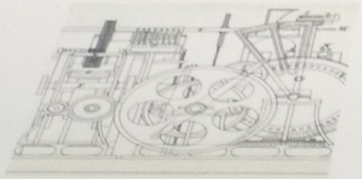
Today's topics

Components of a computer

- CPU
- (Memory)
- (Input)
- (Output)

The world's first working Difference Engines



Drawing of the original Scheutz Difference Engine, 1830

Charles Babbage invented the Difference Engine in 1821, but never built a full example. The only complete Difference Engine built during Babbage's lifetime was made by Swedish engineers George and Edvard Scheutz.

Inspired by Babbage's ideas, and encouraged by Babbage himself, they printed the first ever mathematical tables calculated by machine. The Scheutz brothers went on to sell two further Difference Engines of which this is the second.

Practical and financial problems meant that Babbage and his engineer Joseph Clement completed only about a seventh of Babbage's original mechanism, which is on display in the *Making the Modern World* gallery on the ground floor. Known as Difference Engine No. 1, it is one of the finest examples of precision engineering from nineteenth-century England.

On display in this case:

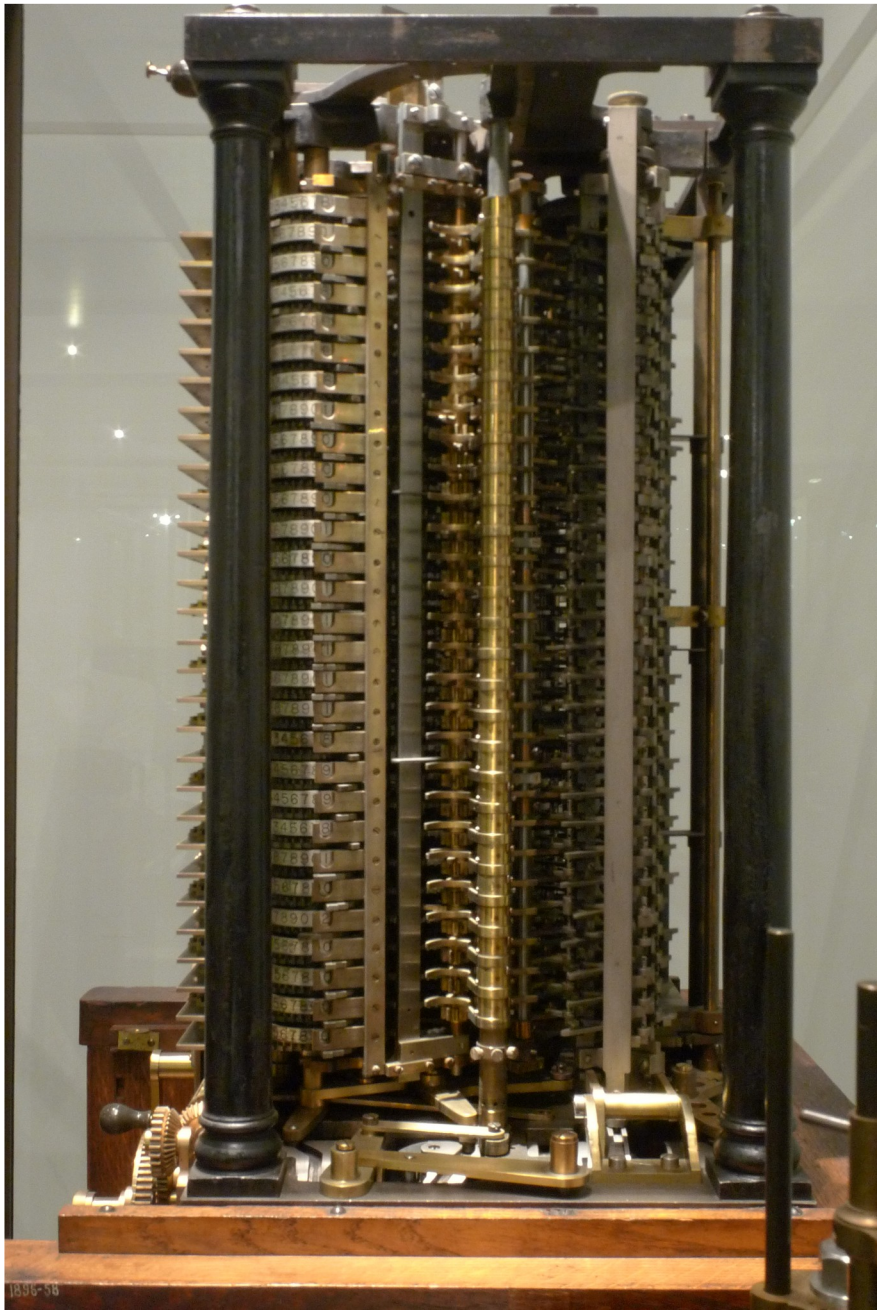
1. Third Scheutz Difference Engine, inspired by Babbage's work, 1859
(1914-132/1)

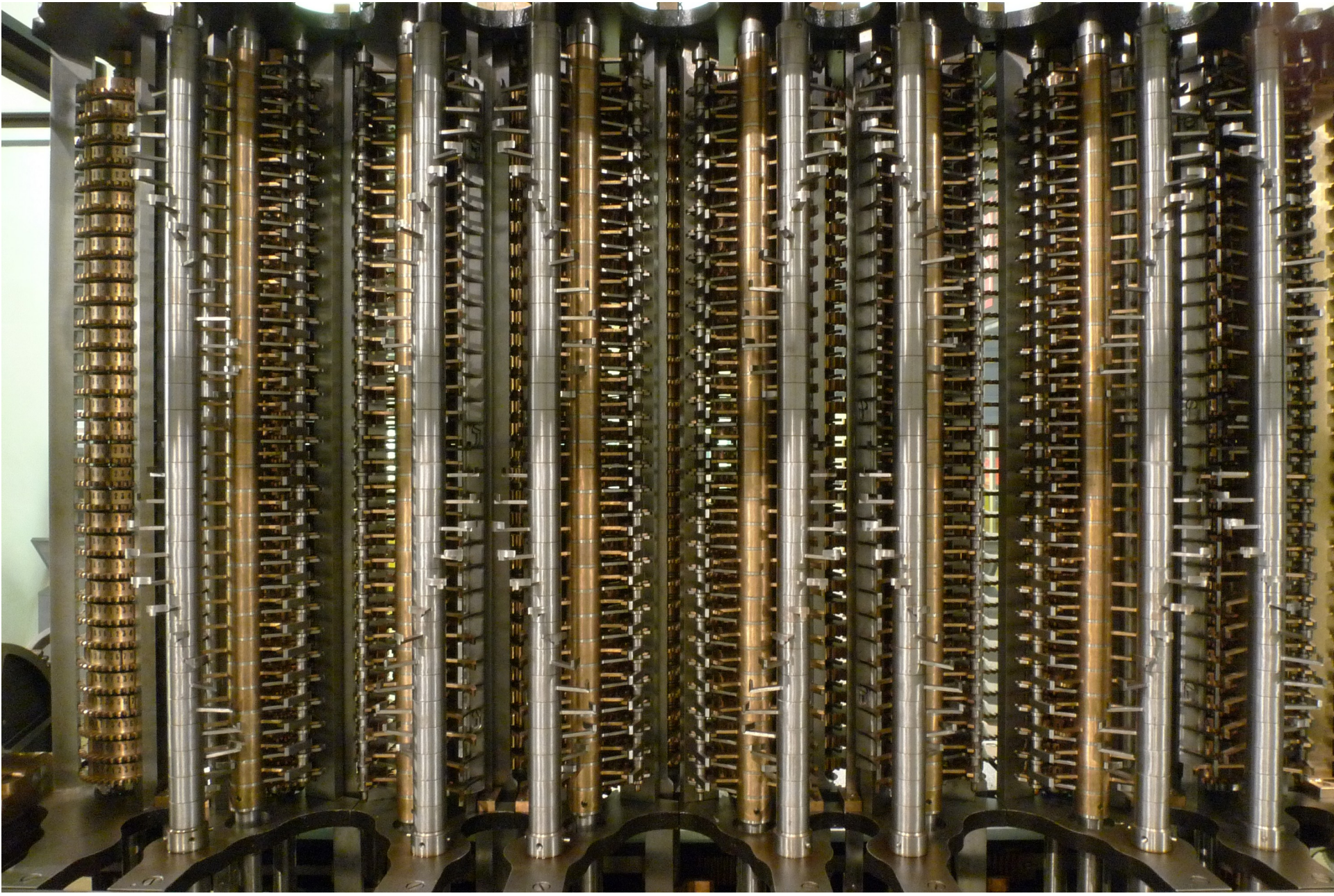
2. Scheutz stereotypes and moulds, c. 1866
To reduce the risk of typesetting errors the Scheutz engines impressed results onto strips of papier-mâché or soft metal. These were then removed from the machine and used as moulds to make stereotype plates for use in a printing press.
(1914-123 pt 9)

Charles Babbage

Calculating Machine

London



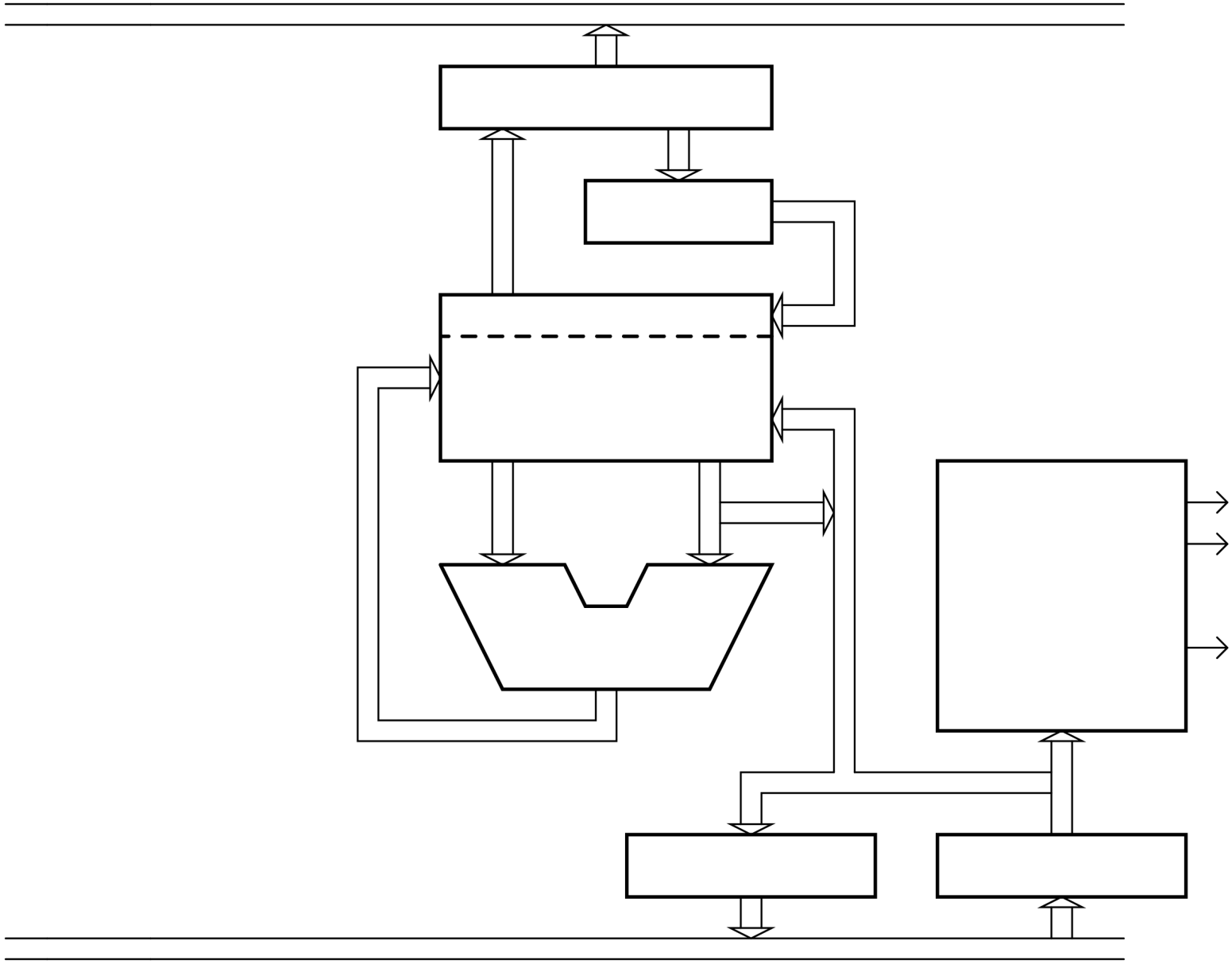


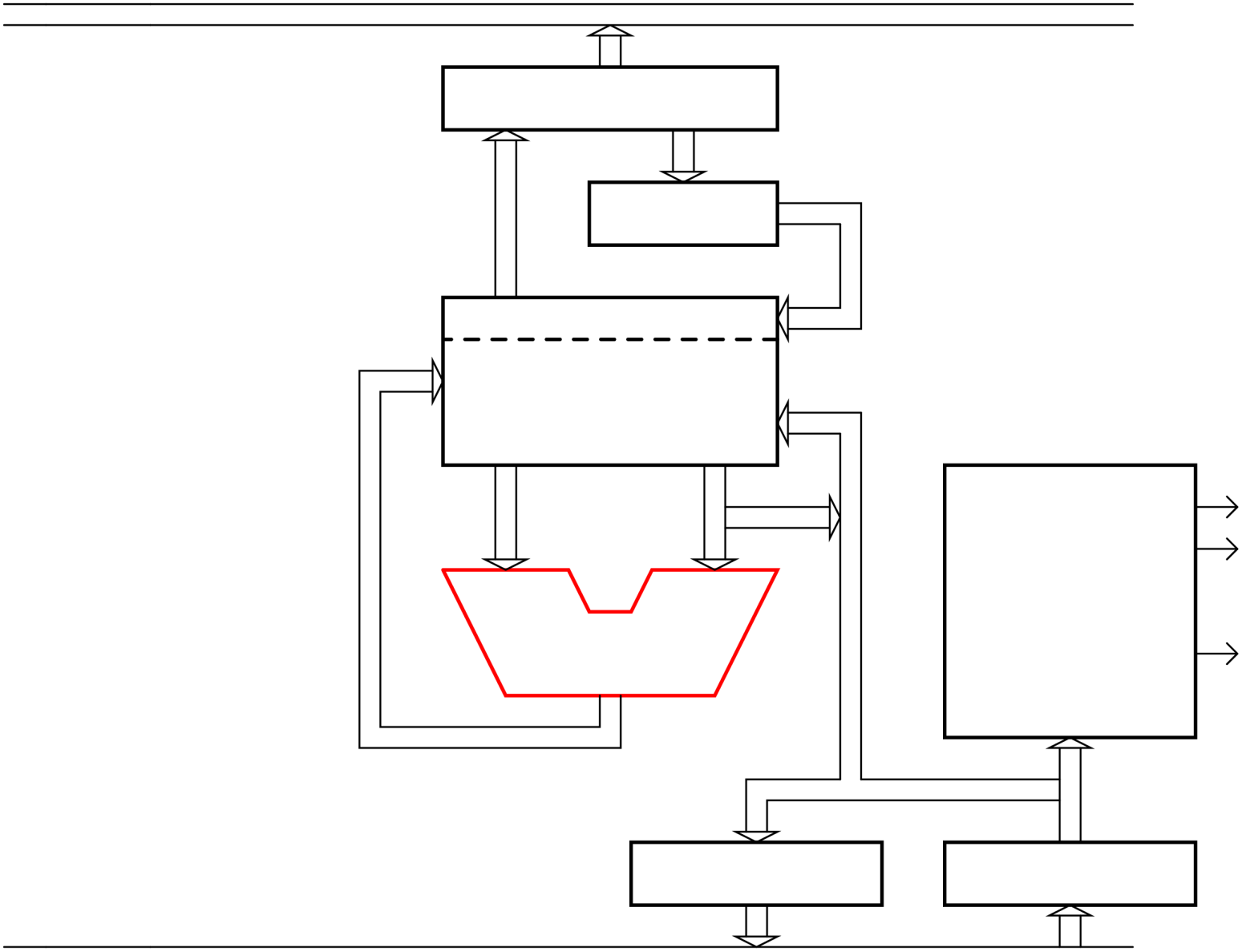
Inside the Processor (CPU)

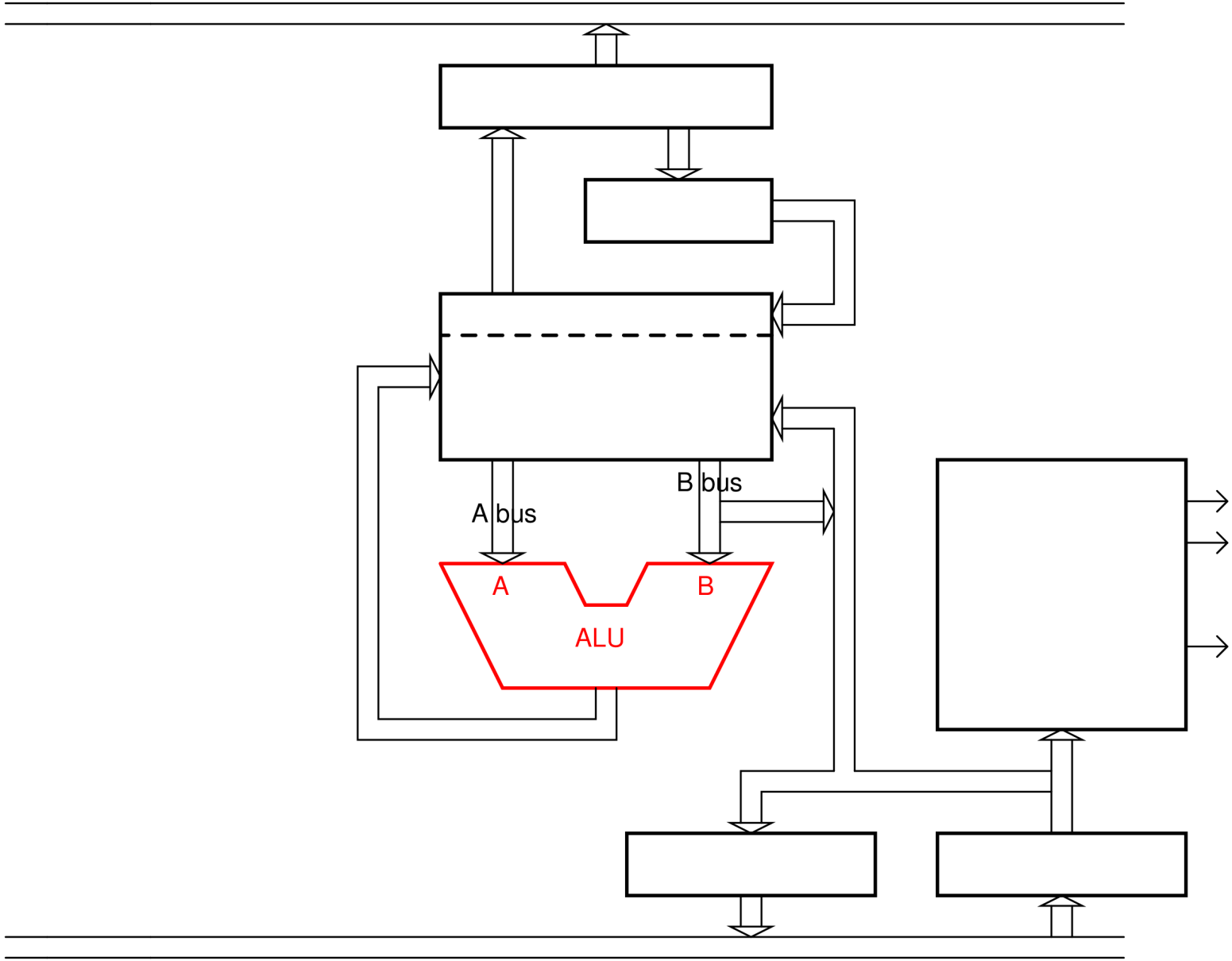
Datapath: performs operations on data
Control: sequences datapath, memory, ...
Cache memory

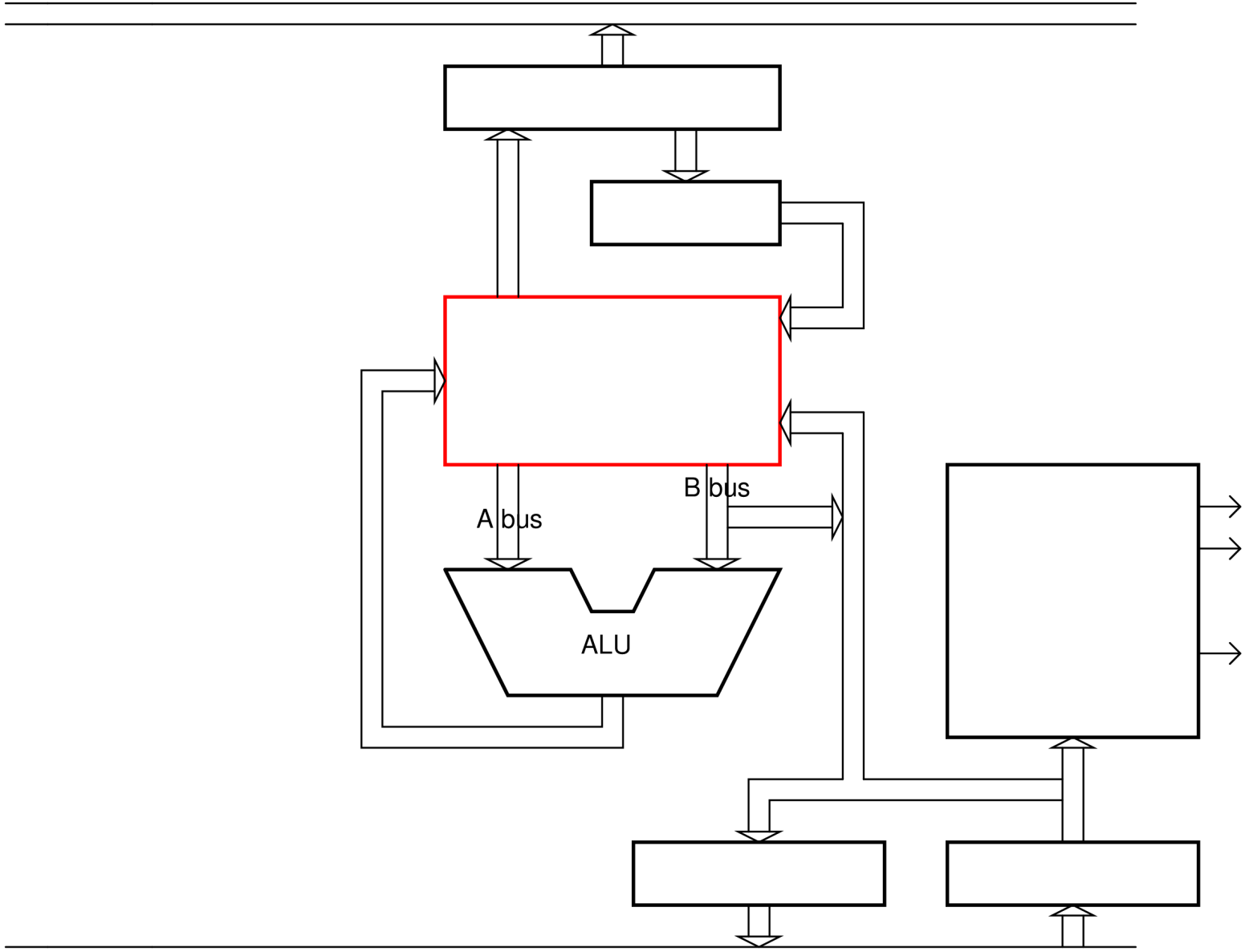
- Small fast SRAM memory for immediate access to data

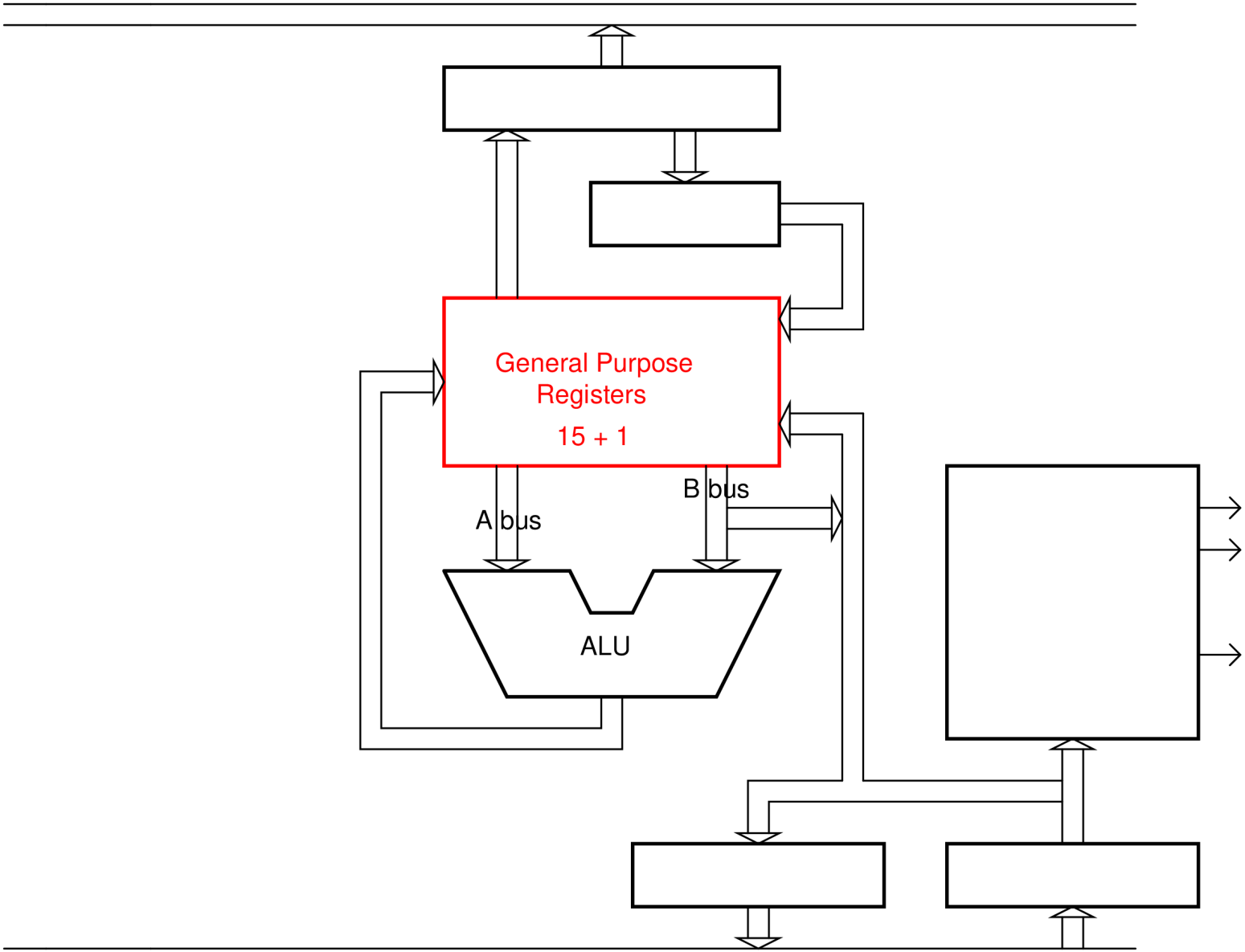
The Aamodt Simple Machine

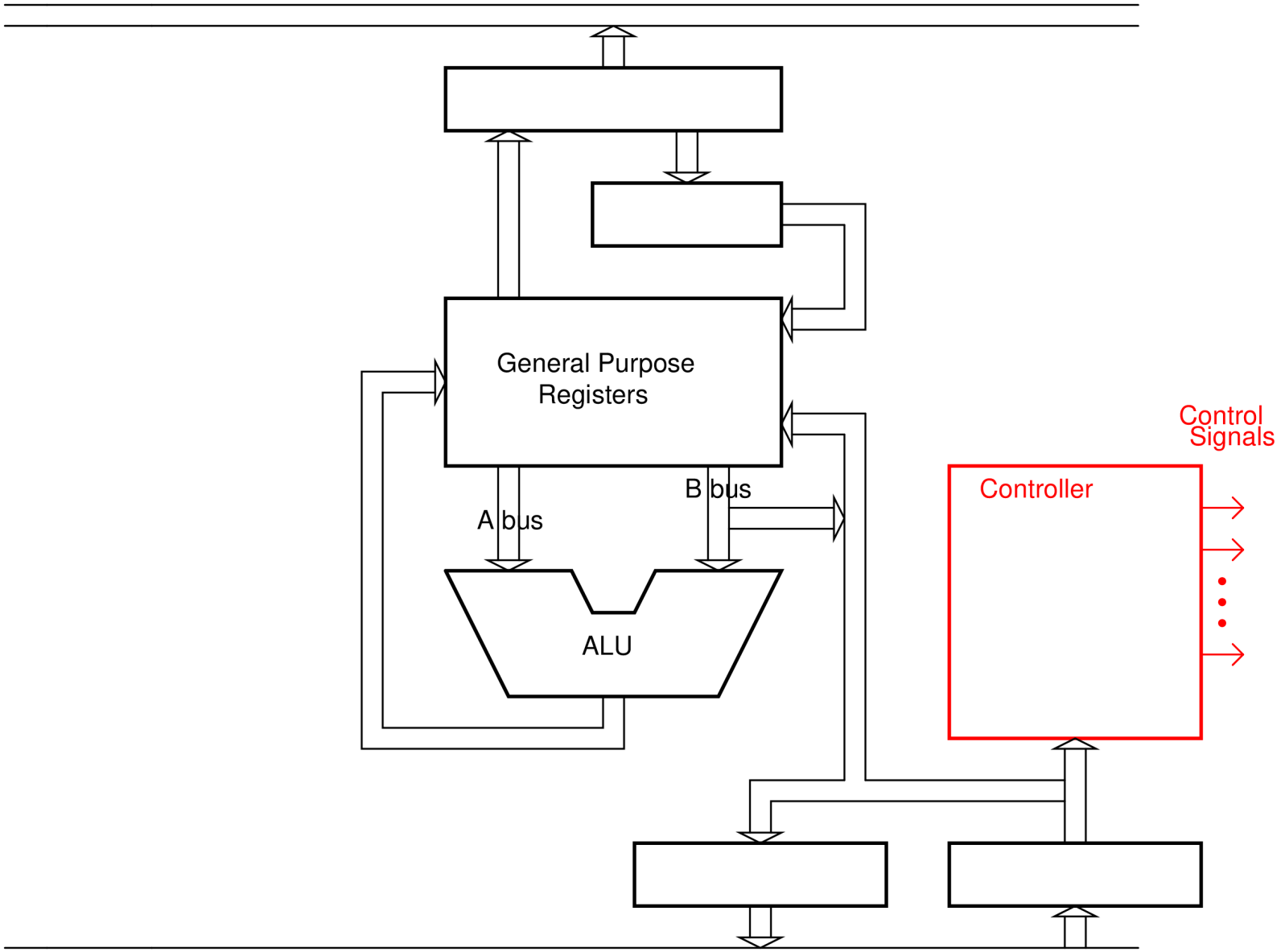


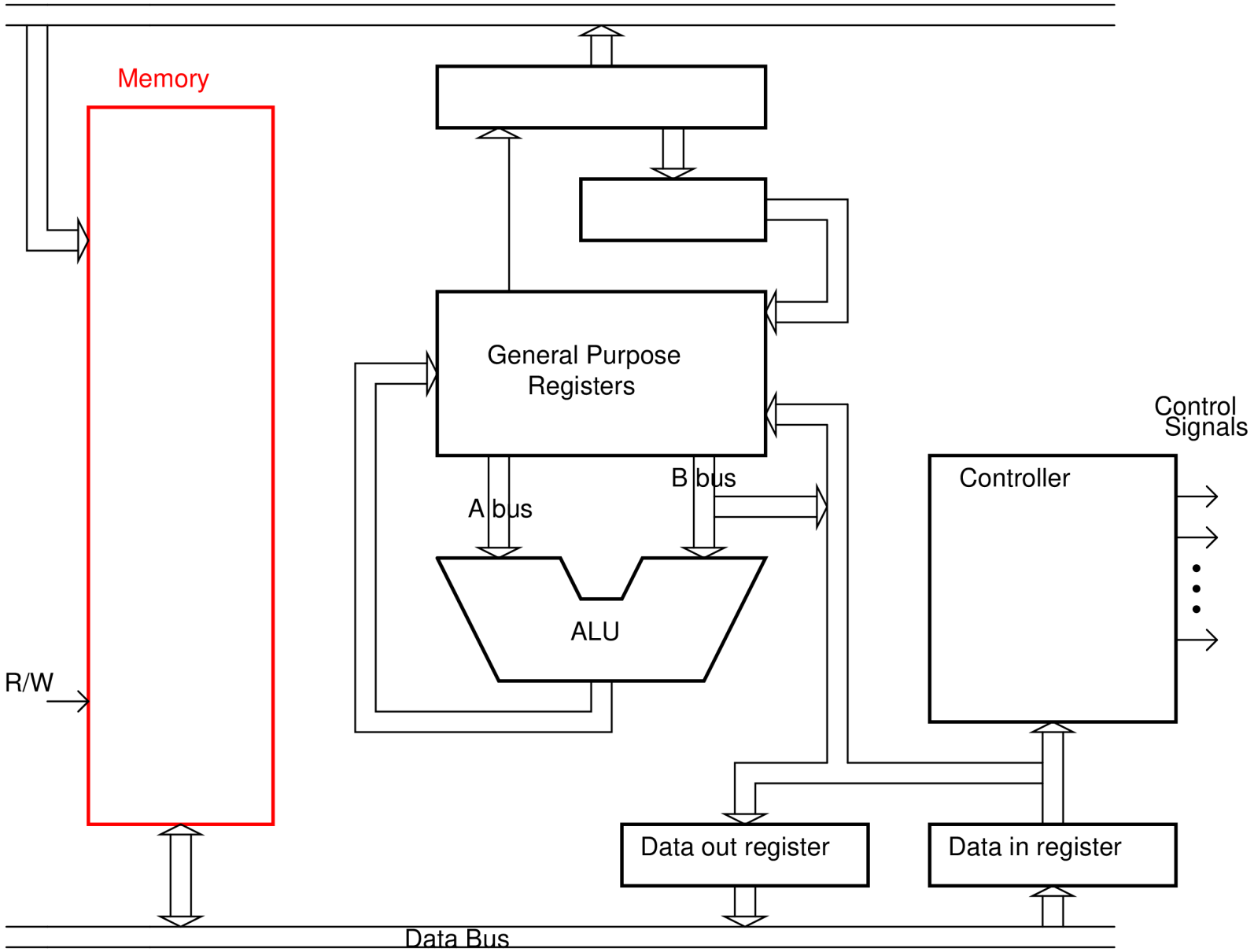


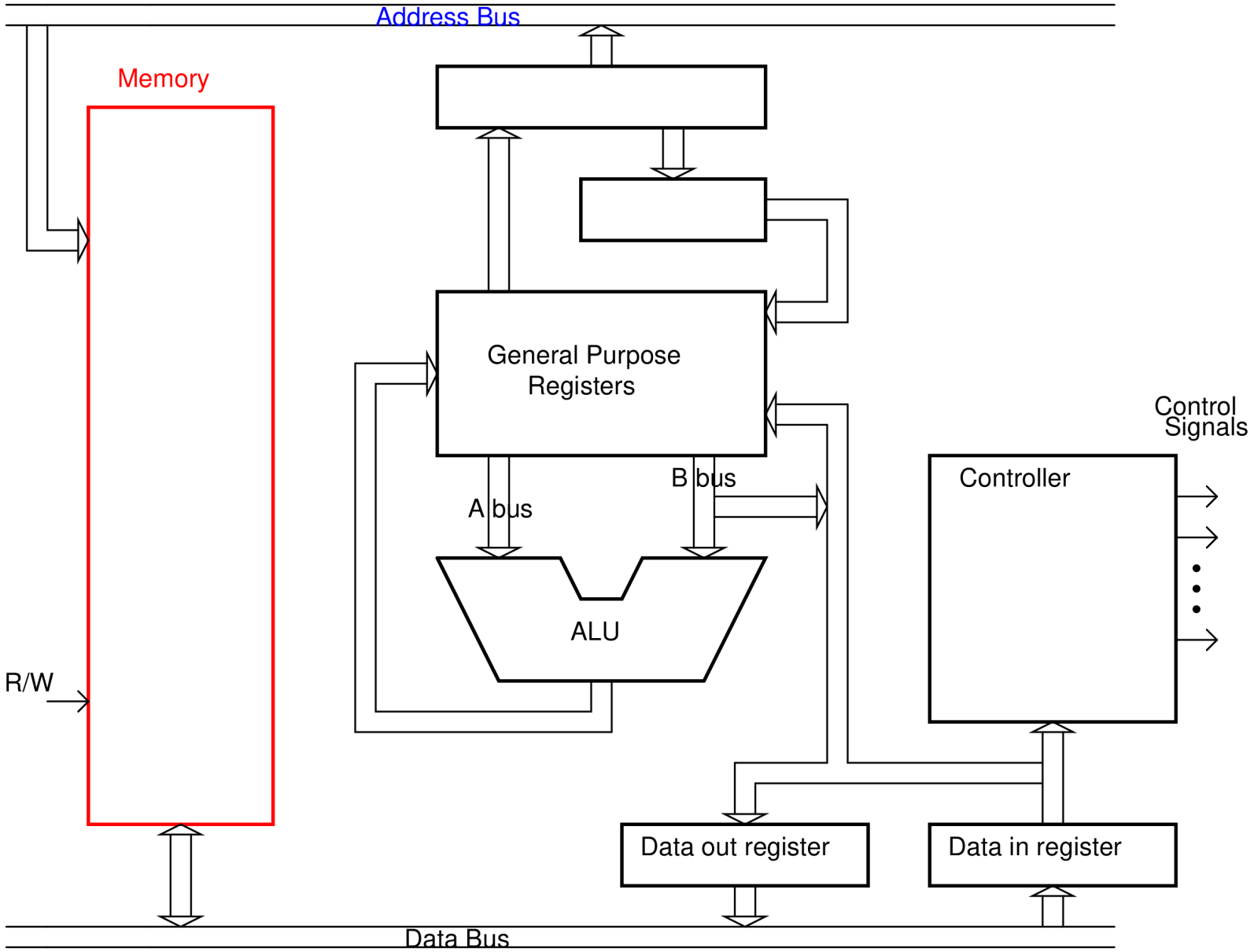

B bus
A bus
A
B
ALU

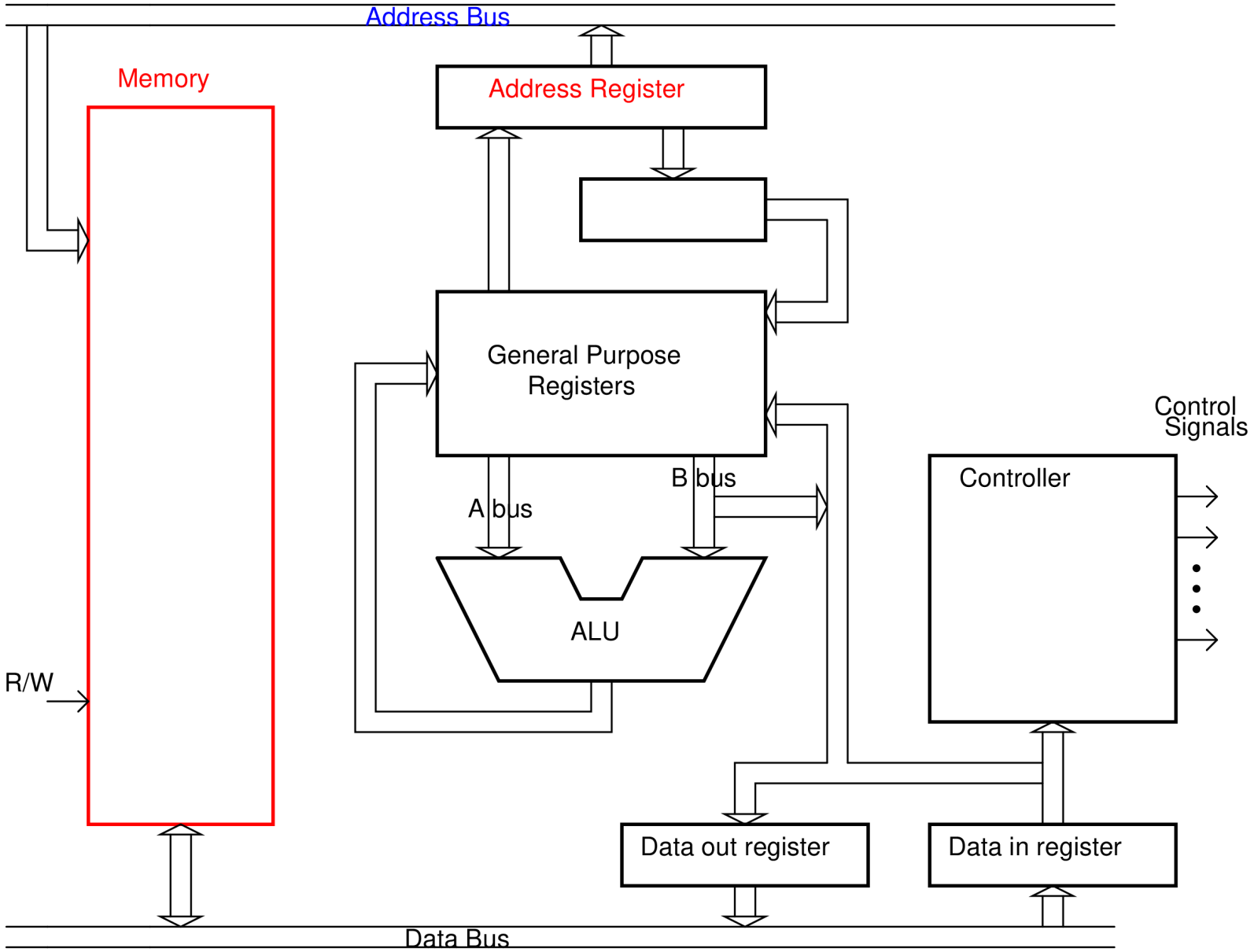

A bus
B bus
ALU

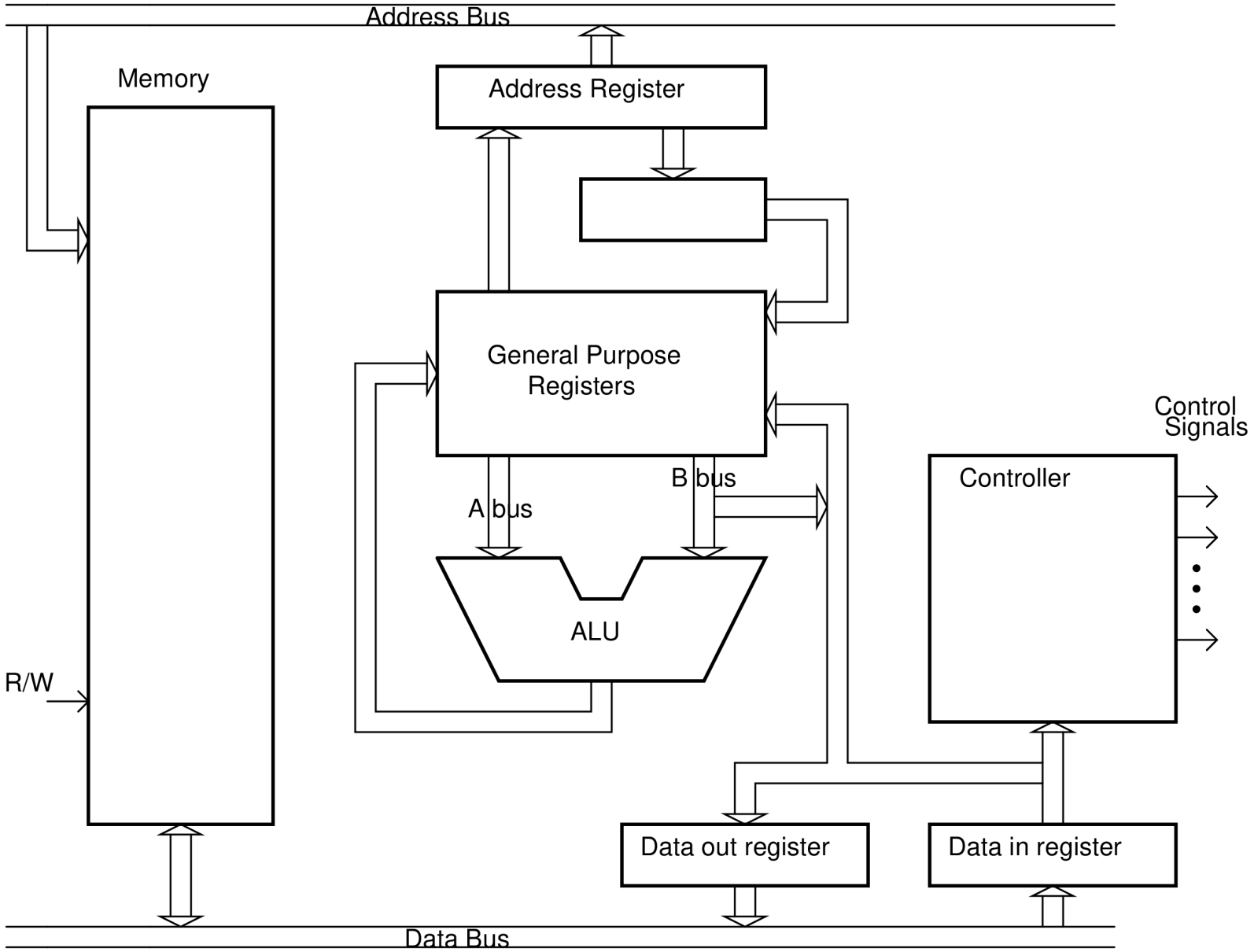

General Purpose
Registers
15 + 1
A bus
B bus
ALU

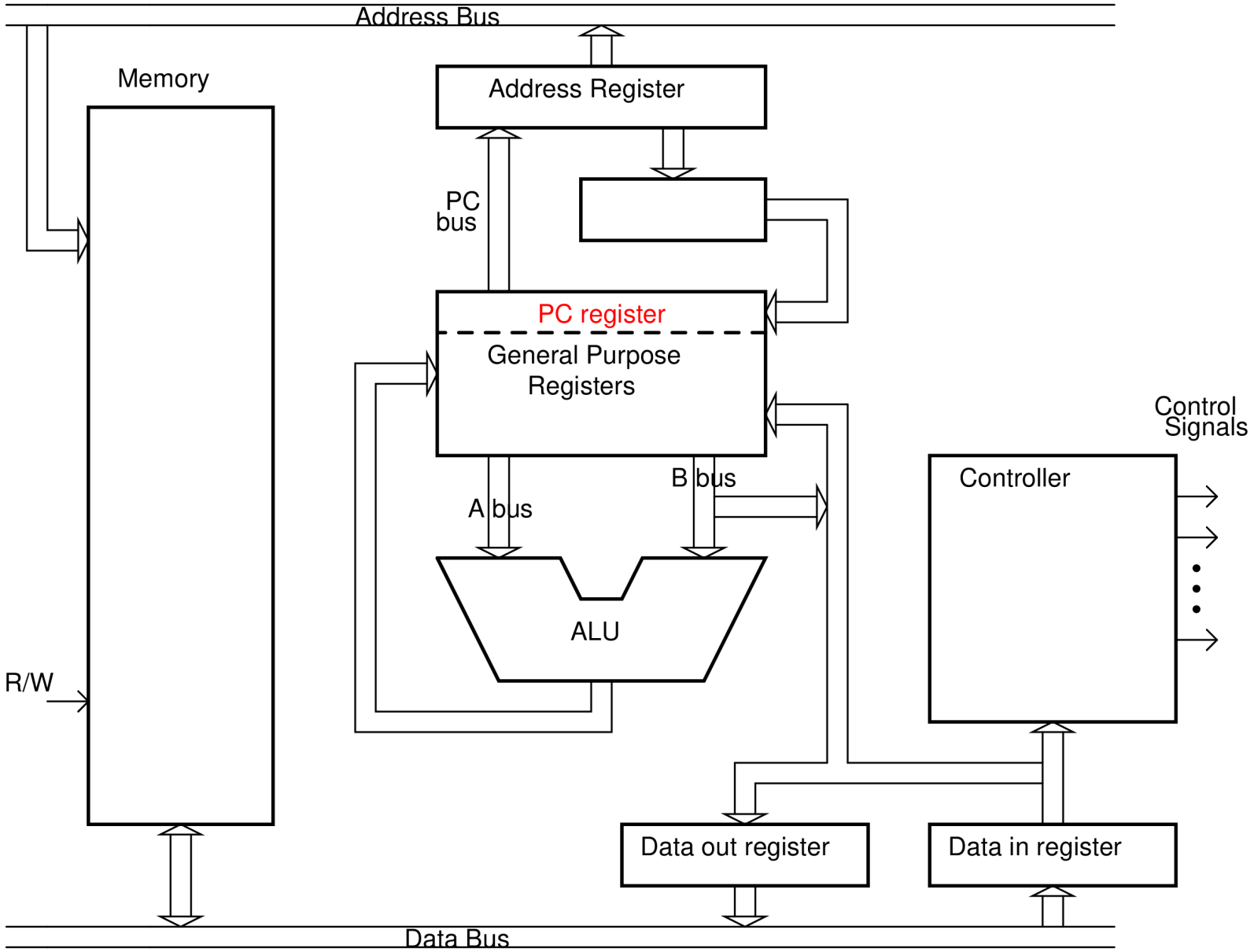

General Purpose
Registers
A bus
B bus
ALU
Control
Signals
Controller

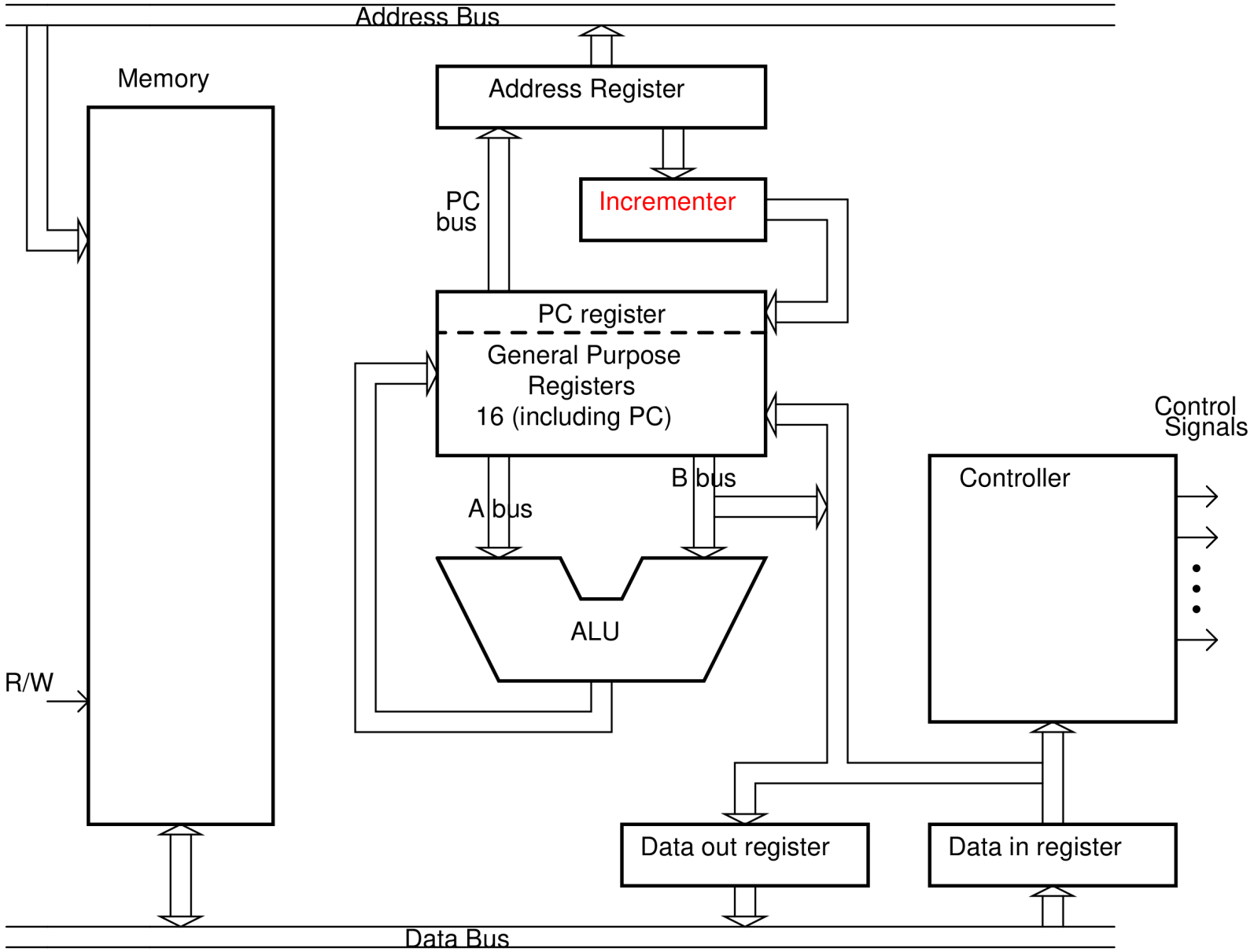

Memory
General Purpose
Registers
A bus
B bus
ALU
Controller
Control
Signals
R/W
Data out register
Data in register
Data Bus

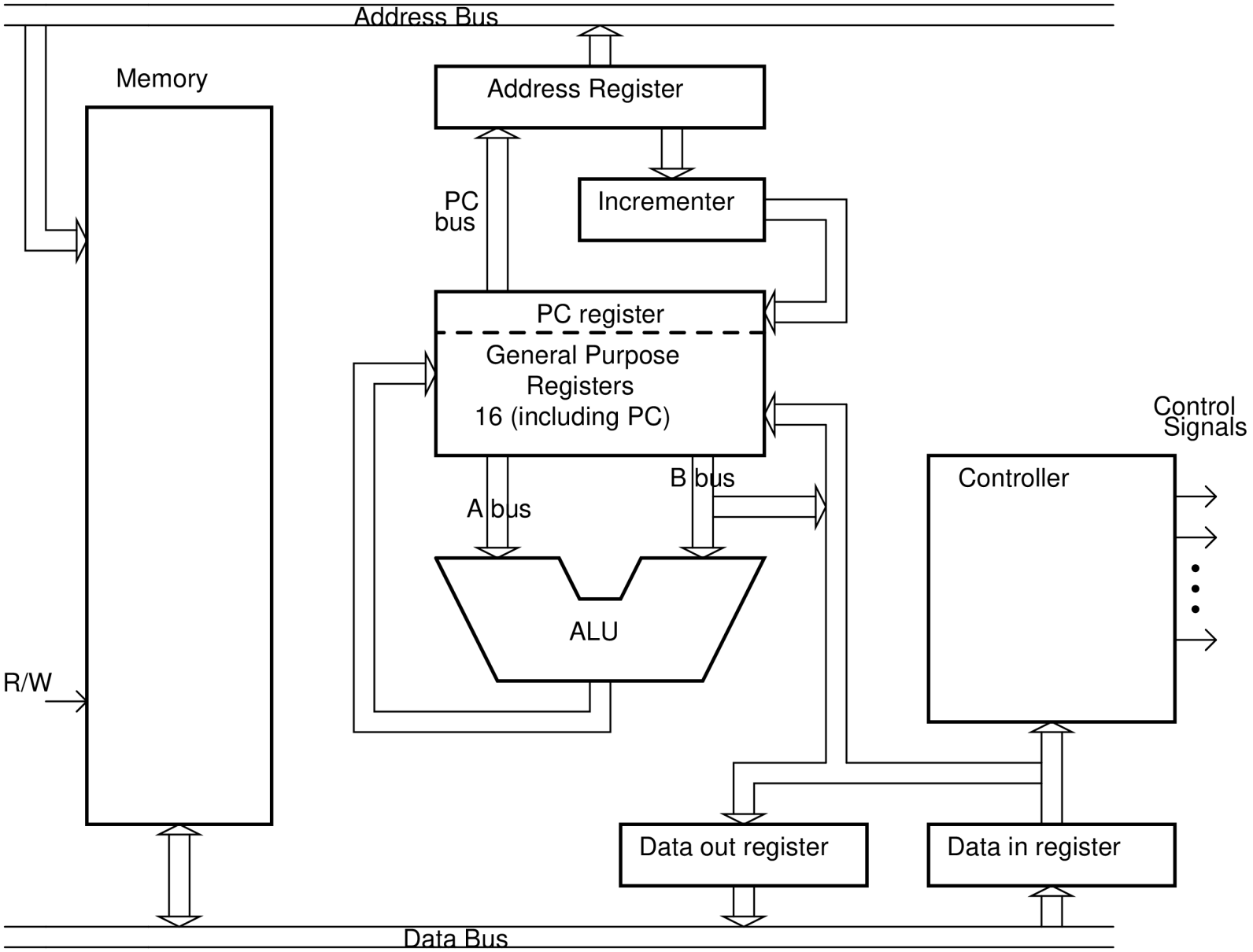

Address Bus
Memory
General Purpose
Registers
A bus
B bus
ALU
Control
Signals
Controller
R/W
Data out register
Data in register
Data Bus

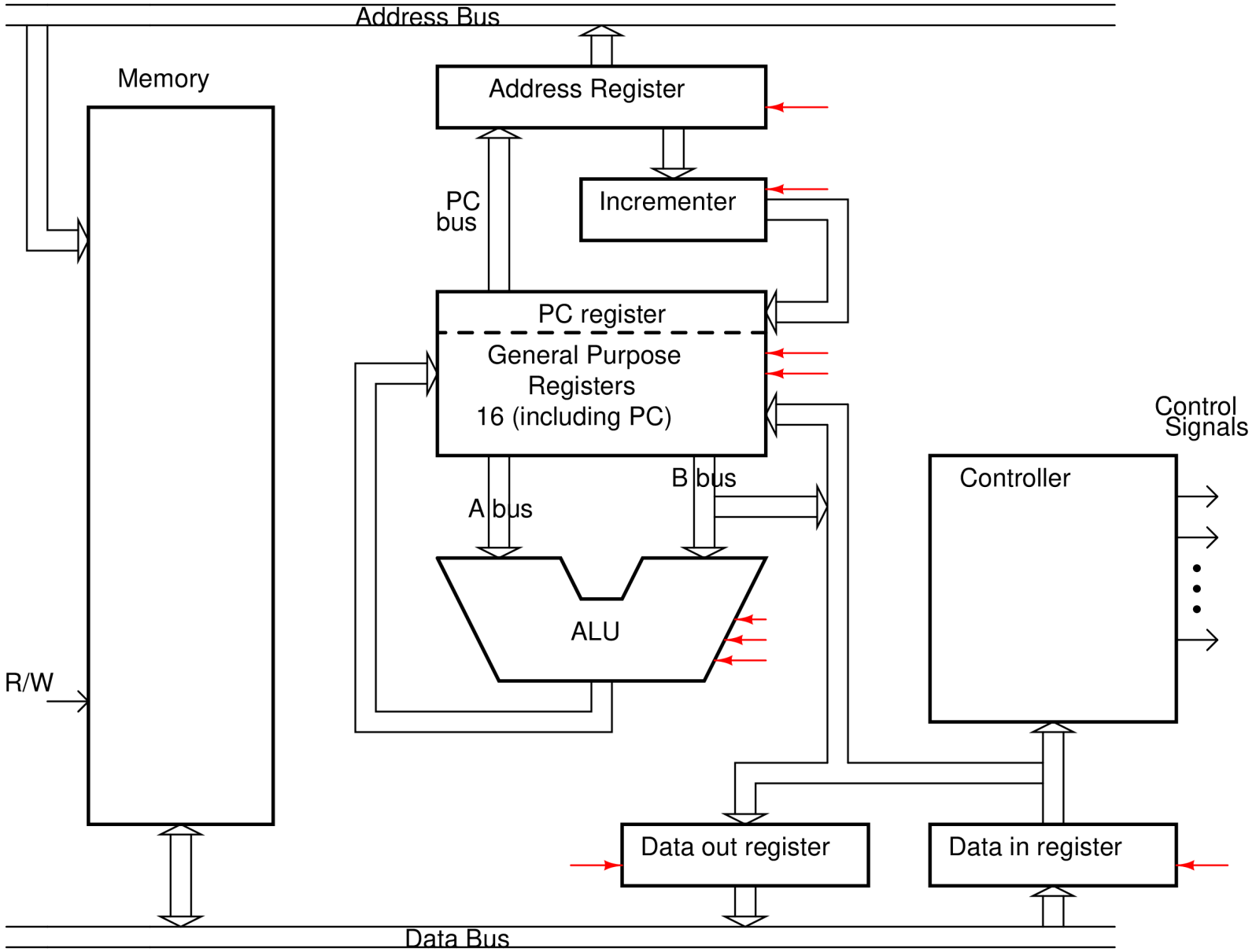

Address Bus
Memory
Address Register
General Purpose
Registers
A bus
B bus
ALU
Control
Signals
Controller
R/W
Data out register
Data in register
Data Bus


Address Bus
Memory
Address Register
General Purpose
Registers
A bus
B bus
ALU
Control
Signals
Controller
R/W
Data out register
Data in register
Data Bus


Address Bus
Memory
Address Register
PC
bus
PC register
General Purpose
Registers
Control
Signals
B bus
A bus
Controller
ALU
R/W
Data out register
Data in register
Data Bus


Address Bus
Memory
Address Register
PC
bus
Incrementer
PC register
General Purpose
Registers
16 (including PC)
Control
Signals
B bus
A bus
Controller
ALU
R/W
Data out register
Data in register
Data Bus


Address Bus
Memory
Address Register
PC
bus
Incrementer
PC register
General Purpose
Registers
16 (including PC)
Control
Signals
B bus
A bus
Controller
ALU
R/W
Data out register
Data in register
Data Bus


Address Bus
Memory
Address Register
PC
bus
Incrementer
PC register
General Purpose
Registers
16 (including PC)
B bus
A bus
ALU
Control
Signals
Controller
R/W
Data out register
Data in register
Data Bus

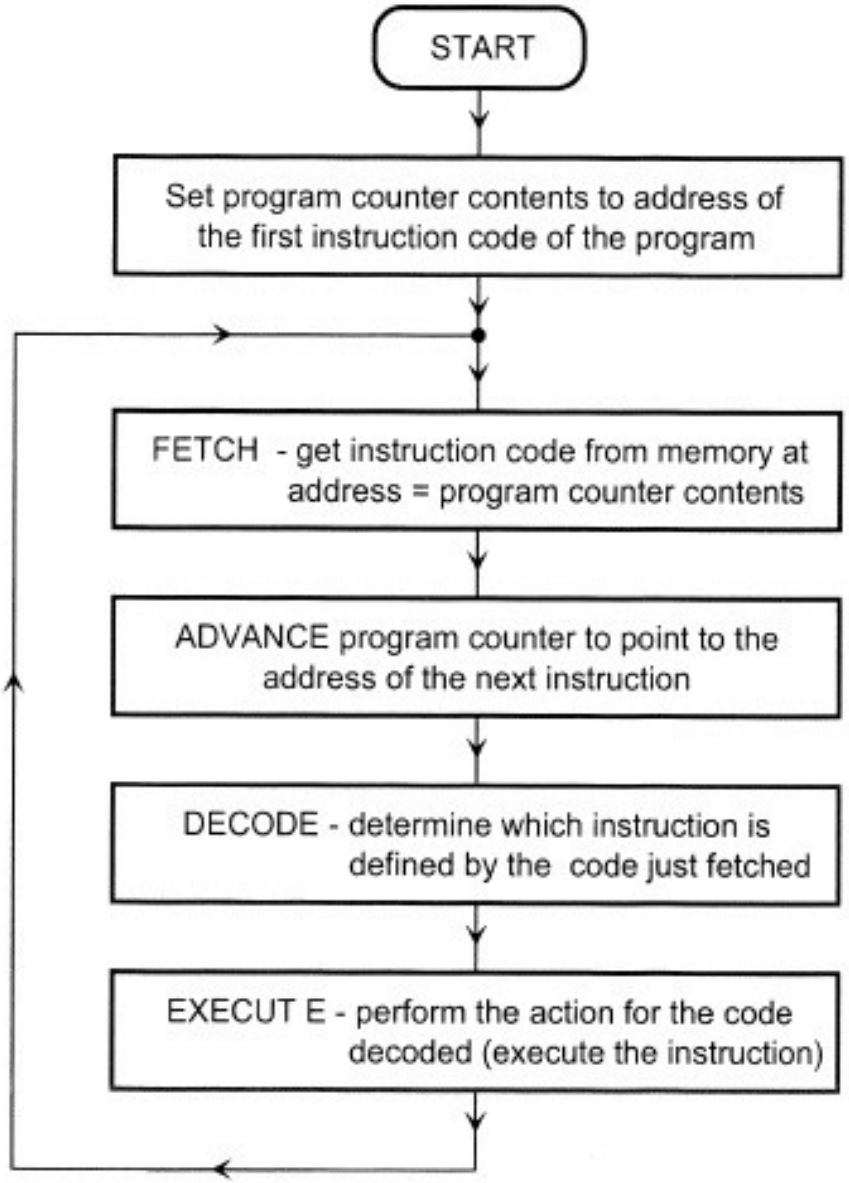


Figure 3.3 Detailed von Neumann cycle sequence

Syntax:
ADD{<cond>}{S} <Rd>, <Rn>, <shifter_operand>

Encoding:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cond				0	0	I	0	1	0	0	S	Rn				Rd				shifter operand											

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Multiply (accumulate)	cond				0	0	0	0	0	0	A	S	Rd				Rn				Rs				1	0	0	1	Rm			
Multiply (accumulate) long	cond				0	0	0	0	1	U	A	S	Rd_MSW				Rd_LSW				Rn				1	0	0	1	Rm			
Branch and exchange	cond				0	0	0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	Rn			
Single data swap	cond				0	0	0	1	0	B	0	0	Rn				Rd				0	0	0	0	1	0	0	1	Rm			
Halfword data transfer, register offset	cond				0	0	0	P	U	0	W	L	Rn				Rd				0	0	0	0	1	0	1	1	Rm			
Halfword data transfer, immediate offset	cond				0	0	0	P	U	1	W	L	Rn				Rd				offset				1	0	1	1	offset			
Signed data transfer (byte/halfword)	cond				0	0	0	P	U	B	W	L	Rn				Rd				addr_mode				1	1	H	1	addr_mode			
Data processing and PSR transfer	cond				0	0	I	opcode				S	Rn				Rd				operand2											
Load/store register/unsigned byte	cond				0	1	I	P	U	B	W	L	Rn				Rd				addr_mode											
Undefined	cond				0	1	1																					1				
Block data transfer	cond				1	0	0	P	U	0	W	L	Rn				register list															
Branch	cond				1	0	1	L	offset																							
Coprocessor data transfer	cond				1	1	0	P	U	N	W	L	Rn				CRd				CP#				offset							
Coprocessor data operation	cond				1	1	1	0	CP opcode				CRn				CRd				CP#				CP			0	CRm			
Coprocessor register transfer	cond				1	1	1	0	CP opc			L	CRn				Rd				CP#				CP			1	CRm			
Software interrupt	cond				1	1	1	1	ignored by processor																							

Inside the Processor

A12 processor