Instruction decoding

This document is the reference one about the instruction decoding module included in the miniMIPS core. It contains every details needed to better understand the sources of the module and of course to understand the functionning way of the core. The reason of this module is also given.

The instruction formats

The goal of this stage is to decode an instruction in order to produce the set of control and data signals for the next stages. The way of functionning is closely related to the format of the processor miniMIPS instructions.

Basically, three types of formats exist. In each format, different fields are defined. The table, following, presents the formats.

Formats	Bits 31 - 26	Bits 25 - 21	Bits 20 - 16	Bits 15 - 11	Bits 10 - 6	Bits 5 - 0
Registers	op	rs	rt	rd	shamt	funct
Immediate	op	rs	rt	imm		
Jump	op	address				

Table 1 : Instruction format

The signification of the different fields are as following :

- The field op contains the type of instructions.
- The fields rs, rt and rd define the address in the banks of registers (system coprocessor or the GPRs). Those fields are the operands of the instruction.
- The field shamt contain the shift of a shifter operation.
- The field funct permits to define a type of registers instruction.
- The field imm contains an immediate value as an operand.
- The field address contains the address where to jump.

The implementation of the decoding

In the current miniMIPS implementation, there are more than fifty instructions. To permit this stage to decode each instruction, a table is defiend containing the different control signals according to the op and funct fields. The table, following, contains the signification of each entry of the instruction table.

Entry	Signification
op_mode	The signal op_mode defines the type of operations of the instruction. There are four different operation modes. It is defined in instruction by the bits 31 to 26. The modes are : ● OP_SPECIAL : those instructions work in general with 3 registers. ● OP_REGIMM : those instructions are branch instructions which compare a register to zero, but not all those instructions are OP_REGIMM. ● OP_COPo : those instructions are instructions working specifically on the coprocessor system. ● OP_NORMAL : all other instructions are in that mode. By determining the mode of operation, the op_code signal can be decoded in the instruction as the op_code is in different place in the instruciton according to the op_mode. With that two signals, the instruction can be found in the table.
op_code	The pair of signals op_code and op_mode defines the instruction in the table.
bra	The signal bra defines if the instruction is branch one.
link	The signal link defines a branch with a link. That means that the return address is saved in a register.
code_ual	The signal code_ual defines the type of operations to execute in the alu.
op_mem	The signal op_mem defines the instruction is a memory operation.
r_w	If the instruction is a memory operation, the signal r_w defines a read or write access.
mode	For a branch instruction, defines if the address must be calculated with the current PC or the first operand of the alu.

Entry	Signification
off_sel	The off_sel signal defines the calculation of the offset for an instruction needing one. Four ways are possible to calculate an offset : ● OFS_PCTRL : the four upper bits of the PC is concatenated with the address field (aligned on a word). ● OFS_NULL : no offset calculated. ● OFS_SESH : the imm field is signed extended and aligned on a word. ● OFS_SEXT : the imm field is signed extended
exc_cause	This signal defines an unconditionnal type of exception.
cop_org1	This signal defines the origin of the first register : general purpose registers or coprocessor system registers.
cop_org2	This signal defines the origin of the second register : general purpose registers or coprocessor system registers.
cs_imm1	This signal defines if the operand 1 is an immediate value or a register value.
cs_imm2	This signal defines if the operand 2 is an immediate value or a register value.
imm1_sel	When the operand 1 is an immediate value, imm1_sel selects zero as operand or the field shamt.
imm2_sel	When the operand 2 is an immediate value, imm2_sel selects the signed extended or not field imm as operand.
level	The signal level defines the stage where the result of the instruction is available.
ecr_reg	Defines if the instruction needs to write a register. This information is necessary to determine the data hazards.
bank_des	Defines which bank of register is selected for the writing of the register.
des_sel	Defines which field of the register contains the register address : ● D_RT : for the field Rd ● D_RS : for the field Rs ● D_31 : for the instructions with link which implicitly use the GPR 31.

Table 2 : Instruction record

Once the instruction decoded, the stage sends the data and control signals to the next stage (Execution stage). Differents possibilities exist :

- if a reset appears, a nop instruction is sent.
- if the signal stop_all is asserted then nothing happened and the last clocked informations are keeped. That signal appears when a data read in memory is not accessible at the moment.
- if a stop_di (data hazards) or a clear (bad branch prediction) signal occurs then there is also a nop instruction on the outputs.
- In the other cases, the current decoded instruction is provided.