COL 216 Assignment 2 Stage 2

Manshi Sagar - 2020CS50429

19 February 2022

1 Introduction

In this assignment we aim to design hardware for implementing a processor that can execute a subset of ARM instructions .The designs are to be expressed in VHDL and then simulated and synthesized.
In Stage 1, we design the skeleton of four modules: ALU , Register File, Program Memory and Data memory.
In Stage 2, we put together the four modules designed in Stage 1 to form a simple processor which can execute the following subset of ARM instructions with limited variants/features. add, sub, cmp, mov, ldr, str, beq, bne, b

2 Program Files

- 8 .vhd files for components: alu, program memory(pm), register file, data memory(dm), pprogram counter(pc), decoder, flag, conditionchecker
- 8 .vhd files for testbench of each component
- Report file
- Gluecode

3 Components

The new components designed in this stage are:

- Program Counter with next address logic
- Flags and associated circuit
- Condition Checker
- Instruction Decoder

The implementation and structure of modules designed in Stage 1 remains same as described in Stage 1 Report

3.1 Program Counter

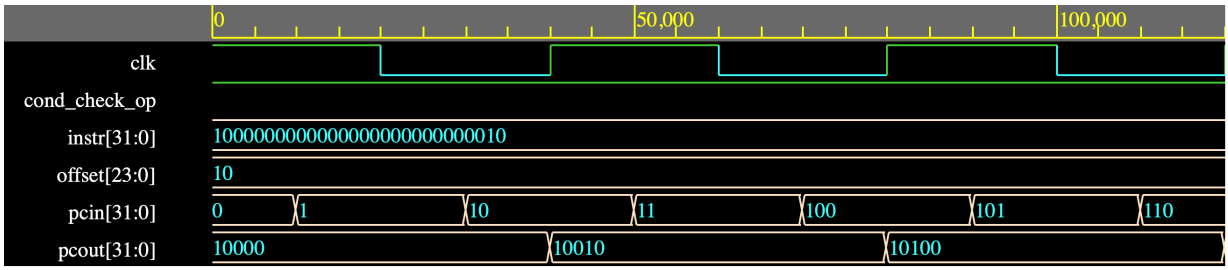
This component increments the pc address by 4 in each rising edge of clock and stores it in a temporary variable.
Then it checks if the given instruction is a branch instruction using the F field ie instruction(27 downto 26). If F= 10 , then the offset given in input is shifted by two bits and added to the temporary variable + 4.
Lastly, pc-out is assigned the value of temporary variable.
Inputs are:

- pc-in: pc input address - std-logic-vector(31 downto 0)
- clock:

- cond-checker-output: boolean input from condition checker - std-logic
- instruction: 32 bit instruction from program memory - std-logic-vector(31 downto 0)
- offset: offset to add for branch instruction - std-logic-vector(23 downto 0);

Outputs are:

- pc-out: pc output address - std-logic-vector(31 downto 0)



3.2 Flags

This component sets the flags according to opcode of instruction. If instruction is of test or compare class, then flags are always set. But if instruction is of DT class , then flags are modified if S bit = '1'. In this stage, from the intruction set that we have to implement, only cmp instruction will modify flags (all flags). Inputs are:

- instr-class: type of instruction (DP, DT, BRN or none)
- DP-subclass : subclass of DP instruction (arithmetic, logical, compare, test or none)
- clock:
- s-bit: 1 if flags need to be modified and 0 if flags need not be modified
- c-out : 1 bit carry out from ALU
- result: 32 bit result from ALU
- msb-oper1: 1 bit MSB of operand 1 of ALU
- msb-oper2: 1 bit MSB of operand 2 of ALU

Outputs are:

- Z, N, C, V flags

3.3 Condition Checker

This component checks the appropriate flags according to the condition field and returns a boolean value (std-logic) = 1 if the instruction is to be executed and 0 if not.
In this component we only check the following conditions:

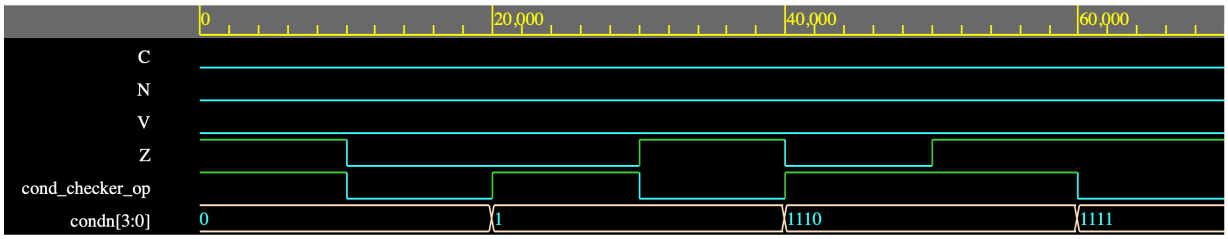
- Condition field = "0000" = eq : if Z flag = 1 then condition-checker-output=1
- Condition field = "0001" = eq : if Z flag = 0 then condition-checker-output=1
- Condition field = "1110" = no flags need to be checked and condition-checker-output=1

Inputs are:

- Z, N, C, V flags
- condition: instruction(3 downto 0) - 4 bit condition field

Outputs are:

- condition-checker-output: 1 bit boolean value



3.4 Instruction Decoder

(uploaded on moodle)
This component extracts fields from the instruction and classifies them into

- Class / type of instruction (DP, DT, BRN or none)
- DP-subclass: subclass of DP instruction (arithmetic, logical, compare, test or none)
- DP-operand-src-type: type of second operand for ALU (register or immediate value of constant)
- Operation type (andop, eor, sub, rsb, add, adc, sbc, rsc, tst, teq, cmp, cmn, orr, mov, bic, mvn)
- load-store instruction
- DT-offset-sign: sign of DT offset
- condition field : std-logic-vector(3 downto 0)

Inputs are:

- instruction : 32 bit instruction

Outputs are:

- instr-class
- operation
- DP-subclass
- DP-operand-src
- load-store
- DT-offset-sign
- condition