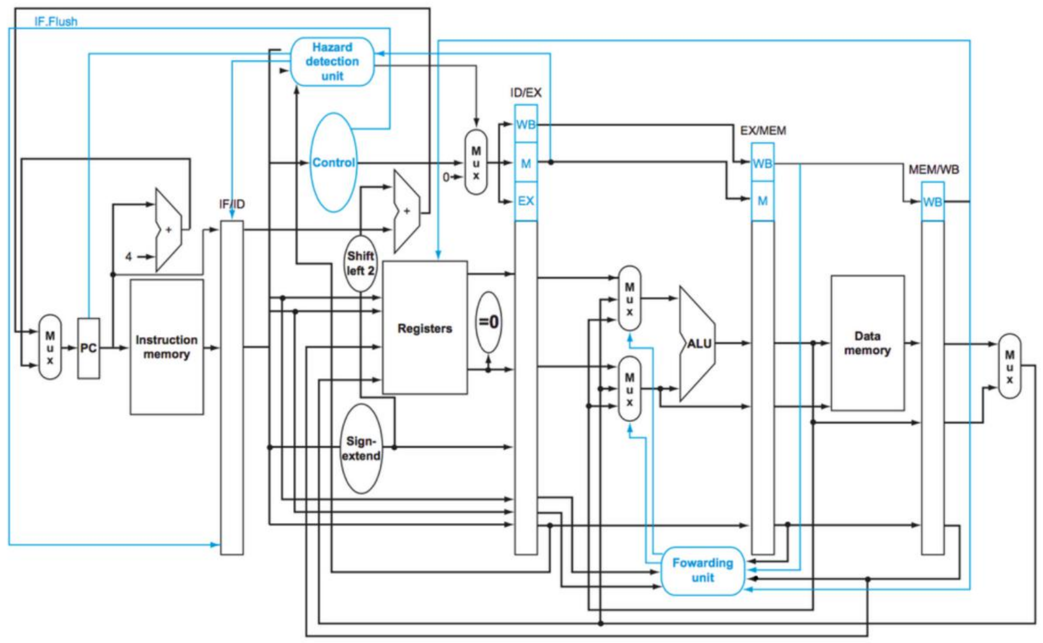
Report

For EE126 assignment6

The goal of this project is to design a pipeline version of a ARM processor that Overcome control hazards by resolving conditional/unconditional branches in ID and using flushing. A pipeline processor is a processor that carries out one instruction in 5 clock cycles, keep every part of the processor busy with current and latter instruction.

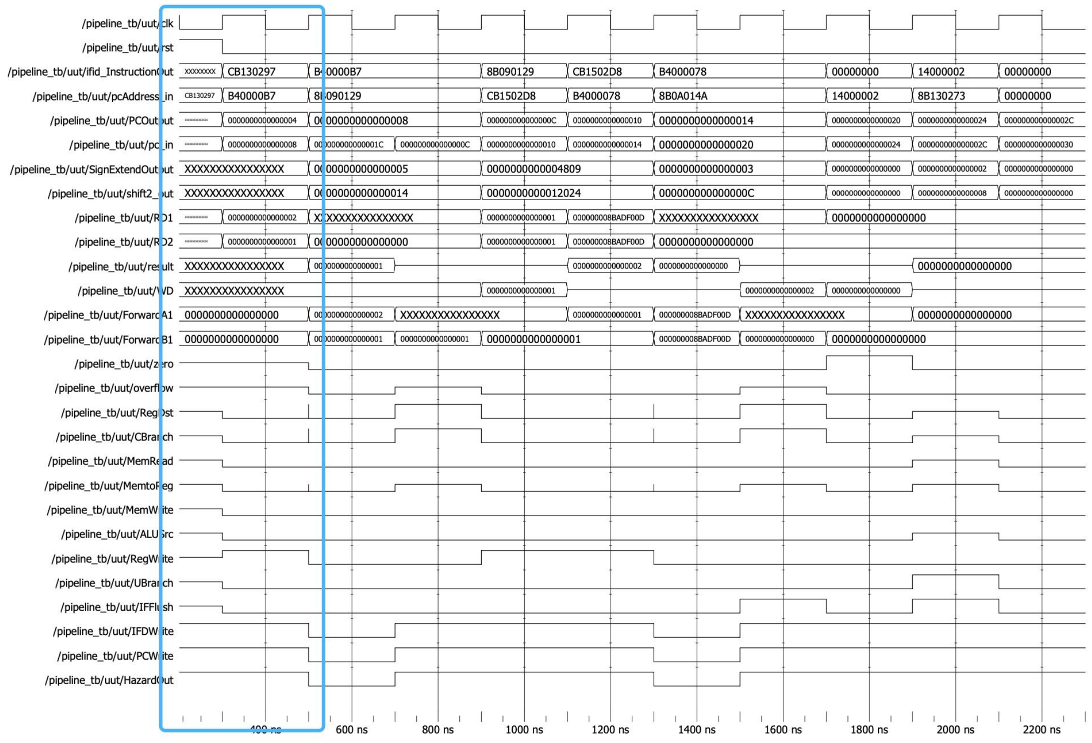
Provide waveforms—along with annotations, labels, and descriptions—that show the successful execution of the program defined later in this document. Be sure to:
Include the most relevant signals in your waveform, including those related to overcoming control hazards Point out the following
– Which instructions are executed, not executed, and which begin execution but are later "flushed" – When a conditional/unconditional branches is resolved
– When/why a branch 'prediction' is correct/incorrect
– When/why a flush occurs and what its effect is
Clearly show what pipeline-stage each instruction is in.



(Processor overview)

In the first CC(Clock Cycle), we set “reset” as ‘1’ to initiate whole processor.

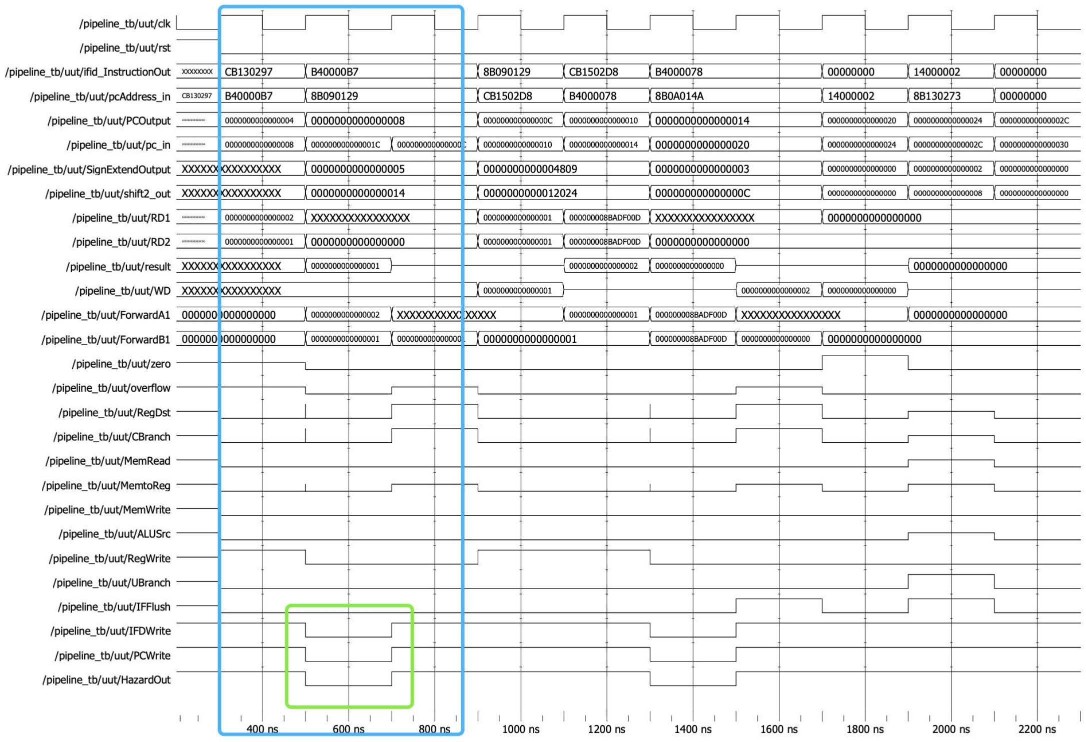
In the second CC, the processor begins to operate test program.
The first stage of SUB X23, X20, X19
Read the PC and find the corresponding 32 bits address and then store them into if/id.



(First two Clock Cycles)

In the third CC, the second instruction(CBZ) begins to operation like the behavior of first one.

The second stage of “SUB”: decode the operand, read data from register and generate control signals.



(The third Clock Cycle)

Combine the forth and fifth CC together to see what's going on under the influence of hazard unit.

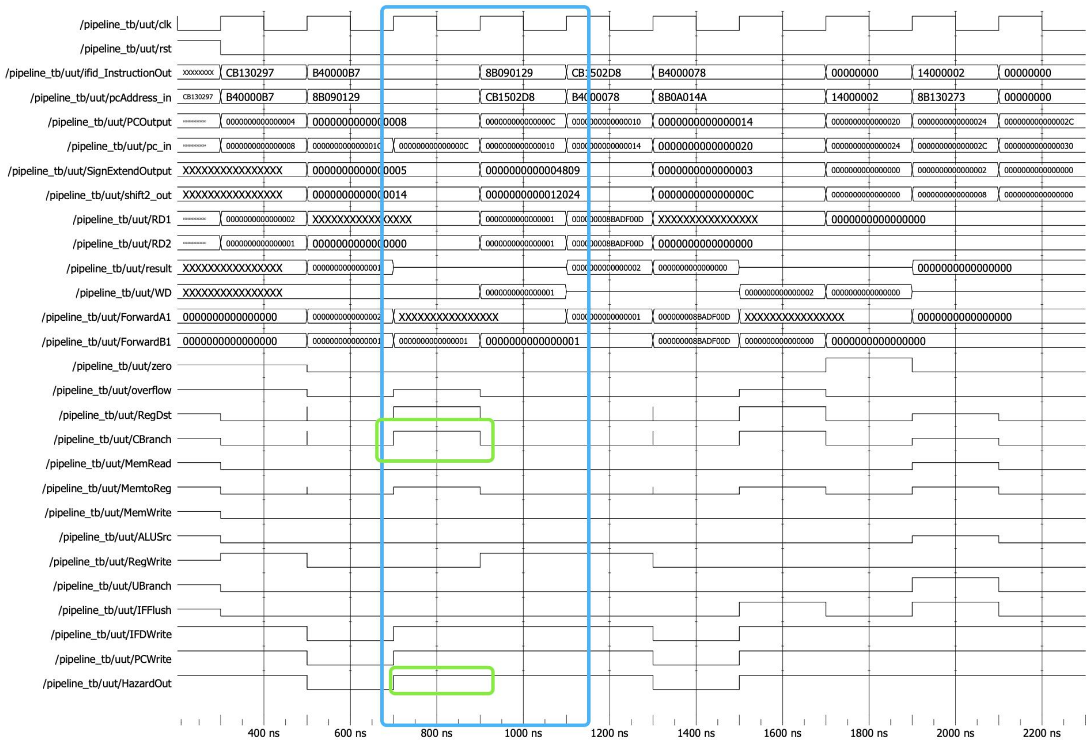
In the third stage of SUB, it starts to compute ALU result, and forth stage. In the meantime, the second instruction(CBZ) should be in ID stage, but because of the hazard unit detect the potential issue, so it stalls it for one cycle, and at the point of 900ns, the second instruction enter ID stage and take effect.

In the next cycle(900ns-1100ns),

The third instruction(ADD) enter if stage.

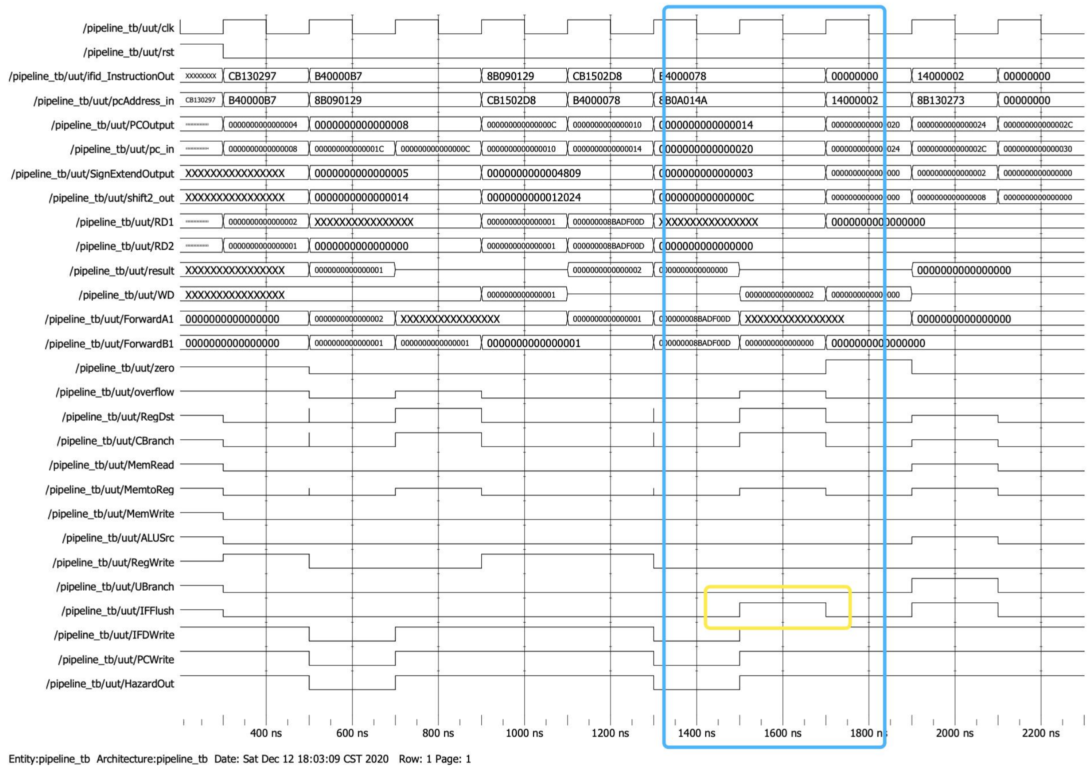
CBZ is in the ID stage, and figure out that RD(x23) is not equal to zero, so the it moves without any stall or flush.

SUB keeps running and enter MEM stage.



(The forth and fifth Clock Cycles)

In the eleventh CC,(from fifth to tenth work normally like assignment 5),
SUB is in the MEM(SUB X24, X22, X21),
NOP occupies EX,
CBZ enter ID, by computing finds out that X24 is equals to zero, branch process works.
Generate a signal to show the result of comparison then send to control to generate "IFFLUSH" to clean the next "ADD"(ADD X10, X10, X10) instruction.
New instruction "ADD"(ADD X10, X10, X10) is entering IF stage(will be cleaned in the next cycle).



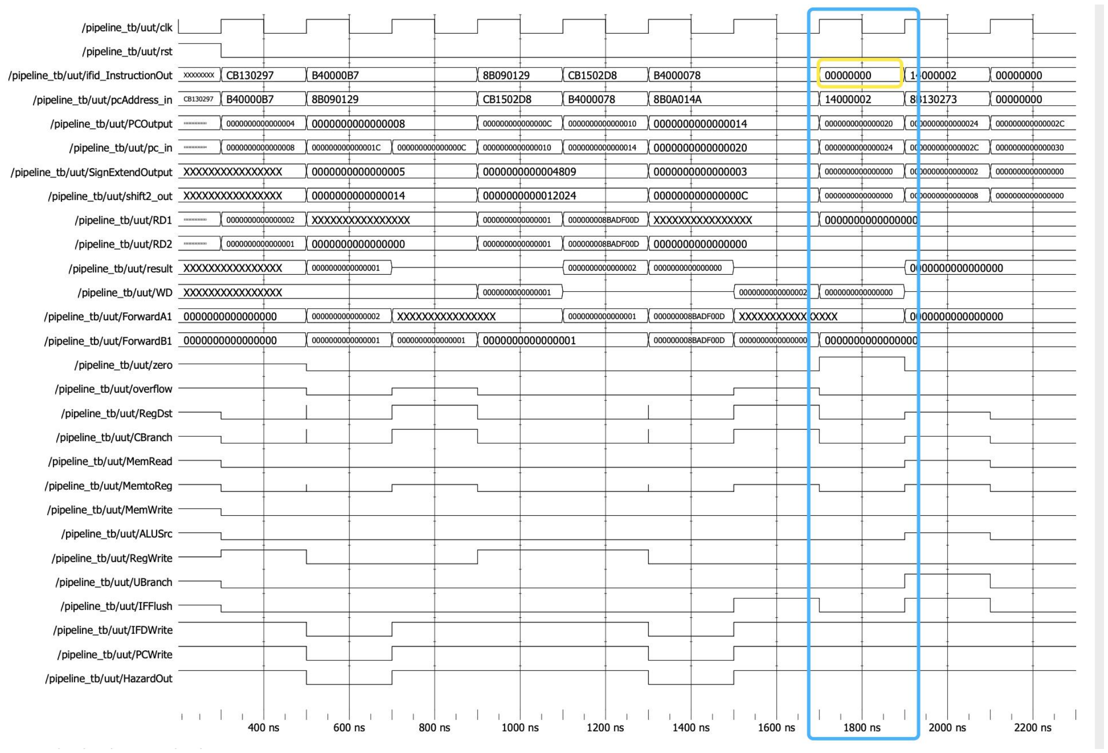
(The sixth Clock Cycle)

In the twelfth CC, the next cycle after branching,

CBZ is in the EX stage(finished),

The one “ADD”(ADD X10, X10, X10) should be in ID stage, while because of FLUSH generated in the eleventh cycle, now the ID stage is occupied by NOP.

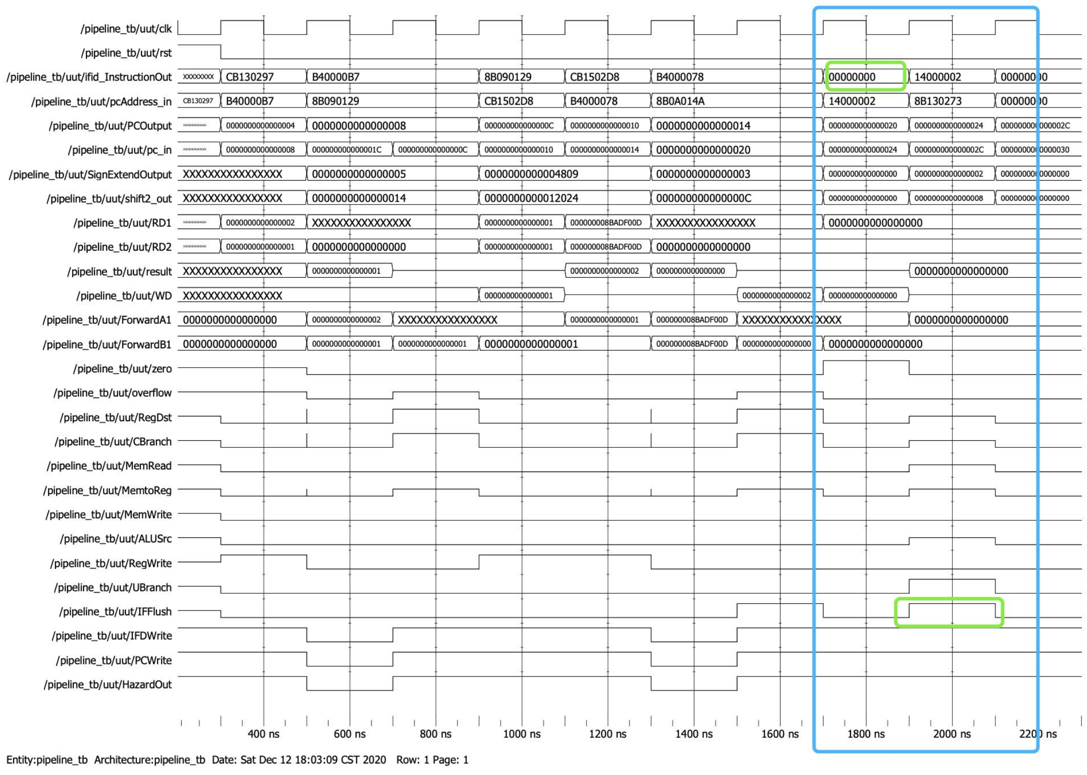
The newer instruction B 2 comes, stay in IF stage.



(The seventh Clock Cycle)

In the thirteenth and fourteenth CC,

Unconditional branch (B 2)is in the ID stage and take effect immediately.
Control generated FLUSH signal again to clean the next instruction which is in the IF stage.
In the fourteenth CC, the instruction NOP is executed(add x19, x19, x19 & add x20, x20, x20 are ignored).
The sequential cycle is keep working to finished all previous instructions.



(The seventh Clock Cycle)