1.2.8 Computational Unit

The essence of the operation of the computational unit is captured in the block diagram on page 28. Some helpful nomenclature is given below.

Data Registers and zero flag

The 4-bit registers referred to as data registers are $x_0,\ x_1,\ y_0,\ y_1,\ o\_reg,\ m,\ i$, and dm. The r register is called a result register and the zero flag is considered to be an extension of the result register.

Registers $x_0,\ x_1,\ y_0,\ y_1,\ o\_reg,\ m$, and i are synchronously cleared when `sync_reset` is high, `but the sync_reset input to the computational unit does not do this.` The reset action is set up by the instruction decoder. It enables all registers and forces the source register select multiplexer to select 4'd10 while `sync_reset` is high.

The `sync_reset` input to the computational unit only affects the ALU circuit. While `sync_reset` is 1'b1 the ALU outputs `alu_out` and `alu_out_eq_0` are forced to 4'H0 and 1'b1, respectively. This action will set the zero flag and clear the r register as the enables for both will be 1'b1 while `sync_reset` is 1'b1.

index Register

One of the data registers is also used as an index register. The index register is register i. It is a post "auto increment" index register. It behaves exactly like the other registers for every instruction that writes to register i. It behaves differently than the other registers for instructions that read or write data memory, with one exception, which is the "move data memory to register i" instruction. For all instructions where data memory is read or written, except the instruction mentioned above, register i is to be written with $i+m$, i.e. $i = i+m$, on the clock edge that executes the instruction. Obviously register i must be written with dm on the "move data memory to register i" instruction.

Offset Register

The m register is a general register that is also used as the offset register. Register m determines the size of the auto-increment of the i register.

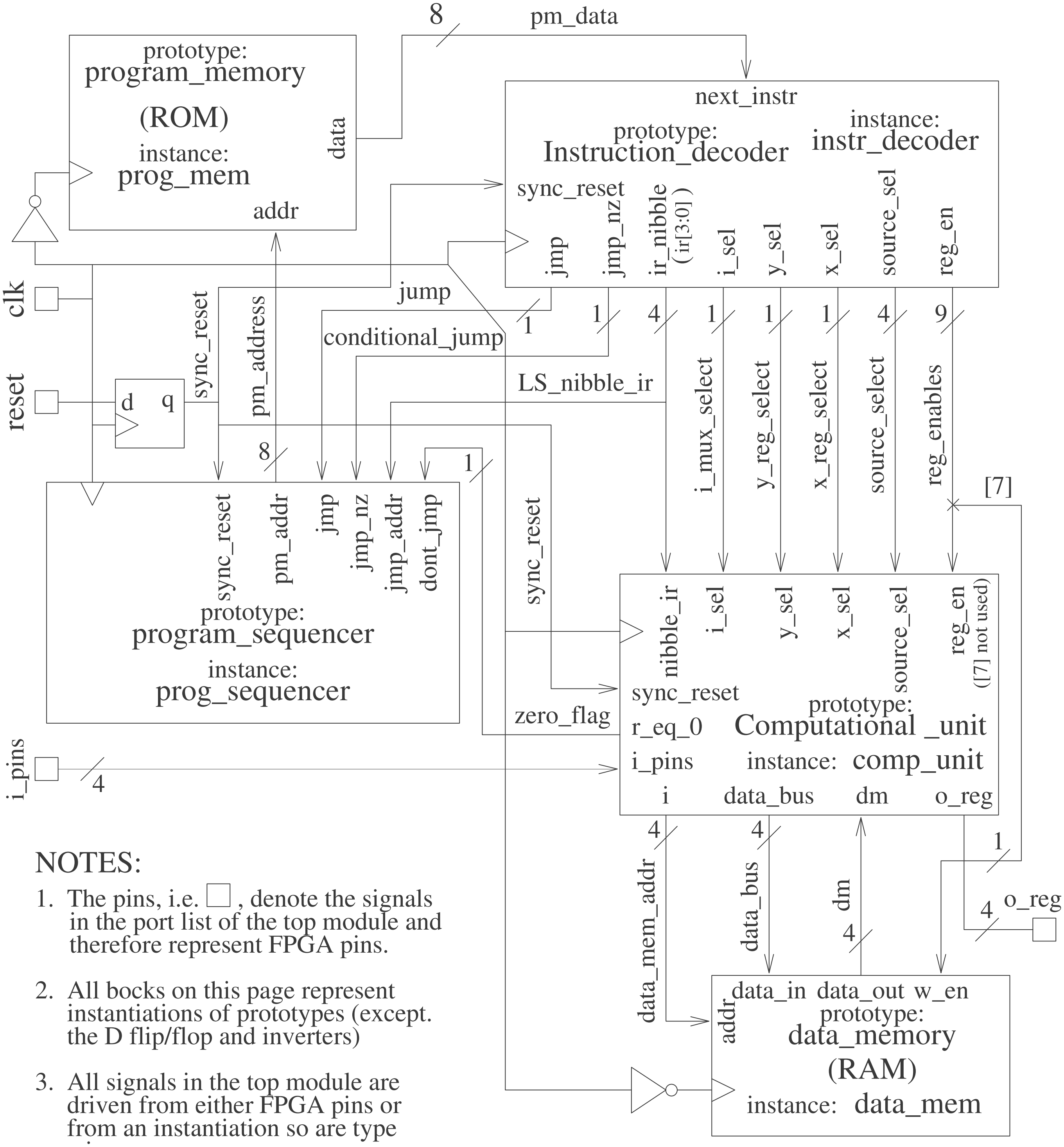
Data Memory

Data memory is connected to the data bus and is treated like a register by the computational unit. The address for the data memory is the i register. A read or write to dm is treated the same as a read or write to any other register.

1.2.9 Data Memory

Data memory is synchronous random access memory. Its inputs are registered (both data and address) and it outputs are not. It is write-enabled with the "dm" register enable and clocked with the negative edge of "clk". Its address is the output of the i register.

1.3 Schematic Diagrams for the CME341 Microprocessor



NOTES:

1. The pins, i.e. □ , denote the signals in the port list of the top module and therefore represent FPGA pins.

2. All bocks on this page represent instantiations of prototypes (except. the D flip/flop and inverters)

3. All signals in the top module are driven from either FPGA pins or from an instantiation so are type wire.

4. The two inverters can be implemented implicitly inside the conection list with the association ".clk(~clk)".

Microprocessor page 1/5
Block Diagram of Microprocessor

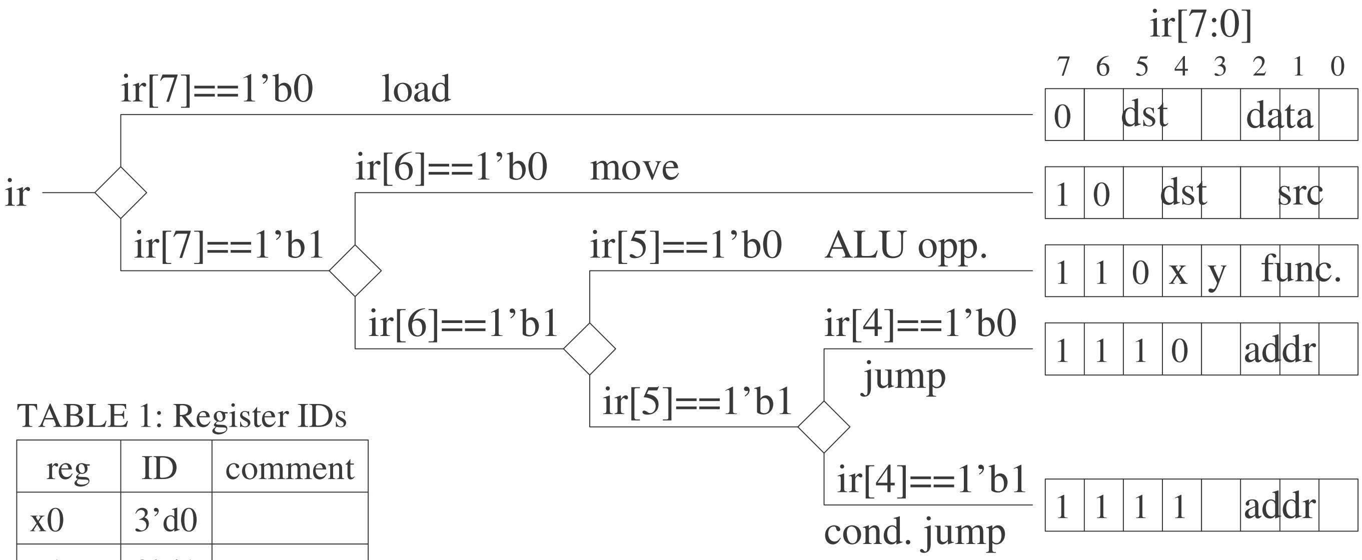


TABLE 1: Register IDs

reg	ID	comment
x0	3'd0	
x1	3'd1	
y0	3'd2	
y1	3'd3	
r	3'd4	src field
o_reg	3'd4	dst field
m	3'd5	
i	3'd6	
dm	3'd7	

Notes: when ID 3'd4 is used in the ''src'' field it refers to the ''r'' register. When it is used in the ''dst'' field it refers to the''o_reg'' register.

Special Cases for the Move Instruction:

(i.e. exceptions to the rule)

1. Move i_pins to register with ID 'dst'

If the ''src'' and ''dst'' fields in a move instruction are such ''dst==src'' and ''dst'' is not the ID for o_reg, then the move instruction moves ''i_pins'' to the register with ID ''dst''.

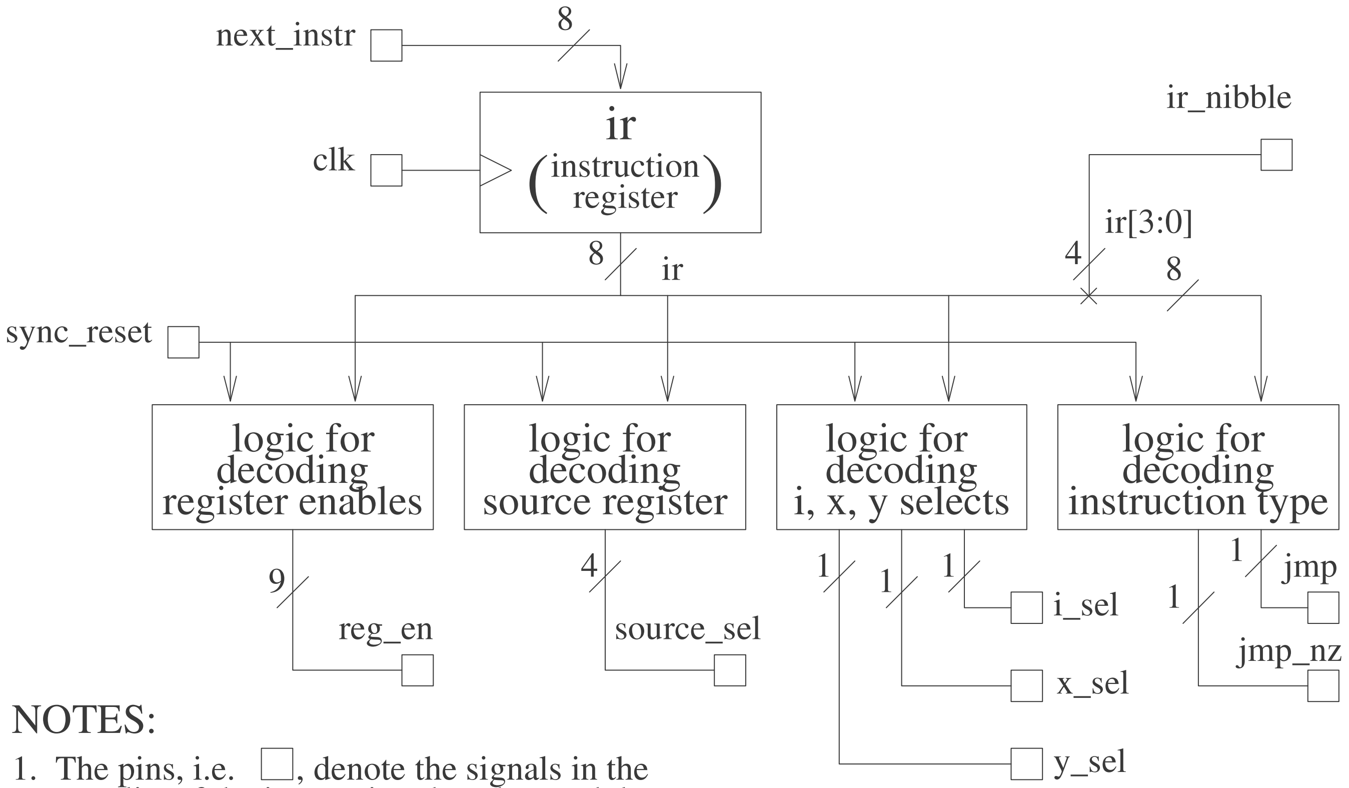
2. Move r to o_reg

If the ''src'' and ''dst'' fields in a move instruction are such ''dst==src==3'd4'', then 'r' is moved to o_reg.

Auto Increment of i register:

The i register is automatically incremented by the value in the the m register upon execution of any load or move instructions where ''dm'' is in the ''src'' or ''dst'' field except the move instructions where ''dm'' is the ''src'' and ''i'' is the ''dst''.

Microprocessor page 2/5
Instruction set for the Microprocessor


next_instr
8
ir
(instruction register)
clk
ir_nibble
ir[3:0]
4
8
8
ir
sync_reset
logic for decoding register enables
logic for decoding source register
logic for decoding i, x, y selects
logic for decoding instruction type
9
reg_en
4
source_sel
1
1
1
i_sel
x_sel
y_sel
1
1
jmp
jmp_nz

NOTES:

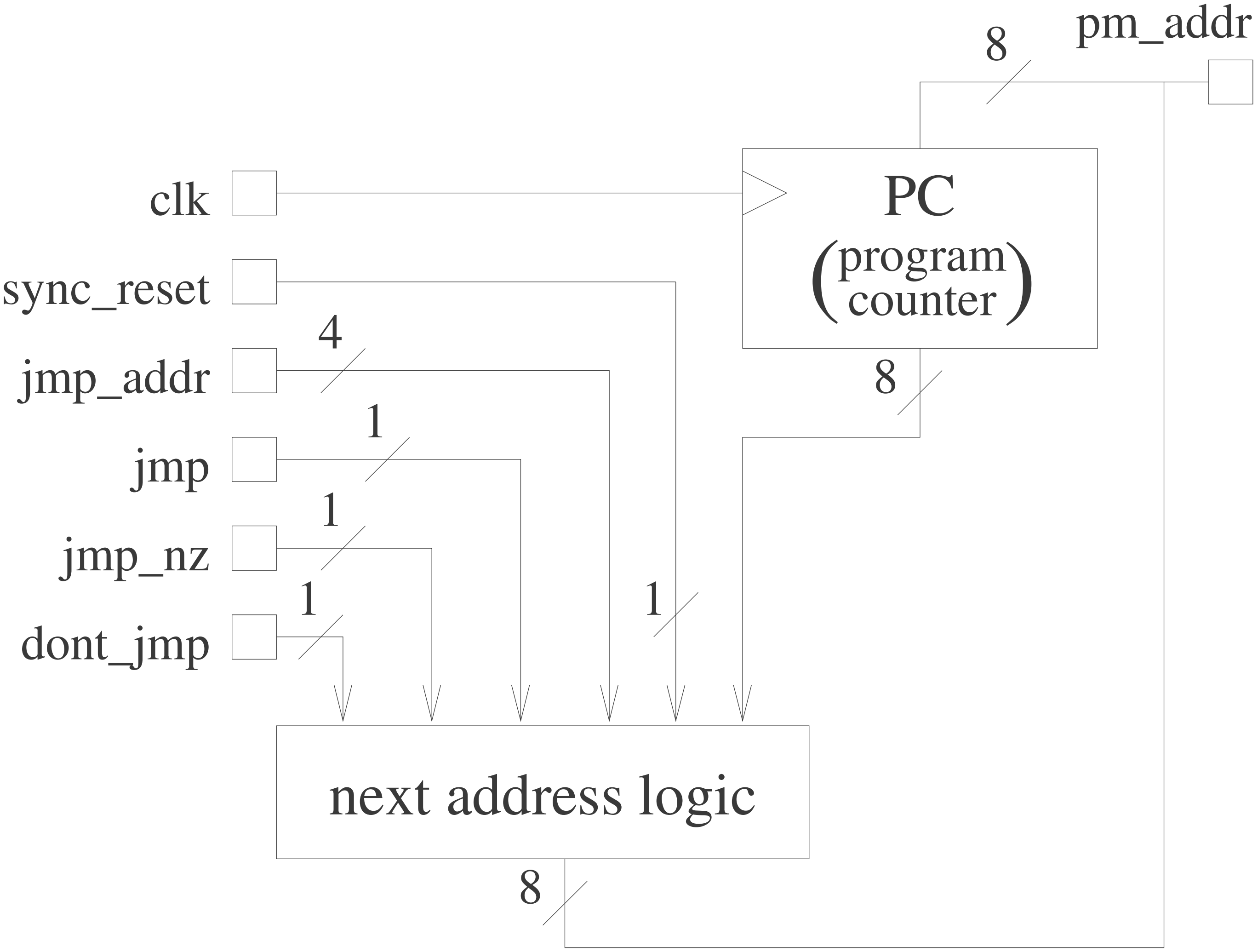
1. The pins, i.e. □, denote the signals in the port list of the instruction decoder module.

2. While sync_reset is high: reg_en must be 9'H1FF, source_sel must be 4'd10, i_sel, x_sel and y_sel must be 1'b0, jmp and jmp_nz must be 1'b0

TABLE 1: Assignments for reg_en[8:0]

reg_en								
[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
o_reg	dm	i	m	r	y1	y0	x1	x0

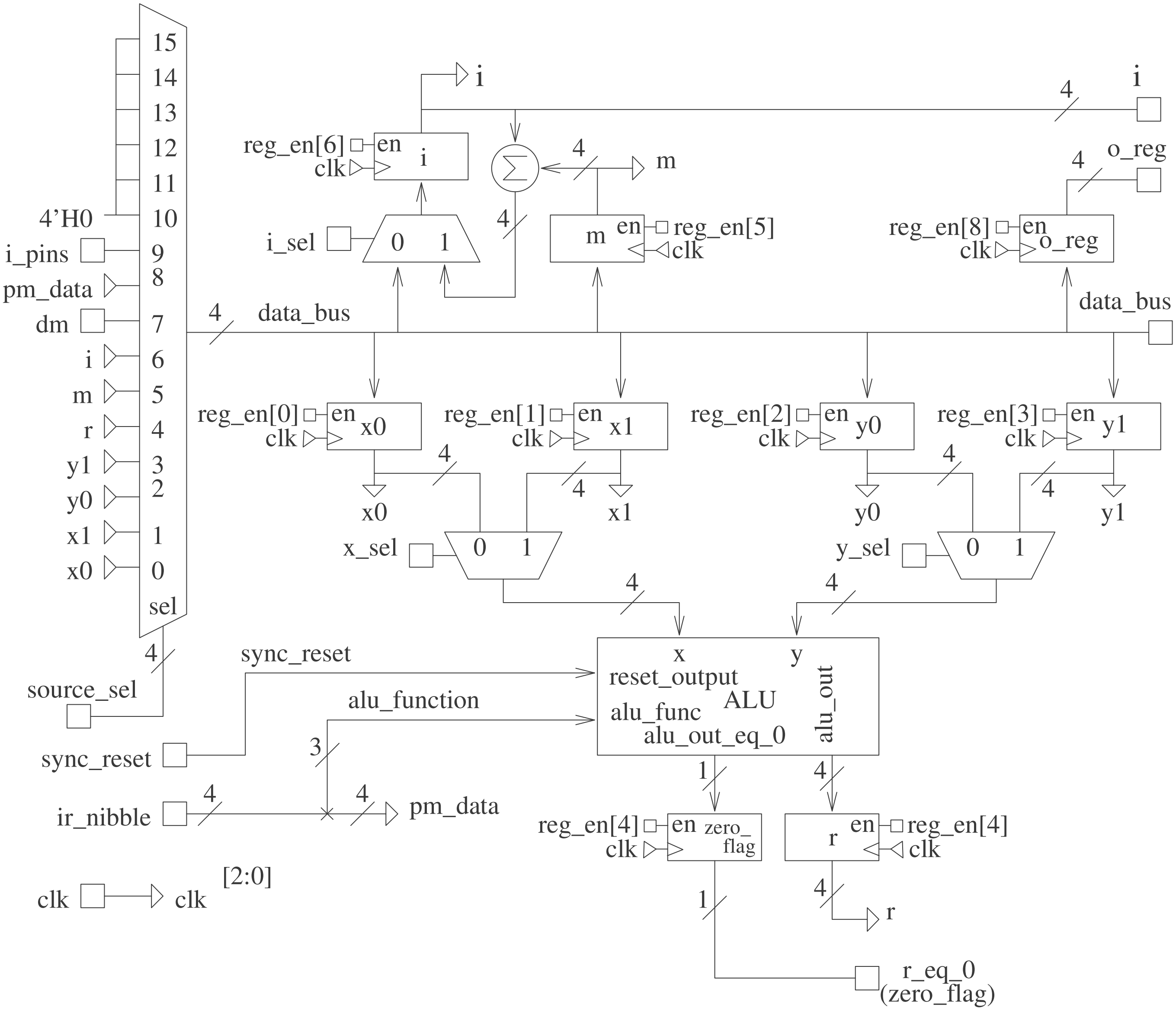
Microprocessor page 3/5
Block Diagram of Instruction Decoder



NOTES: The pins, i.e. □ , denote signals from the port list of the program sequencer module.

Microprocessor page 4/5

Block diagram of program sequencer



NOTES: 1. The pins, i.e. □, denote signals in the port list of the computational unit module

2. While sync_reset is high, alu_out must be 4'H0 and alu_out_eq_0 must be 1'b1

Microprocessor page 5/5

Block Diagram of Computational Unit