Modifications:

1. Inhibit "Reserved Instruction Exception".
   In module *m14k_mpc_dec*, change:

```
assign spec_ri_e = … … (mpc_ir_e[5:1] == 5'b101_00) ||
```

   For:

```
assign spec_ri_e = … … (mpc_ir_e[5:0] == 6'b101_001) ||
```

2. Select the SLT result (in which we will include also SEQ result) instead of the ALU result in *_res_m_31_0_* multiplexer (module *m14k_edp_buf_misc*):

```
mvp_mux2 #(32) _res_m_31_0_(res_m[31:0],mpc_udislt_sel_m, asp_m, {31'h0, bit0_m});
```

   This multiplexer is governed by signal *mpc_udislt_sel_m*, which is 1 for a *slt* (*slt_sel_m*=1) or *udi* (*mpc_udisel_m*=1) instruction and 0 otherwise.

```
assign mpc_udislt_sel_m = slt_sel_m | mpc_udisel_m;
```

   Thus, we must change computation of *slt_sel_m*. At module *m14k_mpc_dec*, add one line to computation of this signal. Change:

```
assign slt_sel_e =    (mpc_ir_e[31:27] == 5'b001_01)    ||         // slti, sltiu
special_e && (mpc_ir_e[5:1] == 5'b101_01) ||        // slt, sltu
special_e && (mpc_ir_e[5:2] == 4'b110_0)  ||        // tge, tgeu, tlt, tltu
regimm_e && (mpc_ir_e[20:18] == 3'o2);              // tgei, tgeiu, tlti tltiu
```

   For:

```
assign slt_sel_e =    (mpc_ir_e[31:27] == 5'b001_01)    ||         // slti, sltiu
special_e && (mpc_ir_e[5:1] == 5'b101_01) ||        // slt, sltu
special_e && (mpc_ir_e[5:1] == 5'b101_00) ||        // SEQ
special_e && (mpc_ir_e[5:2] == 4'b110_0)  ||        // tge, tgeu, tlt, tltu
regimm_e && (mpc_ir_e[20:18] == 3'o2);              // tgei, tgeiu, tlti tltiu
```

3. We have different options for computing the equality condition at module m14k_edp:
   a. The obvious solution is to include a new hardware for computing the equality condition (==) among the two operands.
   b. The SLT instruction is computed by substracting rs-rt in the "Carry Propagate Adder" included in module *m14k_edp*, called *edp_add*. We can use this operation for computing the equality condition, by ORing the 32 bits resulting from the substraction. In case the result is 0, both operands are equal, otherwise they are different.
   c. We can reuse a comparator already available in the pipeline. This will probably be the cheapest option, so it is the one that we are going to use:
      At module *m14k_edp*, the following hardware is implemented:

```
assign edp_cndeq_e = acmp_e == edp_bbus_e;
```

We can reuse this hardware, with some small changes:

- Signal *edp_bbus_e* can be used as it is.
- Signal *acmp_e* requires small changes. It is assigned as:

```
mvp_mux2 #(32) _acmp_e_31_0_(acmp_e[31:0],mpc_cmov_e, edp_abus_e, 32'h0);
```

  In this case, we need *edp_abus_e* to be selected. Thus, we will include as control signal a new one which is 1 when we have a *seq* instruction.

```
mvp_mux2 #(32) _acmp_e_31_0_(acmp_e[31:0],(mpc_cmov_e && ~seq_instr),
edp_abus_e, 32'h0);           // SEQ
```

  where signal *seq_instr* is computed at module m14k_mpc_dec as follows:

```
assign seq_instr = special_e && (mpc_ir_e[5:1] == 5'b101_00);       // SEQ
```

  and is registered to the M-Stage:

```
mvp_cregister #(1) _seq_instr_m(seq_instr_m,mpc_run_ie, gclk, seq_instr);
// SEQ
```

- This signal must be registered to be used at the M-Stage. We add:

```
mvp_cregister #(1) _edp_cndeq_m(edp_cndeq_m,mpc_run_ie, gclk,
edp_cndeq_e);  // SEQ
```

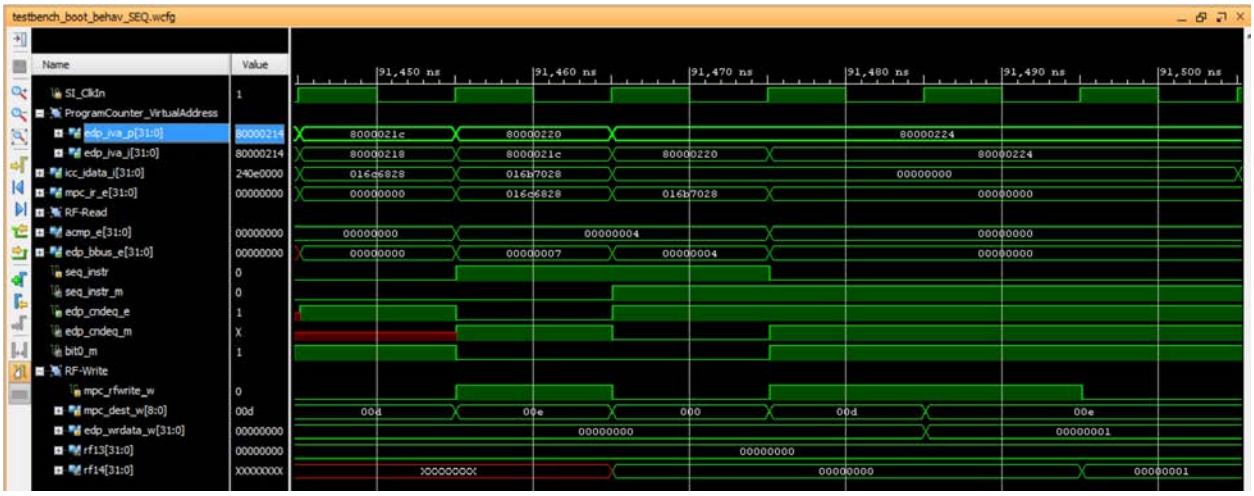
4. At module *m14k_edp*, we must incorporate the SEQ result to the SLT result (bit0_m). For that purpose, we must change:

```
assign bit0_m =       mpc_signed_m ? (pro31_m ^ car31_m) : ~car31_m;
```

   For:

```
assign bit0_m = seq_instr_m ? edp_cndeq_m : (mpc_signed_m ? (pro31_m ^ car31_m) :
~car31_m);            // SEQ
```

Simulation:



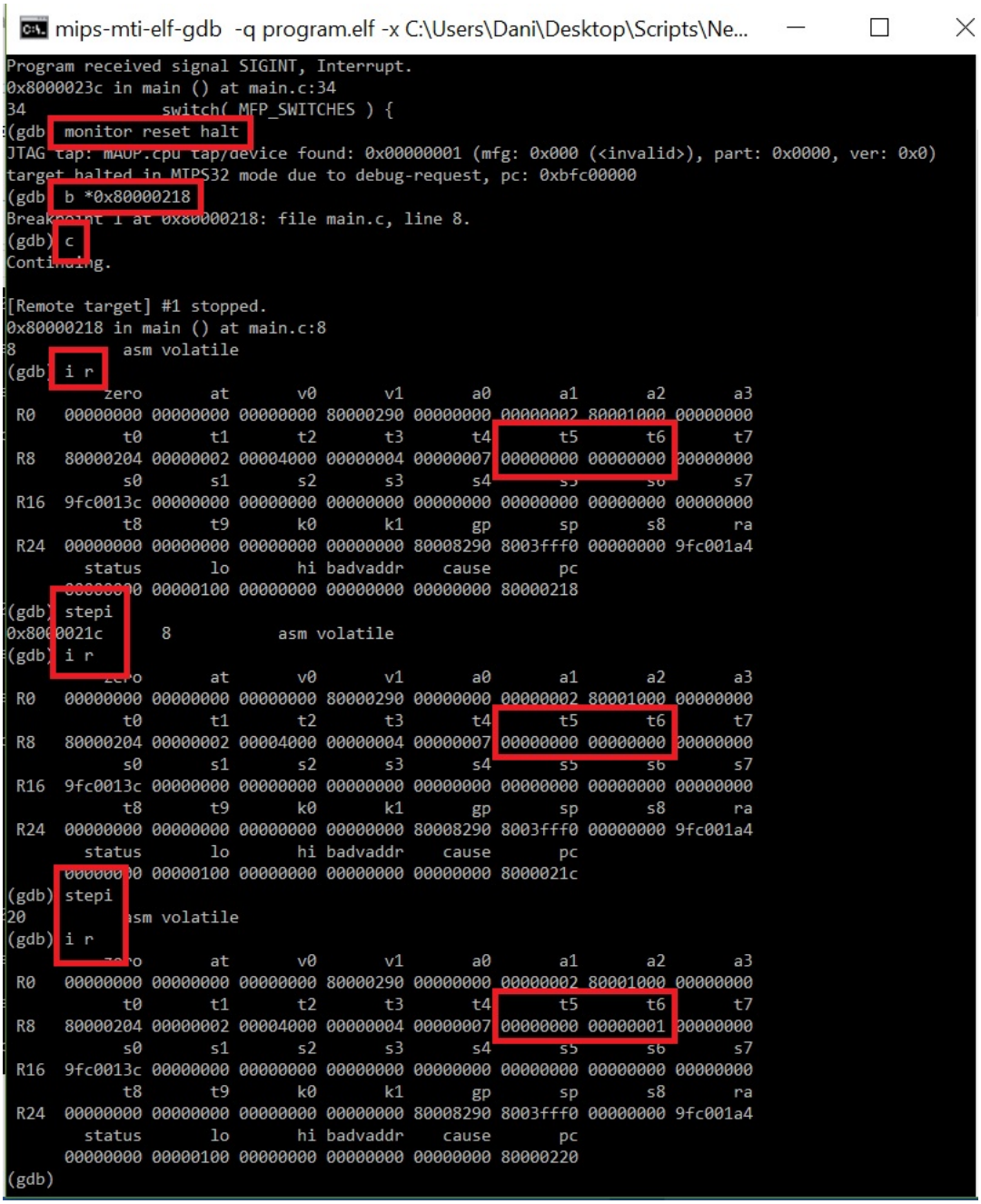
Observe that, in the fifth cycle rf13($t5)=0 and in the sixth cycle rf14($t6)=1.

Execution on the Board:

When the program is downloaded on the board, you should see on the 7-seg displays:

- Switches=0 → 7-seg displays=$t5, which in our example is 0x0
- Switches=1 → 7-seg displays=$t6 which in our example is 0x1
- Any other value for the switches → 7-seg displays=0x0

Then, when you debug the program following the steps stated in the document, you should observe the following:


mips-mti-elf-gdb -q program.elf -x C:\Users\Dani\Desktop\Scripts\Ne...
Program received signal SIGINT, Interrupt.
0x8000023c in main () at main.c:34
34 switch(MFP_SWITCHES) {
(gdb) monitor reset halt
JTAG tap: mAUP.cpu tap/device found: 0x00000001 (mfg: 0x000 (<invalid>), part: 0x0000, ver: 0x0)
target halted in MIPS32 mode due to debug-request, pc: 0xbfc00000
(gdb) b *0x80000218
Breakpoint 1 at 0x80000218: file main.c, line 8.
(gdb) c
Continuing.
[Remote target] #1 stopped.
0x80000218 in main () at main.c:8
8 asm volatile
(gdb) i r
zero at v0 v1 a0 a1 a2 a3
R0 00000000 00000000 00000000 80000290 00000000 00000002 80001000 00000000
t0 t1 t2 t3 t4 t5 t6 t7
R8 80000204 00000002 00004000 00000004 00000007 00000000 00000000 00000000
s0 s1 s2 s3 s4 s5 s6 s7
R16 9fc0013c 00000000 00000000 00000000 00000000 00000000 00000000 00000000
t8 t9 k0 k1 gp sp s8 ra
R24 00000000 00000000 00000000 00000000 80008290 8003fff0 00000000 9fc001a4
status lo hi badvaddr cause pc
00000000 00000100 00000000 00000000 00000000 80000218
(gdb) stepi
0x8000021c 8 asm volatile
(gdb) i r
zero at v0 v1 a0 a1 a2 a3
R0 00000000 00000000 00000000 80000290 00000000 00000002 80001000 00000000
t0 t1 t2 t3 t4 t5 t6 t7
R8 80000204 00000002 00004000 00000004 00000007 00000000 00000000 00000000
s0 s1 s2 s3 s4 s5 s6 s7
R16 9fc0013c 00000000 00000000 00000000 00000000 00000000 00000000 00000000
t8 t9 k0 k1 gp sp s8 ra
R24 00000000 00000000 00000000 00000000 80008290 8003fff0 00000000 9fc001a4
status lo hi badvaddr cause pc
00000000 00000100 00000000 00000000 00000000 8000021c
(gdb) stepi
20 asm volatile
(gdb) i r
zero at v0 v1 a0 a1 a2 a3
R0 00000000 00000000 00000000 80000290 00000000 00000002 80001000 00000000
t0 t1 t2 t3 t4 t5 t6 t7
R8 80000204 00000002 00004000 00000004 00000007 00000000 00000001 00000000
s0 s1 s2 s3 s4 s5 s6 s7
R16 9fc0013c 00000000 00000000 00000000 00000000 00000000 00000000 00000000
t8 t9 k0 k1 gp sp s8 ra
R24 00000000 00000000 00000000 00000000 80008290 8003fff0 00000000 9fc001a4
status lo hi badvaddr cause pc
00000000 00000100 00000000 00000000 00000000 80000220
(gdb)