipeline until simulation reaches start marker)

executable to simulate

statistics output

(redirect from stdout to file)

```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ sim_G4plus -a -p forloop.pipeout forloop_lptrace > forloop.stats
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ ls
GNUmakefile                forloop_lptrace
README.examples            forloop_lptrace.c
forloop.pipeout            hello.tte
forloop.reference.output  runexample_accel_exemode.sh
forloop.stats              runexample_exemode.sh
forloop_accel_exemode.txt  runexample_lptrace.sh
forloop_exemode.c
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$
```

View Statistics (1/3)

\$ less forloop.stats

```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ less forloop.stats  
sim_G4plus expiration date: Sun Nov 14 23:59:59 2004
```

```
sim_G4plus v0_7_1
```

```
The performance model for the MPC7447A  
Copyright 2003 Motorola.  
Motorola Confidential Proprietary.
```

```
Run on host : debian  
Current Time : Mon Jun 7 22:58:08 2004
```

```
# *****  
# *** BEGIN SIMULATION PARAMETERS ***  
# *****  
  
# *** General Simulation Parameters ***  
sim.trace_type = auto  
sim.max_cycles = 0  
sim.pipeout_file = forloop.pipeout  
sim.nineout_start = 1
```

simulation parameters

View Statistics (2/3)

\$ less forloop.stats

```
# *** MSS Parameters ***  
# l2cache.mode = infinite  
# l2cache.first_beat_latency = 9  
# l2cache.subsequent_beat_latency = 1  
  
# *****  
# **** END SIMULATION PARAMETERS ****  
# *****
```

simulation parameters

```
939524095 bytes are available for your program's heap and stack.  
Running forloop_lptrace ...  
SPORTAL: Recieved program exit System Call with exit status: 1  
Program exited with status 1
```

messages during simulation

```
Usage for Buffer Instruction Buffer, size=12  
Instruction Buffer [ 0 ]      24      492      4.8780%  
Instruction Buffer [ 1 ]      67      492     13.6179%  
Instruction Buffer [ 2 ]      30      492      6.0976%  
Instruction Buffer [ 3 ]      36      492      7.3171%  
Instruction Buffer [ 4 ]      76      492     15.4472%  
Instruction Buffer [ 5 ]      97      492     19.7154%
```

output statistics

View Statistics (3/3)

\$ less for loop.stats

LSO GET Data MUX Output Utilization	231	492	40.9512%
----- Completion Performance -----			
Cycles per CPU second	492.0000	1.5300	321.5686
IPS (Insts Completed per CPU second)	364.0000	1.5300	237.9085
Number of completed instructions (includes micro-ops from load/store multiples/strings, does not include folded branches)	364		
Average IPC	364	492	0.7398
Usage for Queue Completion Queue, size=16			
Completion Queue [0]	17	492	3.4553%
Completion Queue [1]	1	492	0.2038%

instructions

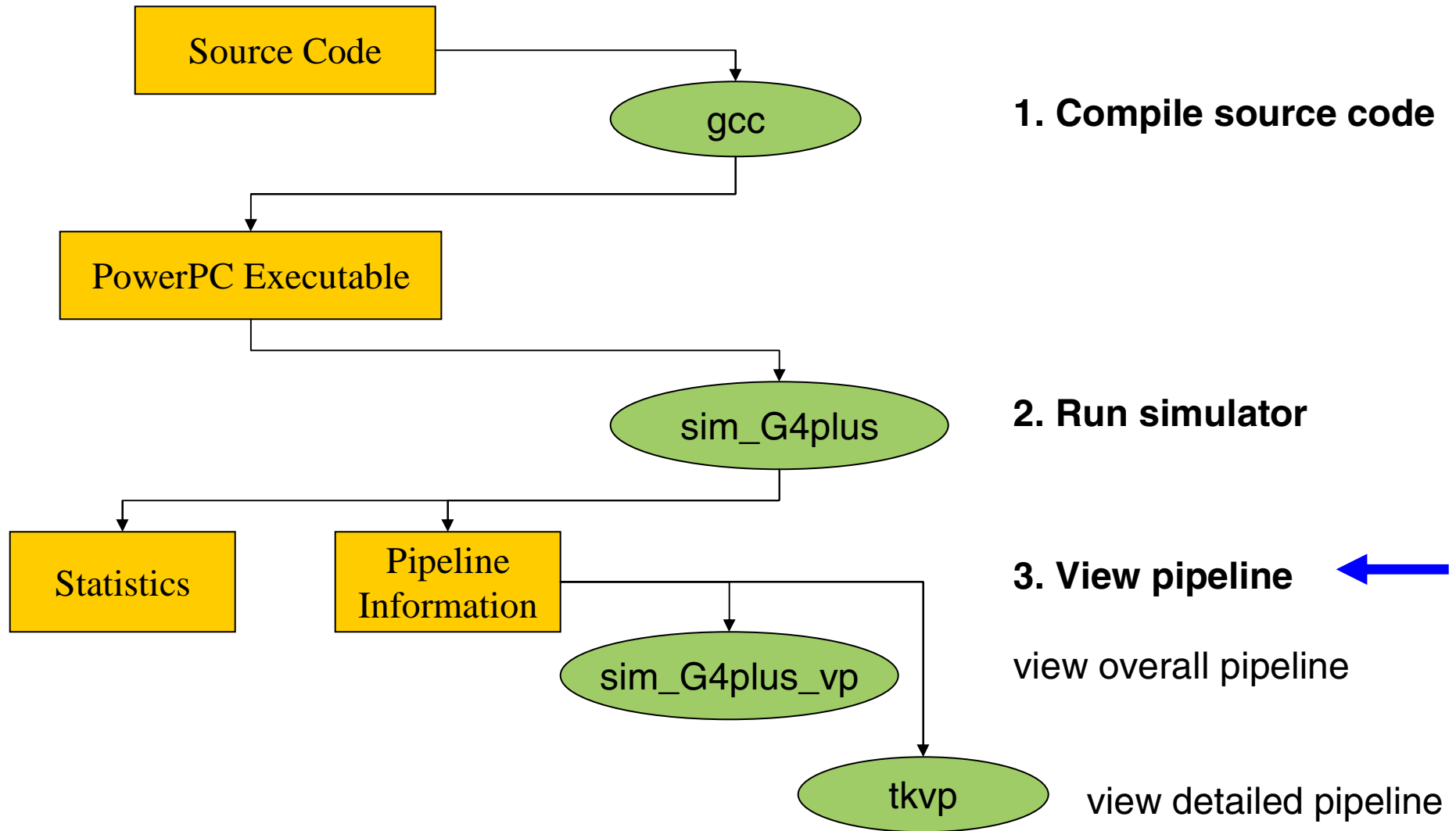
cycles

IPC
(instructions per cycle)

Pipeline Output File

- **Pipeline Output file:**
 - Is a text file
 - Shows status of simulated processor on each clock cycle
- **Use a viewer**

SimG4+ Tool Flow



Step 3: View Pipeline

Overall Pipeline Viewer Options

\$ sim_G4plus_vp -h

```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples$ sim_G4plus_vp -h

Usage /home/guest/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/bin/spftools/tksnp.pl [options]
] <snapshot_def_file> <spf_file>

Where options are one of:
  -f | --firstcycle <num> /* First cycle to display (Non-TK mode only) */
  -e | --endcycle <num> /* Last cycle to display */
  -s | --showcols <list> /* Colon-separated list columns to display */
  -c | --cycwidth <num> /* Number of characters for the cycle column */
  -m | --mapwidth <num> /* Number of characters for the map column */
  -r | --rows <num> /* Number of lines per page */
  -d | --deffile <file> /* Used the deffile (instead of on cmd line) */
  -i | --ignoresize /* In non-Tk mode, ignore the size of the
                    terminal in which we were invoked */
  -n | --notk /* Don't use the TK interface */
  -p | --printall /* Print all cycles on a page in Non-TK mode.
                 Default is to try to only print the number
                 of rows */
  -D | --Delimiter <s> /* Output using the specified
                       string as the delimiter between columns. */
  -w | --warnings /* Give warnings if we can't find a symbol
                 in the snapshot file */
  -l | --list /* List all known columns, then exit
              NOTE: This option requires both a
              snapshot_def_file AND an spf_file. This
              is the only way that the columns for
              a specific spf can be properly displayed */
  -F | --Font <font> /* Use the specified font. */
  -P | --P <Program> /* Run Program on the input file */
  -h | --help /* Print this message, then exit */

The snapshot_def_file should be provided when this script
is delivered. The spf_file is the pipeline output of the simulation.
```

Run Overall Pipeline Viewer

\$ sim_G4plus_vp forloop.pipeout &

pipeline output file generated

run viewer as background process

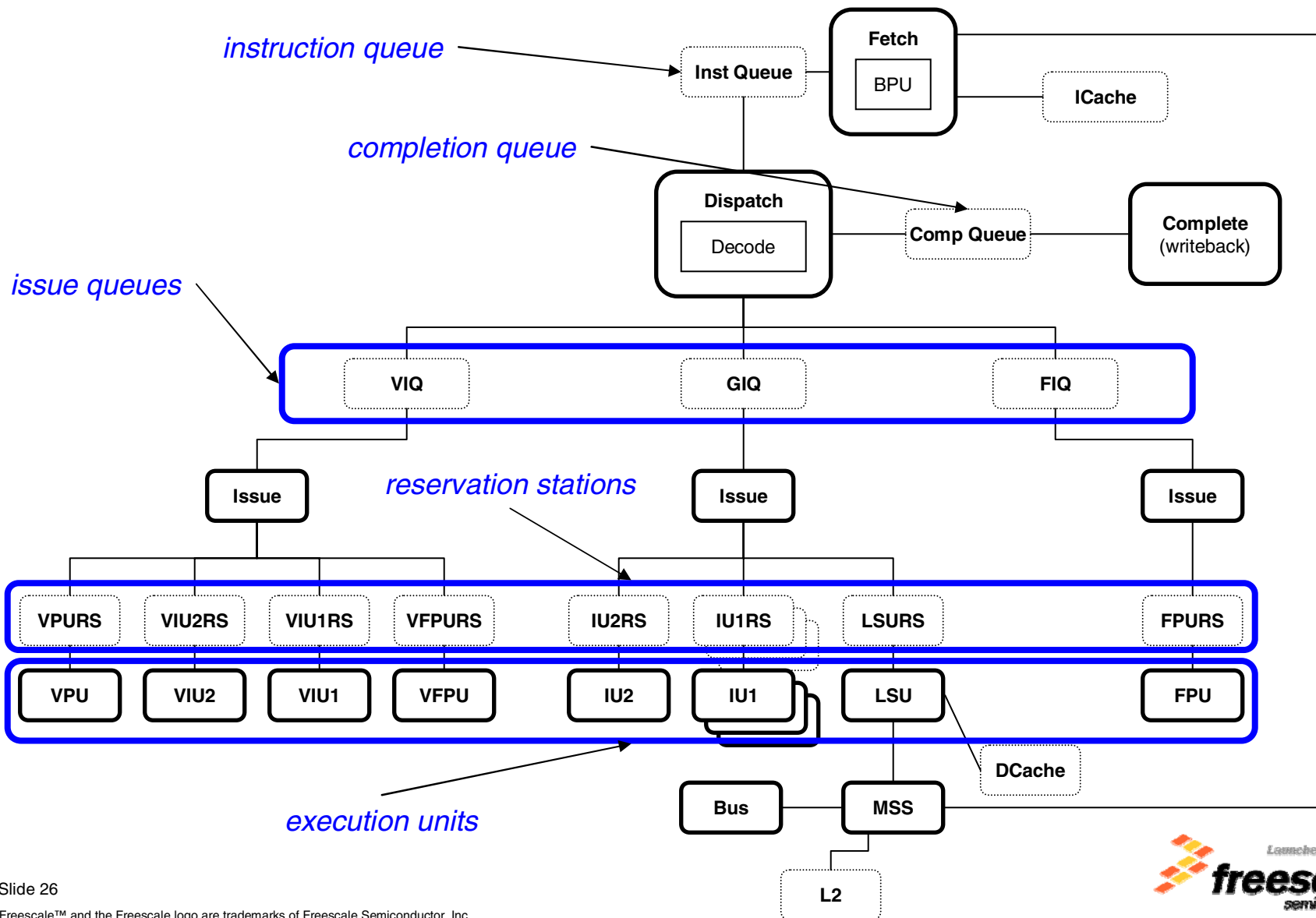
```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ sim_G4plus_vp forloop.pipeout &
[1] 2519
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ Reading pipeline file `/home/guest/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/bin/spftools/babehspf WriteBack.Buffer forloop.pipeoutl'opened via `/home/guest/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/bin/spftools/babehspf WriteBack.Buffer forloop.pipeoutl'...
Reading SPF 2.x file
Updating data values...
File read complete.
```

Overall Pipeline Viewer Display

\$ sim_G4plus_vp forloop.pipeout &

InstAddr	Inst	Instructions	TraceNum	InstNum,Op	Cycle	Fetch0	Fetch1	InstB	GIQ	VIQ	FI	RR	EE	EE	RR	II	II	IIII	II
(A) 0x100002b0	addi	r0,r0,0		1	1	10x100002b0	CR	109876543210	543210	3210	10	10	01	0	0				
(B) 0x100002b4	stw	r0,12(r31)		2	2	210x100002c0	FS 0x100002b0	CR											
(C) 0x100002b8	addi	r0,r0,0		3	3	310x100002d0	FS 0x100002c0	FS											
(D) 0x100002bc	stw	r0,8(r31)		4	4	410x100002d0	FS	DCBA											
(E) 0x100002c0	luz	r0,8(r31)		5	5	5+0x100002e0	FS+	LKJHGF+											
(F) 0x100002c4	cmplwi	r0,0x001F		6	6	610x10000308	BR												
(G) 0x100002c8	ble-	0x8	PFA-	7	7	710x10000318	FS 0x10000308	BR											
(H) # 100002cc	b	60	P-A-	8	8	810x10000318	FS												
(I) # 100002d0	luz	r0,16(r31)		9	9	910x10000318	FS	HI											
(J) # 100002d4	rlwinm	r0,r0,0,31,31		10	10+	10+													
(K) # 100002d8	cmplwi	r0,0x0000		11	11	11													
(L) # 100002dc	beq-	0x10	P-A-	12	12	12													
(M) # 10000308	lptrace_stop			13	13	13													
(N) 0x100002d0	luz	r0,16(r31)		14	14	14													
(O) 0x100002d4	rlwinm	r0,r0,0,31,31		15	15+0x100002d0	BR+													
(P) 0x100002d8	cmplwi	r0,0x0000		16	16	1610x100002e0	FS 0x100002d0	BR											
(Q) 0x100002dc	beq-	0x10	PFA-	17	17	1710x100002f0	FS 0x100002e0	FS											
(R) # 100002e0	luz	r9,12(r31)		18	18	1810x100002f0	FS	QPON											
(S) # 100002e4	addi	r0,r9,1		19	19	1910x10000300	FS	UTSRQ											
(T) # 100002e8	stw	r0,12(r31)		20	20+	20+	+0x10000300	FS+	YXWVU										
(U) # 100002ec	luz	r0,16(r31)		21	21	21		YXWVU+	T+										
(V) # 100002f0	rlwinm	r0,r0,31,1,31		22	22	22													
(W) # 100002f4	stw	r0,16(r31)		23	23	23													
(X) # 100002f8	luz	r9,8(r31)		24	24	24													
(Y) # 100002fc	addi	r0,r9,1		25	25	25													

G4+ Block Diagram



Overall Pipeline Viewer Display: In-Depth (1/2)

The image shows a screenshot of the Overall Pipeline Viewer Display. The display is divided into several sections:

- Legend:** A list of instructions on the left side, including their addresses and mnemonics. Examples include:
 - (A) 0x100002b0 addi
 - (B) 0x100002b4 stw
 - (C) 0x100002b8 addi
 - (D) 0x100002bc stw
 - (E) 0x100002c0 lzw
 - (F) 0x100002c4 cmplwi
 - (G) 0x100002c8 ble-
 - (H) # 100002cc b
 - (I) # 100002d0 lzw
 - (J) # 100002d4 rlwinm
 - (K) # 100002d8 cmplwi
 - (L) # 100002dc beq-
 - (M) # 10000308 lptrace_stop
 - (N) 0x100002d0 lzw
 - (O) 0x100002d4 rlwinm
- Pipeline View:** A central area showing the pipeline stages and registers. The stages are Fetch, Issue, Memory Access, and Write Back. The registers are InstB, GIQ, VIQ, FI, RR, EE, and E. The pipeline registers contain hexadecimal values, such as 109876543210 for InstB and 5432103210 for GIQ.
- Controls:** A control box at the bottom right, containing a button with a right-pointing arrow and a text field with the value '1'.

Blue boxes and arrows highlight these components: 'legend' points to the instruction list, 'pipeline' points to the central view, and 'controls' points to the control box.

Detailed Pipeline Viewer: LSU (1/2)

```
$ ../bin/spftools/tkvp ../fmts/lsu.fmt forloop.pipeout &
```

detailed pipeline viewer

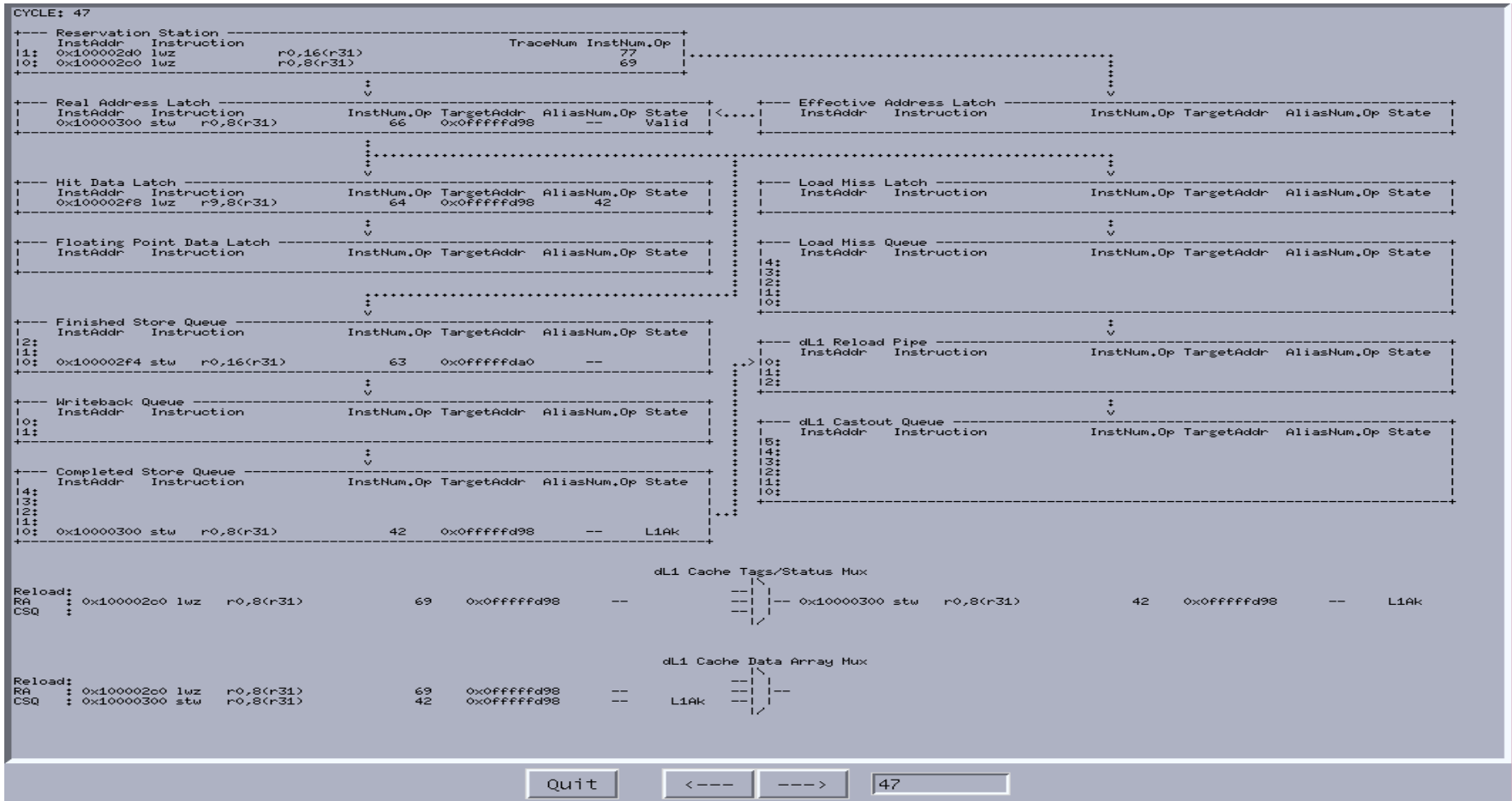
LSU format file

pipeline output file generated

```
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ ../bin/spftools/tkvp ../fmts/lsu.fmt forloop.pipeout &
[1] 2304
guest@debian:~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
$ Reading pipeline file `forloop.pipeout' opened via `<forloop.pipeo
ut' ...
Reading SPF 2.x file
Updating data values...
File read complete.
```

Detailed Pipeline Viewer: LSU (2/2)

\$../bin/spftools/tkvp ../fmts/lsu.fmt forloop.pipeout &



Detailed Pipeline Viewer: BPU

\$ **../bin/spftools/tkvp ../fmts/bpu.fmt forloop.pipeout &**

```

CYCLE: 91
+--- Instruction Buffer -----+
|11:                               |
|10:                               |
| 9:                               |
| 8:                               |
| 7:                               |
| 6:                               |
| 5:                               |
| 4: 0x100002cc b                 60          P-A-      184
| 3: 0x100002c8 ble-              0x8          PTA-      183
| 2: 0x100002c4 cmlwi             r0,0x001F
| 1: 0x100002c0 lwz                r0,8(r31)
| 0: 0x10000300 stw                r0,8(r31)
+---+

+--- Execution Stall Status -----+
| Branch executed                 |
| 0x100002c8 ble-                 0x8          PTA-      183
+---+

+--- Branch Spec Stream -----+
| 0x100002c8 ble-                 0x8          PTA-      183
| 0x100002dc beq-                 0x10         PTA-      164
| 0x100002c8 ble-                 0x8          PTA-      155
+---+

+--- Oldest Spec Branch Connection -----+
| 0x100002c8 ble-                 0x8          PTA-      155
+---+

+--- Taken Flushed Connection -----+
| 0x100002c8 ble-                 0x8          PTA-      183
+---+

+--- Mispredict Drain Connection -----+
|
+---+

+--- Link Stack ---+
| 7: (0x00000000) |
| 6: (0x00000000) |
| 5: (0x00000000) |
| 4: (0x00000000) |
| 3: (0x00000000) |
| 2: (0x00000000) |
| 1: (0x00000000) |
| 0: (0x00000000) |
+---+

Quit <--- <---> 91
    
```

Review of Commands

Preparation

```
$ which sim_G4plus
```

```
$ cd ~/fae-training-04/sim_G4plus_v0_7_1_linux_ppc/examples
```

```
$ less runexample_accel_exemode.sh
```

```
$ less forloop_lptrace.c
```

1. Compile source code

```
$ gcc -o forloop_lptrace forloop_lptrace.c -static
```

```
$ file forloop_lptrace
```

2. Run simulator

```
$ sim_G4plus -a -p forloop.pipeout forloop_lptrace > forloop.stats
```

```
$ less forloop.stats
```

3. View pipeline

```
$ sim_G4plus_vp -h
```

```
$ sim_G4plus_vp forloop.pipeout &
```

```
$ ../bin/spftools/tkvp ../fmts/lisu.fmt forloop.pipeout &
```

```
$ ../bin/spftools/tkvp ../fmts/bpu.fmt forloop.pipeout &
```

Analysis Example

View Pipeline

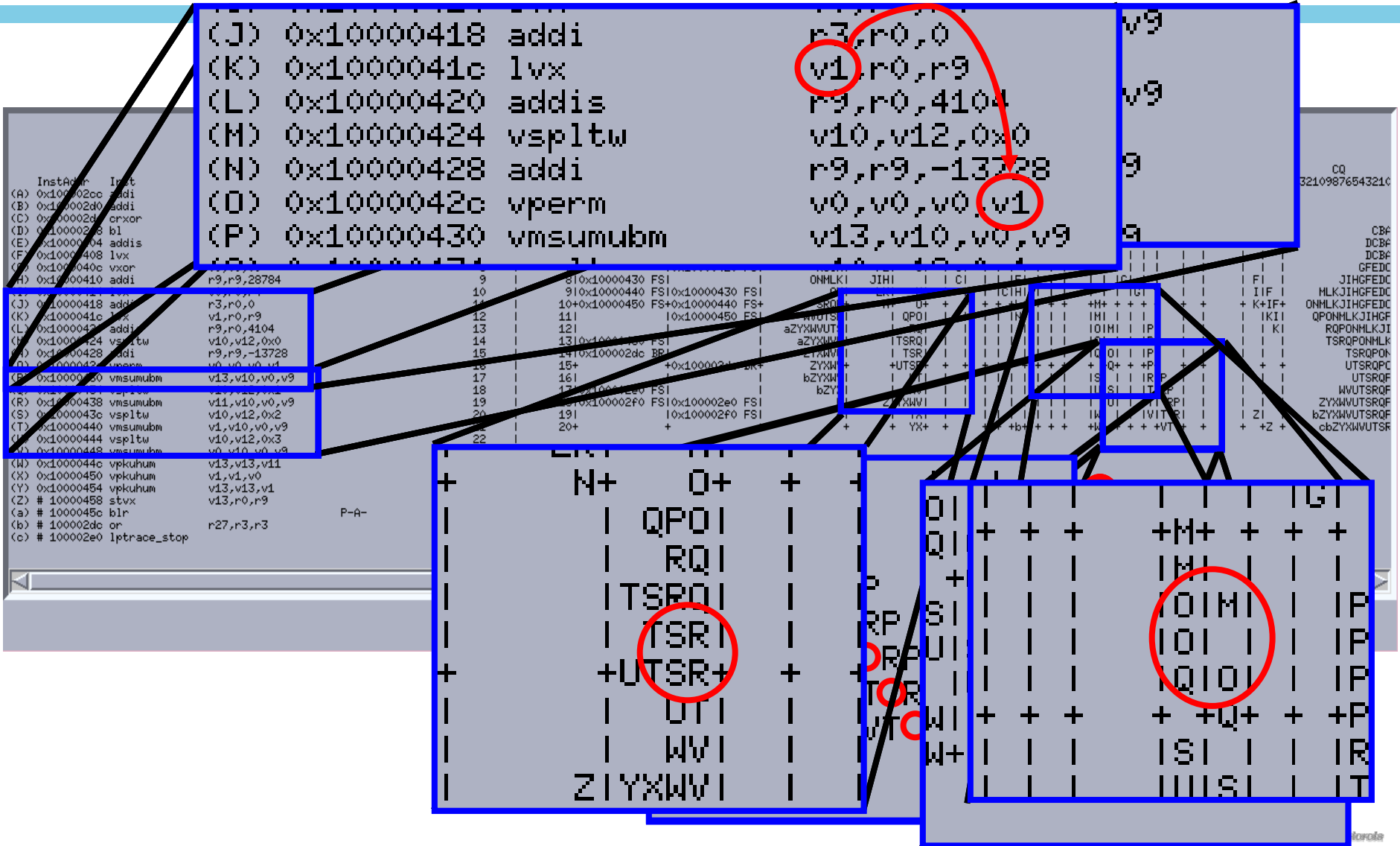
```
$ cd ~/fae-training-04/library/matrix_mul/sndfdemo-slow/
```

```
$ ls
```

```
$ sim_G4plus_vp sndfdemo-slow.execute.pipeout
```

```
$ cd ~/fae-training-04/library/matrix_mul/sndfdemo-slow/
guest@debian:~/fae-training-04/library/matrix_mul/sndfdemo-slow
$ ls
,build.sh          sndfdemo-slow.execute.pipeout
,cleanup.sh       sndfdemo-slow.execute.stats
,execute-and-sim.sh  sndfdemo-slow.lptrace.pipeout
,trace-and-sim.sh  sndfdemo-slow.lptrace.stats
,viewpipe.execute.sh  sndfdemo-slow.traceme
,viewpipe.lptrace.sh  sndfdemo-slow.tte
sndfdemo-slow.c    ttt
guest@debian:~/fae-training-04/library/matrix_mul/sndfdemo-slow
$ sim_G4plus_vp sndfdemo-slow.execute.pipeout
Reading pipeline file '/home/guest/fae-training-04/sim_G4plus_v
0_7_1_linux_ppc/bin/spftools/babehspf WriteBack.Buffer sndfdemo
-slow.execute.pipeout|' opened via '/home/guest/fae-training-04/
sim_G4plus_v0_7_1_linux_ppc/bin/spftools/babehspf WriteBack.Buf
fer sndfdemo-slow.execute.pipeout|'...
Reading SPF 2.x file
Updating data values...
File read complete.
```

Pipeline to Analyze



Analysis

1. **There is a bubble in the VIU2 pipeline between R and T**
2. **T waits in the VIU2 reservation station 1 extra cycle**
3. **T depends on the result in register v10 from S**
4. **S is delayed from entering VPU reservation station because it is blocked in VIQ by R**
5. **R is held in VIQ for 1 extra cycle because the VIU2 reservation station is blocked by P**
6. **P depends on the result in register v0 from O**
7. **O stalls in VPU reservation station because it depends on v1 from K**

Conclusion: If we can insert instructions between K and O, then O will not stall waiting for K