```
typedef struct packed {
    logic [31:0] instruction,
    rv32i_opcode opcode,

    alu_ops aluop,
    cmpop,

    alumux1_sel,
    alumux2_sel,
    regfilemux_sel,
    cmpmux_sel,
    rdmux_sel,

    logic load_regfile,

    logic data_mem_read,
    logic data_mem_write,
    logic [3:0] mem_byte_enable


} rv32i_control_word;
```

All Control Signals:

- Instruction Fetch
    - Determined by hardware (pcmux_sel)
- Instruction Decode
    - none
- Execute signals:
    - Alumux1_sel, alumux2_sel- used to determine the alu operand
    - Rdmux_sel- used to determine the correct destination register encoding from Instruction based on instruction
    - cmpmux_sel- used to determine operand for cmp unit
    - Cmpop - used to determine the type of operation for cmp unit
    - aluop - used to determine the type of operation for alu unit
- Memory signals:
    - data_mem_write - used to signal memory write
    - data_mem_read - used to signal memory read

    - Control_word - used to check if it was branch instruction, in which case it will change pcmux_sel
    - Mem_byte_enable - tells memory which bytes to write
- Write Back signals:
    - regfilemux_sel - used to determine the data that is going to be written to a register in the regfile. Can be output of alu, output of comparison unit, or output of mem

Opcode / Control Signals

	aluop	cmpop	alumux1_sel	alumux2_sel	regfilemux_sel	cmpmux_sel	load_regfile	data_mem_read	data_mem_write
op_lui	funct3	funct3	rs1_out	i_imm	u_imm	rs2_out	1	0	0
op_auipc	alu_add	funct3	pc_out	u_imm	alu_out	rs2_out	1	0	0
op_jal	alu_add	funct3	pc_out	j_imm	alu_out	rs2_out	0	0	0
op_jalr	alu_add	funct3	rs1_out	i_imm	pc_plus4	rs2_out	1	0	0
op_br	alu_add	funct3	pc_out	b_imm	alu_out	rs2_out	0	0	0
op_load	alu_add	funct3	rs1_out	i_imm	funct3*	rs2_out	1	1	0
op_store	funct3	funct3	rs1_out	s_imm	alu_out	rs2_out	0	0	1
op_imm	funct3*	funct3*	rs1_out	i_imm	alu_out	funct3*	1	0	0
op_reg	funct3*	funct3*	rs1_out	rs2_out	alu_out	rs2_out	1	0	0
op_csr	funct3	funct3	rs1_out	i_imm	alu_out	rs2_out	0	0	0

NOTE: many signals are filled in with the default value- this means some signals may be unused, but are here for completion

NOTE: some signals like funct3* means the signal is based off funct3. Often this means the default is a funct3 cast (like `cmpop = branch_funct3_t'(funct3)` and `aluop = alu_ops'(funct3)` but there are cases dependent on the funct3 which may change this value. The logic for this is shown below.

op_load

load_funct3	regfilemux_sel
lb	lb
lbu	lbu
lh	lh
lhu	lhu
lw	lw
default	lw

op_imm

funct3	aluop	cmpop	regfilemux_sel	cmpmux_sel
slt	funct3	blt	br_en	i_imm
sltu	funct3	bltu	br_en	i_imm
sr	if(funct7[5] == 1'b1) aluop = alu_sra;	funct3	alu_out	rs2_out
default	funct3	funct3	alu_out	rs2_out

op_reg

funct3	aluop	cmpop	regfilemux_sel

add	if(funct7[5] == 1'b1)
	  aluop = alu_sub;
	else
	  aluop = alu_add;
		funct3	alut_out

sr	if(funct7[5] == 1'b1)
	  aluop = alu_sra;
	else
	  aluop = alu_srl;
		funct3	alu_out

slt	funct3	blt	br_en
sltu	funct3	bltu	br_en
default	funct3	funct3	alu_out