* **EXPLANATION**:
  1. All changes related to the new instruction are tagged with comment: *// BEQC* in the soft-core. We must perform the following actions:
     + Annul the DS when the branch is not taken: There is a specific signal (***annul\_ds\_i***) which annuls the DS. For Branch Likely instructions, the DS is annulled when the condition for the branch is not met (!***mpc\_eqcond\_e***). For Branch Compact instructions, the DS is annulled when the condition for the branch is met (***mpc\_eqcond\_e***). Note that we only change the functionality of the BEQL instruction.

Change:

*assign annul\_ds\_i = (br\_likely\_e && mpc\_irval\_e && !mpc\_eqcond\_e) ||*

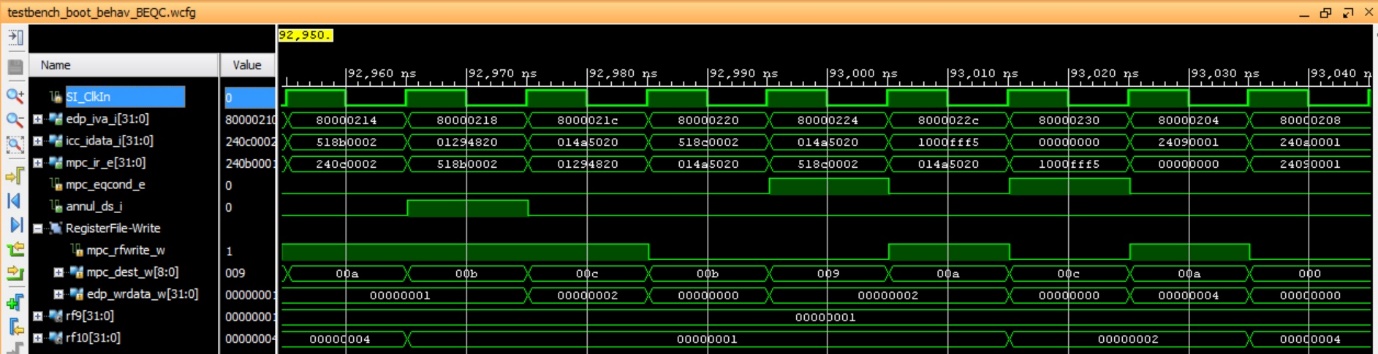
For:

*assign annul\_ds\_i = ((br\_likely\_e && mpc\_ir\_e[27:26]!=2'b00) && mpc\_irval\_e && !mpc\_eqcond\_e) || // BranchCompact*

*((br\_likely\_e && mpc\_ir\_e[27:26]==2'b00) && mpc\_irval\_e && mpc\_eqcond\_e) || // BranchCompact*

* **EXAMPLE - SIMULATION**:

**Results of the simulation on the original core (BEQL instruction):**



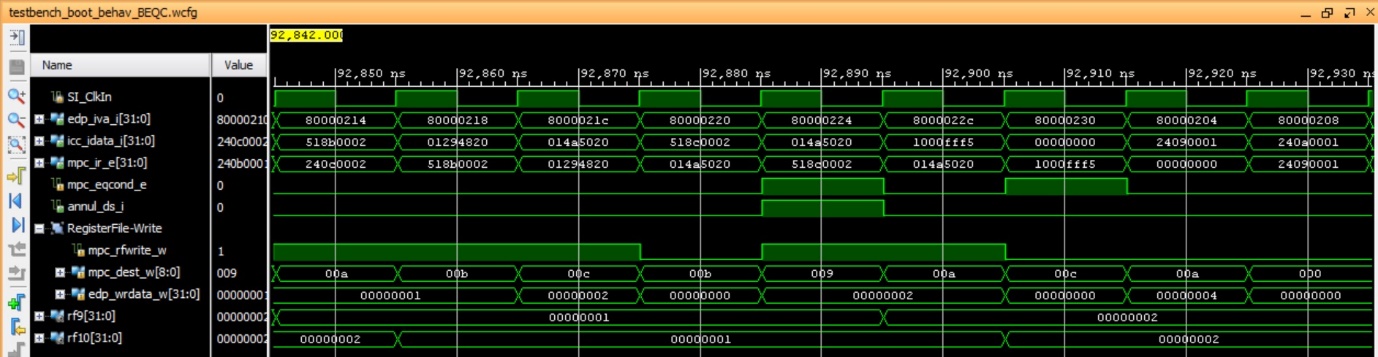
In the second cycle, the first beql is executed. The condition is not met (mpc\_eqcond\_e=0), thus the branch is not taken. In this case, the instruction in the branch delay slot must not be executed (annul\_ds\_i=1). Observe that, in cycle 5, the register file is not written.

In the fifth cycle, the second beql is executed. The condition is met (mpc\_eqcond\_e=1), thus the branch is taken. In this case, the instruction in the branch delay slot must be executed (annul\_ds\_i=0). Observe that, in the eighth cycle, the register file is written with the result of the instruction in the delay slot.

The result in registers $t1 and $t2 is:

* $t1(rf9)=1
* $t2(rf10)=4

**Results of the simulation on the modified core (BEQC instruction):**



In the second cycle, the first beql is executed. The condition is not met (mpc\_eqcond\_e=0), thus the branch is not taken. In this case, differently to the previous case, the instruction in the branch delay slot must be executed (annul\_ds\_i=0). Observe that, in cycle 5, the register file is written.

In the fifth cycle, the second beql is executed. The condition is met (mpc\_eqcond\_e=1), thus the branch is taken. In this case, differently to the previous case, the instruction in the branch delay slot must not be executed (annul\_ds\_i=1). Observe that, in the eighth cycle, the register file is not written.

The result in registers $t1 and $t2 is:

* $t1(rf9)=2
* $t2(rf10)=2
* **EXECUTION ON BOARD**:

**Results of the execution on the original core (BEQL instruction):**

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

* Switches=0 🡪 7-seg displays=$t1, which in our example is 0x1
* Switches=1 🡪 7-seg displays=$t2, which in our example is 0x4

**Results of the execution on the modified core (BEQC instruction):**

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

* Switches=0 🡪 7-seg displays=$t1, which in our example is 0x2
* Switches=1 🡪 7-seg displays=$t2, which in our example is 0x2

**Debug of the modified core:**

