Control Unit DRAC specification version v0.1

Rubén Langarita Benítez

March 13, 2020

Contents

1 General purpose of the module

Person in Charge; Guillem López

The Control Unit module is placed in the datapath module alongside the stages of the pipeline. It is responsible of producing the necessary signals to control the stages and the caches.

This module receives signals from the different stages and the caches, and it outputs the following signals:

- Stall the stages when something blocks the pipeline.
- Flush the stages when something invalidates the instructions in the pipeline.
- Select the next PC to be fetched.
- Invalidate the cache requests.
- Write into the register bank.

2 Design placement

The Control Unit module is placed inside the datapath module. There is only one instance per core. We can differentiate two type of signals: the ones that are connected to the stages and the ones that are connected to the caches.

3 Parameters

All parameters, enums and types used in this module are defined in drac_pkg or risc_pkg.

4 INTERFACE

Signal name	Width	Type	Description
clk_i	1	in	Clock
rstn_i	1	in	Reset
valid_fetch	1	in	The instruction in Fetch is valid
id_cu_i	struct	in	Signals from Decode stage
rr_cu_i	struct	in	Signals from Read Register stage
exe_cu_i	struct	in	Signals from Execution stage
wb_cu_i	struct	in	Signals from Write Back stage
csr_cu_i	struct	in	Signals from the CSRs
correct_branch_pred_i	1	in	A branch instruction in the Execution stage has been predicted correctly
pipeline_ctrl_o	struct	out	Signals to stall the stages and to select PC in case of jump
pipeline_flush_o	struct	out	Signals to flush the stages
cu_if_o	struct	out	Select next PC: PC, PC+4 or jump
invalidate_icache_o	1	out	Invalidate ICache request
invalidate_buffer_o	1	out	Invalidate the ICache buffer
cu_rr_o	struct	out	Signals to Read Register stage

5 RESET BEHAVIOUR

Since this module has no sequential logic, no clock or reset signals are needed.

6 WHAT COULD NOT HAPPEN

The Control Unit is a combinational block and accept all combinations of the inputs. The module should handle all the possible combinations of inputs given priority to some of them. For example, if the execution stage is blocked and the write back stage commits a mispredicted branch, it should give priority to the write back stage.

7 BEHAVIOUR

7.1 DESCRIPTION

In this section we describe the behaviour of the Control Unit module. For each output signal, the behaviour should be the following:

- **pipeline_ctrl_o:** This structure has 6 signals. 5 of them to control when the stages should be stalled (*stall_{if,id,rr,exe,wb}*), and another one to select the next PC in case

of jump (*sel_addr_i*) from the following options:

  - From the CSRs. An exception has been produced.
  - From the execution stage. A branch misprediction has been produced.
  - From the decode stage. A JAL instruction has been executed.

- **pipeline_flush_o:** This structure has 5 signals. Each signals controls if a stage should be flushed (*flush_{if,id,rr,exe,wb}*).
- **cu_if_o:** From this structure only one signal is used: *next_pc*. It controls the multiplexer that selects which PC should be selected in the fetch stage:
  - Same PC as the last cycle. The fetch stage is stalled.
  - Current PC plus 4. The fetch stage is not stalled.
  - Jump to another address. Select PC from *pipeline_ctrl_o.sel_addr_if*.
- **invalidate_icache_o:** Invalidate ICache request.
- **invalidate_buffer_o:** Invalidate the ICache buffer.
- **cu_rr_o:** Write the register bank, when the instruction in the write back stage wants to do it.

8 SPECIAL CASES, CORNER CASES

The Control Unit is a combinational block, it accepts all possible inputs, and it has not special cases.