Report

For EE126 assignment4

The goal of this project is to design a pipeline version of a ARM processor that can deal with all the instruction like “add”, “store” and “sub”.
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.Great attention should be paid to the time and signal control in pinpoint time.

The process of whole processor is as below:

First stage:
PC send the address 1. to find the current instruction from Instruction memory According to file inside the Imem. 2. directly to register “IF/ID”, and store there. The information will be used to 1.Control which is used to generate various signals to control different behavior or logistics of other components. 2.Register which is an very important one, very fast speed provide strong support for different instructions and operations. Register also use the internal register file to find specific value and transmit them to ALU for computing or sent address/value to Data memory for storing or loading. 3. Sign-extend which is used for extend immediate to 64-bit for further calculation.

Second stage:
Because all the instruction are competed with the below components, it’s very important to make them operate different at different times, control is the answer. It generate different control signals according to every individual instruction.
Register, at the same time, decode the instruction and send data to ALU for computing.
Sign-extend expand the immediate number to 64 bits to match the data coming from the register.
The control signals and register data are stored in ID/EX for the next cycle(Stage).

Third stage:
ALU get the data from Register, calculate it immediately. The operations like adding or subtracting are decided according to the value of ALU control which is also controlled by Control.
The adder is used to calculate destination of branch instruction.
The control signal used for WB and Data Memory are transmitted and stored in EX/MEM.

Forth stage:
Data memory find data internally according to ALU result which works as address here to load data and send back to Register. Surly, if storing operation happen, Data memory extract nothing but receive the data from Register and save it.
The data and control signals are also stored in MEM/WB for the last stage.

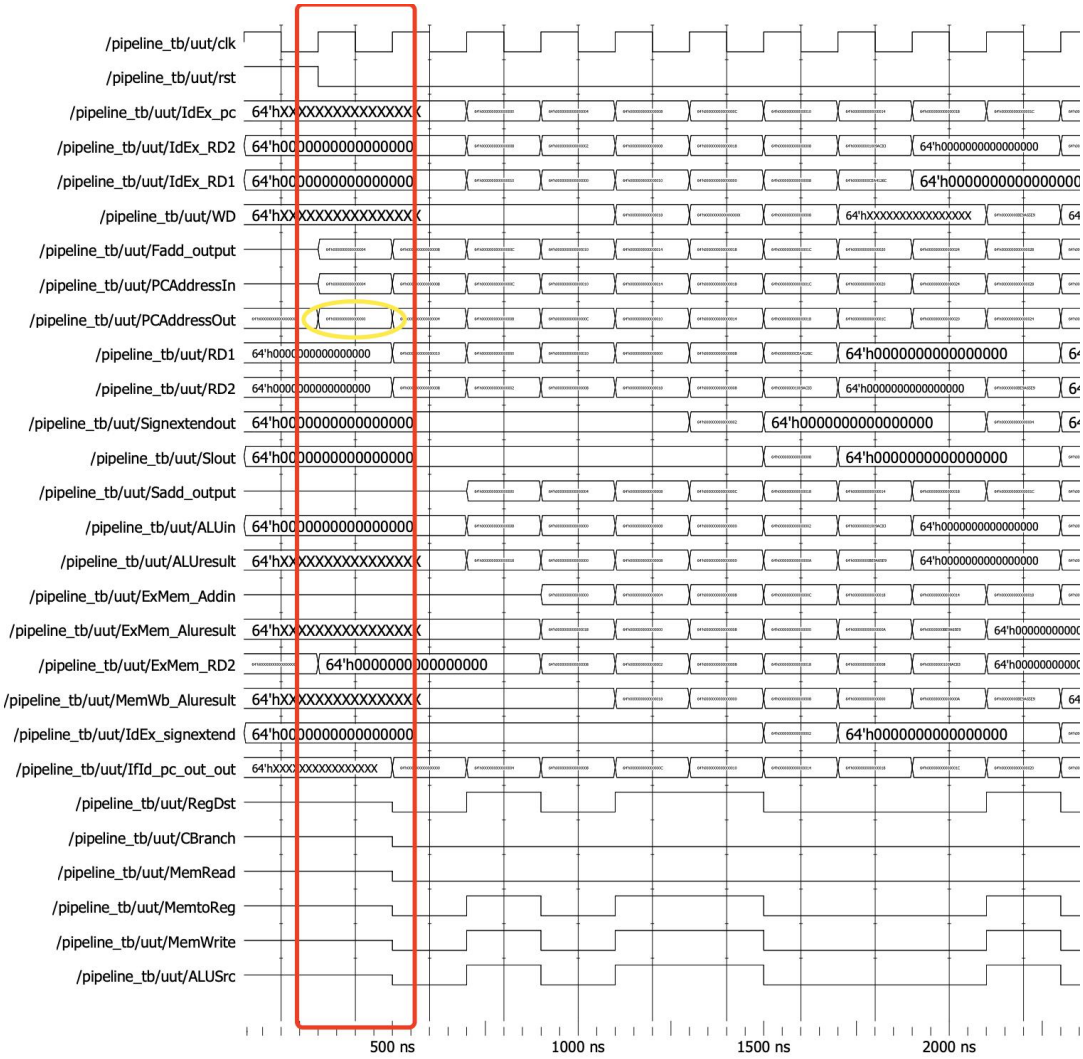
Fifth stage:
Determine which data(Read data or ALU result) should be send back.

Compare with single cycle processor, pipeline use extra five register to store temporary data, and depends on witch to separate different stages.

• Clearly demonstrate what is going on each cycle using waveform-annotations and descriptions.
• Include only the most relevant signals, such as the value of the PC, key control signals, any values read/written, etc.
• Show that correct output is stored in registers and data memory

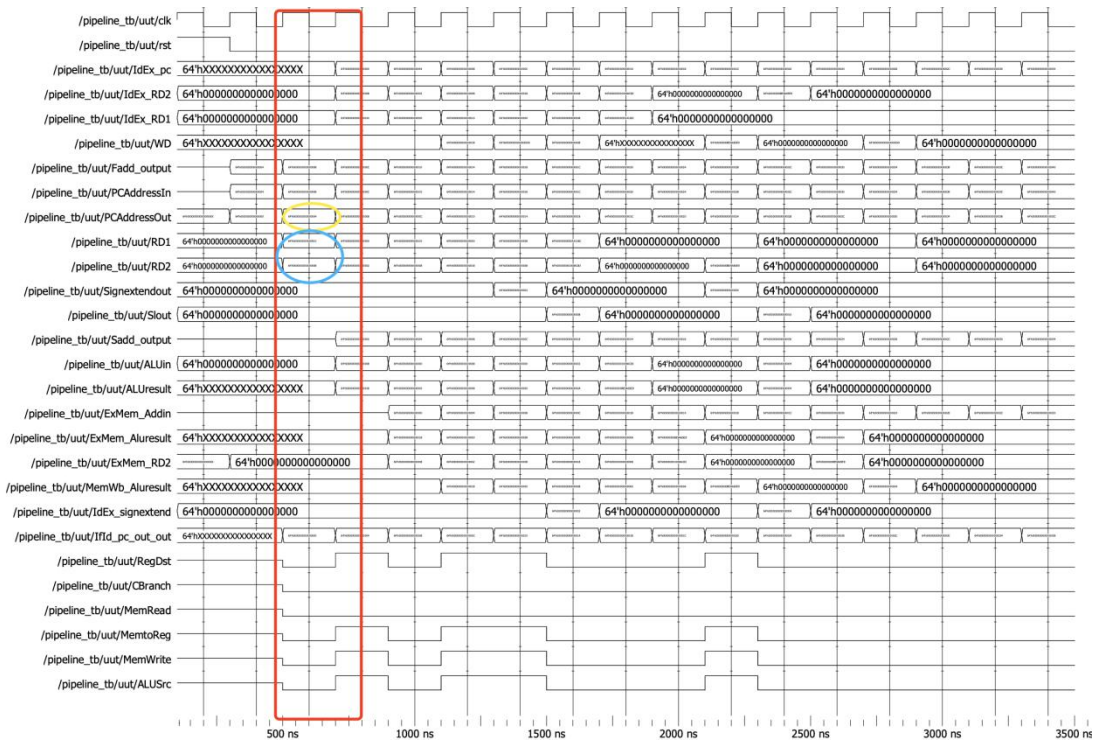
In the first CC(Clock Cycle), we set “reset” as ‘1’ to initiate whole processor.

In the second CC, the processor begins to operate test program.
The first stage of ADD X11, X9, X10:
Read the PC and find the corresponding 32 bits address and then store them into if/id.



(The second Clock Cycle)

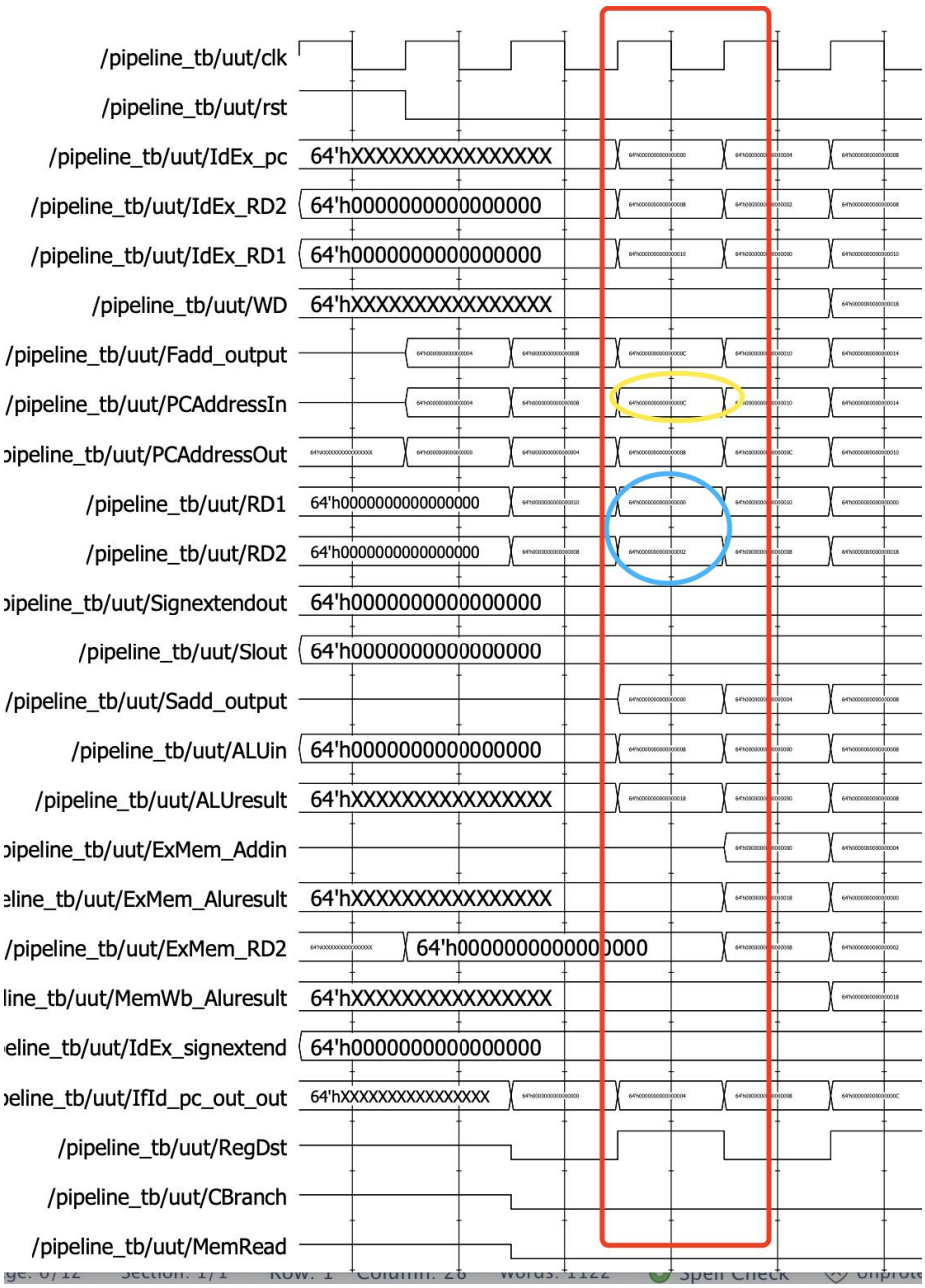
In the third CC, the second instruction begins to operation like the behavior of first one in the second CC(yellow circle shows the PC value of STUR).
The second stage of “ADD”:
Register find the value in x9(0x10) and x10(0x08) and send to ID/EX for storing.
Meanwhile, control resolute the value of each signals and storing in the ID/EX for next few cycles using.



(The third Clock Cycle)

In the forth CC, the third instruction begins to operate like the behavior of first one in the second CC(yellow circle shows the PC value of SUB).
The third stage of “ADD”:
Calculate the ALUresult as 0x18, and send it to EX/MEM.

The second stage of “STUR”:
The value of x11 is 0x02 which will be used to store in the memory’s address “0x00”.
This address is representative as immediate number, so it’s extended in sign-extend and then send ID/EX with the valuex11.

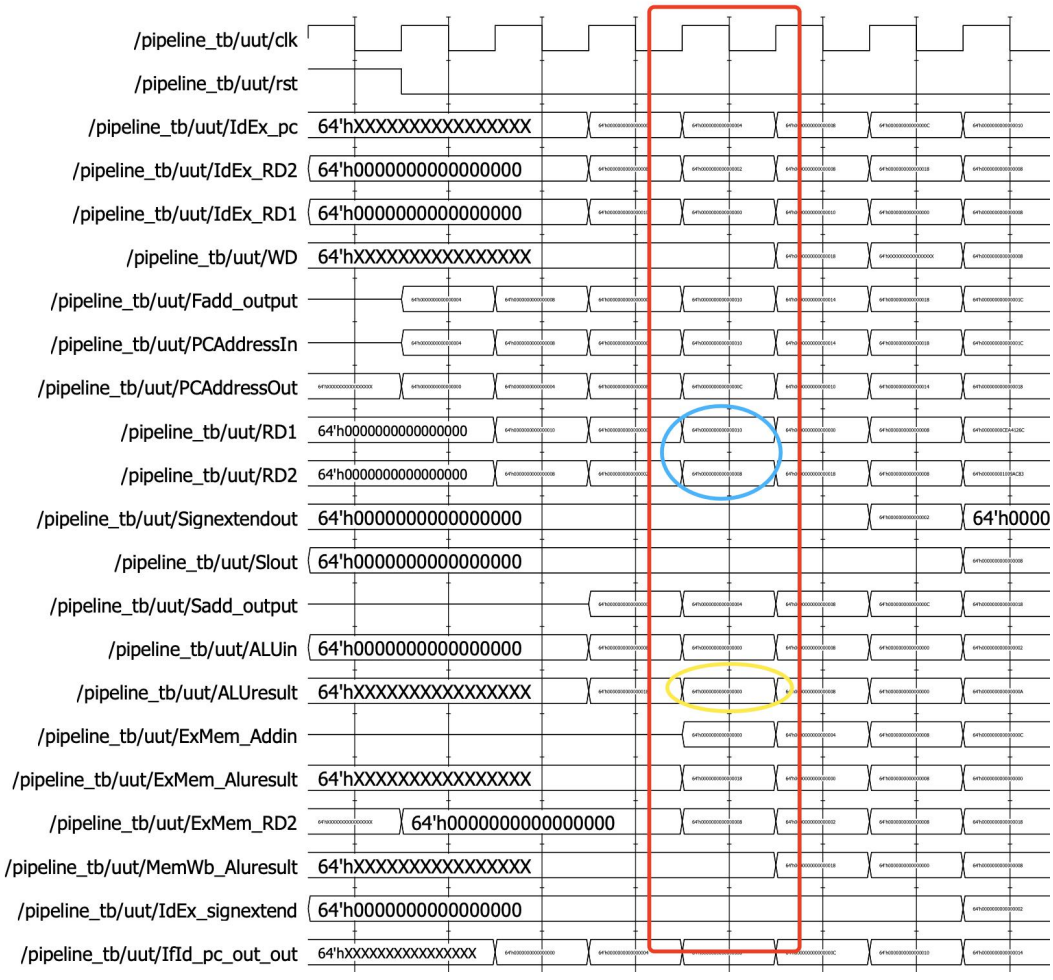


(The forth Clock Cycle)

In the fifth CC, the forth instruction begins to operate like the behavior of first one in the second CC.
The forth stage of “ADD”:
Because “ADD” is R-Type instruction, in this stage, data memory needn’t to be used. But because of the property of pipeline, it cannot send the value back to register like single cycle processor. As a result, the value of ALU result(0x18) is again sent to MEM/WB.

The third stage of “STUR”:
The x11 value is directly sent to EX/MEM for later storing, the address(which is 0 with yellow circle) of it in data memory is computed by ALU in this stage.

The second stage of “SUB”:
Register read the value of x9(0x10) and x10(0x08), send them to ID/EX.



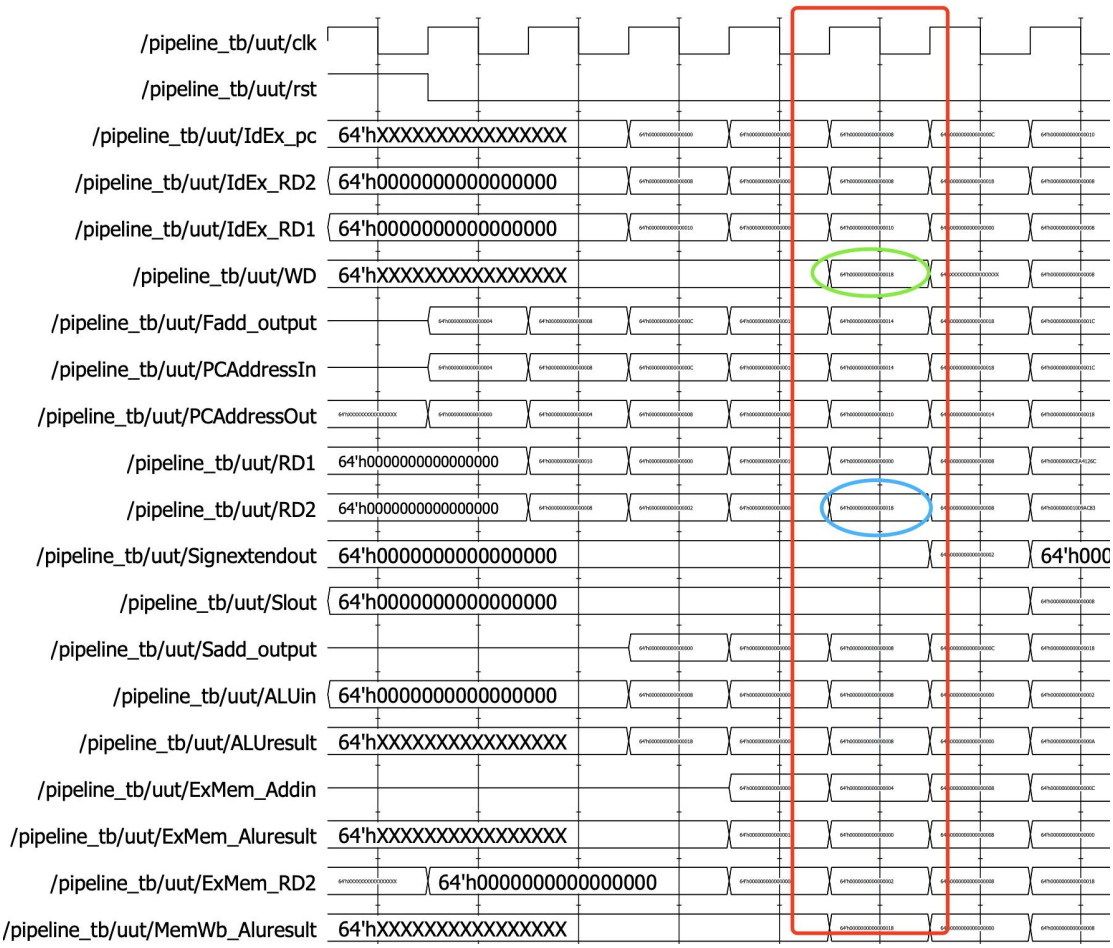
(The fifth Clock Cycle)

In the sixth CC, the fifth instruction begins to operate like the behavior of first one in the second CC.
The fifth stage of “ADD”:
The value in MEM/WB is sent back to WD and store the value in register.

The forth stage of “STUR”:
Save the value from EX/MEM to data memory.

The third stage of “SUB”:
The result of x9(0x10) and x10(0x08) subtraction is implemented which is 0x08. store it in EX/MEM

The second stage of “STUR”:
In this stage, the value x11 is read as 0x18 which was just stored in the fifth stage of “ADD”



(The sixth Clock Cycle)

In the seventh CC, the sixth instruction begins to operate like the behavior of first one in the second CC. At this time, the first instruction “ADD” is implemented.
The fifth stage of “STUR”:
There is in fact no any actual behavior of instruction “STUR” in this stage, due to it’s finished within 4 stages. But in order to make sure every instruction share the same time slot(steps), at least in the process perspective, so it is here.

The forth stage of “SUB”:
Because “SUB” is R-Type instruction, in this stage, data memory needn’t to be used. So the value of ALU result is sent to MEM.WB for later being feedback to register.

The third stage of “STUR”:
The destination value is directly sent to EX/MEM for later storing in data memory, the address of it is also computed by ALU in this stage.

The second stage of “STUR”:
In this stage, the value x11 is read, and the address which is immediate is sent to sign-extend for processing to 64-bit.

Summary of pipeline:

In the above 7 cycles, showing the whole process of pipelining.
From the every beginning, the first instruction “ADD” enters and the second, third instruction begins sequentially, Until the fifth instruction begins to process, all stages are filled, in other word, All parts of pipeline is fully occupied. In this moment, the efficiency of the processor is max.
After that time, the first instruction quit because it is finished , the next instruction, the sixth one comes up. So the pipeline keeps the full occupied status.

• Point out where critical pipeline events occur, such as writing an incorrect value due to an unhandled hazard, overcoming a hazard with a NOP, etc.

Please see the below:

ADD X11, X9, X10
//Working perfectly normal as it's the first instruction.

STUR X11, [XZR,0]
//X11 is the stale data, because when this instruction happen, the value of X11 is not available. To avoid it, there should be at least 2 NOP between ADD and STUR.

SUB X12, X9, X10
//Working perfectly normal.

STUR X11, [XZR,0]
//Correct X11 is read. So correct value stored.

STUR X12, [X12,8]
//Incorrect X12 value, which can be handled by adding a NOP before it.

STUR X12, [X12,8]
//Correct.

ORR X21, X19, X20
//Correct.

nop
nop
//These two instructions used to overcome data hazard of x21.

STUR X21, [XZR,16]
Because of previous two NOPS, the value of X21 is correctly read, the result of storing is also right.

nop
//the following nops is used to prevent any data hazard that may occur later.